energies

MDPI

Article

A Hybrid Channel-Communication-Enabled CNN-LSTM Model for Electricity Load Forecasting

Faisal Saeed [1], Anand Paul [1] and Hyuncheol Seo [2,*]

[1] Department of Computer Science and Engineering, Kyungpook National University, Buk-gu, Daegu 41566, Korea; bscsfaisal821@gmail.com (F.S.); paul.editor@gmail.com (A.P.)
[2] School of Architectural, Civil, Environmental and Energy Engineering, Kyungpook National University, Daegu 41566, Korea
* Correspondence: charles@knu.ac.kr

Abstract: Smart grids provide a unique platform to the participants of energy markets to tweak their offerings based on demand-side management. Responding quickly to the needs of the market can help to improve the reliability of the system, as well as the cost of capital investments. Electric load forecasting is important because it is used to make and run decisions about the power grid. However, people use electricity in nonlinear ways, which makes the electric load profile a complicated signal. Even though there has been a lot of research done in this field, an accurate forecasting model is still needed. In this regard, this article proposed a hybrid cross-channel-communication (C3)-enabled CNN-LSTM model for accurate load forecasting which helps decision making in smart grids. The proposed model is the combination of three different models, i.e., a C3 block to enable channel communication of a CNN (convolutional neural networks) model, two convolutional layers to extract the features and an LSTM (long short-term memory network) model for forecasting. In the proposed hybrid model, Leaky ReLu (rectified linear unit) was used as activation function instead of sigmoid. The channel communication in CNN model makes the proposed model very light and efficient. Extensive experimentation was done on electricity load data. The results show the model's high efficiency. The proposed model shows 98.3% accuracy and 0.4560 MAPE error.

Keywords: cross-channel communication; Convolutional Neural Networks; LSTM; electricity; load; forecasting

Citation: Saeed, F.; Paul, A.; Seo, H. A Hybrid Channel-Communication-Enabled CNN-LSTM Model for Electricity Load Forecasting. *Energies* **2022**, *15*, 2263. https://doi.org/10.3390/en15062263

Academic Editor: Marco Pau

Received: 16 February 2022
Accepted: 17 March 2022
Published: 20 March 2022

1. Introduction

The growing understanding of the importance of modernizing the energy grid in order to enable innovative power consumption and generation patterns has shown itself in the infrastructure of the idea of smart grids [1]. Smart grids enable energy to be delivered more cheaply, sustainably, securely, and effectively by combining revolutionary concepts, models, and auxiliary services from production, transmission, and distribution to customer devices with highly sophisticated communication, sensing, and control technologies [2]. Customers can regulate their demand in response to price variations using smart grids and DSM models. DSM is described as the process of putting policies in place to control energy usage [3]. Typically, DSM identifies the numerous activities carried out by an electric utility and its customers and utilizes this information to control the amount and timing of energy usage. The reference [4] conducts an in-depth examination of DSM's role in smart grids.

Similarly, smart grid is a smart power system which has achieved huge popularity due to its capabilities of demand response, load forecasting, and load scheduling [5]. In this field, plenty of research ideas have been proposed; however, maturity is still required to ensure the accuracy of the forecasting models. When we see its application in the decision making and controlling of grids, accurate load forecasting has great importance and benefits for both utility companies and customers [6]. However, climate change, variable

temperatures, humidity, calendar indicators, occupancy patterns, and social conventions are major obstacles in electric-load forecasting. It is very challenging to achieve the appropriate mapping of these factors due to the nonlinear and stochastic behavior of users. Smart grids' deployment of communications technology, sensing methodologies, and advanced metering infrastructure allows us to monitor, record, and analyse the influence of these elements on load forecasts [7]. In the literature, classical methodologies such as time series methods and computational intelligence methods have been used for electrical load forecasting [8]. Both strategies have their drawbacks. The limitations of the previous classical approaches in dealing with nonlinear data has urged many researchers to provide a better solution. Moreover, computational intelligence methodologies are condemned for flaws such as handcrafted features, low learning capability, ineffective learning, inaccurate assessment, and inadequate guiding importance. However, there are already current machine learning modules used for load- and energy-price forecasting that partially address the aforementioned issues and have better performance owing to better design [9]. To address the aforementioned issues, a proper strategy is needed, since poor prediction accuracy results in significant economic loss. An inaccuracy increment of as little as 1% in forecasting can cause a USD 10 million increase in overall utility costs. As a result, electric companies are attempting to build a short-term electric-load forecasting models that should be quick, accurate, resilient, and easy to implement. Furthermore, precise forecasting can aid in the detection of possible problems and the operation of a dependable grid [10].

In this article, we proposed a novel hybrid cross-channel-communication (C3)-enabled CNN-LSTM model for day-ahead electricity-load prediction to make decisions on the grid side. The proposed C3-enabled CNN-LSTM model is the combination of two major parts, where three models work together. The C3-enabled CNN part works as a feature-extraction module and LSTM for forecasting the day-ahead load. We put the C3 block between two convolutional layers where the C3 block enables channel communication within a single layer. A single cross-channel-communication block has three more modules, i.e., feature encoder, message-passing using graph neural networks, and the feature decoder. Initially, the preprocessed data is fed into the first convolutional layer where it is separated into different channels. Then, these channels are sent to the C3 block where they communicate and update the feature map. Next, this updated feature map is inputted to the second convolutional layer. At last, the LSTM layers predicted the load from these extracted features of the load data. The main contributions of proposed model are manifolded as follows.

- A novel hybrid cross-channel-communication (C3)-enabled CNN-LSTM is proposed for electricity-load prediction;
- For feature extraction, a shallow convolutional neural network is proposed which has only two convolutional layers;
- C3 block is used inside the convolutional neural network to enable the channel communication which makes the network more shallow;
- PMSprop alogorithm is used to optimize the whole network;
- The proposed model is trained and tested on historical electricity-load data. The presented results show the high accuracy of the proposed model.

2. Related Work

The authors of [11] present an efficient method for rapid and precise load forecasting in the day-ahead energy market, which is critical for the proper functioning of SGs with significant demand-side flexibility. They proposed an SPLNF model that can retain linearity while also learning-from-data in LMs. They improved the overall effectiveness for faster model training by lowering the input vector dimensionality. In [12], the authors present an IoT-based deep learning system that automatically extracts characteristics from acquired data and, as a result, provides an accurate prediction of future load value. Their model is an individually constructed two-step forecasting technique, which enhances forecasting precision greatly. Additionally, the proposed model can statistically investigate the impacts

of several main attributes, which is very effective in choosing attribute patterns and deploying onboard sensors for smart grids with large territories, varying climates, and societal customs.

Ayub et al. [13] proposes SVM classifier to tackle the problem of load forecasting accuracy. The forecasting model is divided into two stages: feature engineering and SVM classification. For feature selection, a mixture of two approaches (XGBoost and DTC) is used to choose the finest features from the dataset. The SVM classifier is fine-tuned using three super factors until the desired accuracy is obtained. The SVM classifier has achieved 98% accuracy rate.

Another research study [14] offered a novel method for smart meter client load prediction by converting nonlinear smart meter data into linear system profiles. The approach's resilience was demonstrated using extremely fluctuating smart meter customer demand data. The study demonstrated the advantages of employing the suggested technique over neural networks, particularly when dealing with extremely fluctuating smart meter consumer needs. The combination of the cluster forecast provided a more precise prediction while keeping the information's variability. Usman et al. [15] proposed a modified RNN for short-term pricing and predictive modelling to forecast electricity load and price using data analytics. Data preprocessing techniques such as RFE and DTC are used to eliminate extraneous characteristics to decreases redundancy. LSTM is used to train and test the suggested model. The experimental findings demonstrate the efficacy of the suggested strategy. The analytical findings reveal that their suggested system has a lower MAPE than FFNN and RNN. The study [16] compares three different machine learning techniques on a real-world example based on the daily data from an Aarhus-based DHN (Denmark). In the analysis, support vector regression depending on the climatic parameters and calendar events outperforms other models in the 15–38 h prediction ranges. Wang et al. [17] increased the accuracy of load forecasting by presenting a novel load-forecasting system called VMD–CISSA–LSSVM. The system includes the data preparation approach variational modal decomposition (VMD), the sparrow searches algorithm (SSA), and the least squares support vector machine (LSSVM). To solve the drawbacks of the SSA method, which is susceptible to local optima and sluggish convergence, they also developed a multistrategy improved chaotic sparrow search algorithm (CISSA).

The authors of [18] proposed a fuzzy logic-based controller, which is extremely appropriate for reducing disruptions caused by variations in STLF. The challenge is designed to optimize RER utilization in order to improve the dependability of the power network. To identify any unpredictability in the power system caused by overloading and faults, an effective fuzzy control strategy is used. Their results showed that the network becomes stable in a shorter amount of time than the other methods due to the controller's quick response time to unplanned disruptions. In the suggested method in [19], researchers estimated load using accessible big data, using Apache Spark and Apache Hadoop as big data platforms for distributed computing. This study assessed the development of ML techniques utilizing Apache Spark's MLib package. According to the findings, distributed computing of load prediction delivered good precision and calculation times. Yang et al. [20] proposed a deep scalable and adaptable ensemble learning system for individualized probabilistic load forecasting. To increase uncertainty measurement efficiency, customer categorization and multitask pattern recognition were applied. The ensemble projections were refined using the LASSO-based quantile combination strategy. They also performed case studies on residential and SME clients with two forecasting horizons, showing their superiority and efficacy when compared to state-of-the-art benchmarking approaches.

3. Proposed C3-Enabled CNN-LSTM Model

This paper presents a novel hybrid cross-channel-communication-enabled CNN-LSTM model for electric-load forecasting on the grid side as displayed in Figure 1. This work presents the day-ahead electricity-load forecasting as well as minutely load forecasting. Our proposed model is a hybrid framework which a the combination of CNN and LSTM models.

From Figure 1, we can see that the model works in three phases, i.e., (i) data preprocessing where data cleaning, data normalization, irrelevancy filter, and redundancy filters are applied (ii) a very lightweight C3 block-based convolutional neural network (iii) and a long short-term memory model with PMSprop-based optimization model for accurate predictions. After selecting the dataset, the abovementioned data preprocessing techniques were applied and prepared the data to feed into the lightweight C3 block-based CNN model where the important features are selected. The main purposes of using a C3 block is to enable channel communication between the channels after each convolutional layer. In the C3 block we used an encoder, message-passing algorithm and simple decoder. The detail of the C3 block is presented in coming sections. After the feature selection from the CNN module, these features foraged to the LSTM module for better prediction where a total of four LSTM layers were used. The RMSprop algorithm is used for optimization of the model. The complete implementation procedure is discussed in detail in the following section.

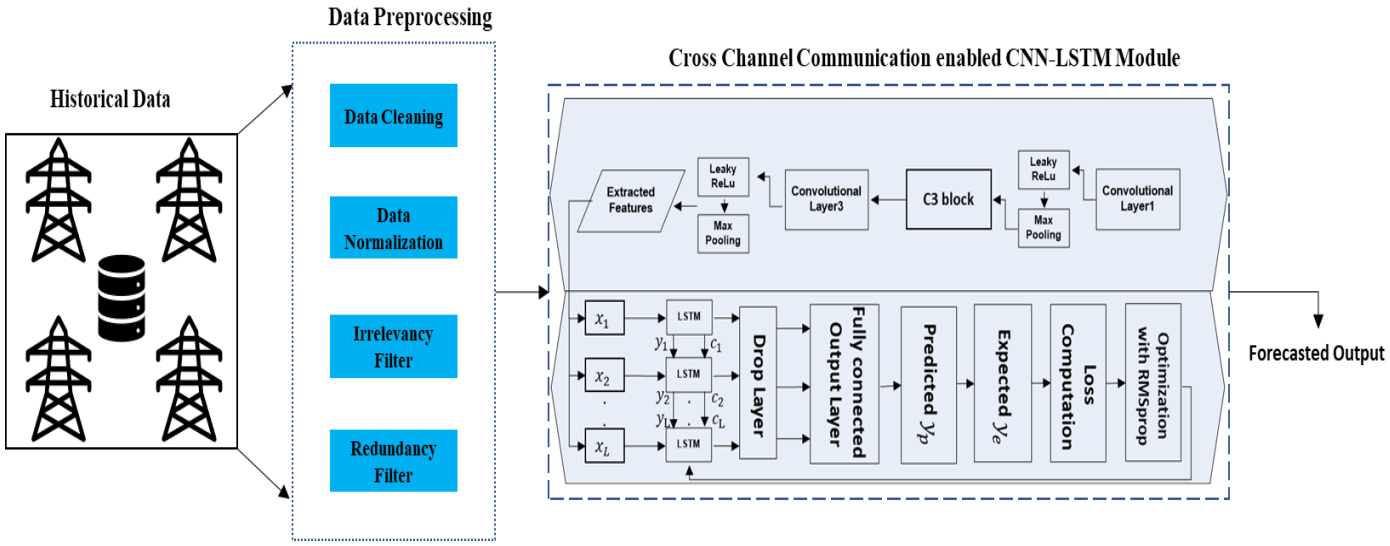

Figure 1. Pictorial view of the proposed C3-enabled CNN-LSTM Model.

3.1. Formulation of C3-Enabled Convolutional Neural Network

To avoid high computation times and memory use, we proposed a lightweight C3-enabled CNN model for feature extraction. The proposed C3-enabled convolutional neural network contains two convolutional layers along with two pooling layers and one C3 block. After the first convolutional layer, there is a maxpooling layer to normalize the output channels. We put the C3 block after the first maxpooling layer where the channels can communicate with one another. The detail of each block is described as follows.

3.1.1. Convolutional Layer

Convolutional layers conduct a complex process on the input image, and the output is passed to its following layer. At each position in the convolutional layer, there is a responsive region with a set of units from the previous levels. The neurons may acquire elementary visual properties in the immediate receptive field such as corners, endpoints, and orientated edges. This convolutional layer has numerous feature maps from which different properties can be extracted. Every unit has the same weightage and bias in every individual feature map. As a result of this, the identified properties are same for all possible input locations. This mathematical formulation is commonly used to indicate the equation of a convolutional layer:

$$X_j^I = f\left[\sum_{i \varepsilon M_j}\left[X_i^{I-1} * k_{ij}^I\right] + b_i^I\right] \quad (1)$$

where X_j^I denotes the output feature map, M_j symbolizes the number of input channels, k_{ij}^I represents the kernels and b is bias term.

3.1.2. Pooling Layer

In order to lower the complicated resolution of each feature map, this layer employs a mix of subsampling and local averaging. It also disregards output reactivity. The mathematical formulation of this layer is given below.

$$X_j^I = f\left[\beta_{jdown}^I\left(X_j^{I-1}\right) + b_i^I\right] \tag{2}$$

The *down* in the above equation represents the subsampling function. In practice, this function performs a sum over each individual block of input picture to reduce the dimensions.

3.1.3. Cross-Channel-Communication Block

We explained the C3 structure within a CNN in this part, along with associated formulations of cross-channel interaction between channels. C3 is a cross-channel-communication block which is published in [21]. We used this block to make our CNN model shallow. The network's sketch map is shown in Figure 2. Learning the time series data is a critical process in the nonlinear electricity-load forecasting process. In the proposed model, a sliding-window technique has been used to learn the features with a fixed window size, in which streams of time series data are often divided into continuous sub-sequences called windows, each of which is linked with a particular feature. We can then insert the C3 block to a few convolutional layers to enable the channels' communication. Mathematical formulation of C3 block is discussed as follows. Let us assume that a neural network has L layers, and each layer contains n_l filters. So, the feature response of the l_{th} will be $X_{l=\{x_l^1,.....x_l^n\}}$. Generally, the updated response after the channel's interaction can be calculated as:

$$\hat{x}_l^i = x_l^i + f_l^i\left(x_l^1, \ldots\ldots x_l^n\right) \tag{3}$$

where, f_l^i is a function has the functionality of collecting all the feature responses of all channels. Simultaneously, it updates the encoded features of the channels. This cross-channel communication enables communication between all sides of the network.

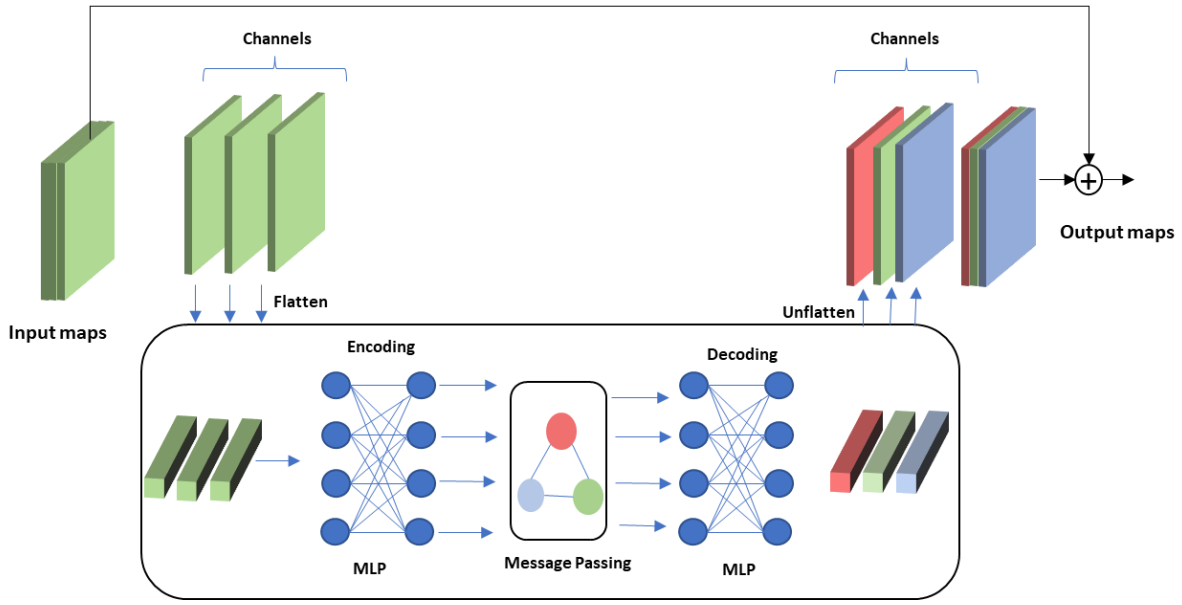

Figure 2. Working of cross-channel-communication Block.

The feature encoding, message passing, and feature decoding are the three main parts of cross-channel communication network.

Feature Encoder

This module is responsible for extracting global information from each channel response map. Particularly, the response map x_l^i as discussed earlier, is flatten into simple one-dimensional vector and then passes it two FC layers, i.e.,

$$y_l^i = f_{enc}^{in}\left(x_l^i\right),$$
$$z_l^i = f_{enc}^{out}\left(\sigma\left(y_l^i\right)\right) \tag{4}$$

There are two fully connected layers where f^{in} and f^{out} are the linear functions and σ is a ReLu activation function.

Message Passing

The message passing module is used to make sure that all channels communicate with each other so that the different feature responses can be represented in different ways by updating the final feature responses. Graph convolutional network (GCN) [22] is a good way to learn the channel interaction. Specifically, we proposed a graph attention network [23] for enabling channel interaction between load data, which has a built-in soft attention mechanism the same as GCN. Our model has the same cross-channel interaction ability as the block intension module. In our model, we construct an undirected graph where $Z = \left\{z_l^i\right\}$ are nodes and $s_{ij} = f_{att}\left(z_l^i, z_l^j\right)$ is the edge strength between two nodes. There are number of methods available to learn f_{att} [23–25] but we used the following method to learn it.

$$\overline{z}_j^i = \sum_{k=1}^{h_l w_l} \frac{z_l^i[k]}{(h_l w_l)},$$
$$s_{ij} = -\left(\overline{z}_j^i - \overline{z}_j^j\right) \tag{5}$$

where h_l and w_l represents the hight and width of a layer. $z_l^i[k]$ represents the k_{th} element of z_l^i 1-D vector. To allow more communication between the similar channels, we computed negative square distance. This way, group of similar channels were formed which becomes more harmonizing and distinct. Then the SoftMax layer normalized the attention score.

Feature Decoder

This module is responsible for obtaining the information for all repaired channels and reshaping it to the original input's dimensions. The feature decoder employs a standard convolution technique to transmit the data to the subsequent layers. After acquiring updated channel wise output z_l^i, the decoder module reshapes it to the original dimension by applying simple convolutional process. All three modules enable communication for balance across all the neurons at the same level.

3.2. Long Short-Term Memory Network

LSTM [25] is designed to be a developed version of RNN (Recurrent Neural Network), which might be useful for sorting, processing, and forecasting the time series data. A pictorial depiction of LSTM is shown in Figure 3. The purpose of developing the LSTM network is to resolve the issue of slope exploding or vanishing gradient problems, which occurs in traditional recurrent neural networks. LSTM brings value with two qualities on comparison to RNN, as discussed below:

1. Cell State c_t: In LSTM, cell state is the new state which is used to list the reliance amongst the subsequent components. Cell state c_t at time period t provides the historical information (memory);

2. Gates: This property of LSTM in the network assists to manage the distribution of information. This mechanism is comprised of three gates: the input gate i_t, the forget gate f_t, and the output gate O_t;
3. The said gates in LSTM helps to restrict the quantity of information flows. The value is expressed between 0 and 1, where the value 0 refers that no transmission of information is authorized, and the value 1 means that total communication of information is accomplished.

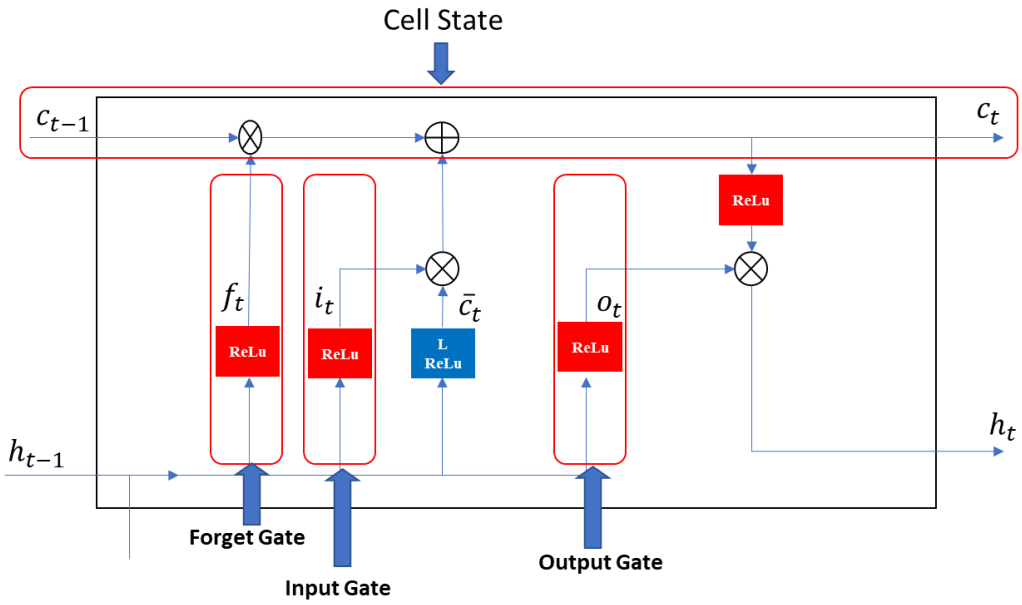

Figure 3. Pictorial description of proposed LSTM cell.

In the proposed LSTM model, despite the conventional LSTM network, we used ReLu and Leaky ReLu [26] activation function instead of traditional sigmoid and hyper tanh functions as shown in Figure 3. As we described earlier, learning the nonlinear behavior of load data is a little challenging for activation functions such as sigmoid and tanh because of their low output limit. To overcome this challenge, we implemented the LSTM model which uses *ReLu* as activation function as shown in the function. The mathematical form of all the cells of LSTM is shown in following equations.

$$f_t = ReLu\left(w_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{6}$$

$$i_t = ReLu(w_i \cdot [h_{t-1}, x_t] + b_t) \tag{7}$$

$$\widetilde{c_t} = \text{L.}ReLu(w_c \cdot [h_{t-1}, x_t] + b_c) \tag{8}$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \widetilde{c_t} \tag{9}$$

$$O_t = ReLu(w_o \cdot [h_{t-1}, x_t] + b_t) \tag{10}$$

$$h_t = \text{L.}Relu(c_t) \tag{11}$$

where f_t, i_t, c_t, O_t, and h_t are representing the forget gate, input gate, cell state, output gate, and hidden state. The proposed approach is divided into four primary sections: preparing the data, training the LSTM system, verifying the system, forecasting the load, and calculating the value or cost based on the testing data. The processes for cost prediction are detailed in the following phases. For the first phase, the historical price and load vectors are normalized using the following computation.

$$p_n = \frac{p - \mu(p)}{\sigma(p)} \tag{12}$$

where μ and σ denotes mean and standard deviation respectively. In the equation (p_n), Mean and Std are used to calculate the standard deviation of the standardized load data. Z-score normalization is the term used to describe this process. The data is separated in hourly manner for our convenience. Algorithm 1 is used to divide the data into training, validating, and testing parts. The system is then trained using the training dataset and validated using the validating dataset in the subsequent phase. A trained neural network is tested using a dataset including anticipated data of load for a day ahead. Root mean square error (RMSE) estimate is used to test the model's efficiency.

Algorithm 1: #This algorithm separated the electricity load dataset into training, validation and testing sets

Input: Electricity load dataset
Output: 65% Training, 15%Validation and 10% Testing sets
1. Data_Size ← Data_length (time-series) × 0.65
2. Data for Training ← time-series [0 . . . Data_Size]
3. X ← length (time-series) × 0.1
4. Validation-Data ← time-series (Data-Size . . . X)
5. Testing-Data← time-series(X . . . length(Data-Size) + length(X))
6. Return Train-Data, Validation-Data, Testing Data

4. Implementation Detail

The electricity datasets of the energy load were taken from different data sources, e.g., the power consumption dataset was from Independent System Operator New England (ISO NE) [27] and New York Independent System Operator (NYISO) [28]. ISO NE controls the creation and distribution system for New England. ISO NE yields and spreads nearly 30,000 MW electrical energy every day. At ISO NE, per annum USD 10 million of business is accomplished by a total of 400 electrical consumers in the market. The facts consist of ISO NE zone's limits of system load per hour and adjusting capacity clearance value of 21 states in USA for the past 8 years that is starting from January 2011 to March 2018. The dataset shows about 63,224 estimations. New York Independent System Operator is a nonprofit establishment that works with an American city's electricity grid and is in charge of an entire state's comprehensive energy markets. The evidence collected from New York Independent System Operator comprises the hourly utilization and value in the city. It contains 13 years' worth of data which is from January 2006 to October 2018 and has a total of 112,300 estimations.

To train the model, we used the minibatch method. The minibatch approach divides the data into many batches and updates the variable for each batch individually. Minibatch avoids the massive number of finds produced by the traditional training strategy of criss-crossing the whole data variable. We must perform gradient steps for all training sets as a single batch in batch gradient descent. In contrast to batch gradient descent, minibatch gradient descent allows a dataset to be split into many little datasets, such as one batch of data into many small vectors of data called minibatches. The training datasets are trundled synchronously between X and Y using the minibatch gradient descent technique. This shuffle ensures that samples are divided into tiny batches at random. The shuffled batch is then divided into several smaller batches. Each micro batch is usually a power of two in size (64, 128, 256, 512, 1024, etc.). The minibatch approach infuses adequate chaos to each gradient update while obtaining relative rapid convergence, because minibatch updates weights on each minibatch gradient. Adam optimizer is used to avoid this disadvantage. The Adam algorithm is not to be confused with the traditional conditional gradient descent algorithm. The classic gradient descent technique maintains a single iteration rate while updating all weights. Throughout the training, the learning rate remains constant. The Adam algorithm calculates the gradient's first instant approximation and second raw instant approximation. For various variables, the instant approximation is built as an independent adaptive learning rate, which may be changed throughout the training process.

Our proposed prediction C3-enabled CNN-LSTM model is implemented in Python using keras and TensorFlow libraries. The model is trained on a system with the specifications described as Intel Core i5-3570 CPU @ 3.40 GHz 3.80 GHz, NVIDIA GeForce GTX1070 GPU, and our operating system was Windows 10, 64-bit. After the normalization of the load dataset, it is fed into C3-enabled CNN model for feature extraction. Leaky ReLu function is used as an activation function every after convolutional layer. Minibatch technique with Stochastic Gradient Descent (SGD) was used where the momentum value is 0.001 to train the model. A total of 4 LSTM blocks were used for predicting the final output. Dropout layers were also used to reduce the factor of overfitting the model. Initially the learning rate was of 0.001 but it decreased by 1/10 every after 30,000, 60,000, 60,000, and 30,000 iterations. All the hyperparameters used in the proposed network are summarized in Table 1.

Table 1. Used hyperparameters.

No.	Hyperparameter's Name	Hyperparameter's Value
1	Learning Rate	0.001
2	Step Decay Rate	1/10
3	Momentum	0.6
4	Dropout in hidden layers	0.5
5	Dropout in input layer	0.8
6	Epoch	50

5. Results and Discussion

As per previous discussion, the proposed model is trained on ISO NE and NYISO datasets. In total, 65% of the data was used for training, 15% of the data for validating the model, and the remaining 20% of the data was used to test the model. Both datasets have records of many grids, but we chose only four grids' data for prediction. The actual data is visualized in Figure 4 for four grids. As discussed earlier, the model is trained in Python with leaky ReLu as an activation function and the step-decay algorithm as a learning function. Initially, the test data is passed to the data preprocessing part to prepare for the model where different filters such as data cleaning, data normalization, irrelevance filter, and redundancy filter were applied. Then, the C3-enabled CNN model was trained for feature extraction purposes.

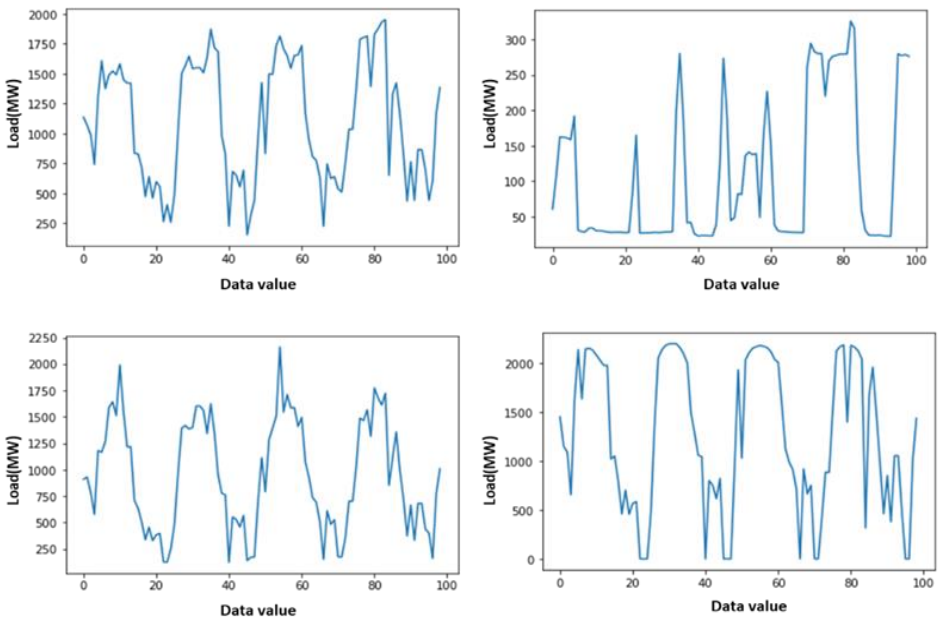

Figure 4. Actual data values of four different cells.

Then, the four-block LSTM model was trained on extracted features. To check the model efficiency, we visualized the learning curves for four different runs. The model performance can be examined over several epochs on training and testing data using a learning curve. It can be said after looking at the learning curve that the model is picking up new information from the data or simply memorizing it. The high error rate in training and testing and the fast convergence because of the high learning rate and bias results the learning curve being skewed, and the model does not learn from its errors. Similarly, when the gap between training and testing errors is high, the high variance develops. In both ways, the model has problem and results in inaccurate generalization. When the test error increases while the training error decreases, this phenomenon is called overfitting. This demonstrates that the model is memorizing, but not learning. Consequently, in these situations it is impossible to generalize from the model. After applying dropout method and early termination of learning can avoid overfitting. For the proposed model, however, the testing/validation error gradually diminishes alongside the training error for the electricity grids as shown in Figure 5. Our model handled overfitting issues quite well.

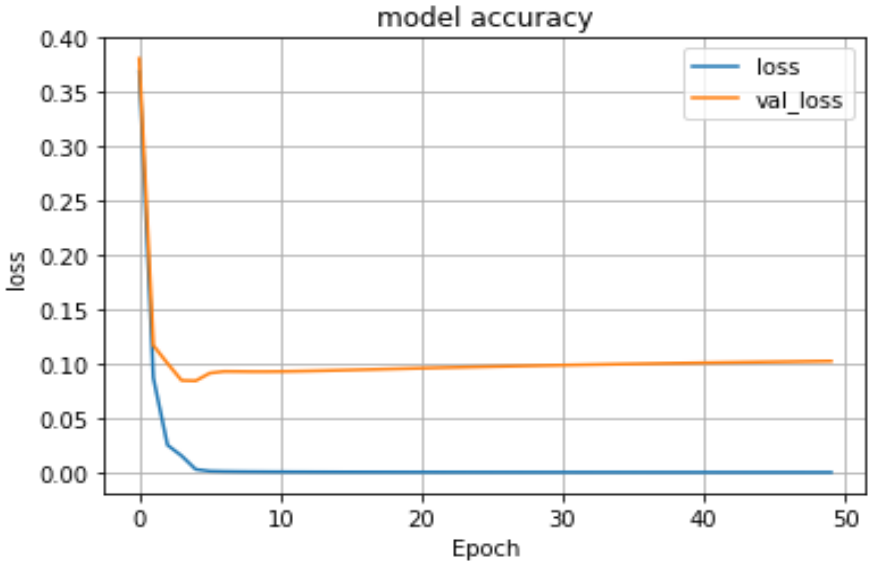

Figure 5. C3-enabled CNN LSTM model loss.

Table 2 presents the first three epochs of each four runs: their time of execution, loss, and accuracy. Meanwhile, while testing the C3-enabled CNN-LSTM model, the model achieved 98.3% accuracy, as shown in Figure 6, while Figure 7 shows the ROC curve of the model. We depicted the predicted loads of four grids' data in Figure 8a–d for the four grids, respectively, which shows the complete load forecasting of our model. From the figure, it can be seen that the model is performing better and efficiently. In Figure 8, blue lines represent the actual value of the load, the yellow line shows the prediction on training data, and green lines show load forecasting. It is noticeable from the presented graphs that the proposed C3-enabled CNN-LSTM model can capture nonlinear behavior from the past data and, on this learned behavior, it can forecast the load very efficiently.

Table 3 provides the comparison of the proposed model with existing models in terms of MAPE. We showed this table for one grid station. This table lists the numerical findings of benchmark models such as LSTM [29], CNN-LSTM [30], Bi-LSTM [31], and our proposed model. It also shows the day-ahead forecasted load based on the proposed model. Our model has a MAPE error of 0.4560%, while the Bi-LSTM model has a MAPE error of 2.5397%, the CNN-LSTM model has a MAPE error of 2.3123%, and the LSTM model has a MAPE error of 4.3664%. When compared to state-of-the-art models, our proposed model shows lower MAPE, which means the proposed model has more accurate results. In terms of accuracy, CNN-LSTM-projected load forecasting outperforms LSTM, while Bi-LSTM outperforms CNN-LSTM. The CNN-LSTM model employed RMSprop for optimization,

but the Bi-LSTM model utilized DEA, which improves prediction accuracy by decreasing error. At the expense of greater execution time, this higher precision is achieved. Due to the inclusion of the C3-based CNN model for feature selection and RMSprop-based optimization module in LSTM framework, the proposed C3-enabled CNN-LSTM model outperforms Bi-LSTM, CNN-LSTM, and LSTM models. Table 1 shows the statistical results of our model with state-of the art models for a single power-grid station in terms of MAPE. We conclude that the proposed C3-enabled CNN-LSTM outperforms benchmark models based on the findings and discussion. In terms of MAPE, the average numerical findings for a power grid are 0.4560% which are lower than the benchmark models.

Table 2. Training time, loss, and accuracy of first three epochs of each run.

Epochs	Training Time	Loss	Accuracy
	First Run		
1/50	2 s	0.3183	0.6817
2/50	0.03 s	0.2153	1.7847
3/50	0.05 s	0.1411	0.8511
	Second Run		
1/50	1.5 s	0.4785	0.5215
2/50	0.8 s	0.2673	0.7327
3/50	0.5 s	0.1588	0.8412
	Third Run		
1/50	1.3 s	0.2541	0.7459
2/50	0.02	0.1770	0.8230
3/50	0.05 s	0.1228	0.8772
	Fourth Run		
1/50	1.1 S	0.4677	0.5323
2/50	0.004 s	0.3508	0.6492
3/50	0.002 s	0.2603	0.7397

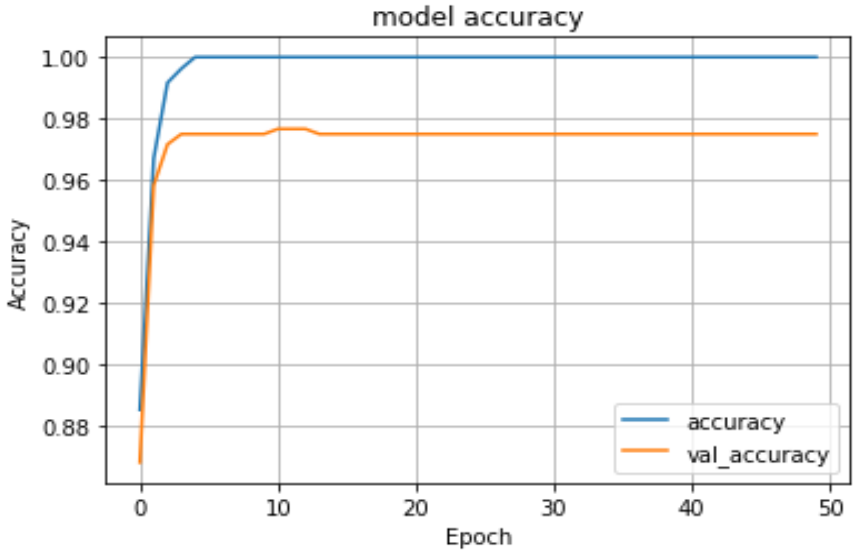

Figure 6. C3-enabled CNN LSTM model accuracy.

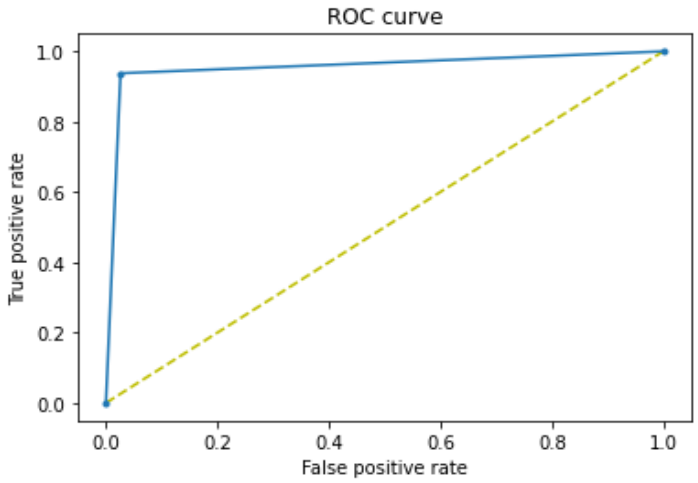

Figure 7. ROC curve of proposed model.

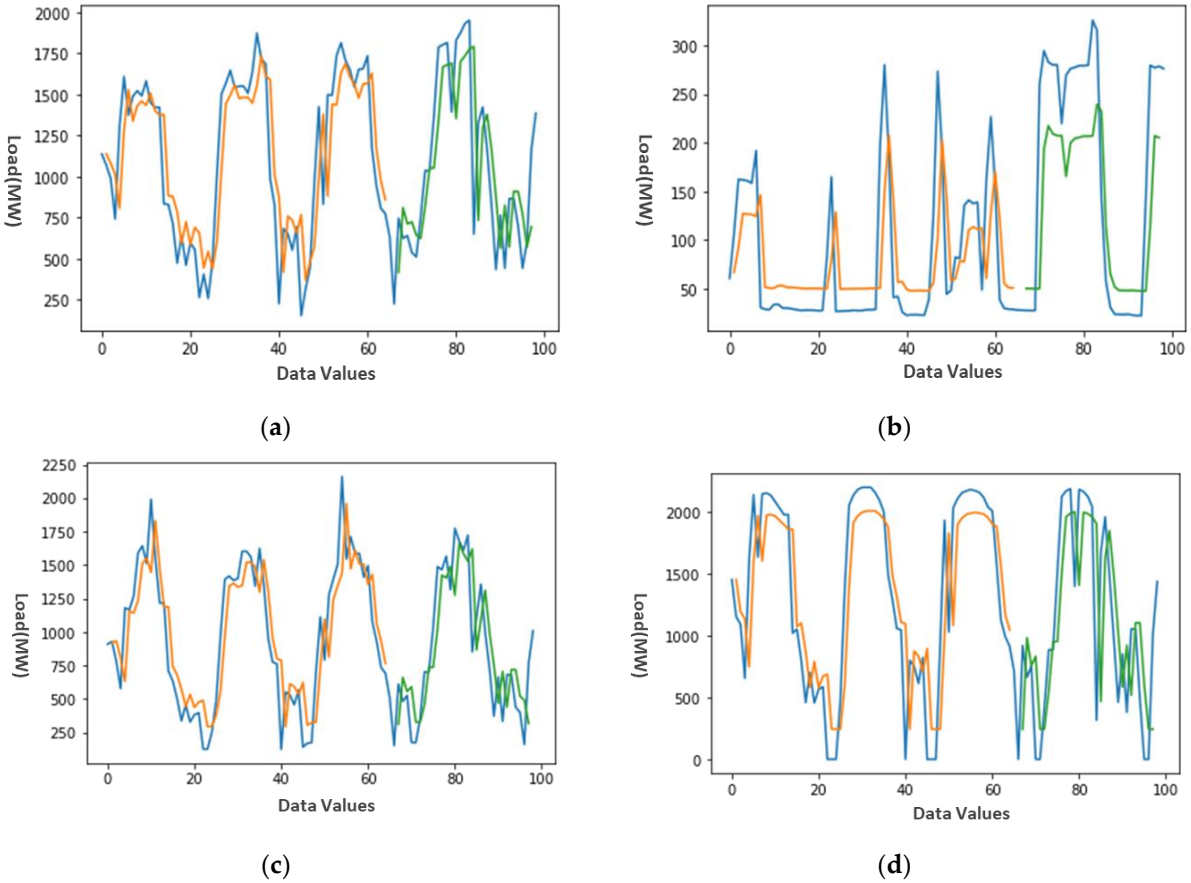

Figure 8. (**a**–**d**) Load forecasting results of proposed C3-CNN-LSTM model on four different grids.

The comparison of the proposed model with respect to time execution with benchmark models is depicted in Figure 9. Sometimes the accuracy of the Bi-LSTM model increased because of the DEA optimization algorithm, but it came at the cost of a higher execution time due to its greedy nature. From Figure 8, it is obvious our model becomes more accurate with lower execution times. The reasons behind the low execution time is the C3 block, which lessens the convolutional layers to only three. There are two main reasons for the low execution time of the C3-enabled CNN-LSTM model, i.e., cross-channel-communication block which enabled the channel communication within layer and the use of the ReLu activation function instead of sigmoid in the LSTM model.

Table 3. Comparison of proposed model with other state of the art algorithms using mean absolute percentage error.

Hours	Actual Load	Proposed Model		Bi-LSTM		LSTM		CNN-LSTM	
		P. Load	MAPE	P. Load	MAPE	P. Load	MAPE	P. Load	MAPE
1	1035	1042	0.4513	1054	2.5186	1012	4.3371	1044	2.4741
2	1370	1374	0.4851	1377	2.5874	1387	4.3251	1376	2.6514
3	1785	1789	0.4752	1798	2.6584	1742	4.3587	1787	2.2541
4	1801	1807	0.4852	1835	2.3658	1826	4.3574	1847	2.3547
5	1813	1820	0.4125	1875	2.1245	1885	4.2541	1880	2.1254
6	1392	1398	0.4456	1421	2.3548	1432	4.6985	1428	2.1458
7	1828	1832	0.4712	1842	2.5847	1852	4.3521	1845	2.6514
8	1874	1877	0.4951	1898	2.6954	1907	4.3887	1903	2.3521
9	1930	1935	0.4325	1965	2.6587	1978	4.3002	1962	2.2514
10	1950	1965	0.4125	1986	2.6874	1995	4.3698	1972	2.1245
11	650	670	0.4215	685	2.3587	695	4.8541	675	2.2154
12	1326	1333	0.4562	1375	2.3658	1384	4.8854	1338	2.5484
13	1421	1430	0.4754	1463	2.1478	1473	4.2514	1435	2.9542
14	1163	1170	0.4751	1187	2.3658	1198	4.3652	1176	2.1124
15	822	832	0.4124	871	2.3214	885	4.3665	840	2.0254
16	434	440	0.4214	487	2.5846	497	4.2154	448	2.1245
17	764	772	0.4239	792	2.3648	801	4.0035	784	2.3652
18	442	450	0.4732	482	2.8487	475	4.2514	455	2.4412
19	865	873	0.4859	890	2.9547	892	4.3251	874	2.3215
20	698	703	0.4854	742	2.6687	748	4.3257	712	2.0024
21	442	451	0.4125	495	2.8745	502	4.3223	455	2.1024
22	601	613	0.4815	654	2.5474	665	4.6587	614	2.1143
23	1167	1178	0.4583	1402	2.8412	1408	4.0021	1184	2.1325
24	1384	1492	0.4954	1528	2.4752	1540	4.2254	1499	2.6512
Average			0.4560		2.5397		4.3664		2.3123

We have made a scalability analysis in Figure 10. This analysis allows us to make assumptions whether the proposed C3-enabled CNN-LSTM model is scalable for the huge dataset or in other said scenarios. We changed the input sample, bias of the model, changed some weights and tried some different features and then analysed the model performance. In the scenario where we changed the weights of the model, but the input remained constant, the proposed model was not affected. Figure 9 shows the impact of these factors on the execution time of the models. We compared our model execution time with other benchmark models in this scenario. This analysis shows, even in the said scenario, that our model outperforms and shows a lower execution time because of the inclusion of the C3 block. Figure 11 shows the comparison of the poposed model with other hybrid models in terms of MAPE error. From the figure we can observe that MAPE error rate of proposed model is lower than WTNNEA [32], WGMIPSO [33] and another hybrid model [34]. This result demonstrates that the proposed model outperforms these hybrid models.

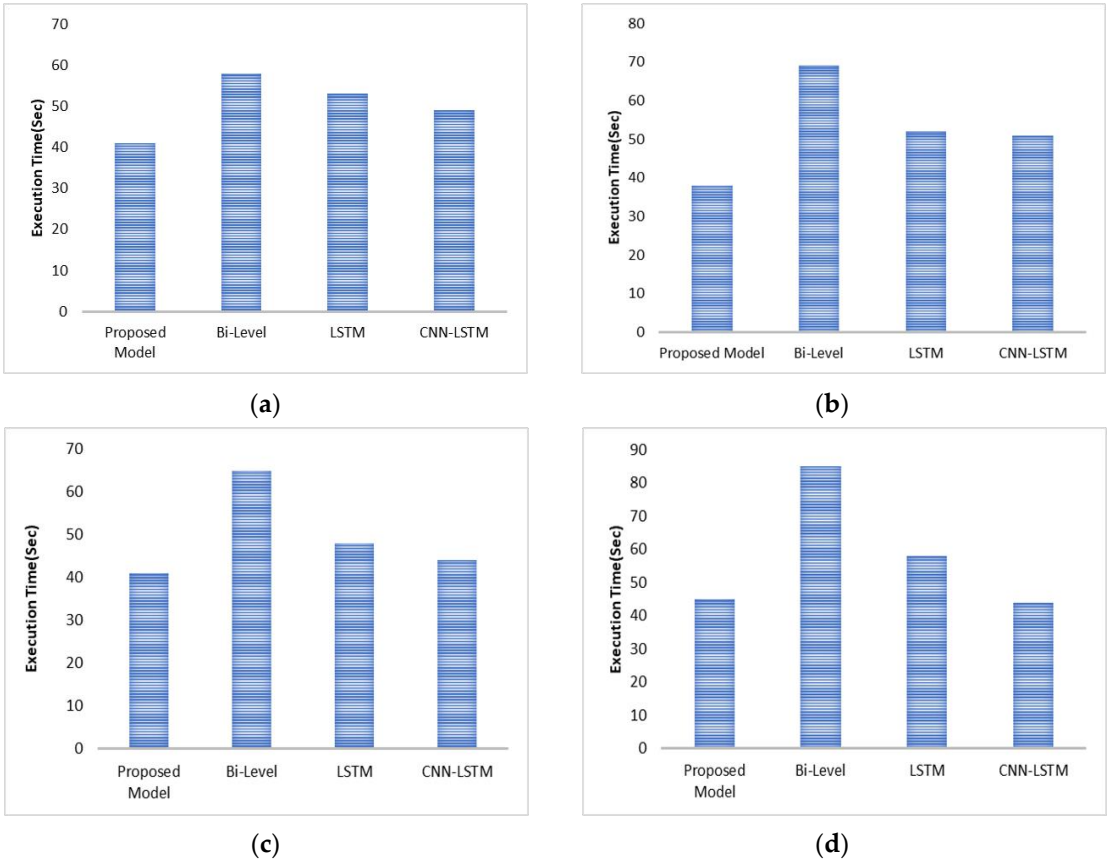

Figure 9. Comparative analysis of proposed model with respect to time execution. (**a**–**d**) shows the execution time of four different test results respectively for four grids.

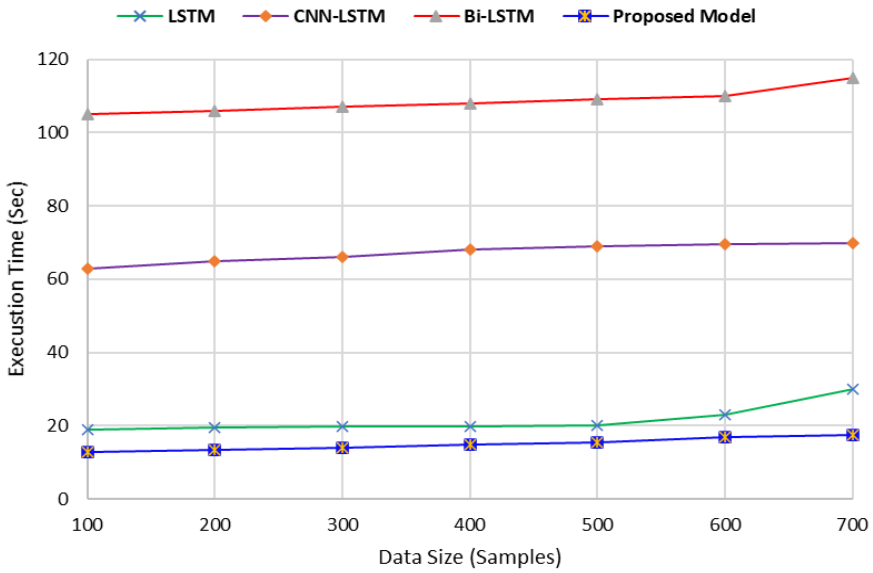

Figure 10. Scalability analysis of our model with other state-of-the-art models.

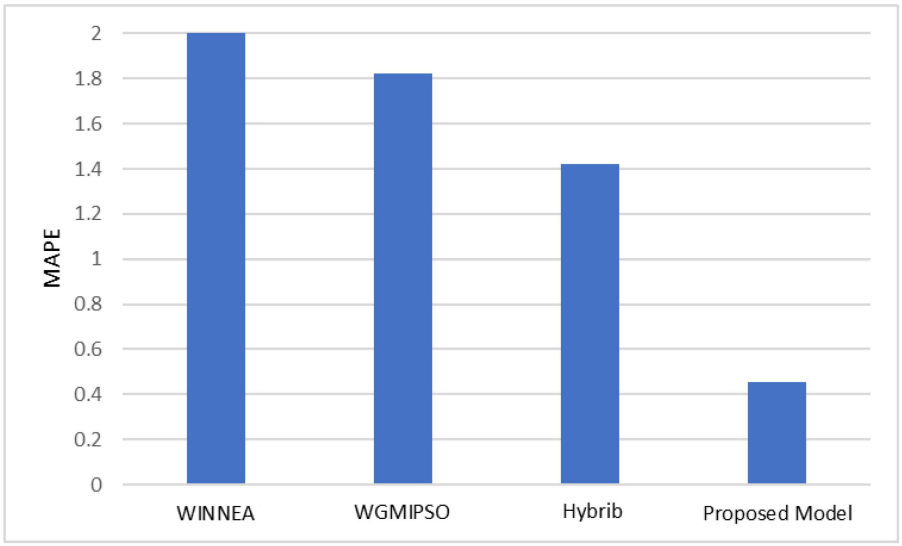

Figure 11. Comparison of proposed model with other hybrid models in terms of MAPE. WTN-NEA [32], WGMIPSO [33] and an other hybrid model [34].

6. Conclusions

Accurate electric-load forecasting is critical for decision making and system functioning in electricity power grids. With efficient forecasting of load demand, operators may create an ideal market strategy to maximize the economic benefits of energy management. In this manuscript, a hybrid C3-enabled CNN-LSTM model for load forecasting is proposed. The proposed model contains three parts, i.e., convolutional layers, a C3 block and LSTM layers. The convolutional layers and C3 block worked to extract the important features from the load data and LSTM layers were used to predict the load. Two different datasets of electricity load were used, named as NYISO and ISO NE. In the model, ReLu functions were used as activation functions. The presented experiments show that the proposed model gained 98.3% accuracy in prediction. The proposed model is compared with other state-of-the-art methods, i.e., LSTM, CNN-LSTM, and Bi-LSTM based on MAPE and execution time. The proposed model showed a 0.4560% error rate while LSTM showed 4.3664%, CNN-LSTM showed 2.3123%, and Bi-LSTM showed 2.5397%. As the proposed model used a C3 block inside the CNN network, making the model shallow, the execution time of the proposed model is comparatively less than other benchmark models.

Author Contributions: F.S. performed conceptualization, prepare the methodologies, performed the experiments and validation of the model while A.P. prepared the first draft, completed the writing process and carried out formal analysis. H.S. supervised the work and with provided all the resources. All authors have read and agreed to the published version of the manuscript.

Funding: This article received no external funding than NRF.

Acknowledgments: This research is supported by the National Research Foundation of Korea. Grant funded by Korean Government (MSIP, South Korea) Number: 2020R1C1C1007127).

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

Nomenclature

BP	Back Propagation
DR	Demand Response
LSTM	Long short-term memory
MAPE	Mean absolute percentage error
RMSE	Root mean square error
SVM	Support vector machine
ARIMA	Auto-regressive integrated moving average
BPNN	BP neural network
ELM	Extreme learning machine
HEMS	Home energy management system
RNN	Recurrent neural networks
WNN	Wavelet neural network
ReLu	Rectified Linear Unit
SG	Smart grid

References

1. Colak, I.; Fulli, G.; Sagiroglu, S.; Yesilbudak, M.; Covrig, C.F. Smart grid projects in Europe: Current status, maturity and future scenarios. *Appl. Energy* **2015**, *152*, 58–70. [CrossRef]
2. Jian, L.; Zheng, Y.; Xiao, X.; Chan, C.C. Optimal scheduling for vehicle-to-grid operation with stochastic connection of plug-in electric vehicles to smart grid. *Appl. Energy* **2015**, *146*, 150–161. [CrossRef]
3. Yu, M.; Hong, S.H. Supply–demand balancing for power management in smart grid: A Stackelberg game approach. *Appl. Energy* **2016**, *164*, 702–710. [CrossRef]
4. Siano, P. Demand response and smart grids—A survey. *Renew. Sustain. Energy Rev.* **2014**, *30*, 461–478. [CrossRef]
5. IEEE Xplore. Energy Efficient Integration of Renewable Energy Sources in the Smart Grid for Demand Side Management. *IEEE J. Mag.* Available online: https://ieeexplore.ieee.org/abstract/document/8443332 (accessed on 15 February 2022).
6. Xiao, L.; Shao, W.; Wang, C.; Zhang, K.; Lu, H. Research and application of a hybrid model based on multi-objective optimization for electrical load forecasting. *Appl. Energy* **2016**, *180*, 213–233. [CrossRef]
7. Alahakoon, D.; Yu, X. Smart Electricity Meter Data Intelligence for Future Energy Systems: A Survey. *IEEE Trans. Ind. Inform.* **2016**, *12*, 425–436. [CrossRef]
8. Hernandez, L.; Baladron, C.; Aguiar, J.M.; Carro, B.; Sanchez-Esguevillas, A.J.; Lloret, J.; Massana, J. A survey on electric power demand forecasting: Future trends in smart grids, microgrids and smart buildings. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 1460–1495. [CrossRef]
9. Rahman, A.; Srikumar, V.; Smith, A.D. Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks. *Appl. Energy* **2018**, *212*, 372–385. [CrossRef]
10. Saeed, F.; Paul, A.; Ahmed, M.J.; Gul, M.J.J.; Hong, W.H.; Seo, H. Intelligent implementation of residential demand response using multiagent system and deep neural networks. *Concurr. Comput. Pract. Exp.* **2021**, *33*, e6168. [CrossRef]
11. Tavassoli-Hojati, Z.; Ghaderi, S.F.; Iranmanesh, H.; Hilber, P.; Shayesteh, E. A self-partitioning local neuro fuzzy model for short-term load forecasting in smart grids. *Energy* **2020**, *199*, 117514. [CrossRef]
12. Li, L.; Ota, K.; Dong, M. When weather matters: IoT-based electrical load forecasting for smart grid. *IEEE Commun. Mag.* **2017**, *55*, 46–51. [CrossRef]
13. Ayub, N.; Javaid, N.; Mujeeb, S.; Zahid, M.; Khan, W.Z.; Khattak, M.U. Electricity Load Forecasting in Smart Grids Using Support Vector Machine. *Adv. Intell. Syst. Comput.* **2019**, *926*, 1–13.
14. Khan, Z.A.; Jayaweera, D. Approach for forecasting smart customer demand with significant energy demand variability. In Proceedings of the IEEE International Conference on Power, Energy & Smart Grid (ICPESG), Mirpur Azad Kashmir, Pakistan, 9–10 April 2018; pp. 1–5. [CrossRef]
15. Usman, M.; Ali Khan, Z.; Khan, I.U.; Javaid, S.; Javaid, N. Data Analytics for Short Term Price and Load Forecasting in Smart Grids using Enhanced Recurrent Neural Network. In Proceedings of the Emerging Technologies Blockchain and IoT: ITT 2019—Information Technology Trends, Ras Al Khaimah, United Arab Emirates, 20–21 November 2019; pp. 84–88. [CrossRef]
16. Dahl, M.; Brun, A.; Kirsebom, O.S.; Andresen, G.B. Improving Short-Term Heat Load Forecasts with Calendar and Holiday Data. *Energies* **2018**, *11*, 1678. [CrossRef]
17. Wang, G.; Wang, X.; Wang, Z.; Ma, C.; Song, Z. A VMD–CISSA–LSSVM Based Electricity Load Forecasting Model. *Mathematics* **2021**, *10*, 28. [CrossRef]
18. Ali, M.; Adnan, M.; Tariq, M. Optimum control strategies for short term load forecasting in smart grids. *Int. J. Electr. Power Energy Syst.* **2019**, *113*, 792–806. [CrossRef]
19. Syed, D.; Refaat, S.S.; Abu-Rub, H. Performance evaluation of distributed machine learning for load forecasting in smart grids. In Proceedings of the 2020 Cybernetics & Informatics (K&I), Velke Karlovice, Czech Republic, 29 January–1 February 2020. [CrossRef]

20. Yang, Y.; Hong, W.; Li, S. Deep ensemble learning based probabilistic load forecasting in smart grids. *Energy* **2019**, *189*, 116324. [CrossRef]
21. Yang, J.; Ren, Z.; Gan, C.; Zhu, H.; Parikh, D. Cross-channel Communication Networks. *Adv. Neural Inf. Process. Syst.* **2019**, *32*.
22. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017. Available online: https://arxiv.org/abs/1609.02907v4 (accessed on 15 February 2022).
23. Veličković, P.; Casanova, A.; Liò, P.; Cucurull, G.; Romero, A.; Bengio, Y. Graph Attention Networks. In Proceedings of the 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 April–3 May 2018. Available online: https://arxiv.org/abs/1710.10903v3 (accessed on 15 February 2022).
24. Yang, J.; Lu, J.; Lee, S.; Batra, D.; Parikh, D. Graph R-CNN for Scene Graph Generation. *ECCV* **2018**, 670–685. Available online: https://openaccess.thecvf.com/content_ECCV_2018/papers/Jianwei_Yang_Graph_R-CNN_for_ECCV_2018_paper.pdf (accessed on 15 February 2022).
25. Wang, X.; Girshick, R.; Gupta, A.; He, K. Non-Local Neural Networks. *ECCV* **2018**, 7794–7803. Available online: https://openaccess.thecvf.com/content_cvpr_2018/papers/Wang_Non-Local_Neural_Networks_CVPR_2018_paper.pdf (accessed on 15 February 2022).
26. Xu, B.; Wang, N.; Kong, H.; Chen, T.; Li, M. Empirical Evaluation of Rectified Activations in Convolutional Network. Available online: https://arxiv.org/abs/1505.00853v2 (accessed on 15 February 2022).
27. ISO_NE_Network Electricity Markit Data. Available online: https://www.iso-ne.com/isoexpress/web/reports/pricing (accessed on 23 January 2022).
28. NYISO. NYISO Market Opration Data. 2019. Available online: http://www.nyiso.com/public/markets_operations/market_data/custom_report (accessed on 23 January 2022).
29. 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* **2019**, *10*, 841–851. [CrossRef]
30. Alhussein, M.; Aurangzeb, K.; Haider, S.I. Hybrid CNN-LSTM Model for Short-Term Individual Household Load Forecasting. *IEEE Access* **2020**, *8*, 180544–180557. [CrossRef]
31. Gul, M.J.; Urfa, G.M.; Paul, A.; Moon, J.; Rho, S.; Hwang, E. Mid-term electricity load prediction using CNN and Bi-LSTM. *J. Supercomput.* **2021**, *77*, 10942–10958. [CrossRef]
32. Amjady, N.; Keynia, F. Short-term load forecasting of power systems by combination of wavelet transform and neuro-evolutionary algorithm. *Energy* **2009**, *34*, 46–57. [CrossRef]
33. Li, S.; Goel, L.; Wang, P. An ensemble approach for short-term load forecasting by extreme learning machine. *Appl. Energy* **2016**, *170*, 22–29. [CrossRef]
34. Zhang, J.; Wei, Y.M.; Li, D.; Tan, Z.; Zhou, J. Short term electricity load forecasting using a hybrid model. *Energy* **2018**, *158*, 774–781. [CrossRef]