Hybrid Deep Learning Approach for Load Profile Prediction

Fatema Tamanna, K. M. Merajul Arefin, Md. Abdul Masud and Md. Samsuzzaman

February 14, 2025

2024 6th International Conference on Sustainable Technologies for
Industry 5.0 (STI), 14-15 December, Dhaka

Hybrid Deep Learning Approach for Load Profile Prediction

Fatema Ferdous Tamanna*, K. M. Merajul Arefin†, Md. Abdul Masud‡, Md. Samsuzzaman§
*Faculty of Computer Sc. and Engr., Patuakhali Sc. & Tech. University
† Dept of Computer Sc. and Engr.,University of Dhaka
‡Dept of Computer Sc. and Information Technology, Patuakhali Sc. & Tech. University
§Dept of Computer and Communication Engr., Patuakhali Sc. & Tech. University
Email: fatimatamannaah@gmail.com, merajularefin@gmail.com, masud@pstu.ac.bd, sobuz@pstu.ac.bd

Abstract—**Predicting load profiles is crucial for efficient electricity management, but traditional methods often struggle with the complexity of real-world data. While deep learning models have been explored for load forecasting, achieving high accuracy remains challenging. This paper presents a streamlined ensemble approach that combines bidirectional LSTM (BiLSTM), bidirectional GRU (BiGRU), and temporal convolutional network (TCN) layers to capture intricate temporal patterns in load profiles. A self-attention mechanism enhances the model's focus on the most relevant features, improving overall representation. The outputs of these components are combined using an XGBoost regressor to produce the final prediction. In testing, this hybrid model achieved notably higher accuracy up to 93% and faster processing times than other advanced models, showing strong promise for real-time load forecasting in smart grid systems.**

Index Terms—**Temporal Convolutional Network (TCN), Bidirectional Gated Recurrent Unit (BiGRU), Bidirectional Long Short-Term Memory (BiLSTM), Self Attention(SA), Hybrid, Load-profile datasets.**

I. Introduction

Accurate prediction of future load profiles is essential for effective decision-making in energy management, impacting everything from grid stability to resource planning. In the realm of electricity consumption, precise load forecasting enables better operational strategies and resource allocation. However, traditional forecasting techniques often struggle to capture the complexities and nuances present in real-world load data. These methods can fall short when it comes to understanding temporal patterns, non-linear behavior, and the variability inherent in power consumption, highlighting the need for more advanced approaches.

In this study, we present a new methodology for predicting load profiles through a hybrid ensemble model. Unlike conventional approaches that rely on a single forecasting model, our proposed system combines multiple models to enhance accuracy and robustness while making the predictions more interpretable. Our model integrates BiLSTM networks, BiGRUs, and TCNs as foundational elements and then uses extreme gradient boosting (XGBoost) to consolidate their strengths.

Furthermore, our hybrid model includes a self-attention mechanism within the combined layer, followed by dense layers to refine the predictions. This approach is designed to capture complex temporal patterns and learn from a variety of data representations, improving overall predictive performance.

The primary contributions of this paper are: 1. We propose a hybrid ensemble model combining BiLSTM, BiGRU, TCN, and XGBoost to achieve higher predictive accuracy in load forecasting. 2. We integrate a self-attention mechanism to capture complex temporal dependencies more effectively. 3. We conduct extensive experiments on real-world load profile datasets to demonstrate the effectiveness of our approach. Additionally, we provide a detailed account of the model's structure, training process, and evaluation metrics, and offer insights into overcoming challenges in load profile prediction.

The remainder of this paper is organized as follows: Section II discusses related work in load forecasting and hybrid modeling. Section III describes the architecture and methodology of our hybrid model in detail. Section IV presents the experimental setup, datasets used, and evaluation metrics. Finally, section V analyzes the results, comparing our model's performance with traditional and individual approaches, along with the conclusion and future works.

II. Related work

Load profile prediction is essential for efficient energy management, allowing precise demand forecasting and maintaining grid stability. It enhances resource allocation, reduces costs, and guides infrastructure planning. Additionally, it facilitates the integration of renewable energy sources, improving reliability and supporting personalized services for consumers. Effective load forecasting also helps achieve sustainability targets and informs policy development. Numerous studies by various researchers- Hippert [1], Sajjad [2], Brodowski [3], Zuo [4] has been done on load profile prediction. In this paper, we have considered the papers by Abumohsen [5], Lindberg [6], [7], [8] etc. We have examined a few cutting-edge models that outperformed all other conventional or hybrid versions. Xiuyun [9] proposed a short-term load forecasting model based on GRU. We found out by thorough research that l2 regularised GRU does much better than the traditional GRU. Therefore we continued our experiment on l2 regularised GRU with other models. J Xu, P Zeng [10]have a paper on Short-term Load Forecasting by BiLSTM Model. Y. Tian [11]presented the hybrid model GRU-FCN, which is built on a fully convolutional network and a gated recurrent unit. It takes a very long time to execute and this performs poorly on tiny datasets. Taking this much time has made it impractical and also the poor performance in small datasets has made it necessary to make a better model. For energy load forecasting, Kumar [12] developed a hybrid model in combination of LSTM and GRU that they said, performed better than LSTM but about the same as GRU. Chiu [13] proposed a hybrid model CNN-GRU that predicts better in some small datasets but results worse than even conventional models in large datasets. So we targeted these problems to create a model that can capture both short-term and long-term dependencies with better accuracy and in less time.

III. Methodology

In the proposed model we have used, LSTM units, GRU units, and TCN units and combined these all along with XGBRegressor. We have changed the sizes according to the dataset's size.

A. LSTM Cell

LSTM was introduced by Sepp Hochreiter and Jurgen Schmidhuber in 1997 [14]. In this cell, we have used LSTM with Bidirection. Suppose we have an input sequence $\mathbf{x} = [x_1, x_2, ..., x_n]$ of length n.

Input Gate:

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

Forget Gate:

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

Output Gate:

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

Memory Cell Update:

$$\widetilde{C}_t = \tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c) \tag{4}$$

Where: x_t is the input at time step t, h_{t-1} is the previous hidden state, i_t, f_t, and o_t are the input, forget, and output gate activations, C_t is the cell state, $\widetilde{C}_t$ is the candidate cell state, σ is the sigmoid activation function and W is the weight matrix. We used LSTM units with sizes of 64 and 100, batch sizes ranging from 128 to 3200 based on dataset size, and a dropout rate of 0.2 to prevent overfitting.

B. GRU Cell

LSTM has separate memory cells, input, forget, and output gates, whereas GRU has fewer gates, merging the forget and input gates into a single update gate. Additionally, LSTM maintains a separate cell state, allowing it to better control the flow of information over time, whereas GRU combines the cell state and hidden state into a single vector, potentially simplifying the model architecture [15]. For a given time step t,

Update Gate:

$$z_t^f = \sigma\left(W_{zx}^f x_t + W_{zh}^f h_{t-1} + b_z^f\right) \tag{5}$$

Reset Gate:

$$r_t^f = \sigma\left(W_{rx}^f x_t + W_{rh}^f h_{t-1} + b_r^f\right) \tag{6}$$

Candidate Hidden State:

$$\tilde{h}_t^f = \tanh\left(W_{hx}^f x_t + r_t^f \odot \left(W_{hh}^f h_{t-1}\right) + b_h^f\right) \tag{7}$$

Hidden State Update:

$$h_t^f = \left(1 - z_t^f\right) \odot h_{t-1} + z_t^f \odot \tilde{h}_t^f \tag{8}$$

Where: x_t is the input, h_t is the hidden state at time step t, z_t^f is the update gate activation, r_t^f is the reset gate activation, $\tilde{h}_t^f$ is the candidate hidden state, σ is the sigmoid activation function, $\odot$ represents element-wise multiplication, and W and b are weight matrices and bias vectors, respectively, for each gate and transformation, with appropriate subscripts denoting forward (f) or backward (b) direction. In the R.GRU model, we configured the model with a learning rate of 0.001, a dropout rate of 0.2 to reduce overfitting, and an L2 regularization with 0.001 to enhance model generalization.

C. TCN Cell

TCNs are designed to handle both local and long-range dependencies through dilated convolutions, which expand the receptive field exponentially [16]. It is used in the proposed model to efficiently capture dependencies with fewer parameters compared to recurrent networks, enhancing the model's ability to learn temporal patterns over extended periods. It complements LSTM and GRU layers by providing a different approach to sequence modeling, improving overall predictive performance. Suppose we have a 1D input sequence $\mathbf{x} = [x_1, x_2, ..., x_n]$ of length n. The TCN convolutional layer equation is:

$$z_i = \sum_{j=0}^{k-1} w_j \cdot x_{i+d_j} \tag{9}$$

$$y_i = \text{LeakyReLU}(z_i + b) \tag{10}$$

Where: *Where:* x_i is the input sequence, w_j are the convolutional filters, d is the dilation rate, b is the bias term, and LeakyReLU is the rectified linear unit activation function. We took tcn filters 64, kernel size 3, dilation rate 4,8, conv1D with causal padding and relu activation layer. Then we applied GlobalMaxPooling1D, Dense layers, and AdaBeliedOptimizer.

D. Combined Model

We developed three separate models, BiLSTM, BiGRU, and TCN, configured to process the input data with identical dimensions. These models were then executed in parallel, each receiving the same input data and producing outputs in a uniform shape. The outputs from the three models were then fed into a combined model that integrates these outputs. Each individual model applies its unique mechanism to capture patterns and trends within the data, and by combining the outputs, the model leverages multiple perspectives on the same input. The combined model's performance is refined by comparing its output against actual data in the validation set, allowing it to learn the degree and manner in which it should follow the patterns identified by each individual model. That's why it can capture and perform better than the other models. In the combined model after concatenating the models, the output is then passed through self-attention, globalmaxpooling, and then dense layers for further processing. Let h_{LSTM}, h_{GRU},

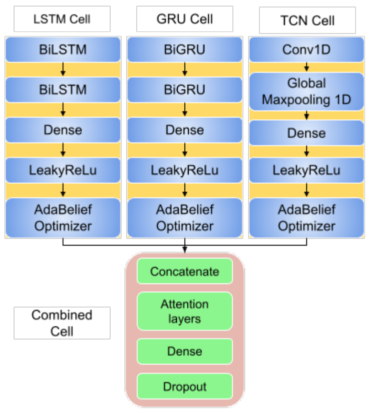

Fig. 1: Model Architecture

and h_{TCN} represent the outputs of the combined, LSTM, GRU, and TCN models respectively. After concatenation, we get the combined output

$$\text{concatOutput} = [\text{concat}(h_{LSTM}, h_{GRU}, h_{TCN})]$$

Where $[\text{concat}(\cdot)]$ denotes concatenation operation. The shape of concatOutput is (m, d), where m is the number of samples and d is the combined feature dimensionality.

E. Attention Layer

The attention mechanism [17] in time series prediction learns temporal dependencies by weighing the importance of different time steps using queries, keys, and values derived from the input sequence. This helps the model focus on the most relevant past data points to improve forecasting accuracy. In this case, we have used Temporal Multihead Attention layer, which allows the model to focus on different time steps within sequences. It uses separate linear transformations to generate query, key, and value matrices, splits these into multiple heads for parallel processing, and applies scaled dot-product attention to capture temporal dependencies. Finally, the outputs from all heads are concatenated, reshaped, and passed through a dense layer to produce the final result.

$$\text{attention} = [(TempMultiHead(numHeads, dModel))]$$

$$\text{attention} = (GlobalMaxPooling1D()) \cdot \text{attention}$$

Inside that TempMultiHead function we have applied dense layers to query, key, and value to create wq, wk, and wv (linear projections of query, key, and value), split each of wq, wk, and wv into numHeads for parallel processing, reshaped the tensors to separate numHeads and depth. Then,

$$\text{score} = \text{query} \cdot \text{key} \,/\, \sqrt{\text{depth}}$$

$$\text{weight} = \text{softmax(score, axis)}$$

After this, we multiplied the attention weights with the value for each head, concatenated and reshaped back to the original dimensionality. We took number of heads as 4 and dimensionality as 128 and changed this according to the dataset size. We then created feature set using the combined models and fit it into the XGBmodel. XGBoost is an ensemble learn-

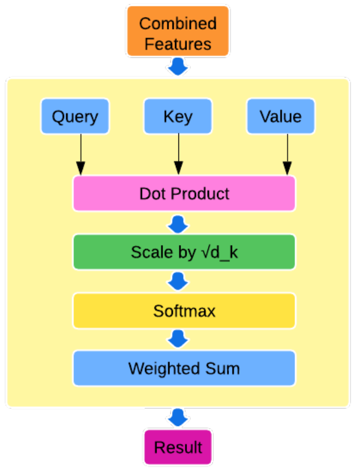

Fig. 2: Attention Mechanism

ing method that combines multiple weak learners, typically decision trees, to build a strong predictive model [18] The `XGBoostRegressor` model learns a function $F(x)$ that maps input features x to the target variable y. At each iteration t, the model fits a new weak learner (a decision tree) to the negative gradient of the loss function $L(y, F(x))$ with respect to the current prediction $F(x)$. The final prediction is the sum of predictions from all weak learners, weighted by a learning rate η. Mathematically, the prediction at iteration t can be represented as:

$$\hat{y}^{(t)} = \sum_{k=1}^{K} \eta \cdot h_k(x) \tag{11}$$

Where: $\hat{y}^{(t)}$ is the predicted value at iteration t, $h_k(x)$ is the prediction of the k-th weak learner, K is the total number of weak learners, η is the learning rate.

The final prediction is obtained by summing up all the iterations. The XGBRegressor is trained on these combined features, a new model is created and updated repeatedly, and then predict the test data with that model.

F. Evaluation

After training the XGBoost model, it is evaluated using test data. The predictions made by the model ($\hat{y}$) are compared against the actual target values (y). The XGBoost model combines predictions from multiple weak learners to make a final prediction.

Evaluation metrics, such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), are calculated to assess the model's performance. Mathematically, MAE and RMSE are defined as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{12}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{13}$$

These metrics are commonly used to evaluate forecasting model accuracy [19].

We have also calculated the Accuracy with the help of sMAPE(symmetric Mean Absolute Percentage Error). While comparing these RMSE, MAE, and Accuracy of different models on different datasets, we have seen that the 'GRU-FCN' hybrid model's RMSE and MAE were better in the large(2M) datasets than the BiLSTM and R. GRU,

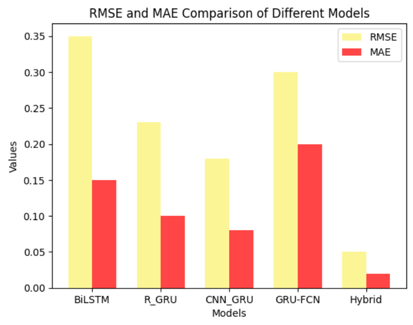

Fig. 3: prediction comparison of different models on Plastic Industry Load Profile(1 Lakh)

But in small datasets(50k) they were resulting worse. The main strategy of our model was the temporal multi-head attention method applied to the combined model. It increases the model's performance drastically.

IV. Experiment

This section presents the performance comparison of the proposed ensemble hybrid model experimental result with other hybrid models CNN-GRU, GRU-FCN, and conventional models BiLSTM, R.GRU.

A. Dataset

Here we have taken 6 datasets, from which 5 were real-time data from UCI repository [7], the other one was taken from UCI too but not realtime, household power consumption dataset [20] that contains every minute's power consumption of several years, and it is about 2M data. The real-time load profile data taken from the UCI repository contains every 15-minute load consumption of various industries. For each dataset, we have taken the first 60% of the data for training, 20% for validation, and 20% for testing. We focused solely on the feature that we are predicting, without considering any other features or their correlations.

TABLE I: A part of Plastic industry Load Consumption dataset

Date	Time	Global Active Power (kW)
2012-01-01	00:00:00	37.98
2012-01-01	00:15:00	41.94
2012-01-01	00:30:00	38.16
2012-01-01	00:45:00	38.64
2012-01-01	01:00:00	39.00
2012-01-01	01:15:00	37.44

For this dataset, we considered the previous global active power consumption rate, trained the model with its trends and patterns, and predicted the future global active power consumption for this industry.

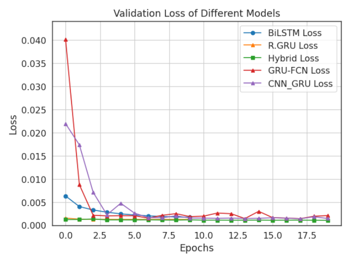

Fig. 4: Epoch loss on plastic industry dataset

B. Preprocessing

At first, we removed the data that we were not considering such as reactive powers, sub-metering, voltage, intensity etc and only active powers were taken from the datasets. Outliers were removed and the data was scaled by minmaxscaler to normalize the feature. Kurtosis and Skewness of the normal distribution are calculated to detect departures from normality. Then we distributed the date time according to weekday, month, year, quarter, etc to remove the ineffective data. We have followed both short-term load forecasting [9] and long-term load forecasting [6] [1]. As we are working on time series data we are focusing on those features that are time dependent so that the trend and pattern can be easily followed to create the future prediction. We are using past records of active power from the grids of these industries to predict future values, without considering any additional features.

C. Experimental Settings

We have tried LSTM, BiLSTM, GRU, L2 regularised GRU, BiGRU, TCN, GRUFCN, CNNGRU but out of all these experiments we saw our hybrid model that is made in combination of BiLstm-BiGRU-TCN-SA and XGBRegressor did the best. LSTM, GRU and BiGRU were removed from the list since their early results were too poor. Different hybrid models has different experimental settings such as:

1) GRU-FCN: Y. Tian [11] proposed a hybrid model GRU-FCN on industrial load data augmentation. We collected the GRU-FCN code from github. Here they employs a GRU layer with 8 units to capture temporal dependencies, it permutes the input for the FCN component, applying three convolutional layers with 128, 256, and 128 filters and kernel sizes of 8, 5, and 3 respectively, each followed by batch normalization and ReLU activation. The GRU and FCN outputs are concatenated and passed through a dense layer to produce the final predictions. The model is compiled with the Adam optimizer and trained for 20 epochs with a batch size of 64.

2) CNN-GRU: We have followed this from some papers [2] [13]. It starts with convolutional layers (e.g., Conv1D) to capture local patterns and features from the input sequences, followed by pooling layers (e.g., MaxPooling1D) to down-sample and reduce dimensionality. Batch normalization and dropout layers are employed to stabilize training and prevent overfitting. The output from the CNN layers is fed into GRU layers to capture temporal dependencies and long-term relationships within the data. The model concludes with a Dense layer with a sigmoid activation for the final prediction, providing a robust architecture for modeling complex sequential data.

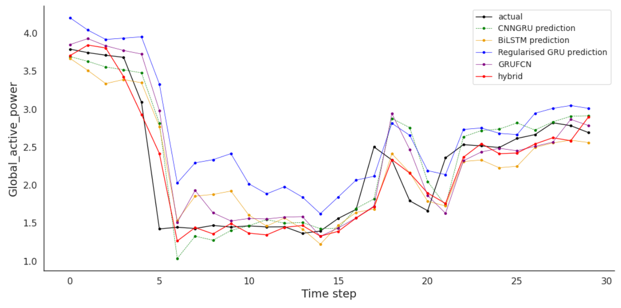

Fig. 5: prediction comparison of different models on Textile Industry Load Consumption

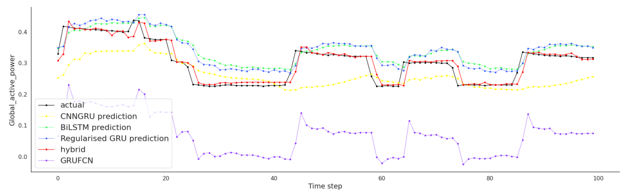

Fig. 6: prediction comparison of different models on Household power consumption

3) Proposed model: We have followed several papers to generate this model. It involves creating and training three separate neural network models LSTM, a GRU, and a TCN each designed to capture different aspects of time-series data parallelly. These models results are then combined using a concatenation layer, followed by additional dense layers to refine the integrated features. The combined model is trained with the AdaBelief optimizer and a learning rate scheduler to enhance performance. Finally, predictions from the combined model are concatenated with the original features and fed into an XGBoost regressor, which is trained to make the final predictions.

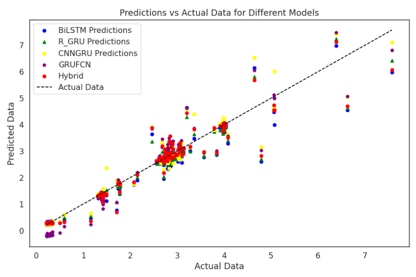

Fig. 7: Scatter graph of predictions comparison of different models on a dataset

D. Experimental Result

TABLE II: Test result of different methods on different datasets

DataSets	Method	Accuracy		RMSE	MAE
		Testing	Training		
Automotive Industry Dataset	A	85.72	88.71	2.67	1.93
	B	84.30	86.19	3.41	2.11
	C	87.50	88.72	2.53	2.01
	D	**91.10**	93.17	0.85	0.70
	E	89.30	90.47	1.24	1.11
Non Metalic Mineral Industry LoadCons.	A	75.32	78.32	2	1.34
	B	82.12	82.89	1.76	1.18
	C	72.88	73.41	2.17	1.4
	D	**89.22**	91.31	1.19	0.79
	E	73.11	76.02	2.12	1.34
Textile Industry Load Consumption	A	72.55	73.14	1.33	1.10
	B	80.31	81.95	1.11	0.80
	C	72.19	73.85	1.45	1.10
	D	**82.65**	83.77	1.10	0.83
	E	70.10	70.32	1.70	1.2
Household Electricity Consumption	A	90.96	92.55	0.221	0.079
	B	89.82	91.04	0.220	0.078
	C	85.91	86.23	0.258	0.095
	D	**93.24**	93.10	0.207	0.076
	E	91.23	92.13	0.100	0.070
Plastic Industry Load Consumption	A	68.43	70.41	15.01	10.86
	B	69.87	70.82	14.43	10.55
	C	67.11	68.01	15.10	11.66
	D	**74.58**	75.86	13.78	10.10
	E	68.11	70.13	14.43	10.89
Paper Industry Load Consumption	A	87.10	89.31	6.56	4.73
	B	88.31	89.11	6.15	4.33
	C	80.29	82.45	7.98	5.82
	D	**90.85**	91.89	5.70	4.01
	E	83.43	84.05	7.7	5.31

In this table, the method names are defined as follows: A represents BiLSTM, B represents L2-regularized GRU, C stands for GRU-FCN, D denotes the Proposed Model, and E corresponds to CNN-GRU. We took a variety of datasets, including household load profile prediction, paper industry, automotive industry load prediction, and datasets having between 20 lakh and 50k data points.

At first, we experimented with LSTM, BiLSTM, and GRU models across multiple datasets, using a household load consumption dataset as our primary test set [20]. Through further analysis, we observed that these conventional models were insufficient for capturing all complex dependencies in the data. This led us to test the BiGRU and L2-regularized GRU models. While BiGRU produced poorer results on our base dataset compared to BiLSTM, the L2-regularized GRU yielded better accuracy than BiLSTM.

TABLE III: Time taken per epoch by different methods for small and large datasets

D.S.Size	BiLSTM	RGRU	CNNGRU	GRUFCN	Hybrid
S (50k)	6s	5s	11s	49s	10s
L (2M)	260s	220s	255s	2600s	271s

Seeking more advanced hybrid approaches, we identified CNN-GRU and GRU-FCN as potential solutions. Applying these models to the datasets improved results overall, but on smaller datasets, they proved less accurate and more time-consuming than the traditional BiLSTM and L2-regularized GRU models. Consequently, we pursued the development of a hybrid model optimized for both small and large datasets. To achieve this, we combined the strengths of LSTM, GRU, and TCN to simultaneously capture both local and global dependencies. This led to our proposed hybrid model, which outperforms all previously tested models in terms of both accuracy and time complexity.

Our proposed model offers significant benefits over individual GRU-FCN (GRU with Fully Convolutional Network) and CNN-GRU (Convolutional Neural Network with GRU) models. On small datasets, the model performed best without the XGBRegressor, while on larger datasets, the addition of XGBRegressor enhanced performance. By utilizing LSTM's ability to handle long-term dependencies, GRU's efficiency with temporal data, and TCN's effective modeling of long-range dependencies through causal convolutions and dilations, our model delivers robust feature extraction and improved predictive capability.

V. Conclusion and Future Work

We introduced a deep learning hybrid model that integrates BiLSTM, BiGRU, TCN, Self-attention and XGBoost regressor for superior load profile forecasting. Our experimental results show that the proposed hybrid model significantly outperforms individual models like BiLSTM, L2 regularized GRU, GRU-FCN, and CNN-GRU, achieving the best performance across key metrics, including the lowest RMSE and MAE, and an accuracy of 93.24%. This demonstrates the model's robustness and effectiveness in capturing complex temporal dependencies. Future work is to focus on optimizing hyperparameters, deploying the model for real-time forecasting, incorporating additional data sources for enhanced accuracy, and ensuring robustness across various scenarios. By extending the evaluation to diverse datasets and real-world applications, the model's generalizability and practical utility can be further validated, paving the way for more reliable and accurate load profile predictions.

VI. Acknowledgement

This paper was supported by the Research and Training Center at Patuakhali Science and Technology University.

References

[1] H. S. Hippert, C. E. Pedreira, and R. C. Souza, "Neural networks for short-term load forecasting: A review and evaluation," *IEEE Transactions on power systems*, vol. 16, no. 1, pp. 44–55, 2001.

[2] M. Sajjad, Z. A. Khan, A. Ullah, T. Hussain, W. Ullah, M. Y. Lee, and S. W. Baik, "A novel cnn-gru-based hybrid approach for short-term residential load forecasting," *IEEE Access*, vol. 8, pp. 143 759–143 768, 2020.

[3] S. Brodowski, A. Bielecki, and M. Filocha, "A hybrid system for forecasting 24-h power load profile for polish electric grid," *Applied soft computing*, vol. 58, pp. 527–539, 2017.

[4] C. Zuo, J. Wang, M. Liu, S. Deng, and Q. Wang, "An ensemble framework for short-term load forecasting based on timesnet and tcn," *Energies*, vol. 16, no. 14, p. 5330, 2023.

[5] M. Abumohsen, A. Y. Owda, and M. Owda, "Electrical load forecasting using lstm, gru, and rnn algorithms," *Energies*, vol. 16, no. 5, p. 2283, 2023.

[6] K. B. Lindberg, P. Seljom, H. Madsen, D. Fischer, and M. Korpås, "Long-term electricity load forecasting: Current and future trends," *Utilities Policy*, vol. 58, pp. 102–119, 2019.

[7] M. A. Masud, J. Z. Huang, M. Zhong, and X. Fu, "Cluster survival model of concept drift in load profile data," *IEEE Access*, vol. 6, pp. 51 269–51 285, 2018.

[8] M. A. M. Fatema Ferdous Tamanna, Tamanna Akter Sonaly, "Enhancing prediction accuracy using ensemble deep learning methods on time series data," in *International Conference on Innovations in Science, Engineering and Technology (ICISET 2024)*, 2024, p. Accepted and Presented.

[9] G. Xiuyun, W. Ying, G. Yang, S. Chengzhi, X. Wen, and Y. Yimiao, "Short-term load forecasting model of gru network based on deep learning framework," in *2018 2nd IEEE Conference on Energy Internet and Energy System Integration (EI2)*. IEEE, 2018, pp. 1–4.

[10] J. Xu and P. Zeng, "Short-term load forecasting by bilstm model based on multidimensional time-domain feature," in *2024 4th International Conference on Neural Networks, Information and Communication (NNICE)*. IEEE, 2024, pp. 1526–1530.

[11] Y. Tian, L. Shao, A. Wang, S. Ma, H. Jin, Y. Yao, and H. Zhang, "Recursive full convolutional network hybrid model based on industrial load data augmentation," in *International Conference on Optoelectronic Information and Functional Materials (OIFM 2023)*, vol. 12781. SPIE, 2023, pp. 480–485.

[12] S. Kumar, L. Hussain, S. Banarjee, and M. Reza, "Energy load forecasting using deep learning approach-lstm and gru in spark cluster," in *2018 fifth international conference on emerging applications of information technology (EAIT)*. IEEE, 2018, pp. 1–4.

[13] M.-C. Chiu, H.-W. Hsu, K.-S. Chen, and C.-Y. Wen, "A hybrid cnn-gru based probabilistic model for load forecasting from individual household to commercial building," *Energy Reports*, vol. 9, pp. 94–105, 2023.

[14] S. Hochreiter, "Long short-term memory," *Neural Computation MIT-Press*, 1997.

[15] K. Cho, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[16] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 156–165.

[17] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[18] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[19] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance," *Climate research*, vol. 30, no. 1, pp. 79–82, 2005.

[20] G. Hebrail and A. Berard, "Individual Household Electric Power Consumption," UCI Machine Learning Repository, 2012, DOI: https://doi.org/10.24432/C58K54.