mathematics

MDPI

Article

Hybrid Deep Learning Applied on Saudi Smart Grids for Short-Term Load Forecasting

Abdullah Alrasheedi * and Abdulaziz Almalaq

Department of Electrical Engineering, Engineering College, University of Ha'il, Ha'il 55476, Saudi Arabia; a.almalaq@uoh.edu.sa
* Correspondence: aalrasheedi900@gmail.com

Abstract: Despite advancements in smart grid (SG) technology, effective load forecasting utilizing big data or large-scale datasets remains a complex task for energy management, planning, and control. The Saudi SGs, in alignment with the Saudi Vision 2030, have been envisioned as future electrical grids with a bidirectional flow of power and data. To that end, data analysis and predictive models can enhance Saudi SG planning and control via artificial intelligence (AI). Recently, many AI methods including deep learning (DL) algorithms for SG applications have been published in the literature and have shown superior time series predictions compared with conventional prediction models. Current load-prediction research for the Saudi grid focuses on identifying anticipated loads and consumptions, on utilizing limited historical data and the behavior of the load's consumption, and on conducting shallow forecasting models. However, little scientific proof on complex DL models or real-life application has been conducted by researchers; few articles have studied sophisticated large-scale prediction models for Saudi grids. This paper proposes hybrid DL methods to enhance the outcomes in Saudi SG load forecasting, to improve problem-relevant features, and to accurately predict complicated power consumption, with the goal of developing reliable forecasting models and of obtaining knowledge of the relationships between the various features and attributes in the Saudi SGs. The model in this paper utilizes a real dataset from the Jeddah and Medinah grids in Saudi Arabia for a full year, 2021, with a one-hour time resolution. A benchmark strategy using different conventional DL methods including artificial neural network, recurrent neural network (RNN), conventional neural networks (CNN), long short-term memory (LSTM), gated recurrent unit (GRU), and different real datasets is used to verify the proposed models. The prediction results demonstrate the effectiveness of the proposed hybrid DL models, with CNN–GRU and CNN–RNN with NRMSE obtaining 1.4673% and 1.222% improvements, respectively, in load forecasting accuracy.

Keywords: artificial neural network; convolutional neural network; deep learning; energy consumption; predictive models

MSC: 62J02; 62J99

Citation: Alrasheedi, A.; Almalaq, A. Hybrid Deep Learning Applied on Saudi Smart Grids for Short-Term Load Forecasting. *Mathematics* **2022**, *10*, 2666. https://doi.org/10.3390/math10152666

Academic Editor: Victor Leiva

Received: 11 June 2022
Accepted: 26 July 2022
Published: 28 July 2022

1. Introduction

For over a century, the Saudi electrical infrastructure has remained unaltered. The hierarchical grid's components are nearing their end of life. Saudi demand for power has progressively grown as the electrical system has deteriorated. The Saudi electric power distribution network is extremely complicated and unsuitable for 21st-century demands. A lack of automated analysis, limited visibility, the use of mechanical switches that cause delayed response times, and a lack of situational awareness are just a few of the flaws that are raised in the current electrical system [1,2]. Furthermore, the power and transportation industries have been major sources of greenhouse gas emissions on the planet [3]. As a result, to solve these issues, a new grid infrastructure in Saudi is required immediately.

The SG paradigm is a modern method of electricity transmission, and when it is checked, updated, and integrated with a bidirectional flow of energy and data, it can be employed. The conventional power system has emerged with the development of the SGs due to the rise of smart sensors, and information and communication technology [1]. According to the Saudi Vision 2030, new technologies have been used to create and improve the infrastructure of today's power systems in many ways. SGs are a novel idea in power system grids that attempts to construct durable, dependable, and efficient grids while lowering production costs. SGs may be made more efficient, reliable, and safe by enhancing them with renewable energy resources, automated control, and communication technology. By investing in the bidirectional flow of electricity and data, SGs aim to significantly enhance the use of technology and communication. The smart grid infrastructure is made up of enhanced sensing, communication, and computing capabilities that operate together in diverse sections of the power system, generating, and distribution [4]. Saudi Arabia is one of the few countries working on an ambitious plan to shift from traditional to smart power networks. The Saudi Arabian government's efforts in and goals of integrating SGs into the energy framework are reflected in the ambitious Vision 2030 plan. By the end of the next decade, it is expected that Saudi Arabia will have established itself as a global hub for smart energy technology.

Load forecasting is a critical activity that predicts future energy consumption for SGs to fulfill their principal functions at any given moment. Forecasting is a critical and fundamental factor in defining a future distribution system's capabilities needed to plan, operate, and manage the power system. Saudi utility companies have always relied on load prediction for planning and operational decisions. Future load prediction is much more critical in Saudi Arabia now that the energy sectors have been deregulated. A load estimate is critical for utilities since supply and demand fluctuate, and weather conditions and energy prices can increase by a factor of ten or more during peak periods. Short-term load forecasting can assist in estimating load flows and in making decisions to avoid overloading. Implementing such decisions in a timely manner improves network reliability and reduces the frequency of equipment failures and blackouts in Saudi Arabia. Load prediction is also crucial in terms of Saudi electrical systems for contract assessments and for evaluating the market's numerous sophisticated financial instruments on energy prices. Capital expenditure decisions based on long-term forecasting are also more essential in a deregulated economy than in a non-deregulated environment, where interest rate rises might be justified by investment proposals [5]. If the prediction is carried out incorrectly, it will influence all subsequent phases in the planning of future loads, putting the entire planning and operation in danger. An accurate load estimate not only aids in the optimization of future producing units but also assists in the identification of risk factors in planning, operation, and control activities. Furthermore, utilizing a designed bidding mechanism, electricity price forecasting gives important information to power suppliers and customers. To create their bidding strategy and to optimize their earnings and advantages, both suppliers and consumers require precise pricing projections. As a result, accurate and efficient load and price prediction have become a critical approach for achieving the aims of Saudi SGs. In the load-estimate challenge, a variety of AI approaches and machine learning algorithms are currently insufficient for properly anticipating the load in the required form [6]. Furthermore, most of these models are based on tiny datasets, with very significant prediction errors. Adding deep learning approaches to smart grid load forecasting will result in more accurate forecasts and efficient forecasting.

Deep learning (DL) is a kind of algorithm in which the structure contains deeper inner hidden layers. Its objective is to duplicate the grid of human brain connections to make machines such as computers comprehend as humans do. Many sectors have looked at AI for automation processes such as automated labor, image and audio recognition, decision-makers in crucial domains, and science research assistants [6]. Machine learning (ML) methods are AI's core techniques for extracting patterns from raw data to make subjective choices. Unsupervised learning and supervised learning are two types of machine learning algorithms. A supervised learning method is applied to a dataset comprising features, each of which has a label [1]. Another type is the unsupervised learning method, which learns from valuable qualities of the dataset structure and many features in the dataset [3]. In learning methods, the dataset is always divided into training and testing sub-datasets. The training and the validation datasets help the model learn from the raw dataset and validate the model performance, whereas the test dataset assists in examining the model's predictions based on unseen datasets. The inefficiency of learning models for high-dimensional datasets is a drawback of machine learning techniques. The ML is frequently utilized in load forecasting across a wide range of papers and applications. The typical neural network has one input layer, one hidden layer, and one output layer. Because it contains more layers and calculations, the structure of DL is so much more complicated than other neural network models [1]. DL is a multi-layered computational model that uses features as inputs to represent data in various representations. DL algorithms are employed in a variety of learning applications, particularly unsupervised learning. Many publications in the field have used DL techniques to help with Saudi load forecasting. The main contributions of the paper can be summarized as follows:

- An SG planning model of Saudi cities, which increases the planning and control progress and meets the planning of future Saudi load demands, is proposed.
- A load-forecasting technique based on hybrid DL algorithms is applied to predict the expected load growth.
- The hybrid DL algorithms of the time-series forecasting is adopted to solve the problem.
- The Saudi SGs of Jeddah and Madinah are investigated according to their different loads and characteristics.

The contents of the paper are organized as follows: a literature review is first presented in Section 2. In Section 3, we elaborate on general information about the methods and DL algorithms including (ANN, RNN, CNN, LSTM, BiLSTM, and GRU). Then, we reformulate the forecasting problem from the case study to real datasets of Saudi SGs in Jeddah and Madinah in Section 4. The prediction results are evaluated and compared with regular DL predictive models and conventional models in Section 5. Finally, some conclusions and future work are presented in Section 6.

2. Literature Review

Saudi Arabia just announced its Vision 2030 plan. The plan details long-term goals for converting the Kingdom's oil-based economy into one that is diverse, sustainable, and situated at a global trade crossroads. A strong renewable energy-procurement program might assist the Kingdom in accomplishing its renewable energy goals as well as its economic development and diversification goals [7]. The vision's goals as well as the strategies intended for the future energy management system to achieve them are examined in this article [8]. Electricity shortages and how to satisfy the predicted rise in demand in the foreseeable future are also highlighted. Furthermore, the researchers have applied a comprehensive study of the potential applications of renewable and sustainable energy technologies for developing an energy policy to achieve energy security and cost reduction as well as to ensure the efficiency of renewable and sustainable energy applications in the Kingdom of Saudi Arabia for long-term prosperity and energy security. The authors of [2] suggested a new technique that forecasts peak loads for the following year using hourly daily loads. The method was based on applying multivariable regression to the hourly loads from the prior year. Three regression models were examined in this study: linear, polynomial, and exponential. The proposed models were used to simulate actual demands on the Jordanian electricity grid. The findings acquired utilizing the suggested methods demonstrated that they perform almost as well as results produced using the commonly utilized exponentiation regression technique. Furthermore, the proposed methodology had a peak load forecast accuracy of roughly 90%. The authors of [9] focused on the characterization, active learning, and long-term forecast of power consumption at a single KSA level. The authors claimed that load forecasting is crucial for the electric industry in Saudi Arabia's deregulated economy. The purchase and generation of energy, load switching, contract review, and infrastructure development are only a few examples of applications. A variety of mathematical approaches to load forecasting have been devised. Peak load was shown to be a major determinant to increase in that study [10]. Reference [11] discussed several different approaches to load forecasting.

The researchers in [12] aimed to present an overview of how the load-forecasting performance in SGs might be enhanced using DL methods. According to [12], there is little evidence that researchers have investigated the issue of combining different DL methods for complex large-scale load forecasting in SGs to develop a reliable predictive model and to understand the relationships between the different predictive models and deep learning methods. To address this gap, our article investigates the capability of hybridizing two DL algorithms and studying their prediction accuracy for a real load dataset.

In [13], the authors proposed a novel hybrid prediction methodology based on evolutionary DL combined with the genetic algorithm. The evolutionary DL algorithm was defined as an LSTM method that optimizes time window lags and uses hidden neurons. The results showed that the evolutionary DL model outperforms conventional and regular prediction models.

The authors in [14] used two different sequence-to-sequence (S2S) RNN techniques on a building-level energy consumption dataset to consider hybridizing two DL models. A comparison of five commonly used short-term load forecasting approaches was provided. Multiple linear regression, stochastic time series, general exponential smoothing, state space, Kalman filter, and knowledge-based methods are some of these techniques. Each of these strategies is briefly discussed, along with their relevant equations. For a direct comparison of these distinct forecasting methodologies, algorithms implementing these strategies were written and applied to the same database. The findings were compared to give a sense of the inherent difficulty of each of these strategies and their performances [15]. GRU encodes information for the first RNNs, and the LSTM model successively creates outputs utilizing this information for the second RNN. The new method was compared with traditional DL algorithms in three different prediction-length scenarios. Using same-day selection and an LSTM network for buildings, Yong et al. [16] suggested a short-term load-forecasting technique for buildings. For residential buildings, the LSTM algorithm was used in network-based short-term load forecasting [17]. To produce the feature vector retrieved from lagged load variables, the authors used wavelet decomposition and collaborative representation approaches. The authors in [18] suggested a model for estimating power consumption at a specific period based on empirical mode decomposition (EMD) and an LSTM network. The residential buildings dataset was the subject of a similar study in [19]. For a single user with a one-minute resolution, the authors found that the LSTM-based RNN model outperformed the simple RNN and GRU–RNN models. For short- and medium-term load forecasting, LSTM prediction models with various configurations were built using data from France's metropolitan energy usage. They compared the results to ML models [20]. Furthermore, for a cluster analysis of the load trend, LSTM multi-input, multi-output models were trained and compared with ML models [9]. The LSTM model beat the backpropagation neural network (BPNN) in the task of short-term load forecasting, according to Kong et al. [21]. Using the most recent popular DL models, these previous studies produced respectable and good forecasting outcomes.

Another method for word-level recognition was developed utilizing CNN architecture, which assesses whether a specific n-gram is present in a specific area of the image [19]. Krishnan et al. [20] also utilized a CNN to learn PHOC-like properties for images and embedded the text represented by the PHOC features into a common subspace with the embedded images. Both of these methods were based on the proposed PHOC representation in [9]. Machine learning with artificial neural networks was used in [22–24] to forecast the demand for an electric power substation at a specific hour of the day. From September to November 2018, historical load data at each hour of the day were collected from the 33/11 kV substation at Kakatiya University in Warangal. Based on the approach utilized to forecast the load, a novel artificial neural network architecture was designed. To forecast the load on a 33/11 kV electric power substation, the generated model was run in MATLAB using historical data. According to the analysis, the proposed design estimated the load with more accuracy.

To forecast household EEC, Le et al. integrated a BiLSTM network with a convolutional neural network (CNN). The CNN was utilized to extract the discriminative feature values from the IHEPC dataset first, followed by the BiLSTM network to produce predictions. To anticipate and estimate energy usage and needs, Ishaq et al. proposed a new ensemble-based DL model. Data were preprocessed using transformation, normalization, and cleaning procedures and then input into the ensemble model, which included the CNN and BiLSTM networks, which extracted discriminative feature values. To improve and ensure the prediction performance of the presented model, an active learning approach based on the moving window was developed in that work. A Korean commercial building dataset was used to see how well the model worked, and the results were shown in the next phase. MAPE, RMSE, MAE, and MSE values were used to measure this [25]. In [26], the authors proposed a predictive model based on a sliding-window algorithm and a stacking ensemble neural network for a same–day load predictive method.

For medium- to long-range forecasting for a larger metropolitan region, an LSTM model employing only ideally selected time-delayed features captured all of the properties of complicated time series and demonstrated lower mean absolute errors (MAEs) and root mean square errors (RMSEs) [9]. In Reference [27], why Adam generalizes less well than SGD was investigated and a variation of Adam was proposed to close the generalization gap. Normalized direction-preserving Adam, the suggested method, allows for more precise control of the direction and step size for updating weight vectors, resulting in much-enhanced generalization performance. The authors increased the generalization performance in classification problems by regularizing the softmax logits following a similar logic and shed light on why some optimization techniques generalize better than others by bridging the gap between SGD and Adam. Reference [5] discussed the most common DL methods used in the literature to solve load-forecasting problems in SG and power systems. That study utilized many types of DL methods for the application of power systems and smart-grid load forecasting. It also compared the accuracy results between RMSE and MAE for the applications examined and demonstrated that the use of convolutional neural networks (CNNs) with the k-means algorithm resulted in a significant reduction in RMSE.

3. Forecasting Methods

3.1. Artificial Neural Networks

Neural networks are biologically inspired models of computation. Generally, a neural network consists of a set of artificial neurons, commonly referred to as nodes or units, and a set of directed edges between them, which intuitively represent the synapses in a biological neural network. An artificial neural network (ANN) is a generic network that encompasses any representation of neural networks, e.g., feedforward, recurrent, convolutional, radial bias function, etc. An example of the simple architecture of multi-layered feedforward neural networks is demonstrated in Figure 1. They are also called "processing elements" (PE) because they process data. Weighted inputs, a transfer function, and one output are all included in each PE. A mathematical formula that balances these inputs and outputs is called a PE. Because the connection weights represent the system's memory, ANNs are also known as connectionist models. ANNs, on the other hand, are capable of digesting large volumes of data and making predictions that are sometimes startlingly accurate.

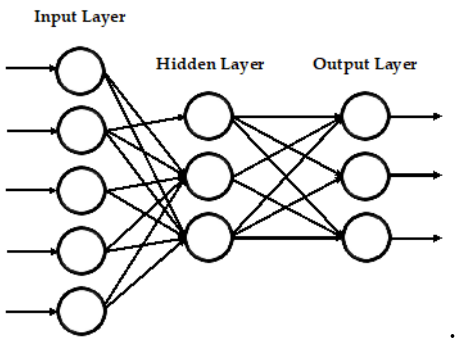


Figure 1. Simple ANN network architecture.

As seen in Figure 1, all artificial neural networks have similar structures or topologies. Some of the neurons in that structure create an interface with the real environment to accept inputs. The network's outputs are delivered to the real world through other neurons. This output could be the character that the network believes it has scanned or the image that it believes is being displayed. The remaining neurons are hidden from view. Many types of generic ANN architectures are developed with different connections and computations such as convolutional neural networks and recurrent neural networks, which are defined in the following subsections.

3.2. Convolutional Neural Networks

In DL applications, the CNN method is typically used for image-recognition tasks. For classification and regression applications, the feature matrix is created automatically from the convolution layer of CNN, unlike traditional approaches. The convolution layers are based on the "convolution" operator's processing principle, and their primary goal is to extract useful features from input data while maintaining correlation between data samples. The feature matrix is created by moving a filter over n-dimensional data, followed by an activation function that replaces all negative values in the feature matrix with zero.

Because CNN models were created for image-classification problems, the input vector is represented in two dimensions, such as image pixels and color information. A feature matrix may be produced via a shifting process with a fixed period over time, and a similar technique can be applied to 1-D sequential data. The 1-D CNN architecture is mostly applied for time-series applications. The structure of data and how the convolution operator traverses across data are the primary differences between multi-dimensional CNN models [28]. According to [29], the authors proposed 1-D CNN and used a one-dimensional time-series signal as input for the fault-diagnosis problem. The authors concluded that the interpretability of 1-D CNN is excellent compared with 2-D CNN for time-series fault diagnosis, has a better feature-extraction mechanism compared with the convolutional kernel analysis, and produces an output in the time-domain. Basically, the architecture of a 1-D CNN includes an input layer such as 1-D time-series dataset, a convolutional layer, a pooling layer, a fully connected layer, and an output layer, as shown in Figure 2. The convolutional layer is responsible for generating a feature map for the 1-D time series dataset, where the convolutional kernels extract the different features of the input. Since the number of features from the convolutional layer increases extensively, a down-sampling techniques is employed in the pooling layer by averaging the feature size according to the pooling window. Finally, the fully connected layer maps all features from the previous layer to the activation function.

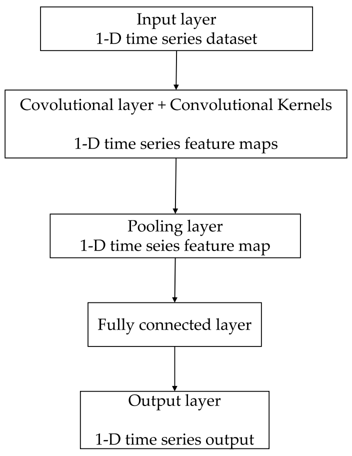


Figure 2. Architecture of a 1-D CNN network.

3.3. Recurrent Neural Network (RNN)

Recurrent neural networks (RNNs) are a kind of ANN with one input layer, many hidden layers, and one output layer. These hidden layers feature recurrent connections, making them appropriate for sequential computing as in [30]. The recurrent connections in the hidden layer are from the output to the input. It is frequently used for time-series sequences because its architecture includes a memory state that aids in the processing of sequential data. The method was used for a variety of load-forecasting situations.

Two equations specify all calculations necessary for computation at each time step on the forward pass in a simple RNN, as shown in Figure 3:

$$s_t = f_\theta(Ux_t + UW_{s(t-1)}) \quad (1)$$

$$h_t = f_\propto(Vs_t) \quad (2)$$

where x_t denotes the input at timestamp t, s_t denotes the state ate timestamp t, h_t denotes the output at t, and the current state s_t is computed based on current input x_t and on the previously hidden state s_{t-1} [23,24].

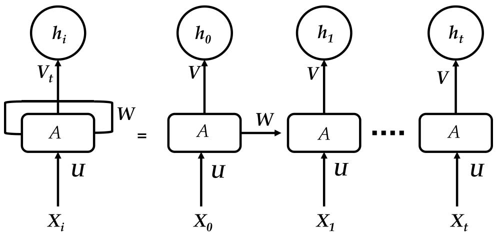


Figure 3. Architecture of an RNN network.

3.4. Long-Short Term Memory (LSTM)

The LSTM model is similar to the RNN model in structure, but Gers et al. [23] replaced each node of the hidden layer unit with a processing unit called "memory cell" and then improved it with an extra unit called "forget cell". Each memory cell has a recurrent node with a set weight to ensure that the gradient does not vanish or explode over time. Memory cells and gate units make up the LSTM cell's basic structure. Input gate it, forget gate ft, and output gate O_t are the three gates that make up the gating mechanism, where σ stands for the standard sigmoid function:

$$\sigma\ (x) = \frac{1}{1+\ e^x} \quad (3)$$

For the objective function, we use the square loss function given by the following formula:

$$e = \sum_{t=1}^{n}(y_t - p_t)^2 \tag{4}$$

where y_t is the actual output, p_t is the anticipated load Adam optimizer, and a modified stochastic gradient descent (SGD) optimizer with adaptive learning rates is used for back-propagation across time to minimize the training error while avoiding local minimal points (BPTT). Figure 4 depicts the structure of a single-cell LSTM memory block as in [31]. The cell states at timestamps t and $t - 1$ are denoted by c_t and c_{t-1}. The forget gate uses the sigmoid layer to select the information to be retained in c_{t-1} and takes x and h as input. The value of c is determined by the input gate i using x and h. Using both the sigmoid and tanh layers, the output gate O_t adjusts the output of the LSTM cell based on c_t. The equations of all nodes in an LSTM cell are given by

$$f_t = \sigma(W_f \cdot [h_{t-1} \cdot x_t] + b_f) \tag{5}$$

$$i_t = \sigma(W_i \cdot [h_{t-1} \cdot x_t] + b_i) \tag{6}$$

$$c_{t'} = \tanh(W_c \cdot [h_{t-1} \cdot x_t] + b_c) \tag{7}$$

$$c_t = f_t \odot c_{t-1} \oplus i_t \odot c_{t'} \tag{8}$$

$$O_t = \sigma(W_o \cdot [h_{t-1} \cdot x_t] + b_o) \tag{9}$$

$$h_t = O_t \odot \tanh(c_t) \tag{10}$$

where $W_{(.)}$ and $b_{(.)}$ are the weight matrices and biases, respectively, and denote the sigmoid activation function. While the input gate is in charge of the memory unit's update mechanism, the output gate and the forget gate determine whether the present information in the unit will be kept or wiped. The internal state of the LSTM structure does not change when the input gate is set to zero, and the memory cell retains its current activity. As long as both the input and output gates are closed, this process continues with intermediate time steps.

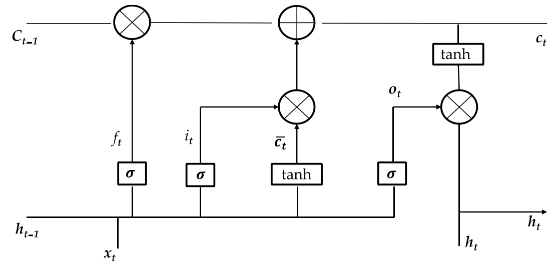


Figure 4. Architecture of the LSTM cell.

3.5. Gate Recurrent Unit (GRU)

The GRU is a variant of the LSTM method and was introduced by K. Cho [11]. The GRU is based on the LSTM unit, but it is thought to be easier to compute and implement. It has the same resistance to the vanishing gradient problem as the LSTM method, but its internal structure is simpler, making it easier to train because it requires fewer computations to update its hidden state. The reset gate r and the update gate z are the two gates that make up a typical GRU cell as in [32]. The hidden state output at time t is calculated by combining the hidden state at time $t - 1$ with the input time-series value at time t, as illustrated in the mathematical functions used to control the gating mechanism in the GRU cell.

$$t = \sigma\ (W_z h_{t-1} + U_z x_t) \tag{11}$$

$$r = \sigma\,(W_r h_{t-1} + U_r x_t) \tag{12}$$

$$c = \tanh(W_c(h_{t-1} \otimes r) + U_c x_t) \tag{13}$$

$$h_t = (z \otimes c) + ((1 - z) \otimes h_{t-1}) \tag{14}$$

Figure 5 depicts the GRU's update and reset gates, which are comparable with the forget and input gates in the LSTM unit. The update gate specifies how much past memory should be retained, whereas the reset gate specifies how the current input should be combined with the previous memory. The main difference is that the GRU uses only integration to fully expose its memory content (but with an adaptive time constant controlled by the update gate).

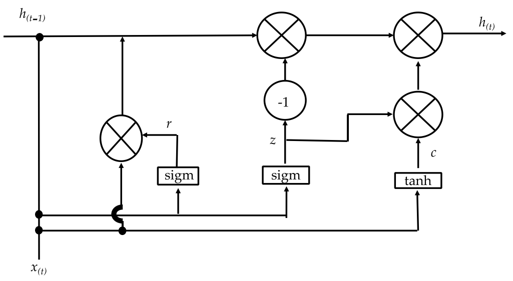


Figure 5. Architecture of the GRU cell.

3.6. Bidirectional Long Short-Term Memory (BiLSTM)

The bidirectional RNN (BiRNN), first proposed by Schuster and Paliwal, uses both past and future data to calculate the output values of a regular time series. Unlike the LSTM network, where only past information has an effect on output, this technique can be trained without the need for a predetermined input length. According to [33], two hidden layer nodes exist in the BiRNN architecture, and these layers are coupled to both the input and output layers. This means that the first hidden layer's recurrent unit is linked to previous time steps in a forward direction. However, the second hidden layer's recurrent unit is linked in the opposite direction. The model structure is shown in Figure 6.

The same procedure used to train a normal RNN may be used to train a BRNN. Due to the lack of interaction between hidden layers, a typical feed-forward network unfolded over time might also be preferred. BRNN's mathematical formulae are as follows:

$$h_t = \sigma\,(w_{h,x} x_t + w_{h,h} h_{t-1} + b_h) \tag{15}$$

$$e_t = \sigma\,(w_{e,x} x_t + w_{e,e} e_{t-1} + b_e) \tag{16}$$

$$y_t = f(x)_i \times (w_{y,h} h_t + w_{y,e} e_t + b_y) \tag{17}$$

The values of hidden layers in both the forward and backward directions of BRNN are represented by h_t and e_t, respectively. As a result, the LSTM and BRNN approaches are complementary and can be combined. The LSTM method can be used to create a new unit that can be employed in the hidden layer, while the BRNN approach connects these units.

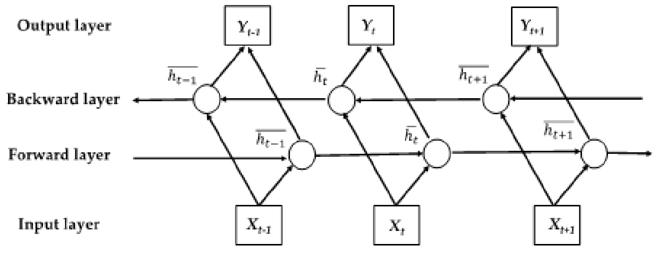
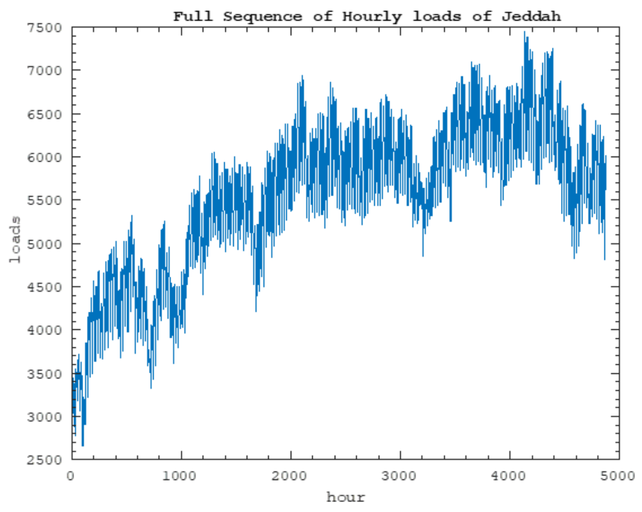


Figure 6. Architecture of the BiLSTM cell.

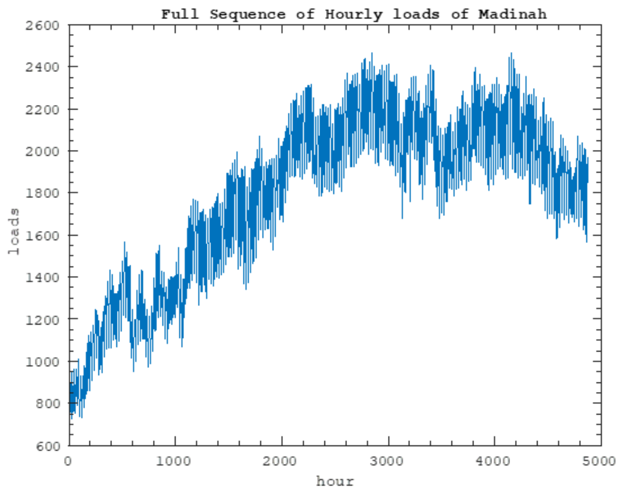
4. Forecasting Modeling

4.1. Data Description

The dataset used in this study is a real dataset of Saudi SGs, consisting of loads for Jeddah city for the year 2021 with a one-hour resolution and thus containing a total of 4876 hourly consumption for two seasons during 2021. Figure 7a,b represent the load data from the beginning of March to the end of September 2021 for the cities of Jeddah and Madinah, which are two of the biggest cities in Saudi Arabia in terms of population, and industrial and commercial density. Each dataset consists of 4876 consumptions and can be used to generate our proposed DL model for each load forecasting model. The input of the prediction model is the historical energy-consumption values of Jeddah's loads in Saudi Arabia, collected by customers' smart meters. This DL method also covers inconsistencies, missing and outlier values, and issues that arise in the datasets of energy consumptions. The output of the proposed short-term load forecasting is a precise future energy-consumption prediction. It is possible to predict the loads using one of the methods of DL and to obtain optimal results in terms of RMSE, NRMSE, and MAPE. The forecasting results can provide ideas for improving Saudi energy systems because the Ministry of Energy changes its physical systems based on the results.



(**a**)



(**b**)

Figure 7. Real dataset loads from the Saudi SGs (March 2021–September 2021). (**a**) Jeddah city load with a one-hour time resolution. (**b**) Madinah city load for the year of 2021 with a one-hour time resolution.

4.2. Hybrid DL Approach

Using energy usage for customer load forecasting in a smart grid is a difficult time-series problem. To handle complicated systems and to reduce prediction errors, we used a hybrid DL strategy reinforced with advanced preprocessing techniques in this study to address the time-series issue. Normally, a time-series dataset consists of a large amount of noise and requires reliable smoothing and denoising techniques for more accurate prediction. In [34], the Savitzky–Golay (SG) filter was proposed to eliminate noise from a dataset. We plan to adopt this technique for our future work. As a result, the suggested method's modeling framework was divided into two parts: an advanced single-layer model and a hybrid DL model.

Figure 8 depicts the architecture of the proposed CNN–LSTM, CNN–GRU, and CNN–BiLSTM-based DL frameworks. The convolutional operation allows the initial layer to learn low-level features in the applied input. A pooling layer is frequently added to mitigate the invariance limitation of the resulting feature map. The activation function is used to improve the model's capacity to learn complicated structures. We used a dropout layer between the CNN feature extraction block and the LSTM, GRU, and BiLSTM sequences of learning in this study. To avoid overfitting, this layer comprises a random selection of neurons and deactivates some of them.

When creating a CNN model, typically, a coarse-to-fine strategy is used. Because there are so many trainable parameters in this structure, it adds to the computational complexity. To avoid overfitting, we chose a kernel size of 32 for the convolution layer. There were 200 hidden units in each layer of the LSTM, GRU, and BiLSTM layers in the sequence-learning block. We used one layer for each of these layers. The return sequence for the LSTM, GRU, and BiLSTM layer was set to true so that the network outputs the whole sequence of hidden states, whereas the return sequence for the final layer was set to false so that the network only outputs the hidden state at the final time step. To avoid overfitting, we employed the dropout layer before the fully linked layer. Figure 9 shows a flowchart of the overall proposed hybrid DL algorithm. The flowchart starts with the data input from the real dataset for Jeddah load or Medinah load. The input data were then fed to the preprocessing stage, which consists of normalizing and splitting the dataset into two main portions: 70% for training and 30% for testing. Then, the training dataset was applied to the hybrid DL model predictive model, which consists of a CNN in the first stage first and RNN–LSTM–BiLSTM–GRU in the second stage. Each experiment of the hybrid DL algorithm consisted of a CNN and one of the conventional DL algorithms in the second stage. Then, the model was tested with the remaining 30% of the dataset. If the prediction performance and the forecasting accuracy was the lowest and satisfactory, the model was terminated; otherwise, another combination of the hybrid DL model was chosen and then trained again with the same training dataset.

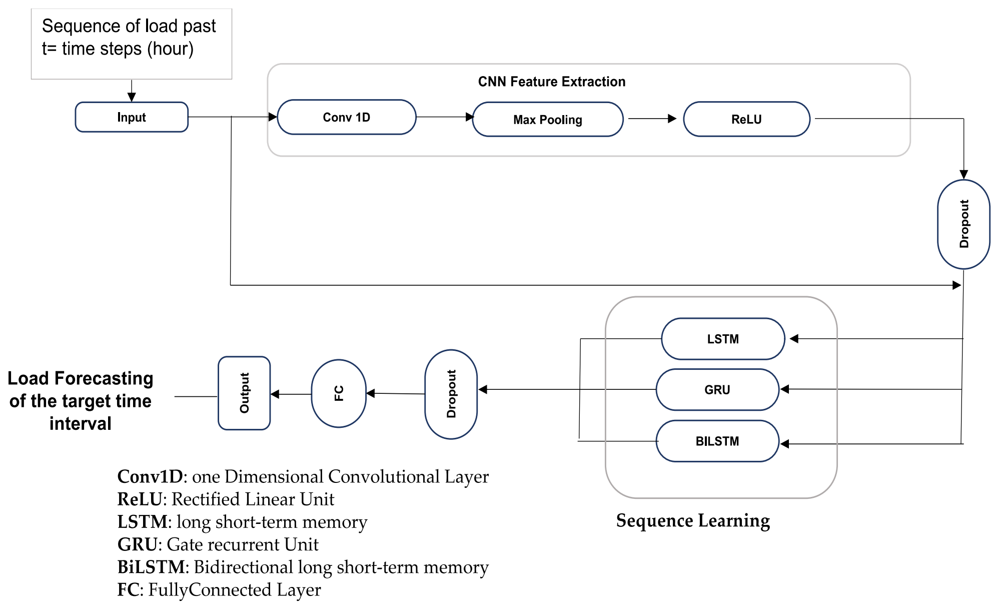


Figure 8. Proposed hybrid DL algorithm architecture for 1-D CNN combined with different DL algorithms.

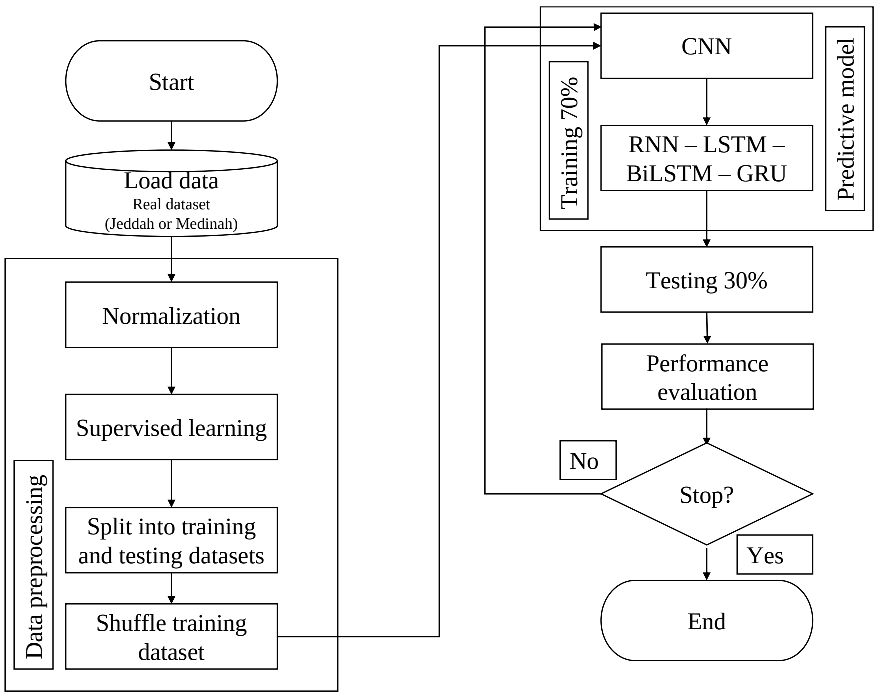


Figure 9. Proposed hybrid DL algorithms flowchart for load-forecasting modeling.

Table 1 shows the parameters for the DL system that was built. In this study, we used Adam, a well-known optimizer, as well as the mean absolute error as a loss function. The proposed hybrid DL model was trained using a 1-D CNN to extract features from time-series data and an LSTM or GRU or BiLSTM model for feature analysis. In the feature extraction and the first phase of the extracted feature analysis (LSTM or GRU or BiLSTM), the model was trained forward, whereas in the second stage of the extracted feature analysis, it was trained backward.

Table 1. Hyperparameters of the proposed hybrid DL model.

Parameter	Setting
Network architecture	LSTM, GRU, BiLSTM, RNN, ANN CNN–LSTM, CNN–GRU, CNN–BiLSTM
Optimizer	Adam
Loss Function	Mean Absolute Error (MAE)
Learning Rate	{0.0005}
Adjustment	learning rate = 1×10^{-6}
Batch Size	64
Epoch	2000
Iteration per epoch	52
Lag	1:50 h look back
TrainFcn	gradient descent momentum (traingdx)
LearnRateDropPeriod	400
LearnRateDropFactor	0.2
CNN, ANN	32 hidden units
LSTM, GRU, BiLSTM layers	400 hidden units

Two layers of the 1-D CNN with a 5-filter size and 32 hidden units were used in the training process to improve the extraction of input features, and two layers of the LSTM or GRU or BLSTM with 400 hidden units were used to analyze the features retrieved and to predict the output. Our model employs a rectified linear unit (ReLU) activation function with a total of 2000 training epochs. For DL fine-tuning and complex stochastic processes, the hyperparameters, which include the network design, the number of hidden units, an activation function, a loss function, and the number of epochs, constitute a nondeterministic polynomial (NP) optimization problem. In our modeling, the hyperparameters were chosen using a trial-and-error method, though 400 hidden units have been noticed as the most optimal architecture for DL algorithms after our many training trials. Although different types of DL algorithms have their own methods of computation, specifying the optimal hyperparameters to all models is a key element of fast-track modeling due to the high computational cost and prediction accuracy.

In [35], a hybrid CNN–RNN model was proposed and was implemented using MATLAB. In our modeling, we adopted the generic algorithm of the CNN and RNN hybridization, and we modified our proposed models based on different CNN hybridization with the RNN, LSTM, BiLSTM, and GRU. A summary of the MATLAB code is represented in Figure 10 as a pseudocode.

The platforms used in our modeling were a MacBook Pro (13-inch, M1, 2020) and 8 GB (4 for performance and 4 for efficiency) as the total number of cores. The developmental environment of our system was MATLAB 64-Bit (maci64), where the DL models were implemented with the deep learning toolbox [36].

```
data = xlsread("data.xlsx");
mu = mean(dataTrain);
sig = std(dataTrain);
if strcmp(NetOption,"CNN-RNN")
elseif strcmp(NetOption,"CNN-LSTM")
 elseif strcmp(NetOption,"CNN-BiLSTM")
elseif strcmp(NetOption,"CNN-GRU")
elseif strcmp(NetOption,"RNN")
     elseif strcmp(NetOption,"GRU")
    elseif strcmp(NetOption,"LSTM")
  elseif strcmp(NetOption,"BiLSTM")
elseif strcmp(NetOption,'FeedForwardNet')
end
YPred = sig.*YPred + mu;
YTest = sig.*YTest + mu;
```

Figure 10. Pseudocode of the proposed hybrid DL algorithms.

5. Results

To evaluate the forecasting-performance results, we utilized 30% of the dataset to test the model and 70% of the dataset to train the model. Conventional metrics were used to evaluate the prediction models and to evaluate the forecasting in our experiments. Traditional metrics such as the root mean squared error (RMSE); the coefficient of variation (CV) of RMSE, known as the normalized root mean squared error (NRMSE); MAE; and MAPE are defined as follows:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - p_i)^2}{n}} \tag{18}$$

$$\text{NRMSE} = \frac{\text{RMSE}}{\bar{y}} \times 100\% \tag{19}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - p_i| \tag{20}$$

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - p_i|}{yi} \times 100 \tag{21}$$

where n represents the total number of data points in the time series, $\bar{y}$ represents the average of the measured time series in the original scale of the dataset, and p_i is the predicted output of the time series.

All of the DL models and empirical approaches shown in Table 2 were evaluated only for comparison. While most of the conventional DL algorithms such as ANN, RNN, LSTM, BiLSTM, and GRU were proposed in the literature only recently [13,14,25,30,37], we have benchmarked our proposed hybrid algorithms to them. Since 30% of the dataset was used to test the trained model with unseen data, the prediction performance of the testing can be used to evaluate the performance and the accuracy of the forecasting model. According to Reference [35], a CNN hybrid with the RNN forecasting model was demonstrated to be useful for medical-forecasting applications in chickenpox cases based on prior months. Our proposed model was benchmarked with their modeling and design. Therefore, the idea of a hybrid CNN–RNN was applied to a dataset to predict the loads of Jeddah and Madinah as benchmarks. In addition, we applied a hybrid method of CNN with LSTM, GRU, and BiLSTM for a comparison of the results. We also used the R, RMSE, NRMSE, and MAPE values. For the performance of the individual ANN, RNN, LSTM, GRU, and BiLSTM models, we see that the R, RMSE, NRMSE, and MAPE of the BiLSTM-based model were 0.9869, 80.5873, 1.526, and 0.9991, respectively, for the optimal forecast. However, the average MAPE of the BiLSTM-based DL model was higher than that of the other DL models. For analyzing the performance of the suggested hybrid CNN–GRU, CNN–LSTM, and CNN–BiLSTM models, we chose a DL model based on the LSTM, GRU, and BiLSTM architectures based on this research. The suggested hybrid CNN–BiLSTM model had average R, RMSE, NRMSE, and MAPE of 0.9872, 78.085, 1.4786, and 0.9497, respectively. According to Table 2, on the other hand, the model based on CNN–GRU, produced the best results, with average R, RMSE, NRMSE, and MAPE of 0.9875, 77.4877, 1.4673, and 0.9505, respectively. State-of-the-art methods such as LSTM and GRU were unsuitable for forecasting compared with the suggested method. RNN-based methods were also not as good as the suggested method. The simple ANN showed better performance than RNN, LSTM, BiLSTM, and GRU because the size of the dataset was not very large. If the dataset contained big data, the DL models would perform better than the simple ANN. In Table 3, the prediction performance of the CNN–RNN model showed the most accurate prediction result. The hybrid CNN–RNN model of Madinah forecasting loads had average R, RMSE, NRMSE, and MAPE of 0.9927, 20.7501, 1.2227, and 0.7591, respectively. In addition, the hybrid model outperforms other DL models in predicting loads in both cases.

Table 2. Prediction error (R, RMSE, NRMSE, and MAPE) for the Jeddah results.

Model	R	RMSE	NRMSE (%)	MAPE
ANN	0.9873	78.3173	1.483	0.9562
RNN	0.98577	95.2624	1.8039	1.1695
LSTM	0.9868	81.6872	1.5468	1.0144
GRU	0.9868	81.3668	1.5407	1.0100
BiLSTM	0.9869	80.5873	1.526	0.9991
CNN–RNN	0.9819	97.321	1.8428	1.2194
CNN–LSTM	0.9873	77.8123	1.4734	0.9511
CNN–BiLSTM	0.9872	78.085	1.4786	0.9497
CNN–GRU	**0.9875**	**77.4877**	**1.4673**	**0.9505**

Table 3. Prediction error (R, RMSE, NRMSE, and MAPE) for the Madinah results.

Model	R	RMSE	NRMSE (%)	MAPE
ANN	0.9749	38.3944	2.2623	1.4723
RNN	0.992	21.4315	1.2628	0.7667
LSTM	0.992	21.6804	1.2775	0.7778
BiLSTM	0.992	21.7257	1.2802	0.7776
GRU	0.9922	21.419	1.2621	0.7614
CNN–RNN	**0.9927**	**20.7501**	**1.2227**	**0.7591**
CNN–LSTM	0.9918	21.9719	1.2947	0.7946
CNN–BiLSTM	0.9918	22.0343	1.2983	0.7989
CNN–GRU	0.9917	22.141	1.3046	0.8074

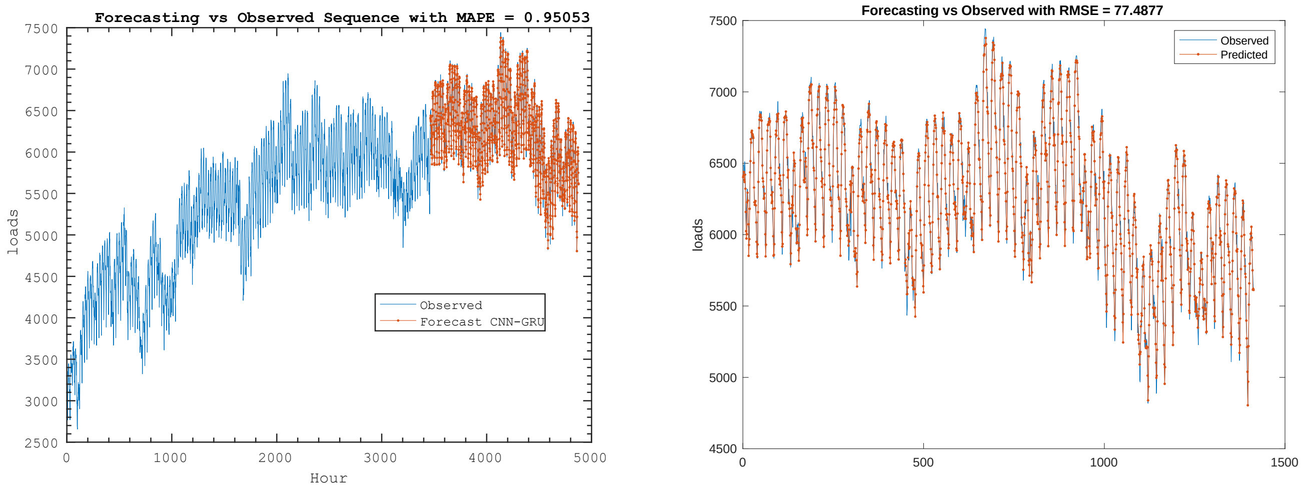
In Table 4, the computational cost of the hybrid DL models and the conventional DL models show a variety of different time complexities for each model according to their architecture and weights. From the table, the computational cost of conventional DL algorithms seems to be less than that of the hybrid models. This concludes that the shortcoming of the proposed hybrid DL algorithm is the critical time cost. A potential solution to this problem is to tune the hyperparameters of each model.

The hybrid CNN–GRU model was created and represented as MAPE, RMSE, and a regression plot. Forecasting in comparison with the observed sequences was assessed and plotted. The forecasting load was observed for an iteration at a different hour. The actual–observed forecasting had an MAPE of 0.95053 and is shown in Figure 11a, while with RMSE it was found to be 77.4877.

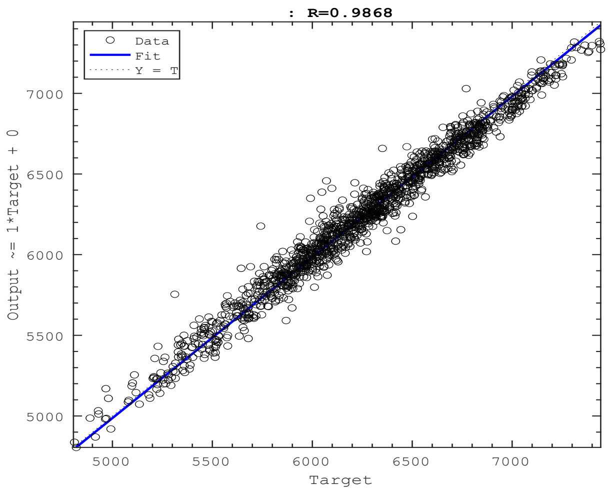
Table 4. Computational cost of different DL and hybrid DL algorithms.

Model	Training Time
RNN	77 min 44 s
LSTM	43 min 25 s
BiLSTM	107 min 59 s
GRU	60 min 40 s
CNN–RNN	91min 54 s
CNN–LSTM	62 min 25 s
CNN–BiLSTM	135 min 10 s
CNN–GRU	**78 min 26 s**

In MATLAB, an ANN architecture was created by considering various hidden neurons in the hidden layer. The network's performance was measured in terms of RMSE and R at various hidden neurons. To predict the load, the architecture with the best performance, i.e., one with a low RMSE and a high R, was chosen. In Table 2, the RMSE was used to evaluate the performance of the best model using 32 hidden units.



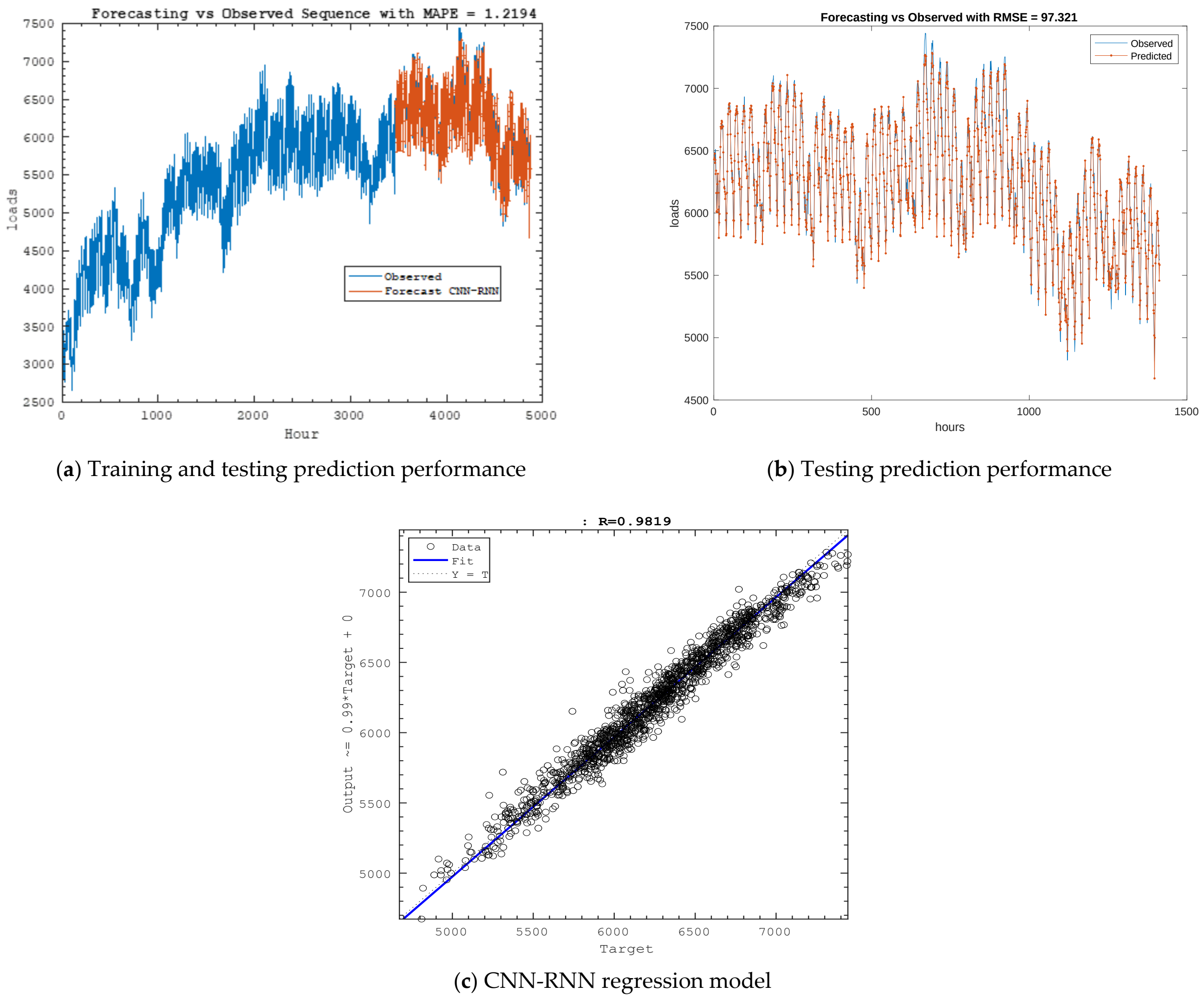
(**a**) Training and testing prediction performance (**b**) Testing prediction performance



(**c**) CNN–GRU regression model

Figure 11. Hybrid CNN–GRU model prediction performance for the Jeddah load: (**a**) training and testing, (**b**) testing, and (**c**) regression plot.

Figure 12 and Figure 13 presents the model's performance at each level of training and testing for the CNN-RNN model and ANN model, respectively. The regression plot containing data from training and testing resulted in a regression coefficient of 0.9873, which is acceptable because it is close to one. Figure 13c shows an error histogram plot with training and testing data, and all of the errors have a normal distribution. Figure 14 shows the results of the real and forecast Madinah loads that were applied in the hybrid CNN–RNN. Figure 14c shows the regression data from training and testing for the hybrid DL model for the Madinah load dataset with a regression coefficient R = 0.99267.

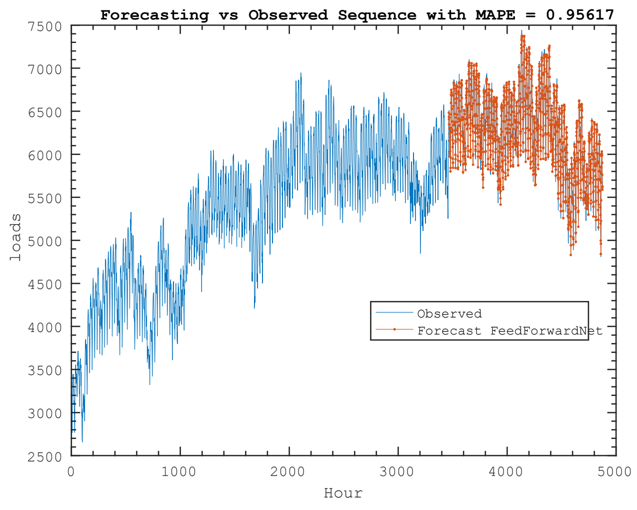


(**a**) Training and testing prediction performance

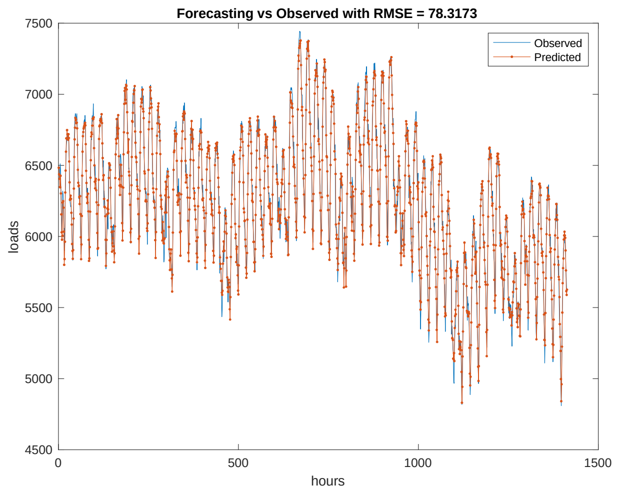
(**b**) Testing prediction performance

(**c**) CNN-RNN regression model

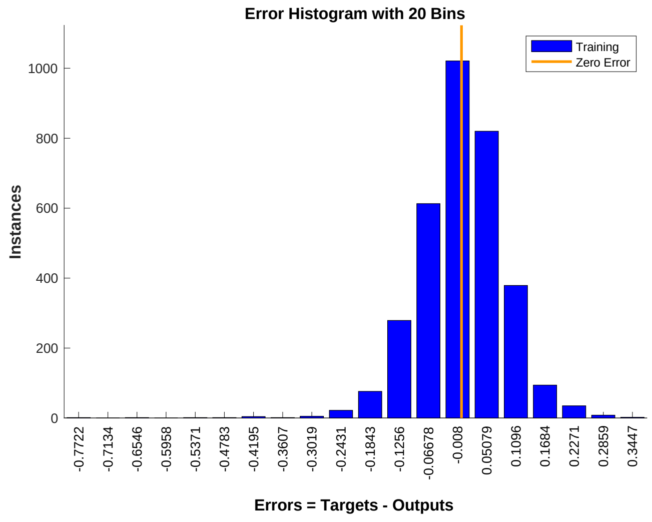
Figure 12. CNN-RNN model prediction performance for Jeddah load: (**a**) training and testing, (**b**) testing, and (**c**) regression plot.



(**a**) Training and testing prediction performance



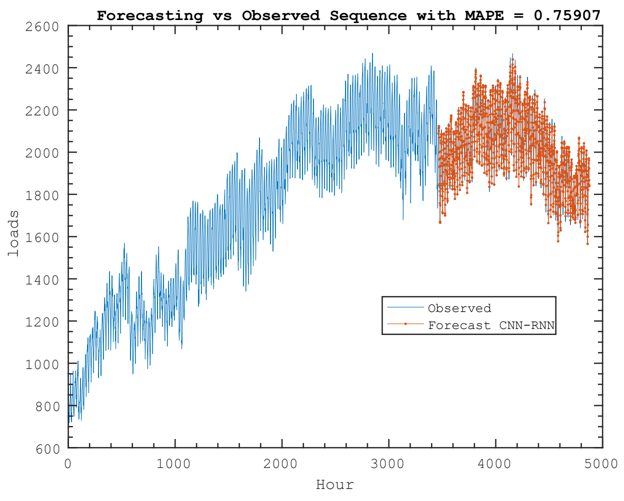
(**b**) Testing prediction performance



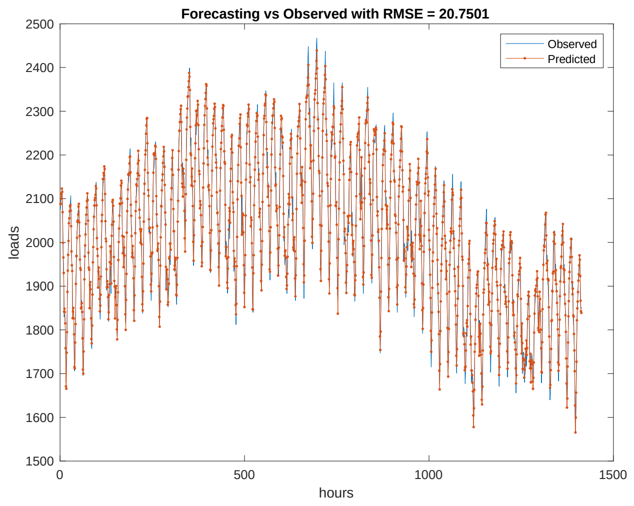
(**c**) ANN model error histogram plot

Figure 13. ANN model prediction performance for the Jeddah load: (**a**) training and testing, (**b**) testing, and (**c**) error histogram plot.

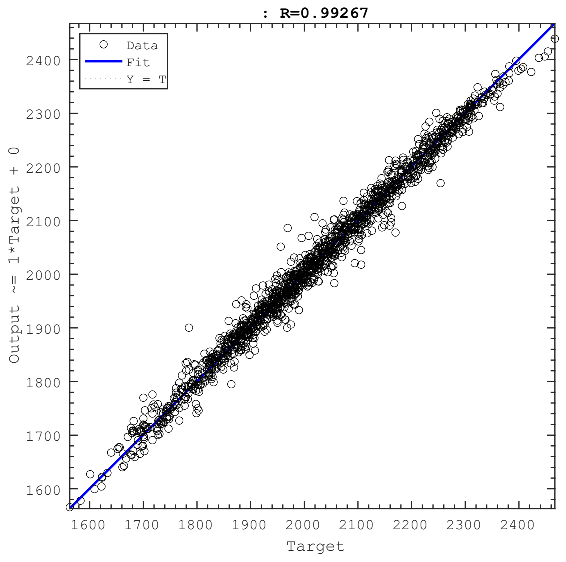
Our proposed model showed promising empirical results in terms of performance and showed that our model outperformed conventional prediction models. Hybrid DL models can effectively combine two DL models to improve the forecasting performance. The prediction accuracy improved by combining CNN model with RNN, LSTM, BiLSTM, or GRU. These short-term load-forecasting models were developed for extracting the relationships between the Jeddah and Madinah SGs. The outcomes of the proposed method could be used for power system planning.



(**a**) Training and testing prediction performance



(**b**) Testing prediction performance



(**c**) CNN–RNN regression model

Figure 14. CNN–RNN model prediction performance for the Madinah load: (**a**) training and testing, (**b**) testing, and (**c**) regression plot

6. Conclusions, Limitations, and Further Research

Recently, load forecasting has been a vital problem in the energy management and cost-effectiveness of SGs due to the increase in energy consumption globally. There have been many attempts to predict future loads efficiently using statistical models; one of those attempts was using DL methods to obtain promising prediction results. This paper proposed a hybridization of DL prediction models to improve the prediction accuracy and network architecture.

A hybrid model based on a combination of two DL algorithms among CNN, LSTM, GRU, BiLSTM, and RNN was proposed. The framework's performance in short-term electric load forecasting was compared with that of other state-of-the-art systems. The results showed that the proposed hybrid CNN–GRU- and CNN–RNN-based DL algorithms outperformed the competition when it comes to predicting how much energy will be used. The LSTM, GRU, and BiLSTM models improve the overall average RMSE, NRMSE, and MAPE of Saudi Arabia's energy-consumption prediction for both single-step and multistep forecasting. The implementation of the prediction system was applied to two real datasets from Jeddah and Madinah in Saudi Arabia. The proposed model presented better performance than the conventional DL prediction methods. The reasoning behind the hybridization concept is that, for DL algorithms, it is faster and more efficient to find the optimal forecasting accuracy and the optimal computational cost. Although the hybrid DL concept is more demanding regarding computational requirements, it notably outperformed the best conventional DL prediction models. To this end, accurate forecasting results can help solve the problem-relevant features for planning engineers and site managers. Thus, the best forecasting performance can help optimize the network scalability and requirements. The relative load-forecasting errors at the level of transmission and distribution substations result in increasingly smooth load shapes. By leveraging the data of smart meters, communication tools and sensors, enhanced load forecasting can play a major role in optimizing SG investments and grid operations. Like any time-series forecasting technique, our proposed approach, the hybrid DL method, does not guarantee optimal solutions or the most accurate prediction. Furthermore, data preprocessing and an optimized DL architecture may not be suitable for accurate prediction for other problems. Therefore, there are still some limitations to our proposed approach, e.g., the computational cost of the hybrid DL methods is considerably higher than that of conventional DL models

Since the proposed hybrid DL approach is a more complex architecture than conventional DL algorithms, utilizing meta-heuristic approaches in future work for DL architecture optimization problems can help find the optimal hyperparameters of DL models such as the number of hidden neurons, the number of hidden layers, epochs, , etc. A genetic algorithm could be a good optimization approach for hyperparameter population search coupled with the proposed approach. The forecasting outcome would be the most optimal architecture. Moreover, time-series datasets commonly consist of noise; therefore, utilizing a noise-cancellation method such as SG filter and wavelet decomposition can enhance the prediction accuracy. In addition, conducting a computation parallel to the proposed hybrid DL model can provide a real-time load-forecasting paradigm that can train historical input variables offline and can update and test recent input variables online.

Author Contributions: A.A. (Abdulaziz Almalaq) and A.A. (Abdullah Alrasheedi) designed the problem under study; A.A. (Abdullah Alrasheedi) performed the simulations and obtained the results; A.A. (Abdulaziz Almalaq) and A.A. (Abdullah Alrasheedi) analyzed the results; A.A. (Abdullah Alrasheedi) wrote the paper, which was further reviewed by A.A. (Abdulaziz Almalaq). All authors have read and agreed to the published version of the manuscript.

Funding: This research has been funded by Scientific Research Deanship at University of Ha'il – Saudi Arabia through project number GR-22 085.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available from the corresponding author upon request. The data are not publicly available due to their large size.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Kousar, S.; Zafar, N.A.; Ali, T.; Alkhammash, E.H.; Hadjouni, M. Formal Modeling of IoT-Based Distribution Management System for Smart Grids. *Sustainability* **2022**, *14*, 4499. https://doi.org/10.3390/ su14084499.
2. Ahssein, Y.H.; Amran, Y.H.; Amran, M.; Alyousef, R.; Alabduljabbar, H. Renewable and sustainable energy production in Saudi Arabia according to Saudi Vision 2030; Current status and future prospects. *J. Clean. Prod.* **2020**, *247*, 119602. https://doi.org/10.1016/j.jclepro.2019.119602.
3. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; The Mit Press: Cambridge, MA, USA, 2016.
4. Erol-Kantarci, M.; Mouftah, H.T. Wireless multimedia sensor and actor networks for the next generation power grid. *Ad Hoc Netw.* **2011**, *9*, 542–551. https://doi.org/10.1016/j.adhoc.2010.08.005.

5. Feinberg, E.; Genethliou, D. LOAD POCKET MODELING. Available online: https://www.semanticscholar.org/paper/LOAD-POCKET-MODELING-Feinberg-Genethliou/db8d19c9e66b8b8e80cbbe88400a90a53eebee32 (accessed on 5 June 2022).
6. Almalaq, A.; Edwards, G. A Review of Deep Learning Methods Applied on Load Forecasting. Available online: https://ieeexplore.ieee.org/document/8260682 (accessed on 5 June 2022).
7. Jurgenson, S.; Bayyari, F.M.; Parker, J. A comprehensive renewable energy program for Saudi Vision 2030. *Renew. Energy Focus* **2016**, *17*, 182–183. https://doi.org/10.1016/j.ref.2016.08.006.
8. Moshashai, D.; Leber, A.M.; Savage, J.D. Saudi Arabia plans for its economic future: Vision 2030, the National Transformation Plan and Saudi fiscal reform. *Br. J. Middle East. Stud.* **2018**, *47*, 381–401. https://doi.org/10.1080/13530194.2018.1500269.
9. Bedi, J.; Toshniwal, D. Deep learning framework to forecast electricity demand. *Appl. Energy* **2019**, *238*, 1312–1326. https://doi.org/10.1016/j.apenergy.2019.01.113.
10. Feinberg, E.A.; Genethliou, D. Load Forecasting. In *Power Electronics and Power Systems*; Metzler, J.B. Ed.; Springer: Boston, MA, USA, 2006; pp. 269–285.
11. Alsaedi, Y.H.; Tularam, G.A. The relationship between electricity consumption, peak load and GDP in Saudi Arabia: A VAR analysis. *Math. Comput. Simul.* **2019**, *175*, 164–178. https://doi.org/10.1016/j.matcom.2019.06.012.
12. Almalaq, A.; Zhang, J.J. *Deep Learning Application: Load Forecasting in Big Data of Smart Grids*; Springer: Cham, Switzerland, 2019; pp. 103–128. https://doi.org/10.1007/978-3-030-31760-7_4.
13. Almalaq, A.; Zhang, J.J. Evolutionary Deep Learning-Based Energy Consumption Prediction for Buildings. *IEEE Access* **2018**, *7*, 1520–1531. https://doi.org/10.1109/access.2018.2887023.
14. Sehovac, L.; Nesen, C.; Grolinger, K. Forecasting building energy consumption with deep learning: A Sequence to Sequence Approach. In Proceedings of the 2019 IEEE International Congress on Internet of Things (ICIOT), Milan, Italy, 8–13 July 2019. https://doi.org/10.1109/iciot.2019.00029.
15. Moghram, I.; Rahman, S. Analysis and Evaluation of Five Short-Term Load Forecasting Techniques. *IEEE Power Eng. Rev.* **1989**, *9*, 42–43. https://doi.org/10.1109/MPER.1989.4310383.
16. Yong, Z.; Xiu, Y.; Chen, F.; Pengfei, C.; Binchao, C.; Taijie, L. Short-term building load forecasting based on similar day selection and LSTM network. In Proceedings of the 2018 2nd IEEE Conference on Energy Internet and Energy System Integration (EI2), Beijing, China, 20–22 October 2018; pp. 1–5. https://doi.org/10.1109/ei2.2018.8582673.
17. Imani, M.; Ghassemian, H. Residential load forecasting using wavelet and collaborative representation transforms. *Appl. Energy* **2019**, *253*, 113505. https://doi.org/10.1016/j.apenergy.2019.113505.
18. Bedi, J.; Toshniwal, D. Empirical Mode Decomposition Based Deep Learning for Electricity Demand Forecasting. *IEEE Access* **2018**, *6*, 49144–49156. https://doi.org/10.1109/access.2018.2867681.
19. Hossen, T.; Nair, A.S.; Chinnathambi, R.A.; Ranganathan, P. Residential Load Forecasting Using Deep Neural Networks (DNN). In Proceedings of the 2018 North American Power Symposium (NAPS), Fargo, ND, USA, 9–11 September 2018; pp. 1–5. https://doi.org/10.1109/naps.2018.8600549.
20. Bouktif, S.; Fiaz, A.; Ouni, A.; Serhani, M.A. Optimal Deep Learning LSTM Model for Electric Load Forecasting using Feature Selection and Genetic Algorithm: Comparison with Machine Learning Approaches †. *Energies* **2018**, *11*, 1636. https://doi.org/10.3390/en11071636.
21. Kong, W.; Dong, Z.Y.; Jia, Y.; Hill, D.J.; Xu, Y.; Zhang, Y. Short-Term Residential Load Forecasting Based on LSTM Recurrent Neural Network. *IEEE Trans. Smart Grid* **2017**, *10*, 841–851. https://doi.org/10.1109/tsg.2017.2753802.
22. Veeramsetty, V.; Deshmukh, R. Electric power load forecasting on a 33/11 kV substation using artificial neural networks. *SN Appl. Sci.* **2020**, *2*, 855. https://doi.org/10.1007/s42452-020-2601-y.
23. Arvanitidis, A.I.; Bargiotas, D.; Daskalopulu, A.; Laitsos, V.M.; Tsoukalas, L.H. Enhanced Short-Term Load Forecasting Using Artificial Neural Networks. *Energies* **2021**, *14*, 7788. https://doi.org/10.3390/en14227788.
24. Son, N. Comparison of the Deep Learning Performance for Short-Term Power Load Forecasting. *Sustainability* **2021**, *13*, 12493. https://doi.org/10.3390/su132212493.
25. Hora, S.K.; Poongodan, R.; de Prado, R.P.; Wozniak, M.; Divakarachari, P.B. Long Short-Term Memory Network-Based Metaheuristic for Effective Electric Energy Consumption Prediction. *Appl. Sci.* **2021**, *11*, 11263. https://doi.org/10.3390/app112311263.
26. Ma, J.; Cheng, J.C.; Jiang, F.; Chen, W.; Wang, M.; Zhai, C. A bi-directional missing data imputation scheme based on LSTM and transfer learning for building energy data. *Energy Build.* **2020**, *216*, 109941. https://doi.org/10.1016/j.enbuild.2020.109941.
27. Zhang, Z. Improved adam optimizer for deep neural networks. In Proceedings of the 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS), Banff, AB, Canada, 4–6 June 2018. https://doi.org/10.1109/IWQoS.2018.8624183.
28. Ünal, F.; Almalaq, A.; Ekici, S. A Novel Load Forecasting Approach Based on Smart Meter Data Using Advance Preprocessing and Hybrid Deep Learning. *Appl. Sci.* **2021**, *11*, 2742. https://doi.org/10.3390/app11062742.
29. Huang, S.; Tang, J.; Dai, J.; Wang, Y. Signal Status Recognition Based on 1DCNN and Its Feature Extraction Mechanism Analysis. *Sensors* **2019**, *19*, 2018. https://doi.org/10.3390/s19092018.
30. Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv* **2014**, arXiv:1406.1078. https://arxiv.org/abs/1406.1078.
31. Hochreiter, S.; Schmidhuber, J. *Long Short Term Memory*; München Inst. Für Informatik: München, Germany, 1995.

32. Fu, R.; Zhang, Z.; Li, L. Using LSTM and GRU neural network methods for traffic flow prediction. In Proceedings of the 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC 2016), Wuhan, China, 11–13 November 2016. https://doi.org/10.1109/yac.2016.7804912.
33. Jiang, Q.; Cheng, Y.; Le, H.; Li, C.; Liu, P.X. A Stacking Learning Model Based on Multiple Similar Days for Short-Term Load Forecasting. *Mathematics* **2022**, *10*, 2446. https://doi.org/10.3390/math10142446
34. Bi, J.; Zhang, X.; Yuan, H.; Zhang, J.; Zhou, M. A Hybrid Prediction Method for Realistic Network Traffic with Temporal Convolutional Network and LSTM. *IEEE Trans. Autom. Sci. Eng.* **2021**, *19*, 1869–1879. https://doi.org/10.1109/tase.2021.3077537.
35. Sanchez, H. Time Series Forecasting Using Hybrid CNN-RNN. MATLAB Central File Exchange. 2022. Available online: https://www.mathworks.com/matlabcentral/fileexchange/91360-time-series-forecasting-using-hybrid-cnn-rnn (accessed on 20 July 2022).
36. Deep Learning Toolbox. Available online: https://in.mathworks.com/products/deep-learning.html (accessed on 22 April 2022)
37. Gers, F.A.; Schmidhuber, J.; Cummins, F. Learning to Forget: Continual Prediction with LSTM. *Neural Comput.* **2000**, *12*, 2451–2471. https://doi.org/10.1162/089976600300015015.