2023 5th International Conference on Energy, Power and Environment: Towards Flexible Green Energy Technologies (ICEPE) | 979-8-3503-1312-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICEPE57949.2023.10201537

Bidirectional LSTM Network-Based Short-Term Load Forecasting Method in Smart Grids

Alok Kumar
Dept. of Electrical Engineering
NIT Warangal
Warangal, India
akee21219@student.nitw.ac.in

Mahamad Nabab Alam
Dept. of Electrical Engineering
NIT Warangal
Warangal, India
mnalam@nitw.ac.in

Abstract— **Load forecasting, including classic time series analysis and more contemporary machine learning techniques, has emerged as one of the most prominent research domains. The primary emphasis of research in this field lies in predicting aggregated power usage. However, the significance of demand-side energy management, encompassing individual load forecasts, is increasingly gaining prominence. This work proposes load forecasting models that rely on deep neural networks (DNNs). These models are applied to a demand-side load database for analysis. The forecasting accuracy of DNN-based load forecasting models is assessed by comparing them with Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Bidirectional Long-Short-Term Memory (B-LSTM) models. The B-LSTM is a new recurrent artificial neural network recommended as the forecasting unit due to its ability to process information from both the past and present hidden layer using memory loops. Performance of the algorithm is check based on the mean absolute error, mean absolute percentage error, and root mean square error.**

***Keywords*—load forecasting, neural network, feature selection, machine learning, CNN, LSTM, B-LSTM.**

I. Introduction

Short-term energy load forecasting (STLF) is a method for anticipating short-term electricity demand, often for a few hours to a few days. With the increased demand for renewable energy sources and the requirement to balance their sporadic nature with conventional energy sources, STLF has grown significantly. Because of the highly variable and dynamic nature of energy demand—influenced by various factors, including weather, holidays, economic activity, and social events—STLF is a problematic issue. Complex mathematical models and data analysis methods must be used to anticipate power consumption accurately.

Deep learning's main benefit is its capacity to automatically learn features from unprocessed data without the requirement for manually engineered feature engineering. By using a precise forecasting technique, electrical energy-producing systems can operate more steadily and dependably [1]. When introducing Convolutional Neural Networks (CNN), a novel approach is adopted by applying the market wavelet function as an activation function instead of the traditional options. Nevertheless, due to a shortage of memory units and the incidence of gradient vanishing, this specific arrangement performs poorly in predicting issues [2].

Hence, the need for more understanding regarding deep learning approaches for predicting multi-step loads in commercial buildings. To achieve this, two conventional deep neural network models, the recurrent neural network (RNN) and the CNN, have been proposed and designed using recursive and direct multi-step techniques [3]. The deep neural network's latest version is the LSTM model that uses memory cells to store data about previous inputs to address the vanishing gradient issue that standard recurrent neural networks have, making it challenging for the network to learn long-term dependencies between inputs. By enabling information to be added to or withdrawn from memory cells selectively, LSTMs were created to address this problem and enable the network to retain information for extended periods [4]. Successful LSTM network-based applications have been described in numerous fields, including speech recognition, image captioning, and natural language translation [5-7]. The suggested framework is put to the test using a set of actual household smart meter data that is publicly available, and the results are thoroughly compared to several benchmarks, such as cutting-edge load forecasting [8].

With the processing of the input sequence both forward and backwards in time, B-LSTM improve the basic LSTM architecture. By analysing the input sequence in both ways, B-LSTM models can capture bidirectional dependencies and increase prediction accuracy. Predicting future data multiple steps ahead, the B-LSTM emerges as a suitable approach. It utilises the forecasted outputs as input data for subsequent steps, thus maintaining a robust memory capable of accurately storing essential past and future features [9].

This paper proposes B-LSTM deep learning based short-term load forecasting for the smart grid applications.

II. Proposed Bi-directional LSTM Algorithm

A. Convolution Neural Network (CNN)

In CNN, the convolutional layer and pooling layer of a CNN can collect useful characteristics from the original input in order

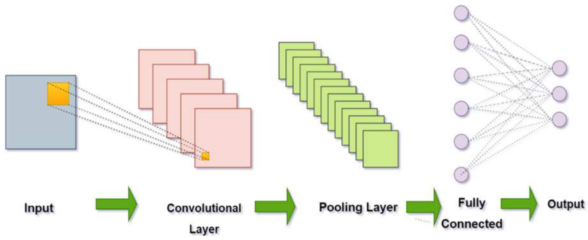

Fig. 1. Block diagram of CNN.

to extract the local properties of the data in an automated manner. Moreover, CNN can create comprehensive and dense feature vectors using the retrieved features. Fig. 1 shows the block diagram of CNN as mentioned.

In the convolutional layer, filters are used to extract important information from the input image, creating a feature map through element-wise multiplication and summation. The pooling layer reduces the spatial dimensionality of the feature maps and selects vital data using techniques such as max pooling or average pooling. The activation function layer adds non-linearity to the model by applying a function to each neuron's output, commonly using ReLU, sigmoid, or tanh. The fully-connected layer establishes connections between neurons in different layers to generate the final output. Lastly, the dropout layer randomly removes neurons to prevent overfitting.

B. Long and short-term memory (LSTM)

LSTM networks have emerged as a viable solution for solving time series forecasting challenges, such as load forecasting, in recent years. LSTM networks, which are a type of RNNs, effectively identify long-term dependencies in time series data by incorporating memory cells and gating mechanisms. The introduction of forget gate, input gate, and output gate functions in LSTM networks has significantly enhanced their suitability for modelling time series data with long-term dependencies. These functions effectively address the issues of gradient explosion and gradient disappearance that commonly arise during RNN training.

Historical load data can be utilised as inputs to the network in order to apply LSTM networks for load forecasting. A supervised learning method can be used to train the network. In this method, the network is given a sequence of historical loads and asked to forecast the loads for a future time period. Many research has shown how well LSTM networks do load forecasting using LSTM networks to anticipate load in a power system with a significant concentration of renewable energy sources. Their findings demonstrated that the LSTM-based model performed better in terms of forecasting accuracy than conventional time series forecasting models. Fig. 2 show block diagram of the LSTM model.

Below, you can find the equations representing the different LSTM components.

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{1}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{2}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{3}$$

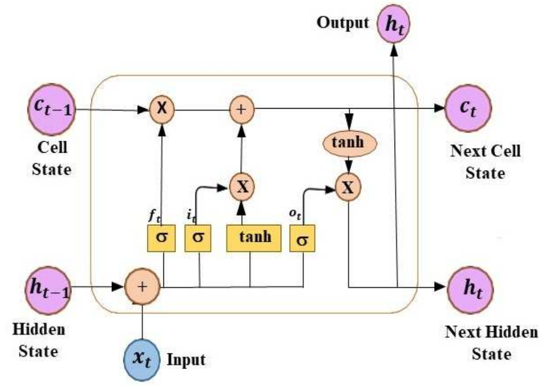

Fig. 2. Block diagram of LSTM.

$$\tilde{c}_t = tanh\,(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{4}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{5}$$

$$h_t = o_t * tanh\,(c_t) \tag{6}$$

Among them, i_t, f_t, o_t represents for t-th input, forget, and output gate. The weight values associated with the input, forget, output gate, and memory unit are symbolized by W_{xi}, W_{xf}, W_{xo}, W_{xc} . W_{hi}, W_{hf}, W_{ho}, W_{hc} are the notations used for the weight connections between the hidden and the input, forget, output gate, and memory unit, respectively. The terms for bias of each gate function are represented by b_i, b_f, b_o, b_c . The activation functions utilized in the LSTM network are $\sigma\ (\cdot)$ sigmoid function for gating and *Tanh* hyperbolic tangent function for memory units. The point-by-point multiplication operation is denoted by the symbol *.

C. Bi-Directional Long and short-term memory (B-LSTM)

Bi-directional Long Short-Term Memory (B-LSTM) is a type of RNN that is frequently employed in natural language processing (NLP) tasks [10]. The traditional LSTM model, which can handle sequential data and maintain long-term dependencies, is expanded by the B-LSTM model. The performance of NLP tasks, including sentiment analysis, machine translation, and speech recognition, is enhanced using B-LSTM networks [11-12]. To perform better, B-LSTM networks take advantage of the bidirectional nature of LSTM networks. The B-LSTM network comprises two LSTM networks, where one input sequence is in a forward direction and the backward direction. The final output of the B-LSTM network is obtained by combining or concatenating the outputs from each individual B-LSTM network.

Fig. 3 illustrates the block diagram representing the B-LSTM network. In the B-LSTM network, the forward layer at

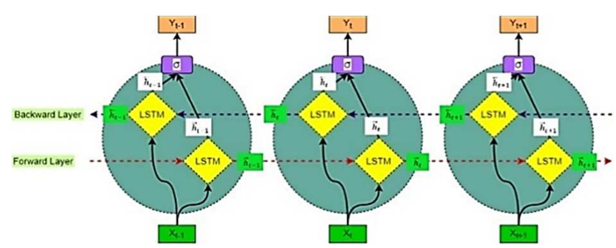

Fig. 3. A bidirectional long short-term memory unit.

time step t for the incoming data sequence X_t and the previous hidden state value $\overrightarrow{h_{t-1}}$. Using this information, the current hidden state value $\overrightarrow{h_t}$ is computed.

Next, the internal equation shown below is applied to update and process the hidden state value.

$$\overrightarrow{f_t} = \sigma\left(\overrightarrow{W_f}\left[\vec{h}_{t-1}, \vec{X}_t\right] + \vec{b}_f\right) \tag{7}$$

$$\overrightarrow{i_t} = \sigma\left(\overrightarrow{W_i}\left[\vec{h}_{t-1}, \vec{X}_t\right] + \vec{b}_i\right) \tag{8}$$

$$\overrightarrow{o_t} = \sigma\left(\overrightarrow{W_o}\left[\vec{h}_{t-1}, \vec{X}_t\right] + \vec{b}_o\right) \tag{9}$$

$$\vec{\tilde{c}}_t = \tanh\left(\overrightarrow{W_c}\left[\vec{h}_{t-1}, \vec{X}_t\right] + \vec{b}_c\right) \tag{10}$$

$$\overrightarrow{c_t} = \overrightarrow{f_t} * \vec{c}_{t-1} * \overrightarrow{i_t} * \overrightarrow{\tilde{c}_t} \tag{11}$$

$$\overrightarrow{h_t} = \overrightarrow{o_t} * \tanh(\overrightarrow{c_t}) \tag{12}$$

To summarize, this the output hidden layer of the forward layer LSTM at time step t can be expressed as follows.

$$\overrightarrow{h_t} = f\left(\vec{X}_t,\ \vec{h}_{t-1},\ \vec{\square}\mathbf{LSTM}\right) \tag{13}$$

In this context, the symbol $\vec{\square}$**LSTM** represents the internal operation of the current state forward LSTM unit. Additionally, the hidden state value $\overleftarrow{h_t}$ of the backward layer in the B-LSTM network is updated by considering the present input data $\vec{X}_t$ and the future hidden state value $\overleftarrow{h}_{t-1}$. Following are all internal updates that occur inside a backward layer:

$$\overleftarrow{f_t} = \sigma\left(\overleftarrow{W_f}\left[\overleftarrow{h}_{t+1}, \overleftarrow{X_t}\right] + \overleftarrow{b_f}\right) \tag{14}$$

$$\overleftarrow{i_t} = \sigma\left(\overleftarrow{W_i}\left[\overleftarrow{h}_{t+1}, \overleftarrow{X_t}\right] + \overleftarrow{b_i}\right) \tag{15}$$

$$\overleftarrow{o_t} = \sigma\left(\overleftarrow{W_o}\left[\overleftarrow{h}_{t+1}, \overleftarrow{X_t}\right] + \overleftarrow{b_o}\right) \tag{16}$$

$$\overleftarrow{\tilde{c}_t} = \tanh\left(\overleftarrow{W_c}\left[\overleftarrow{h}_{t+1}, \overleftarrow{X_t}\right] + \overleftarrow{b_c}\right) \tag{17}$$

$$\overleftarrow{c_t} = \overleftarrow{f_t} * \overleftarrow{c}_{t-1} * \overleftarrow{i_t} * \overleftarrow{\tilde{c}_t} \tag{18}$$

$$\overleftarrow{h}_t = \overleftarrow{o_t} * \tanh(\overleftarrow{c_t}) \tag{19}$$

The backward LSTM layer's output equation for the hidden state can be summed up as follows:

$$\overleftarrow{h}_t = f\left(\overleftarrow{X_t},\ \overleftarrow{h}_{t+1},\ \overleftarrow{\square}\,\mathbf{LSTM}\right) \tag{20}$$

In this context, the notation $\overleftarrow{\square}$**LSTM** signifies the collective internal operation of the backward LSTM layer. By applying an activation function, the hidden states of both the forward and backward layers in a B-LSTM network are combined to produce the output of the hidden state. Consequently, the influence of both previous and future data can be observed in the output of a B-LSTM network.

III. The Load Forecasting Model's Overall Technique Consists of the Following Steps:

1) **Data Collection:** Gathering historical load data is crucial for training the model.
2) **Data Pre-processing and Feature Extraction:** It needs to pre-process the acquired data and extract useful features from it. To do this, the data must be cleaned, normalised, and formatted into a time series.
3) **Model Selection:** The next step is to choose an acceptable model for load forecasting after the data has been pre-processed and the features have been retrieved. For load forecasting, models like CNN, LSTM, and Bi-directional LSTM are frequently utilised. The type of data and amount of precision needed will determine which model is used.
4) **Model Training:** After the model has been chosen, the trained model will be applied to the pre-processed data. During the training phase, the parameters are iteratively adjusted to minimize the discrepancy between the predicted and actual load. This adjustment process takes place by feeding the pre-processed data into the model.
5) **Hyperparameter Tuning:** Adjusting the hyperparameters of the model is necessary to increase the model's accuracy after it has been trained.
6) **Model Evaluation:** After the hyperparameters have been fine-tuned, the model must be assessed on the test set. Several evaluation metrics are commonly employed to assess the performance of the model.
7) **Model Deployment:** Lastly, the model can be used for load forecasting after it has been trained and assessed.

IV. Data Preprocessing and Feature Extraction

Building precise and effective machine learning models, such as CNN, LSTM, and Bi-directional LSTM for electrical load forecasting, requires the pre-processing of data and feature extraction. The procedures for feature extraction and data pre-processing are as follows:

1) **Data Cleaning:** The first step in cleaning the data is to eliminate any inconsistencies or missing information. Making sure the model can learn from reliable and consistent data is crucial.
2) **Data normalisation**: It scaled the data to ensure that all variables have the same range. This enhances the model's functionality and strengthens its stability.
3) **Time Series Data Formatting:** Electrical load forecasting is a time series problem, hence the data must be organised in a way that makes sense for time series analysis. This entails placing the information in a certain time-based chronological arrangement (hourly, daily, weekly, etc.).
4) **Feature Extraction:** After the data has been prepared, pertinent features must be extracted from the data. Among the often-utilised elements for predicting electrical load are:
   - Whether it is a workday or a weekend.
   - The time of day: dawn, noon, dusk, or night
   - Data on the load from the previous week, month, or year

5) **Feature Scaling:** Following feature extraction, the features must be scaled to have the same range across the board. This is crucial to make sure the model can benefit evenly from each feature.
6) **Data Splitting:** The data must be divided into training, validation, and test sets before being processed.

The data can be entered into the CNN, LSTM, or Bi-directional LSTM model for electrical load forecasting when the feature extraction and data pre-treatment processes are finished.

Algorithm 1: Algorithm for B-LSTM

```
1: Initialization
   def initialize weights randomly():
2: Training
   def train(num_epochs, training_dataset, learning_rate):
   for epoch in range(num_epochs):
       for input_sequence, target_output in training_dataset:
           # Forward pass
         forward_hidden_state=initialize_forward_hidden_state()
   backward_hidden_state=initialize_backward_hidden_state()
           forward_outputs = []
           backward_outputs = []
    # Forward pass of the forward LSTM layer
    for input_timestep in input_sequence:
     forward_hidden_state=forward_lstm_cell(input_timestep,
     forward_hidden_state)
    forward_output=get_output_from_hidden_state
     (forward_hidden_state)
    forward_outputs.append(forward_output)
    # Forward pass of the backward LSTM layer
    reversed_input_sequence = reversed(input_sequence)
    for reversed_input_timestep in reversed_input_sequence:
     backward_hidden_state=backward_lstm_cell(reversed_inp
     ut_timestep, backward_hidden_state)
    backward_output=get_output_from_hidden_state
    (backward_hidden_state)
            backward_outputs.append(backward_output)
3: Concatenate the forward and backward outputs
    concatenated_outputs = torch.cat((forward_outputs,
     backward_outputs), dim=1)
4: Calculate loss and update weights (Backward pass)
    loss = calculate_loss(concatenated_outputs, target_output)
           gradients = compute_gradients(loss)
           update_weights(gradients, learning_rate)
```

V. CASE STUDY

A. Data Description

Data from the Canadian province of Ontario are used in this study as a case study [13]. According to linking several actors in the power market, the load data in Ontario exhibits a significant degree of dynamic behaviour. Ten years of load demand data are gathered, from January 1, 2013, to December 30, 2022, at hourly intervals.

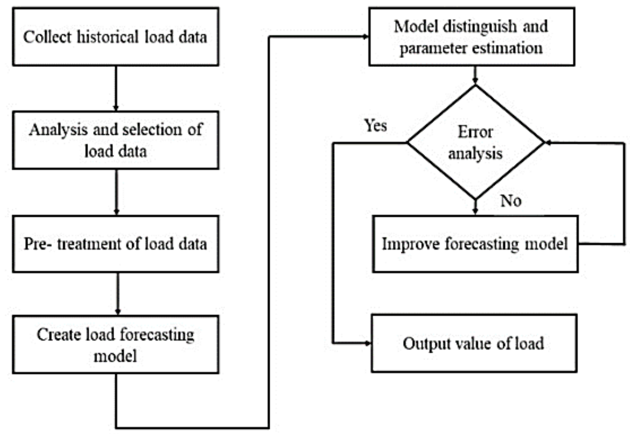

Fig. 4. Detailed flowchart of the suggested approach.

The ten years of load demand data gathered in the Ontario study at hourly intervals can be used to train and validate load forecasting models. By analysing historical data, load forecasting models can identify patterns and trends in energy demand, which can help predict future demand with a high degree of accuracy.

VI. EXPERIMENTAL RESULT AND ANALYSIS

Three noteworthy error criteria—the mean absolute error (e_{mae}), the mean absolute percentage error (e_{mape}), and the root mean square error (e_{rmse}) are taken into account to assess how well the various approaches perform.

A. Evaluating Accuracy Measures for Predictive Analysis

The mean absolute error (e_{mae}) (21), the mean absolute percentage error (e_{mape}) and the relative mean square error (e_{rmse}) are utilised as evaluation indices. e_{mape} stands for the model error (22). It is utilized as a means to evaluate the accuracy of the model. The volatility of the model error is shown by the symbol e_{rmse} (23). It is used to see the stability and robustness of the model. The formulas for e_{mape} and e_{rmse} are in equations (22) and (23).

$$e_{mae} = \frac{1}{n_0} \sum_{g=1}^{n_0} \left(\left| \hat{Y}_g - Y_g \right| \right) \tag{21}$$

$$e_{mape} = \frac{1}{n_0} \sum_{g=1}^{n_0} \left(\left| \frac{\hat{Y}_g - Y_g}{\hat{Y}_{g_{mean}}} \right| \times 100 \right) \tag{22}$$

$$e_{rmse} = \sqrt{\frac{1}{n_0} \sum_{g=1}^{n_0} \left(\left| \hat{Y}_g - Y_g \right| \right)^2} \tag{23}$$

B. Short-Term Forecasting Results

The initial phase in this case study is data pre-processing, which includes numerous activities like filtering outliers and resolving missing values as well as scaling the data to make sure that all characteristics are on a similar scale. The data may need to be split into a time-lagged format after being cleaned

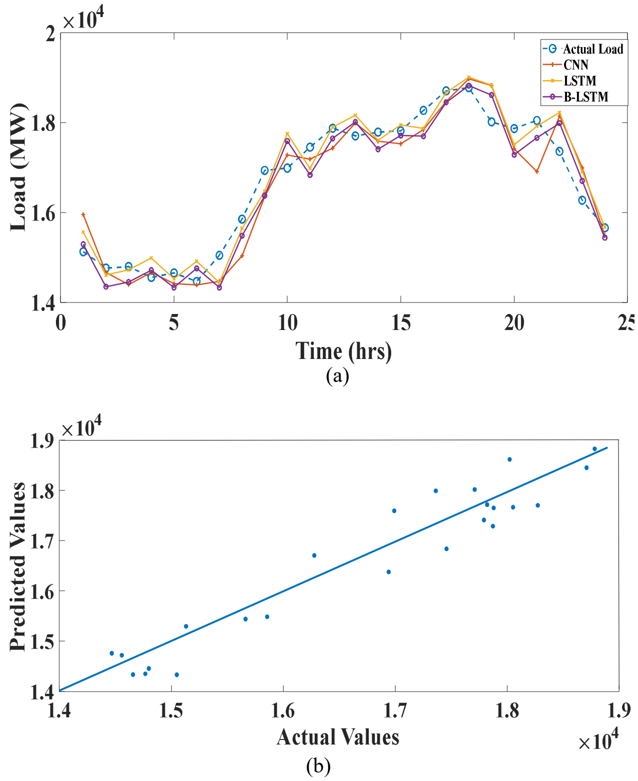

Fig. 5. Forecasting results on December 31, 2022. (a) Load demand profiles. (b) Regression plots for 24 hrs.

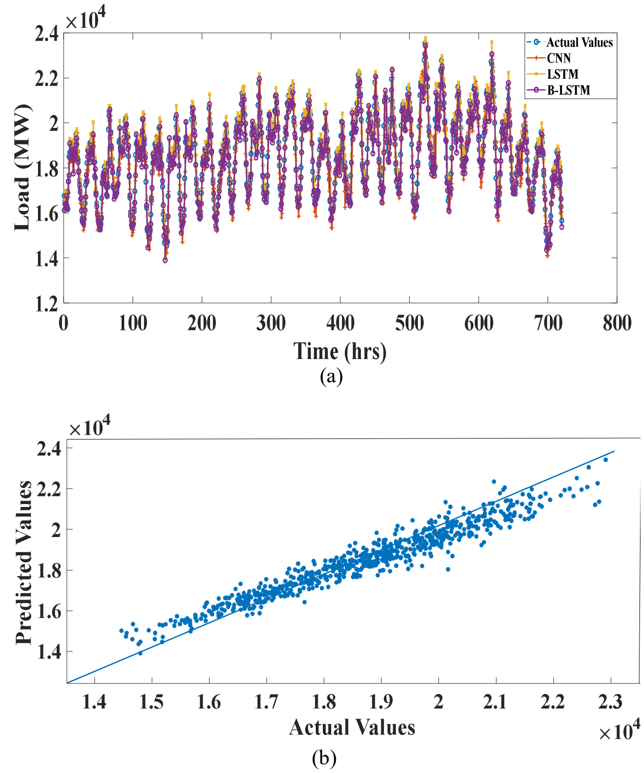

Fig. 6. Forecasting results on December, 2022. (a) Load demand profiles (b) Regression plots for December month.

and converted, which entails developing features and targets that may be utilised for modelling and forecasting.

With univariate time series data, this phase is particularly crucial since it enables us to examine trends and patterns over time and forecast future values. In every data analysis project, pre-processing the data is a crucial stage since it helps to guarantee that the data are correct, trustworthy, and appropriate for analysis. Fig. (4) presents a flowchart illustrating the proposed method.

To build our model, we used the Python programming language together with the libraries Keras, TensorFlow, NumPy, and PyTorch. For both the LSTM and B-LSTM models, a portion of the learning set was aside for validation split. The data was divided, and a separate validation set was created. The model was learned using the "adam" optimizer and the "mae" loss function. By increasing the number of epochs, a more complicated forecasting model can be developed, improving the accuracy of the results. However, as the epoch increases, the problem of model overfitting becomes more critical.

The task involves using B-LSTM networks for forecasting, and these networks are trained with specific datasets. Each LSTM network has 20 hidden layers, which have been found to provide good performance. It is crucial to emphasize that the B-LSTM network exhibits higher sensitivity to the learning rate values than the LSTM network, which can cause saturation during the training process. To avoid this, the LSTM network's initial learning rate is set to 0.01, while 0.005 for the B-LSTM network. In simpler terms, the B-LSTM network is trained using specific datasets, has 15 hidden layers in each LSTM network, and requires careful adjustment to avoid saturation during training.

In order to confirm the robustness of the suggested strategy, a thorough comparison of several methodologies over various time periods (daily and monthly samples) was conducted in December (Tables I and II), is considered. After the completion of training and testing using the complete ten years of data, Fig.5(a) depicts the effectiveness of all models and is presented by comparing their outputs with the actual data and its associated values regression plot Fig.5(b) for December 31, 2012, which serves as an example day. The peak around the 31st day was selected as one of the days in December month, and Upon analysis, it was found that the CNN and LSTM models predicted the lowest and second-lowest peaks, respectively. Conversely, the B-LSTM models exhibited peaks that closely aligned with the actual data. The load profile has a straightforward trend without any significant spikes, as shown by the predicting findings.

Fig. 6. displays the anticipated load demand and its accompanying regression plot for December. The load profile has a straightforward trend without any significant spikes, as shown by the predicting findings. While forecasting is similar to regression, In order to further validate the reliability of the case study, a widely recognized criterion is taken to measures the relation between projected and actual data. This criterion is illustrated in Fig.6(b) to provide additional evidence of the

study's robustness. Thus, load demand analysis has been carried out in order to compare the models overall, as shown in Table I and Table II. Nevertheless, deep neural network topologies have demonstrated superior performance in terms of RMSE and MAPE. Even though, Table I & II numerical study demonstrates that the B-LSTM error is very small (<2% MAPE). The B-LSTM demonstrates a remarkable level of accuracy in predicting results, this indicates that the model was successful in capturing and learning from the input data, leading to accurate and dependable predictions. The low MAPE value observed is evidence of the Bi-directional LSTM architecture's success in capturing both forward and backward temporal dependencies. This promising finding suggests that the Bi-directional LSTM model may be applied to similar forecasting applications.

TABLE I. FORECASTING RESULT ON DECEMBER 31, 2022

Algorithm	Load Demand		
	MAPE (%)	**MAE (MW)**	**RMSE (MW)**
CNN	2.73	457.79	532.39
LSTM	1.96	366	410.5
B-LSTM	**1.42**	**340.63**	**396.34**

TABLE II. FORECASTING RESULT ON DECEMBER, 2022

Algorithm	Load Demand		
	MAPE (%)	**MAE (MW)**	**RMSE (MW)**
CNN	2.88	544.43	711.49
LSTM	2.2	460.52	562.83
B-LSTM	**1.78**	**378.23**	**453.72**

VII. CONCLUSION

This work describes implementing an STLF method for the Ontario case study using the B-LSTM. Using a specialized forecasting network for those sites, this B- LSTM technique includes detecting specific points with different fluctuation rates. The suggested strategy performs better than existing forecasting techniques. The method is shown to be both robust and accurate. It is compared to benchmark algorithms to demonstrate its superiority in terms of accuracy, complexity, training time, and ease of use. We used the deep learning algorithms LSTM and B-LSTM for the load forecasting job, with B-LSTM performing better than LSTM with a 98.22% accuracy rate. This is achieved using sequence networks, which behave like memory elements to retain information from past time steps and incorporate it into future predictions. In other words, the sequence network enables the model to consider the temporal dynamics of the data, allowing it to make more accurate and robust forecasts. The B-LSTM network has been proposed as a dependable approach for forecasting various time-series task-based findings of this work, especially when working with data that exhibits significant stochasticity and abrupt swings. Due to its financial and technical benefits, accurate forecasting is crucial in present-day power systems. With comprehensive knowledge, the distribution network operator may make more intelligent choices with fewer risks and errors.

REFERENCES

[1] H. Jahangir, H. Tayarani, S. S. Gougheri, M. A. Golkar, A. Ahmadian and A. Elkamel, "Deep Learning-Based Forecasting Approach in Smart Grids With Microclustering and Bidirectional LSTM Network", *IEEE Transactions on Industrial Electronics*, vol. 68, no. 9, pp. 8298-8309, 2021.

[2] M. A. Zamee, D. Han and D. Won, "Online Hour-Ahead Load Forecasting Using Appropriate Time-Delay Neural Network Based on Multiple Correlation–Multicollinearity Analysis in IoT Energy Network", IEEE Internet of Things Journal, vol. 9, no. 14, pp. 12041-12055, 2022.

[3] Mengmeng Cai, Manisa Pipattanasomporn, Saifur Rahman, "Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques", Applied Energy, vol 236, pp. 1078-1088, 2018.

[4] N. P. Singh, A. R. Joshi and M. N. Alam, "Short-Term Forecasting in Smart Electric Grid Using N-BEATS", International Conference on Power, Control and Computing Technologies (ICPC2T), pp. 1-6, 2022.

[5] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks", Advance Neural Inference Process System, Montreal, QC, Canada, pp. 3104–3112, 2014.

[6] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator", IEEE Conference Computer Vision Pattern Recognition, Boston, MA, USA, pp. 3156–3164, 2015.

[7] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks", ICML, Beijing, China, pp. 1764-1772, 2014.

[8] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on LSTM recurrent neural network", IEEE Transaction Smart Grid, vol. 10, no. 1, pp. 841–851, 2019.

[9] C. G. Huang, H. -Z. Huang and Y. -F. Li, "A Bidirectional LSTM Prognostics Method Under Multiple Operational Conditions", IEEE Transactions on Industrial Electronics, vol. 66, no. 11, pp. 8792-8802, 2019.

[10] M. K. Negia, R. M. Tamiru and M. Meshesha, "Amharic-Kistanigna Bi-directional Machine Translation using Deep Learning", International Conference on Information and Communication Technology for Development for Africa (ICT4DA), Bahir Dar, Ethiopia, pp. 61-65, 2022.

[11] N. Kapali, T. Tuhin, A. Pramanik, M. S. Rahman and S. R. H. Noori, "Sentiment Analysis of Facebook and YouTube Bengali Comments Using LSTM and Bi-LSTM", 13th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kharagpur, India, pp. 1-6, 2022.

[12] H. Zhang, H. Huang and H. Han, "Attention-Based Convolution Skip Bidirectional Long Short-Term Memory Network for Speech Emotion Recognition", IEEE Access, vol. 9, pp. 5332-5342, 2021.

[13] 2019. [Online]. Available: http://www.ieso.ca/power-data