Niğde Ömer Halisdemir Üniversitesi Mühendislik Bilimleri Dergisi
Niğde Ömer Halisdemir University Journal of Engineering Sciences

Araştırma makalesi / Research article

www.dergipark.org.tr/tr/pub/ngumuh / www.dergipark.org.tr/en/pub/ngumuh

Electricity load forecasting models based on LSTM and GRU with their bidirectional recurrent neural networks

Çift yönlü tekrarlayan sinir ağlarına sahip LSTM ve GRU tabanlı elektrik yükü tahmin modelleri

Khalid Alhashemi[1,*], **Okkes Tolga Altinoz**[2]

[1,2]*Ankara University, School of Natural and Applied Sciences, 06830, Ankara Türkiye*

Abstract

Accurate electricity load forecasting is crucial for power system planning, reliability, and sustainability, enabling more efficient markets and reduced greenhouse gas emissions. This study leverages deep learning algorithms, specifically bidirectional recurrent neural networks, to develop a unified model for predicting one day-ahead electricity demand for the entire year of 2023. The model's performance was evaluated on a monthly basis, allowing for a detailed assessment of its forecasting capabilities across different time periods. Four neural network algorithms were compared: Long Short-Term Memory (LSTM), Bidirectional LSTM, Gated Recurrent Unit (GRU), and Bidirectional GRU. The GRU model demonstrated superior performance, achieving an R-squared value of 0.8526 in October and a Mean Absolute Percentage Error (MAPE) of 2.34% in March. These results highlight the potential of the proposed model as an effective tool for electricity demand forecasting, supporting the integration of renewable energy sources and enhancing grid resilience.

Keywords: Load forecasting, Long short-term memory, Gated recurrent unit, Bidirectional recurrent neural networks

Öz

Doğru elektrik yük tahmini, elektrik sistemi planlaması, güvenilirliği ve sürdürülebilirliği için çok önemlidir ve daha verimli piyasalar ile azaltılmış sera gazı emisyonlarına olanak tanır. Bu çalışma, 2023 yılının tamamı için bir gün önceden elektrik talebini tahmin etmek üzere birleşik bir model geliştirmek amacıyla, özellikle çift yönlü tekrarlayan sinir ağları olmak üzere, derin öğrenme algoritmalarından yararlanmaktadır. Modelin performansı aylık bazda değerlendirilmiş olup, farklı zaman dilimleri boyunca tahmin yeteneklerinin ayrıntılı bir değerlendirmesine olanak sağlamıştır. Dört sinir ağı algoritması karşılaştırılmıştır: Uzun Kısa Süreli Bellek (LSTM), Çift Yönlü LSTM, Gated Recurrent Unit (GRU) ve Çift Yönlü GRU. GRU modeli üstün performans sergileyerek, Ekim ayında 0.8526 R-kare değeri ve Mart ayında %2.34 Ortalama Mutlak Yüzde Hatası (MAPE) elde etmiştir. Bu sonuçlar, önerilen modelin elektrik talep tahmini için etkili bir araç olma potansiyelini vurgulamakta, yenilenebilir enerji kaynaklarının entegrasyonunu desteklemekte ve şebeke dayanıklılığını artırmaktadır.

Anahtar kelimeler: Yük tahmini, Uzun kısa süreli bellek, Geçitli tekrarlayan birim, Çift yönlü tekrarlayan sinir ağları

1 Introduction

Electric energy plays a crucial role in every aspect of modern life, from everyday activities to the implementation of national strategies for sustainable development. As global electricity consumption continues to rise, unregulated growth may lead to excessive energy production and unnecessary resource usage. To address this challenge, accurate load forecasting becomes essential. By estimating future power demand, utility providers can avoid generating surplus electricity, thereby enhancing operational efficiency and reducing financial losses. Beyond minimizing waste, load forecasting is a fundamental component of power system planning and operation. It enables grid operators to balance supply and demand more effectively, schedule generation units optimally, and ensure system stability. Moreover, accurate load predictions contribute to the reliability of electricity delivery and help prevent potential blackouts. A clear understanding of future demand also supports infrastructure development and improves the integration of renewable energy sources into the grid.

Additionally, accurate forecasting plays a pivotal role in electricity markets, where pricing and bidding strategies heavily depend on anticipated demand. Energy producers and distributors depend on these forecasts to guide strategic decisions related to trading volumes, contract management, and maintaining a competitive edge in the market. Therefore, in the face of rising demand and growing system complexity, the ability to forecast load with precision has become a strategic necessity not only for minimizing overproduction and preventing outages, but also for enhancing economic planning, mitigating environmental impact, and ensuring reliable service delivery to consumers.

The estimate of energy consumption can be calculated using a number of methods, including regression analysis processes, exponential smoothing forecast method, straight line basis depreciation calculator, and artificial intelligence technologies. The multiplicity of methods for predicting

[*] Sorumlu yazar / Corresponding author, e-posta / e-mail: kwhalhashemi@ankara.edu.tr (K. Alhashemi)

Geliş / Received: 23.12.2024 Kabul / Accepted: 29.08.2025 Yayımlanma / Published: 15.10.2025
doi: 10.28948/ngumuh.1605395

electrical loads is because electrical loads are non-linear and may be influenced by numerous factors such as weather and economic factors. Because of this, the prediction accuracy of traditional nonlinear models is unable to meet the accuracy requirements of modern energy management systems.

The power industry has developed rapidly over the past few years due to the fast economic expansion. The electric power industry relies heavily on power load forecasting, which forms the foundation for various operations such as energy storage management, strategies for economic dispatching, facility upkeep plans, and future energy contracts [1].

Approaches to estimating electricity consumption have been categorized into two categories: conventional and artificial intelligence techniques. Conventional power load forecasting techniques such as regression analysis used to estimate and detect how the predicted and actual values relate to one another [2]. Moreover, conventional methods are easy to understand and apply, their limitations include low prediction accuracy, high sample data stationarity requirements, and limited ability to handle datasets with many features. Artificial intelligence techniques, on the other hand, is capable of handling several characteristics and many types of data structures. with great prediction accuracy and controllable generalization errors.

There has been a noticeable shift in the world's electrical supply networks within the past several years. These shifts included a trend towards smart grids, cloud computing for data management, and storing clean energy for resilience. Approaches to estimating electricity consumption that start with conventional methods and end with deep learning techniques play a crucial role in this change.

Deep learning technology is based on stacking several layers of neural networks with the use of enormous amounts of well-annotated datasets, such as inventory, financial, and climatic data. It is also applied in enormous fields, such as automated driving, natural language processing, game strategies, and image recognition, and thus, a wide variety of training algorithms of neural networks like backpropagation might be used to adjust the weights during the training models [3]. Activation function can also be used in hidden layers like ReLU [4] or GeLU [5]. Furthermore, by taking advantage of regularization techniques, the dropout technique prevents neural networks from overfitting.

A Recurrent Neural Network (RNN) is a type of very efficient deep learning variation network that has been extensively developed to perform temporal analysis and modeling and has already generated a great deal of interest due to its remarkable versatility in uncovering underlying sequential and non-linear correlations [6]. Because of their unique structural design, RNNs have received a lot of attention lately for their outstanding performance in many fields of prediction approaches.

The vanishing gradient issue that the RNN model ran into made it unable to produce good results and exposed a significant flaw. The reason is that using the similar weights to assess yield at every phase of the data preparation process may produce unreliable results. However, the gradient can decline over layers as a result of using the sigmoid and hyperbolic tangent functions as the activation function in these variations. Another solution has come for the vanishing gradient issue through proposed several RNN variations models with the ability to selectively forget or remember the relevant information such as the Long Short-Term Memory (LSTM) or the Gated Recurrent Unit (GRU) [7]. Bidirectional models are another sort of RNN that analyses sequential data in both forward and backward orientations, they combine the capabilities of a model with bidirectional processing, enabling the model to capture the input sequences' past and future context [8].

This study aims to address the problem of short-term electricity load forecasting by developing and evaluating four advanced deep learning models: LSTM, bidirectional LSTM, GRU, and bidirectional GRU. The main objective is to perform one-day-ahead power demand prediction for each day of the year 2023, using historical hourly consumption data spanning over nine years. Such a long training period allows the models to capture both short and long-term temporal dependencies, including daily, weekly, and seasonal cycles.

A major contribution of this work is the comparative performance analysis between unidirectional and bidirectional architectures. Bidirectional models are designed to process data in both forward and backward directions during training, potentially enhancing learning from complex sequences and improving forecast accuracy. By including these architectures, the study examines whether incorporating future context improves prediction quality for power demand data.

Another key aspect of this work is the application of a rolling a day-ahead forecasting approach across the entire year of 2023. This approach involves generating a prediction for each day based on all prior real data, allowing for a more realistic evaluation that mirrors operational conditions. The use of this method provides a comprehensive view of model performance across the year, enabling the identification of trends and shifts in prediction accuracy over time. It also helps in distinguishing specific periods, particularly those marked by abrupt changes in demand or external factors, where forecast reliability may decline. Furthermore, this study provides valuable insights for energy planners by presenting monthly evaluations and visual comparisons between actual and predicted loads. The results highlight improved accuracy during stable weather periods, underscoring the influence of environmental conditions on model performance. Thus, indicates the potential need to incorporate additional data, in addition to the power demand, for weather factors such as temperature or humidity into future models.

This section in this paper provides an overview of electricity power forecasting, and the sections that follow are organized as follows: Section 2 introduces the algorithms used and earlier researches. Section 3 describes the dataset, tuning the hyperparameters, and the experiments conducted to assess the effectiveness of the suggested approach. Section 4 illustrates the experimental results and compares the outcomes of the used models. Section 5 concludes the study.

2 Neural networks framework for load forecasting

A key tool used by power companies to estimate the power required to balance supply and load demand in the power system is load forecasting. Forecasting techniques such as artificial neural networks are predicated on elementary mathematical models of the human brain, they permit intricate nonlinear correlations between the predictors and the dependent variables. Over the last years, there has been a massive increase in research activity due to the growing interest in employing Artificial Neural Networks (ANNs) for predicting. Although ANNs have a lot of potential, there is also a lot of ambiguity around them. Researchers are still unsure about how important elements affect ANN predicting performance [9]. Although it might seem like a novel notion, the term Deep Learning (DL) dates back to the 1940s and has undergone around three waves of developments. DL was first popularized as cybernetics during the first wave, which lasted from 1940 to 1960. Later, in the 1980s and 1990s, it became known as connectionism [10]. The most recent revival of this concept was initiated by Goodfellow et al. [11]. Attempting to address the many limitations that ANNs have, such as handling large amounts of data that result in the vanishing gradient issue or handling complicated nonlinear data that causes the prediction to perform poorly. Consequently, its benefit over other RNNs, hidden Markov model, and other sequence learning techniques is its relative insensitivity to gap length.

There are now new models available that come to light to attempt to address these limitations of ANNs by offering features like a LSTM model -which is one of tens of other models- that may span thousands of timesteps [12]. These models will be highlighted in the following section.

2.1 Long short-term memory model

LSTM networks are renowned for their remarkable capacity to learn and understand the nuances of order dependency in sequence prediction tasks. Mozer [13] has initially proposed this concept in 1989, his primary area of study was backpropagation algorithms. At that moment, he focused on solving a mathematical problem related to the activation context unit, considering how the residual connection in the constant error loop should correct the actual value to one. Afterwards in 1997, Hochreiter and Schmidhuber [12] cited Mozer's work and presented an effective gradient-based technique called long short-term memory. They have worked on solving the problem of vanishing or exploding gradients that occur in long term dependencies that may cause the forget gate to reset the existing weight, by defining the Constant Error Carousel (CEC) which maintains the state to a fixed weight. The addition of this approach to the initial version of the LSTM, which consisted of only cells with input and output gates, was considered a great success in the RNN architecture and has been very popular in many subsequent applications. Kong et al. [14] assessed actual data from home smart meters and contrasted their findings with many standards, such as the most recent technology in load forecasting using LSTM, the model yields superior short-term load forecasts for individual residential families when compared to competing algorithms. Motepe et al. [15] conducted experiments and found that using both LSTM and neural fuzzy logic for a power distribution network produced better outcomes than alternative methods.

Traditional feedforward neural networks are fundamentally different from RNNs. Since no hidden unit in an RNN is autonomous, the temporal relationships between historical and present data are established through sequences. The fundamental structure of a single RNN unit is depicted in Figure 1. Module (A) of the neural network gets x_i as input, while h_i as the output. In the sequence, the neural network is replicated several times, with each neural network module transmitting data to the next in the queue. The information flows from one step to the next in this manner. Every stage of the RNN procedure was repeated starting from the stage next to the input. This type of methodology significantly reduces the number of parameters the networks must learn and shortens the training period by repeating the product of the training process, which assures precision.

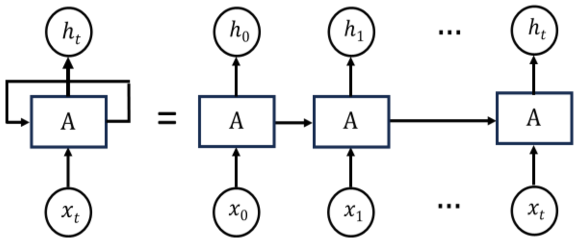

Figure 1. Schematic of the folded (left) and unfolded RNN cell (right)

Time series analysis prerequisites shows that an RNN's decision at time t_{-1} may be impacted by time t. Because of this particular property, RNNs are ideally suited to handle load prediction problems that arise in the setting of individual short-term residential load forecasting [14]. The problem of vanishing gradients, where the long-term components' gradient norm experiences an exponential decline, has been observed to make it difficult to acquire long-term dependencies using RNNs. The solution to this problem was to engineer an LSTM network with an added forget gate.

Structure of the input, output, and forget gate make up an LSTM unit in general. The three gates are linked in such a way that the cell state is linked to them, which is responsible for storing information over time, while the hidden state leads to the transfer of information to the next step. Figure 2 shows the three control gates of a LSTM cell: input, output, and forget gates. The network's output will be impacted by this input continuously and continuously starting at the beginning of the input processing phase. As an example, let us use the normal input sequence for an LSTM is $x_t = \{x_1, x_2, x_3, ..., x_T\}$, where x_t represents each real vector value. The LSTM cell maintains an internal memory state throughout its operation to capture temporal dependencies. Using the current input x_t, the previous hidden state h_{t-1}, and the previous internal cell state c_t, the network determines which parts of the memory should be updated, retained, or erased.

This process allows the LSTM to effectively incorporate both current and past information when managing its internal state.

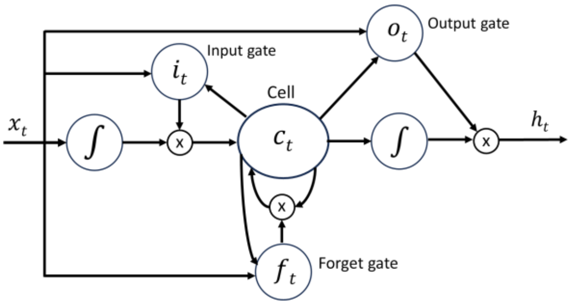

Figure 2. LSTM cell

The LSTM structure consists of the internal state vector as well as the input gate i_t, the output h_t, forget get gate f_t, and output gate o_t. Starting with the forget gate, the LSTM is updating its state with each step according Equation 1:

$$f_t = \sigma\left(W_{fx}\, x_t + U_{fh}\, h_{t-1} + b_f\right) \tag{1}$$

Here, σ denotes the activation function sigmoid, W_{fx} denotes the input weight matrix of the forget gate, x_t denotes the current input vector, U_{fh} denotes the recurrent weight matrix of the forget gate, h_{t-1} denotes the previous output, and b_f denotes the bias of the forget gate. The input gate itself updated by Equation 2:

$$i_t = \sigma\left(W_{ix}\, x_t + U_{ih}\, h_{t-1} + b_i\right) \tag{2}$$

In this context, W_{ix} refers to the input weight matrix of the input gate, U_{ih} refers to the recurrent weight matrix of the input gate, and b_i refers to bias of the forget gate. The output gate o_t is just similar to the forget gate and input gate i_t, all of them are using the sigmoid σ as an activation function. The output gate o_t can be expressed by Equation 3:

$$o_t = \sigma\left(W_{ox} x_t + U_{oh} h_{t-1} + b_o\right) \tag{3}$$

Where W_{ox} denotes the input weight matrix of the output gate, U_{oh} denotes the recurrent weight matrix of the output gate, and b_o denotes the bias of the output gate. The internal state is updating by its self-loop by Equation 4:

$$c_t = f_t\, c_{t-1} + i_t\, \sigma\left(W_{cx} x_t + U_{ch} h_{t-1} + b_c\right) \tag{4}$$

Where W_{cx} denotes the input weight matrix of the internal state, U_{ch} denotes the recurrent weight matrix of internal state, and b_c is the bias of the internal state. The output h_t is regulated by the output gate o_t and the internal state c_t, enabling the LSTM cell to control the flow of information to the next time step. This relationship is mathematically defined in Equation 5:

$$h_t = tanh\,(c_t)\, o_t \tag{5}$$

Here, $tanh$ represents the hyperbolic tangent activation function. The sigmoid activation function σ, is given by Equation 6:

$$\sigma_{(x)} = \frac{1}{1 + e^{-x}} \tag{6}$$

Equation 7 can be used to express the hyperbolic tangent function, or *tanh*:

$$tanh_{\ (x)} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{7}$$

The output ht at the current time t can be calculated by Equation 5, while, h_{t-1} in Equations 1-4 refers to the output at prior time *t-1*.

Referring to Figure 2, each × inside the tiny circles represents the Hadamard product (also known as the entrywise product) carried out between each of its inputs separately [16]. Large circular forms with an S-shaped curve within represent applying a differentiable function to a weighted sum, such as the sigmoid function. In addition, every LSTM layer has a dropout layer added to it to prevent overfitting in neural network modeling, this will be covered in more detail in hyperparameters tuning and optimizers section.

2.2 *Gated recurrent unit model*

In RNNs, the GRUs are a gating mechanism that first introduced by Cho et al. [17]. GRU network has fewer parameters than an LSTM network because it lacks an output gate, and it merges the input and forget gate into a single update gate, additionally, it has a reset gate. However, both models share a gating mechanism for inputting or forgetting particular features. It has been demonstrated that smaller and less frequent datasets get superior results from GRUs [18]. Figure 3 graphically and structurally illustrates the fully gated version of GRU. Reset, update, and a temporary output gate makes up a GRU in general [19].

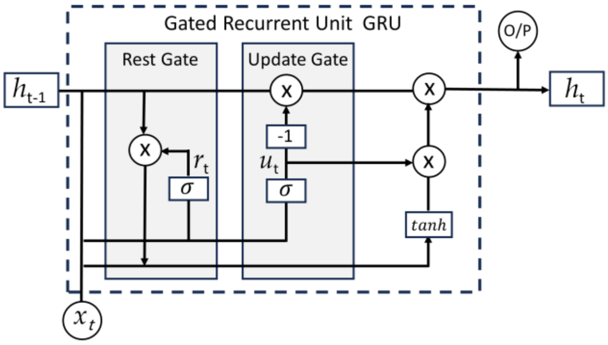

Figure 3. Basic structure of GRU

As an example, let us use the normal input sequence for a GRU model as x_t = {x_1, x_2, x_3, ……., x_T}, where x_t represents each real vector value. The full gated unit of the GRU cell comes with many variations, where gating can be done in different ways utilizing the previous hidden and the biased state. The GRU structure consists of input vector x_t, the output h_t, update gate u_t, and reset gate vector r_t. The GRU mechanism utilizes the previous hidden state h_{t-1} and a bias term to regulate the flow of information. Given input sequence will result the output h_t is defined by Equation 8 [11]:

$$h_t = u_{t-1}\, h_{t-1} + \sigma\,(1 - u_{t-1})\,(W_{hx}\, x_{t-1} + U_{hh}\, h_{t-1}\, r_{t-1} + b_h) \tag{8}$$

In this context, W_{hx} refers to the input weight matrix of the output, U_{hh} refers to the recurrent weight matrix of the output, and b_h is the bias of the output. The update gate u_t controls how much of the past information is retained or updated. Acting like a dimension-wise filter, it allows the state vector to either preserve previous values or incorporate new ones, depending on the input. This controlled integration enables the model to adaptively adjust its internal state. The update gate u_t is defined in Equation 9:

$$u_t = \sigma\,(W_{ux}\, x_t + U_{uh}\, h_t + b_u) \tag{9}$$

Here, W_{ux} denotes the input weight matrix of the update gate, U_{uh} denotes the recurrent weight matrix of the update gate, and b_u denotes the bias of the update gate. Similar to the update gate, the reset gate selectively filters components of the previous state when computing the next state. This adds nonlinearity, enabling the model to emphasize or suppress specific information, allowing for more flexible and context-aware transitions. The reset gate r_t is defined in Equation 10:

$$r_t = \sigma\,(W_{rx}\, x_t + U_{rh}\, h_t + b_r) \tag{10}$$

Where W_{rx} denotes the input weight matrix of the reset gate, U_{rh} denotes the recurrent weight matrix of the reset gate, and b_r denotes the bias of the reset gate.

The sigmoid function is denoted by σ, which is given by Equation 6. Hyperbolic tangent function is denoted by *tanh* which is given by Equation 7. When r_t approaches zero, the reset gate allows the unit to forget the previously computed state by essentially making it behave like it is reading the first symbol of the input sequence.

2.3 Bidirectional recurrent neural networks

Bidirectional Recurrent Neural Networks (BRNNs) are neural network architectures designed to process sequential input. Both forward and backward processing of the input sequences is done by BRNNs so that the network may employ both past and future contexts in its predictions. This is the primary difference that BRNNs and traditional RNNs differ from one another. The input sequence is processed forward by one of a BRNN's two distinguishing recurrent hidden layers, and backward by the other. The outcomes of these hidden layers are then gathered and sent into the last layer, which makes predictions. The BRNN can predict each individual output element by using the data from the complete input sequence because of its bidirectional construction [20]. Every time step, BRNNs update the hidden state based on the input that is now being received and the prior hidden state, operating in a manner akin to traditional recurrent neural networks in the forward direction. In contrast, the backward hidden layer analyses the input sequence oppositely, updating the hidden state according to the current input as well as the hidden state of the subsequent time step. Due to BRNNs ability to handle data in both ways and account for both past and future states, the BRNN has better accuracy when compared to traditional unidirectional RNNs, employing two different hidden layers also provides a kind of model regularization, since the two can enhance each other and provide the final prediction layer with more information.

The gradients are calculated for both the forward and backward passes of the backpropagation using the time approach, which is commonly used to train BRNNs, in order to update the model parameters. During inference, the BRNN processes the input sequence in a single forward pass, and predictions are based on the sum of the outputs from the two hidden layers. The fact that the element of traditional RNNs in each gate vector can only receive input from the element of the cell vector so it considered as one of their limitations. However, BRNNs analyze the data in both ways using two distinct hidden layers fed into same output layer.

It is possible to construct the recurring hidden layers utilizing various kinds of recurrent neural network cells, notably including the LSTM and GRU, providing a myriad of choices for customization in the architectural planning.

2.3.1 Bidirectional LSTM neural network

The bidirectional LSTM neural network consists of dual LSTM layers, each dedicated to calculating the hidden vector in a different direction, with one layer calculating the hidden vector in a forward direction and the other in a backward direction. These two layers regulate the bidirectional LSTM neural network's output [21]. The mechanism of the bidirectional LSTM neural network and the conventional feedforward neural network are not the same. Inside each layer of a bidirectional LSTM, there is no connection between the internal nodes. Moreover, results are committed to memory and kept in the memory unit, which can enhance the correlation between individual data points in various time series, thus, a directed loop is incorporated in the linkage of hidden layers, and preceding data. Combining earlier output and the current input yields a new output in neural network. Nevertheless, when the amount of input data in the time series rises, problems with gradient expansion and disappearance will arise. In contrast, some structures only provide a portion of the input data because of the limitation of the small input windows, the bidirectional structure may adjust during training to make the greatest use of the input information [22].

As seen in Figure 4, the forward layer stores the output of the forward hidden layer at each instant and computes the forward direction from one to *t*, whilst the backward layer stores the output of the backward hidden layer and reversely computes the forward direction. Ultimately, the bidirectional long short-term memory neural network's output is computed by aggregating the corresponding forward and backward layer output values at each time period.

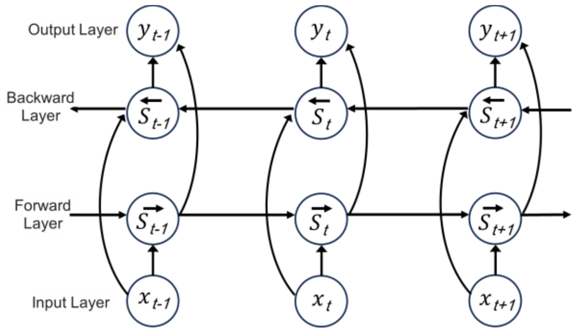

Figure 4. Basic structure of bidirectional LSTM

For a normal input sequence of a bidirectional LSTM neural network model, x_t represents each real vector value. The bidirectional LSTM cell acts like an LSTM cell by generating and maintaining the internal memory cell state across its lifespan in order to make temporal connections, together with the memory cell state S_{t-1}, the subsequent input x_t and intermediate output h_{t-1} discover which internal state vector components are required to update, erase, or maintaining. However, in addition to the standard forward output in traditional LSTM networks, the bidirectional LSTM also incorporates a backward output is considered as $\bar{\bar{h}}_t$. These two outputs capture information from both past and future contexts. The forward output of bidirectional LSTM which considered as $\bar{h}_t$ can be expressed by Equation 11 [23]:

$$\bar{h}_t = H\left(W_{x\bar{h}}\, x_t + W_{\bar{h}\bar{h}}\, \bar{h}_{t-1} + b_{\bar{h}}\right) \tag{11}$$

Where H denotes the hidden layer function, $W_{x\bar{h}}$ denotes the input-forward output weight matrix, $W_{\bar{h}\bar{h}}$ denotes the forward output weight matrix, and $b_{\bar{h}}$ is the bias vector of the forward output. While the backward output of the bidirectional LSTM $\bar{\bar{h}}_t$, can be expressed by Equation 12:

$$\bar{\bar{h}}_t = H\left(W_{x\bar{\bar{h}}}\, x_t + W_{\bar{\bar{h}}\bar{\bar{h}}}\, \bar{\bar{h}}_{t-1} + b_{\bar{\bar{h}}}\right) \tag{12}$$

Where $W_{x\bar{\bar{h}}}$ denotes the input-backward output weight matrix, $W_{\bar{\bar{h}}\bar{\bar{h}}}$ denotes the backward output weight matrix, and $b_{\bar{\bar{h}}}$ is the bias vector of the backward output. With the iterating the forward layer from t=1 to T, and the backward layer from t=T to 1, the final output y_t of the bidirectional LSTM can be expressed in Equation 13:

$$y_t = W_{\bar{h}y}\, \bar{h}_t + W_{\bar{\bar{h}}y}\, \bar{\bar{h}}_t + b_y \tag{13}$$

Here, $W_{\bar{h}y}$ denotes the forward of the final output weight matric, $W_{\bar{\bar{h}}y}$ denotes the backward of the final output weight matric, and b_y denotes the bias of the final output. The forward and backward layers in a bidirectional LSTM operate independently and do not share the state weight matrices. Each layer processes the input sequence separately, and their outputs are computed sequentially [24].

2.3.2 *Bidirectional GRU neural network*

A sequence processing model with two GRUs is called a bidirectional GRU. One is processing the input forward, while the other is processing it backward. This neural network is bidirectional and recurrent, simply utilizing input and forget gates. Bidirectional GRUs are just the result of combining two separate GRUs. For one GRU, the input sequence is fed in forward order for the other, it is fed in reverse order. Every time step, the outputs from the two networks are typically concatenated. Context is better understood when information from the past and future is preserved. A bidirectional GRU and a GRU are different in that a bidirectional GRU contains two distinct hidden states, one for each direction, and before making its final prediction, it concatenates the hidden states from both directions. Because of this, the bidirectional GRU is able to obtain information from the input sequence's past as well as its future, while a conventional GRU can only access data from the past [25-26]. One-way state transfer occurs from front to rear in a conventional recurrent neural network. That being said, there are some issues where the present output is connected to both the prior and subsequent states. The development of the BRNN, for instance, addresses the issue of anticipating the missing words in a phrase, which necessitates the prior judgment for the next state [27]. The bidirectional GRU is a variation of the unidirectional GRU that resolves the unidirectional GRU's issue and produces a more accurate result by depending on the dual impacts of the forward and backward states. Figure 5 depicts the basic structure of bidirectional GRU model [28].

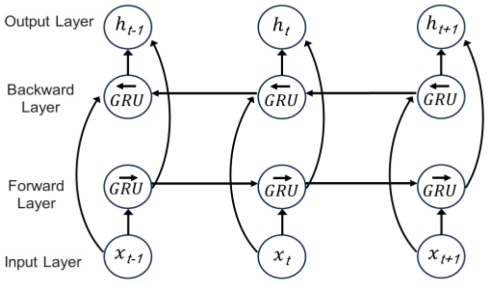

Figure 5. Basic structure of bidirectional GRU

The state information of the bidirectional GRU h_t can be calculated by Equation 14 [29]:

$$h_t = \bar{h}_t \oplus \bar{\bar{h}}_t \tag{14}$$

Where $\bar{h}_t$ is the state information of the forward GRU and it can be defined as a function of the conventional GRU of the previous state, as shown in Equation 15:

$$\bar{h}_t = \overline{GRU}\left(x_t, \bar{h}_{t-1}\right) \tag{15}$$

While $\bar{\bar{h}}_t$ is the state information of the backward GRU and it can be defined as a function of the conventional GRU of the next state, as Equation 16:

$$\bar{\bar{h}}_t = \overline{\overline{GRU}}\left(x_t, \bar{\bar{h}}_{t-1}\right) \tag{16}$$

It is important to highlight that the traditional functionality of the GRU components is fundamentally based on the mathematical formulations previously presented in Equations (8), (9), and (10).

3 Experimental setup and evaluation

The approach suggested in this paper uses four enhanced deep learning models to forecast future load in accordance with an index of time series for individual months of the year on a daily basis for an entire year. These models include LSTM, bidirectional LSTM, GRU, and bidirectional GRU. The training and testing set of the dataset are divided into periods of nine years and one year, accordingly. Certain parameters, such as the number of layers or type of the optimizer, need to be selected. In general, they could be acceptable depending on the dataset and the model that the algorithm is meant to be created in. The precision of the predictions has been evaluated by assessing the errors through the use of several metrics once the regression models have been constructed.

3.1 Data collection and description

One of the greatest challenges for researchers developing forecasting models is obtaining a high-quality dataset—free from typos, missing values, and inconsistencies. Therefore, a preliminary statistical analysis is essential to determine whether the dataset is suitable for accurate forecasting. Various statistical tests may be applied, such as assessing the p-value (probability value), which indicates the likelihood that the observed data could have occurred under the null hypothesis [30]. A reliable dataset is the foundation of any successful prediction process. For example, without a temporal component, time series forecasting would not be possible.

The hourly power demand data, measured in megawatts (MW), utilized in this study was obtained directly from the official website of the PJM Interconnection. PJM is a regional transmission organization (RTO) responsible for coordinating the movement of wholesale electricity across several states in the United States. It encompasses numerous electric utilities involved in electricity generation, transmission, and distribution. Due to historical and administrative changes in regional boundaries, data availability may vary across areas covered by sub-utilities. Accordingly, the dataset used in this work specifically pertains to the region served by Commonwealth Edison (ComEd), the largest electric utility in the state of Illinois. ComEd supplies electricity to the city of Chicago and most of Northern Illinois. The dataset is publicly available online [31].

The dataset, consisting of 87,648 rows, spans the time period from January 1, 2014, at 00:00 to December 31, 2023, at 23:00. The average power demand over this period is 11,021.65 MW. The lowest recorded demand was 6,775.822 MW on March 5, 2020, at 07:00, while the highest recorded demand was 22,467.01 MW on August 24, 2023, at 17:00

Figure 6 presents a graphical representation of the entire dataset, where a slight upward trend can be observed. However, such graphs can sometimes give a misleading impression or fail to convey the complete picture.

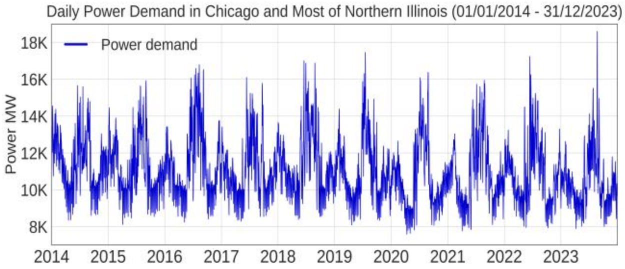

Figure 6. Graphical representation of the dataset [31]

Figure 7 shows the histogram of the power data distribution over time. At first look, it is obvious that the data are not normally distributed; rather, they appear to be skewed (not symmetric). The histogram is said to be balanced when the mean and median are roughly equal in both the left and right tails, thus making a better data distribution. Furthermore, it should be noted that the histogram serves as a visual aid to demonstrate how the data points are spread out in correlation to the vertical red line, serving as a visual marker for the central tendency of the data, which symbolizes the statistical average of the dataset.

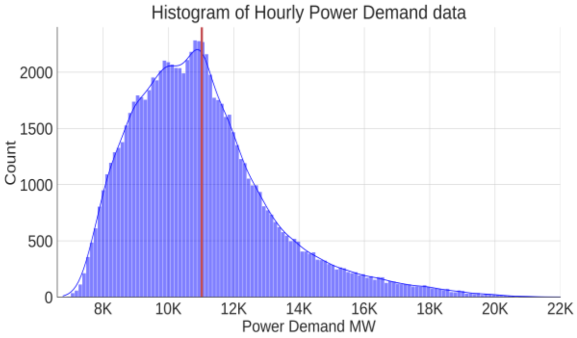

Figure 7. Data distribution

The probability value, or p-value that the observed difference would occur by chance, has been used to assess the significance of the null hypothesis (H_0). A statistical hypothesis known as the null hypothesis (H_0) states that a group of particular observations has no statistical significance and can be accepted or rejected based on the p-value. An optimal dataset with a p-value of less than 0.05 indicates that the data is stationary and does not have a unit root [32]. Within the dataset that was scrutinized in this study, the p-value was computed to be 7.46×10^{-10}.

3.2 Hyperparameters tuning and optimizers

For a deep learning model to be successful and reliable, it must satisfy all necessary conditions that enable the system to learn complex, non-obvious patterns and produce optimal outcomes with minimal error. The dataset used in this study was originally recorded on an hourly basis, as previously mentioned. To reduce the frequency of observations, the data was resampled to a daily resolution using the mean of each day's hourly values. This transformation resulted in a dataset representing the average daily power demand, with a total of 3,287 rows. The training set comprises data from 2014 to

2022, while the test set includes data from the year 2023, consisting of 365 rows (2023 is a common year).

For complicated tasks where machines must interpret unstructured data, deep learning (which is a subset of machine learning) that is based on artificial neural networks has been projected to be the most effective. The word "deep" indicates the network's utilization of many levels. Through deep learning, computational models with several processing layers may learn representations of data with various degrees of abstraction [33]. The basic concept of deep learning is to teach computers to process data in a way inspired by the human brain, where humans do not create feature layers, but rather the data is taught through a general learning process. Several layers have been used in this study to verify the forecasting procedure in order to get the best possible forecasts. The first step in building a deep learning algorithm is to stack multiple identical or different layers so that the output of one layer feeds into the input of another layer. A sequential approach is used to achieve this, which, by iteratively traversing through several neural layers, allows the exact and sequential building of a neural network from input layer to output one.

Each model used in this work has four highest-level building block layers of the same particular structure architecture model. For each layer 100 memory units are used, this is done to ensure that sequences are passed to the next model layer instead of just randomly dispersed input layers. Neural network topologies include a layer known as the dropout layer to prevent overfitting. This process involves eliminating individual nodes using a probability over several training rounds, treating them as though they were not even a part of the network structure, every layer has the dropout layer included in it.

Neural network topologies also include a layer of neurons called the Dense Layer. This layer is fundamental and fully interconnected, each neuron receives input from every other neuron in the previous layer, thus, each dense layer neuron's output is calculated as weighted sum of its inputs from every neuron in the layer before it, dense layer comes to be the neural network's last step.

The proposed models employ a computationally efficient stochastic optimization algorithm called Adaptive Moment Estimation (Adam), which is used for gradient-based optimization of objective functions. Adam combined the advantages of two popular optimization methods: momentum and RMSprop, and added some additional improvements. Every iteration of the models' training process uses the Mean Absolute Error (MSE) as the loss function. Equation 17 is used to compute this metric:

$$\mathrm{MSE} = \frac{\sum_{i=1}^{n}(\hat{yi} - yi)^2}{n} \tag{17}$$

Where $\hat{yi}$ is the expected amount of the electricity needed derived from the actual data, y_i is the real value of the power demand taken from recorded data, and n is the number of samples to fit the model. Efficient training and tuning of the models occurred over 88 epochs, with a batch size of 32 utilized consistently. Eventually, after this many trainings, with every epoch, the model will yield findings that demonstrate a growing level of improvement.

The optimal parameters used for the models are presented in Table 1. The same values were applied across all deep learning algorithms in this study to ensure consistency in comparison.

Table 1. Optimal parameter selection

Parameter	Value	Parameter	Value
No. of layers	4	No. of neurons	100 units each
Optimizer	Adam	Loss function	MSE
Output size	One dense layer	Epochs	88
Batch size	32	Dropout	0.25

3.3 *Performance evaluation metrics*

Given the actual data, the results obtained for each prediction model can be evaluated to determine how well it performs. R-squared and MAPE were used in paper as evaluation metrics. Equation 18 can be used to calculate R-squared value.

$$\mathrm{R-squared} = 1 - \frac{SS_{\mathrm{res}}}{SS_{total}} \tag{18}$$

Here, SS_{res} denotes the residuals summation of squares, and it can be calculated via Equation 19:

$$SS_{\mathrm{res}} = \sum_{i} (yi - \hat{yi})^2 \tag{19}$$

While SS_{total} denotes the total summation of squares, it can be calculated via Equation 20:

$$SS_{\mathrm{total}} = \sum_{i} (yi - \hat{y})^2 \tag{20}$$

Furthermore, y_i is the real value of the power demand taken from recorded dataset, $\hat{yi}$ is the predicted values of the electricity needed derived from the actual data, $\hat{y}$ is the average of the observed data, and it can be calculated by Equation 21:

$$\hat{y} = \frac{1}{n}\sum_{i=1}^{n} yi \tag{21}$$

Where n is the number of predicted samples. For second metric used MAPE, it can be calculated by Equation 22:

$$\mathrm{MAPE} = (\frac{1}{n}\sum_{i=1}^{n} \frac{|yi - \hat{yi}|}{yi}) * 100 \tag{22}$$

The coefficient of determination, or (R-squared) in statistical terms is the percentage of dependent variable variations that can be expected from the independent variable(s) that is accounted for by a linear or nonlinear model. The coefficient of determination normally ranges from zero to one. A value of zero signifies a model that fails to capture any of the variance in the response variable relative to its average. Likewise, a value of one signifies a model that captures all of the variance in the response variable relative to its average. In some cases, this coefficient may provide a negative value, this may occur if the model is predicting worse than using the average of the observed data ($\hat{y}$) as the average of the prediction [34].

MAPE is frequently employed as a loss function in regression tasks and model assessment, owing to its easily understandable representation of relative error. It is also a scale-independent and readily applicable to both high and low values. However, differential frequently results biased forecasting. For traditional load forecasting problems, the drawbacks of MAPE, such as its inability to handle tiny and zero denominators, are not especially significant because the aggregated load is rarely zero or extremely close to a minimal value [35].

4 Results and discussions

In this paper, the power needs for every month of the year 2023 were thoughtfully predicted by applying the various models that have been presented in detail. When aiming to convert time series data from an hourly basis to a daily framework, a particularly useful technique for frequency adjustment and resampling requires the thorough calculation of the mean function value, which is based on 24 separate observations that take place over the duration of one day. The process of assessing the performance of the used deep learning algorithms is carried out in terms of coverage and prediction accuracy after the optimum operating parameters have been set.

Time series models represent a highly effective and powerful methodology for predicting future values based on the analysis of historical data trends. It is of utmost importance to thoroughly evaluate the effectiveness and accuracy of these forecasting models, whether one is employing advanced deep learning techniques or opting for simpler statistical approaches. It may evaluate the precision and dependability of the projections using a variety of assessment indicators that are mentioned in the last section. This section examines and discusses the advantages and disadvantages of several of the most often used metrics for load forecasting. The findings obtained are examined for each model and each month in 2023, separately in order to acquire insight into the influence of the choice of algorithm primarily on the prediction accuracy and insight for at which month the has the highest prediction. To carry out these tests, we employed the metrics that are detailed in the following.

4.1 Experiments using the coefficient of determination

R-squared is a statistical metric that represents the percentage of dependent variable variations that can be expected from the independent variable(s) of the model. Regression analysis frequently uses the R-squared metric to evaluate a model's quality of fitting. However, because it ignores the temporal correlation between the observations, it is limited in its use of time series forecasting. Therefore, in forecasting processes, more than one metric is used to evaluate the forecast and compare the results of other forecasts.

Table 2 shows the evaluation results as a forecast comparison after applying each model over an individual month of 2023 by using R-squared as a metric. The GRU model performs better than the other models, in particular, it excels in October when it achieved 0.8526, thus the percentage value of the coefficient of determination is 0.8526 × 100% = 85.26%, and so on with other results. This value is a percent of variance explained by such a model, moreover, it considers a proportion by which the errors' variance is lower than the dependent variable's variance.

Table 2. Monthly evaluation results of different models using R-squared (2023)

Months of 2023	**Coefficient of determination (R-squared)**			
	LSTM	**Bi-LSTM**	**GRU**	**Bi-GRU**
Jan	0.4221	0.4059	0.6863	-0.0462
Feb	0.7296	0.6968	0.7955	-0.0341
Mar	0.4617	0.3740	0.5577	-0.8237
Apr	0.6483	0.7188	0.6664	-0.6193
May	0.5488	0.5298	0.5541	0.3368
Jun	0.7557	0.7307	0.7525	0.6357
Jul	0.5519	0.4850	0.5200	0.2581
Aug	0.6052	0.6201	0.5809	0.5646
Sep	0.7440	0.8037	0.8202	**0.6999**
Oct	**0.8420**	**0.8324**	**0.8526**	0.4238
Nov	0.7680	0.7019	0.7950	0.2508
Dec	0.7028	0.6776	0.7132	0.5111

The best result of R-squared with using the LSTM model was 0.842 in October. The bidirectional LSTM model obtains best R-squared at 0.8324 in October as well. While it is clear that the bidirectional GRU model has the worst results even it has negative values which explains the model fits the data less well than a horizontal line since it does not follow the data's trend. In other words, when the coefficient of determination is less than zero, there is a negative (inverse) correlation. This suggests that the two variables (the actual and predicted values) are moving against one another.

Figure 8 shows the actual and forecasted demand for the power in MW in October 2023 using the LSTM, bidirectional LSTM, GRU, and bidirectional GRU models. October was chosen because the best R-squared results appeared in this month. The difference between the real and the predicted values obtained from applying the different models can be seen in this figure. LSTM, bidirectional LSTM, and GRU models are making a good prediction as it is clear. When applying the bidirectional GRU model, many extreme values led to a decrease in the R-squared value when it achieved 0.6999 as best result in September.

The evaluation results are presented in Figure 9, with particular emphasis on the R-squared values as a basis for comparing the forecasting performance of each model for every month of the year 2023. To ensure clarity in the visual representation, negative R-squared values were deliberately excluded from the analysis.

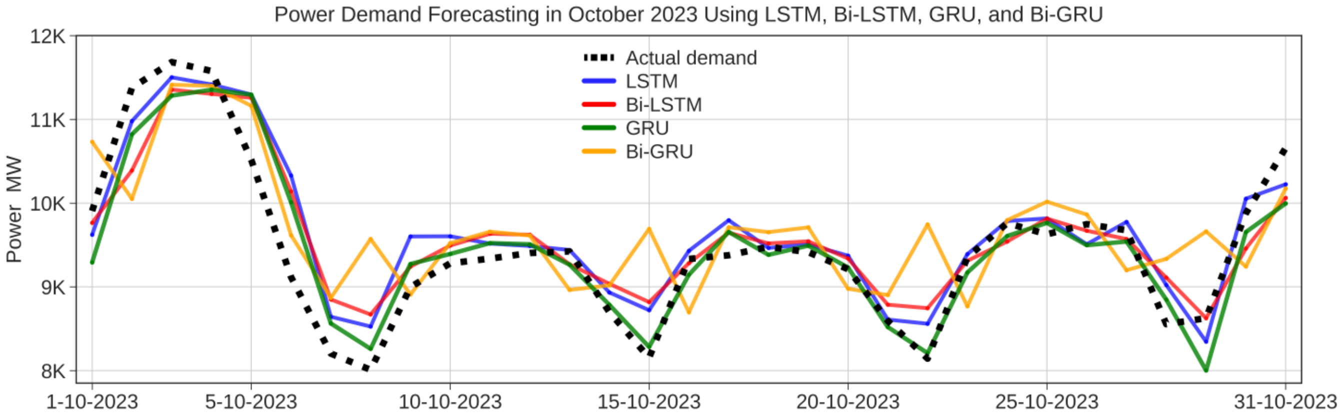

Figure 8. Actual vs. forecasted electricity demand by the models in October 2023

4.2 *Experiments using MAPE metric*

Expression of the accuracy of the regression model as a ratio of the loss function can be done using MAPE. It is commonly used due to its very intuitive interpretation in the context of relative error. In practical settings, the MAPE is commonly used for predicting values that are consistently above zero. For instance, MAPE was used as a performance benchmark in an electricity consumption forecasting competition organized by GDF Ecometering on DataScience.net [36]. In a larger context, it has been contended that the MAPE is very suitable for predicting purposes, particularly when sufficient data are accessible. Table 3 presents the evaluation results for each model's monthly forecasting performance in 2023 using MAPE as the evaluation metric. Once again, the GRU model outperforms the others, particularly in March, where it achieved the lowest MAPE of 2.34%.

A MAPE of 2.34% indicates that, on average, the model's predictions deviate from the actual values by 2.34%. In other words, the predicted values are, on average, 2.34% different from the observed data. The same interpretation applies to the other MAPE results.

The LSTM model achieved its best MAPE result of 2.7887% in April. Similarly, the Bidirectional LSTM model also recorded its lowest MAPE in April. While the Bidirectional GRU model performed worse than the other models overall, its best MAPE result was observed in December.

Table 3. Monthly evaluation results of different models using MAPE (2023)

Months of 2023	**MAPE (%)**			
	LSTM	**Bi-LSTM**	**GRU**	**Bi-GRU**
Jan	4.2769	4.6480	4.0577	6.0642
Feb	2.9722	4.6623	2.6223	6.8404
Mar	2.8239	4.1553	**2.3400**	5.3256
Apr	**2.7887**	**3.8033**	2.6669	5.4996
May	4.6031	6.0017	4.4882	6.0366
Jun	5.5684	4.9782	5.3433	6.3848
Jul	6.2598	7.0162	5.8463	9.3820
Aug	6.8618	6.9236	6.5636	6.9269
Sep	4.7947	5.5765	5.1261	6.0614
Oct	3.2237	4.4820	3.2335	5.9923
Nov	2.9189	4.2818	2.7561	5.4809
Dec	2.8418	3.9583	2.8427	**4.1882**

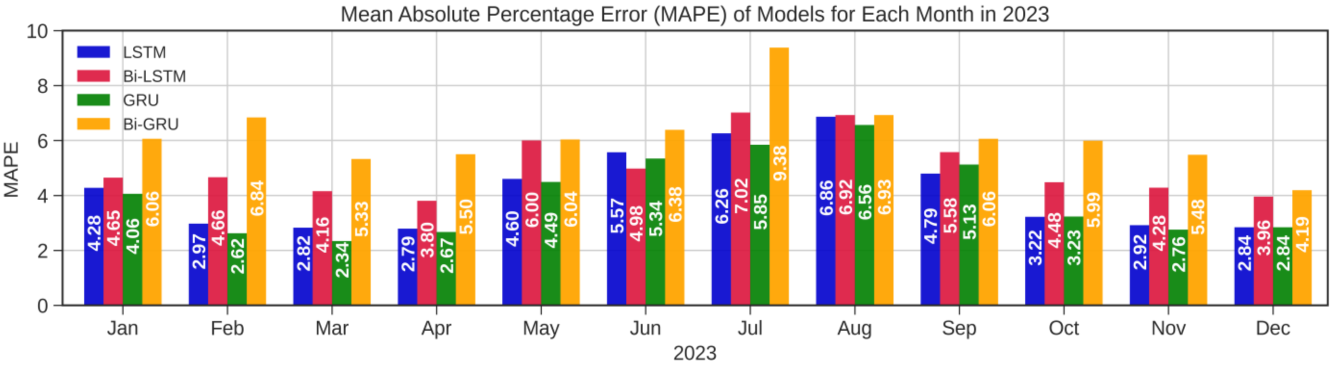

Figure 9. Monthly MAPE scores achieved by the forecasting models in 2023

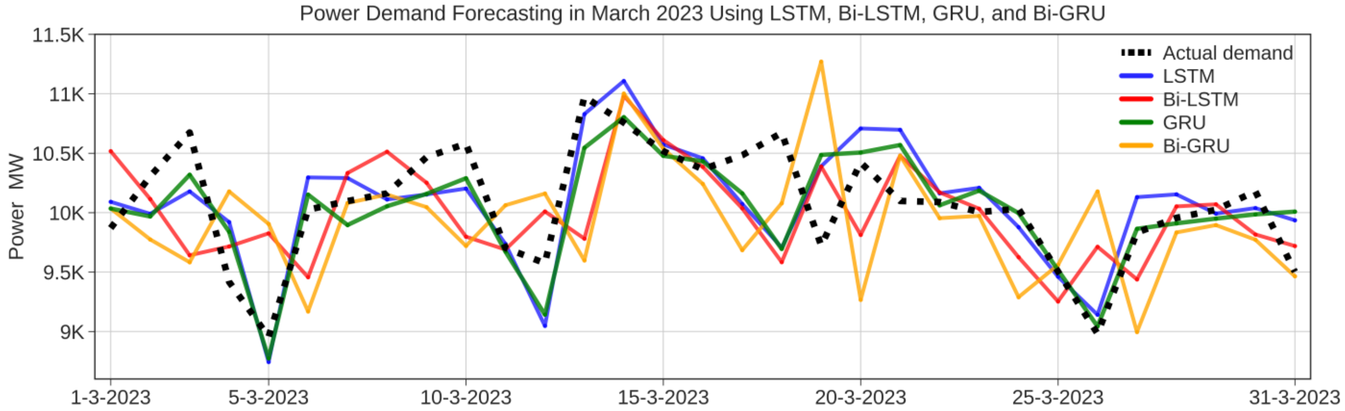

Figure 10. Actual vs. forecasted electricity demand by the models in March 2023

Figure 10 illustrates the discrepancy between the actual and forecasted demand for the power in MW in March 2023 which was achieved by using various models. March was chosen because the best MAPE results appeared in this month. LSTM and GRU models are making a good prediction. When applying both the bidirectional LSTM and bidirectional GRU model, many extreme values increased the MAPE value when their best values achieved 3.8033 in April and 4.1882 in December for bidirectional LSTM and bidirectional GRU, respectively.

Figure 11 shows the evaluation results of MAPE as a forecast comparison after applying each model for 2023.

4.3 Forecasting performance analysis

Based on the results presented in Table 2 and Table 3, the GRU model consistently outperformed the other models across most months. It achieved the highest R-squared value of 0.8526 in October, reflecting a strong fit between the predicted and actual values. Moreover, GRU obtained the lowest MAPE value of 2.34% in March, indicating a high level of prediction accuracy. The LSTM model also demonstrated reliable performance, showing competitive results in several months, although it was generally outperformed by GRU. The bidirectional LSTM model performed reasonably well, particularly in months such as April and September, but lacked the consistency exhibited by GRU. In contrast, the bidirectional GRU model exhibited the weakest performance among the four. It recorded negative R-squared values in multiple months (e.g., March and April), which suggests that its predictions were less accurate than simply using the mean of the observed data. Additionally, bidirectional GRU produced the highest MAPE values, reaching 9.38% in July, further confirming its limited predictive capability. These findings underscore the robustness and effectiveness of the GRU model for short-term electricity demand forecasting, particularly when evaluated on a monthly scale. The results also highlight the potential limitations of bidirectional architectures, especially in the context of highly variable or seasonal time series data

A detailed examination of the analyzed data reveals subtle yet consistent discrepancies between the actual power load values and the forecasts, particularly in the GRU and LSTM models. These two models demonstrate not only high reliability but also notable accuracy in their predictions. In contrast, the Bidirectional LSTM and Bidirectional GRU models exhibit significant outliers, with several extreme deviations between predicted and actual values. Such variability poses a potential risk to grid stability and operational reliability. Notably, the R-squared values for these bidirectional models are negative in several months, indicating that the residual sum of squares (SS_{res}) exceeds the total sum of squares (SS_{total}). Based on the R-squared definition, this outcome implies that the models fail to explain the variability in the data, and that the mean of their predictions performs worse than simply using the mean of the observed values. This highlights a substantial limitation in the effectiveness of bidirectional recurrent neural networks for this forecasting task.

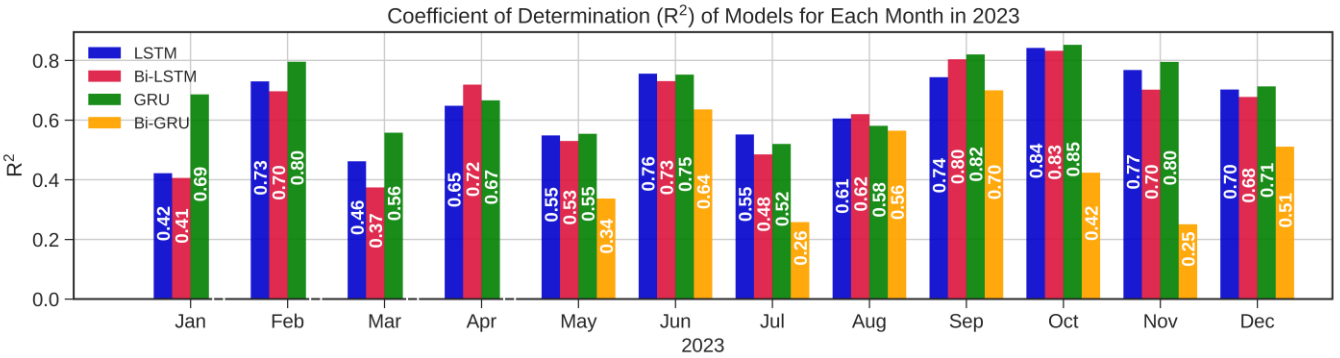

Figure 11. Monthly R-squared scores achieved by the forecasting models in 2023

5 Conclusion

Accurate electricity power demand forecasting is advantageous for calculating electricity generation costs and utility company profits as well. It has also a positive impact on the reliability and performance of power plants. In this paper, four deep learning algorithms (LSTM, bidirectional LSTM, GRU, bidirectional GRU) were utilized to predict the electricity power demand based on the data collected that were taken from PJM Interconnection LLC organization. The dataset used in this study comes from ComEd company which is the largest electric utility in Illinois, United States [31].

A lot of advancements were made when creating the models by tuning the right hyperparameter that allows tweaking model performance for optimal results. Forecasting performances of the used models are compared in terms of R-squared and MAPE metrics. The forecast was conducted on a daily basis for each separate month of the year 2003. To study and compare the best results that will appear based on the data for the particular month.

The GRU model demonstrated a deeper comprehension of the context by employing an update and reset gate mechanism to learn future time steps. Additionally, this model shows that gating is generally beneficial and more efficient than the other models when turned out it gave the most effective results when the R-squared was 0.8526 in October, and the MAPE was 2.34% in March.

According to the experimental findings, the model yielded high regression accuracy by deploying the gating mechanism to input and forget certain features which enabled the best and easier to run and train. It appears from the results of the separate months that March and October are the best months for forecasting. This is due to the mild weather and the lack of electrical energy consumption being affected by weather fluctuations. Here, it is inferred that it is necessary to add a more feature to the dataset that may contain additional data, such as the weather. In contrast, a one-day forecast generally provides only an estimate of the maximum power demand expected over the course of a day, which does not reflect the instantaneous fluctuations in demand that the grid must handle. Furthermore, a limitation of the proposed bidirectional neural network models lies in their computational complexity. While effective in capturing long-range dependencies, these models significantly increase training time for long sequences, making them less suitable for real-time applications particularly when dealing with complex, nonlinear data influenced by diverse factors such as weather conditions, socioeconomic variables, and seasonal variations in daylight hours.

Regarding the applicability of the proposed method to data from different time periods or countries, the deep learning models demonstrated strong performance using datasets from the United States. However, their effectiveness in other regions depends on several factors, including data quality and frequency, demand variability, seasonal patterns, socio-economic influences, and energy consumption behavior—all of which can vary significantly across countries. Nevertheless, since the architectures used in this study are data-driven and designed to capture temporal patterns, they can be generally adaptable to other regions, provided sufficient historical data is available. To support broader generalization, future work may involve testing the models on datasets from different countries to assess their transferability and adaptability to local grid conditions. Our study also emphasizes evaluating different algorithms across various times of the year and highlights the need for adjusting multiple hyperparameters for each dataset. This motivates the development of a more general model configuration capable of adapting to new datasets, which is why multiple forecasting algorithms were proposed in this work.

Conflict of interest

We declare that there is no conflict of interest.

Similarity rate (iThenticate): 20%

References

[1] J. C. Lu, X. Zhang and W. Sun, A real-time adaptive forecasting algorithm for electric power load. In 2005 IEEE/PES Transmission & Distribution Conference & Exposition, Asia and Pacific, pp. 1-5, IEEE, August, 2005.

[2] S. Karthika, V. Margaret and K. Balaraman, Hybrid short term load forecasting using ARIMA-SVM. In 2017 Innovations in Power and Advanced Computing Technologies (i-PACT), pp. 1-7, IEEE, April 2017.

[3] M. E. Sertkaya, M. Durmuş and B. Ergen, Detection of early stage alzheimer's disease in gradient-based MR images using deep learning methods. NÖHÜ Mühendislik Bilimleri Dergisi, 13(3), 2024 https://doi.org/10.28948/ngumuh.139083 0.

[4] X. Glorot and Y. Bengio, Deep sparse rectifier neural networks. In Proceedings of the fourteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, pp. 315-323, June 2011.

[5] D. Hendrycks and K. Gimpel, Gaussian error linear units (GELUs). Scientific Research Publishing, 2016. https://doi.org/10.48550/arXiv.1606.08415.

[6] Y. Liao, H. Tang, R. Li, L. Ran and L. Xie, Response prediction for linear and nonlinear structures based on data-driven deep learning. Applied Sciences, 13 (10), 5918, 2023. https://doi.org/10.3390/app13105918.

[7] S. Li, W. Li, C. Cook, C. Zhu and Y. Gao, Independently Recurrent Neural Network (IndRNN): Building a longer and deeper RNN. IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5457-5466, 2018.

[8] H. Lai, J. Shen, W. Zhang and Y. Yu, Bidirectional model-based policy optimization. In International Conference on Machine Learning, pp. 5618-5627, PMLR, November 2020.

[9] G. Zhang, B. E. Patuwo and M. Y. Hu, Forecasting with artificial neural networks: The state of the art.

international journal of forecasting, 14 (1), 35-62, 1998. https://doi.org/10.1016/S0169-2070(97)00044-7

[10] S. F. Ahmed, M. S. B. Alam, M. Hassan, M. R. Rozbu, T. Ishtiak, N. Rafa, M. Mofijur, A. B. M. Shawkat Ali and A. H. Gandomi, Deep learning modelling techniques: current progress, applications, advantages, and challenges. Artificial Intelligence Review, 56 (11), pp. 13521-13617, 2023. https://doi.org/10.1007/s10462-023-10466-8.

[11] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning. MIT Press, 2016.

[12] S. Hochreiter and J. Schmidhuber, Long Short-Term Memory. Neural Comput, 9 (8), 1735-1780, 1997. https://doi.org/10.1162/neco.1997.9.8.1735.

[13] M. Mozer, A focused backpropagation algorithm for temporal pattern recognition. Complex Systems, 3, 349-381, 1989.

[14] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill and Y. Xu, Short-term residential load forecasting based on LSTM recurrent neural network. IEEE transactions on smart grid, 10(1), 841-851, 2019. https://doi.org/10.1109/TSG.2017.2753802.

[15] S. Motepe, A. N. Hasan and R. Stopforth, Improving Load Forecasting Process for a Power Distribution Network Using Hybrid AI and Deep Learning Algorithms. IEEE Access, 7, 82584-82598, 2019. https://doi.org/10.1109/ACCESS.2019.2923796.

[16] Z. C. Lipton, J. Berkowitz and C. Elkan, A Critical Review of Recurrent Neural Networks for Sequence Learning. arXiv (Cornell University), 2015. https://doi.org/10.48550/arxiv.1506.00019.

[17] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk and Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078. 2014.

[18] N. Gruber and A. Jockisch, Are GRU cells more specific and LSTM cells more sensitive in motive classification of text?. Frontiers in artificial intelligence, 3, 40, 2020. https://doi.org/10.3389/frai.2020.00040.

[19] M. Abumohsen, A. Y. Owda and M. Owda, Electrical Load Forecasting Using LSTM, GRU, and RNN Algorithms. Energies, 16(5), 2283, 2023. https://doi.org/10.3390/en16052283.

[20] Y. Su and C. C. J. Kuo, On extended long short-term memory and dependent bidirectional recurrent neural network. Neurocomputing, 356, 151-161, 2019. https://doi.org/10.1016/j.neucom.2019.04.044.

[21] C. Cai, Y. Tao, T. Zhu and Z. Deng, Short-Term Load Forecasting Based on Deep Learning Bidirectional LSTM Neural Network. Applied Sciences, 11 (17), 8129, 2021. https://doi.org/10.3390/app11178129.

[22] M. Schuster and K. K. Paliwal, Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45 (11), 2673-2681, 1997. https://doi.org/10.1109/78.650093.

[23] G. Alex, A. Mohamed and G. Hinton, Speech recognition with deep recurrent neural networks. In 2013 IEEE international conference on acoustics, speech and signal processing, pp. 6645-6649. Ieee, 2013. https://doi.org/10.48550/arXiv.1303.5778.

[24] J. Du, Y. Cheng, Q. Zhou, J. Zhang, X. Zhang and G. Li, Power load forecasting using BiLSTM-attention. IOP Conference Series: Earth and Environmental Science, 440 (3), 032115, 2020. https://doi.org/10.1088/1755-1315/440/3/032115.

[25] C. Xiong, S. Merity and R. Socher, Dynamic memory networks for visual and textual question answering. In International conference on machine learning, pp. 2397-2406, PMLR, 4 March 2016.

[26] Y. Cheng, L. Yao, G. Xiang, G. Zhang, T. Tang and L. Zhong, Text sentiment orientation analysis based on multi-channel CNN and bidirectional GRU with attention mechanism. IEEE Access, 8, 134964-134975, 2020. https://doi.org/10.1109/ACCESS.2020.3005823.

[27] Y. Cheng, H. Sun, H. Chen, M. Li, Y. Cai, Z. Cai and J. Huang, Sentiment analysis using multi-head attention capsules with multi-channel CNN and bidirectional GRU. IEEE Access, 9, 60383-60395, 2021. https://doi.org/10.1109/access.2021.3073988.

[28] H. He, H. Wang, H. Ma, X. Liu, Y. Jia and G, Gong, Research on short-term power load forecasting based on Bi-GRU. Journal of Physics: Conference Series, 1639 (1), p. 012017, October 2020.

[29] S. Wang, C. Shao, J. Zhang, Y. Zheng and M. Meng, Traffic flow prediction using bi-directional gated recurrent unit method. Urban informatics, 1(1), 16, 2022. https://doi.org/10.1007/s44212-022-00015-z.

[30] A. Grami, Probability, random variables, statistics, and random processes: Fundamentals & applications. John Wiley & Sons, 2019.

[31] PJM, Data Miner 2 [online]. https://dataminer2.pjm.com/feed/hrl_load_metered, Accessed 02 May 2024.

[32] G. Zhao, A Test of non null hypothesis for linear trends in proportions. Communications in Statistics-Theory and Methods, 44 (8), 1621-1639, 2015. https://doi.org/10.1080/03610926.2013.776687.

[33] Y. LeCun, Y. Bengio and G. Hinton, Deep Learning. nature, 521 (7553), 436-444, 2015. https://doi.org/10.1038/nature14539.

[34] A. Colin Cameron and F. A. G. Windmeijer, An R-squared measure of goodness of fit for some common nonlinear regression models. Journal of econometrics, 77 (2), 329-342, 1997. https://doi.org/10.1016/S0304-4076(96)01818-0.

[35] T. Hong and S. Fan, Probabilistic electric load forecasting: A tutorial review. International Journal of Forecasting, 32 (3), 914-938, 2016. https://doi.org/10.1016/j.ijforecast.2015.11.011.

[36] A. de Myttenaere, B. Golden, B. Le Grand and F. Rossi, Mean Absolute Percentage Error for regression models. Neurocomputing, 192, 38-48, 2016. https://doi.org/10.1016/j.neucom.2015.12.114.