Multi-Task Learning for Short-Term Load Forecasting Using State Transfer in Autoencoder-LSTM

Ali Forootani
Department of Electrical and Software Engineering
University of Calgary, Calgary, AB, Canada
Email: Ali.forootani1@ucalgary.ca

Hamidreza Zareipour
Department of Electrical and Software Engineering
University of Calgary, Calgary, AB, Canada
Email: Hzareipo@ucalgary.ca

***Abstract*—Accurate short-term load forecasting (STLF) plays a crucial role in optimizing grid operations and reducing operational costs for utility companies. Among various forecasting methods, Long Short-Term Memory (LSTM) networks are widely adopted due to their ability to capture temporal dependencies within load sequences. Although LSTM models have shown promising results, this paper further investigates their potential to enhance STLF accuracy for the next 24 hours. A multi-task learning framework inspired by Autoencoders is proposed to improve LSTM-based forecasting by jointly reconstructing historical data and predicting future loads. By transferring the encoder's hidden and cell states to the forecasting decoder, the model captures richer temporal dependencies. Experimental results on a publicly available dataset show that the proposed framework improves forecasting performance over the next 24 forecasting steps, across multiple error metrics (MAPE, MAE, RMSE, and R^2) compared with models widely used in STLF.**

***Index Terms*—Load forecasting, Multi-task learning, Autoencoders, LSTM.**

I. INTRODUCTION

Short-term load forecasting (STLF) is essential for the reliable and efficient operation of power systems [1]. Accurate forecasts help minimize the mismatch between electricity generation and consumption, thereby reducing operational costs and improving system stability. Furthermore, precise load forecasts allow utilities to make better bidding decisions in electricity markets [2]. Consequently, STLF, which typically covers forecasting horizons from one hour up to twenty four hours ahead, continues to receive considerable attention from both academia and industry.

Recent advances in artificial intelligence, particularly in deep learning, have significantly accelerated research in electrical load forecasting. This progress has been supported by the deployment of advanced metering infrastructures (AMIs), which provide high-resolution load data at hourly or even sub-hourly intervals [3]. Before the availability of such data, most forecasting practices in both academia and industry relied on statistical or shallow machine learning models, which had limited capability to capture the nonlinear and short-term fluctuations present in load profiles [4]. As a result, deep learning models have emerged as effective tools for representing complex temporal dependencies, leading to improved short-term forecasting accuracy.

Early research in electrical load forecasting primarily employed statistical models, with the autoregressive integrated moving average (ARIMA) approach being among the most widely used [5], [6]. For instance, Cao et al. [5] combined ARIMA with a similar-day method, where target days were matched with historical days of comparable weather conditions, and the corresponding loads were averaged to estimate demand. Although such models are computationally efficient, they struggle to represent nonlinear relationships and complex temporal dependencies in modern load data. Consequently, researchers have turned to machine learning methods to overcome these limitations.

Following this shift, shallow-architecture machine learning models have been extensively applied to load forecasting. In Zhang et al. [7], support vector regression (SVR) integrated with an adaptive genetic algorithm was developed to incorporate meteorological variables into the prediction process. To further enhance accuracy, an improved hybrid SVR approach was proposed, where input features were optimized through the adaptive genetic algorithm. Similarly, the k-nearest neighbor (KNN) method has been used effectively in load forecasting [8], [9]. For example, Zhang et al. [8] introduced a multivariate KNN regression model for forecasting electricity demand in the UK market. Despite their usefulness, shallow machine learning approaches are limited in capturing the nonlinear temporal dependencies that dominate short-term load patterns.

To address these challenges, researchers began exploring deep learning models capable of learning such dependencies directly from raw data. Kong et al. [4] introduced a Long Short-Term Memory (LSTM) recurrent neural network specifically designed for STLF. This model demonstrated superior accuracy compared with traditional statistical and machine learning approaches, establishing LSTM as a strong baseline for STLF. Almost at the same time, the authors in [10] proposed the use of LSTM networks for STLF and demonstrated their superiority over a wide range of existing methods. They showed that an LSTM model composed of two stacked layers,

979-8-3315-8138-1

each containing 512 neurons, followed by a fully connected layer of 512 neurons, can accurately forecast the load for the next hour. While these advancements marked a turning point in the application of deep learning for STLF, open research questions remain regarding the further improvement of LSTM accuracy and the maintenance of reliable performance over extended horizons.

Building upon this foundation, this paper aims to further enhance the performance of LSTM models for STLF from different perspectives. First, previous studies [4], [10] focused primarily on one-hour-ahead forecasting, while the question remains: how do these models perform over a 24-hour horizon? Second, it is important to investigate the capacity of LSTM networks. Specifically, does achieving high accuracy necessarily require stacked hidden layers and wide dense layers with a large number of neurons? Increasing model capacity generally improves performance [11], but there is also potential to achieve comparable or better results by leveraging advanced deep learning techniques such as multi-task learning (MTL).

Accordingly, this paper proposes a methodology that explores the potential of MTL within an LSTM framework for load forecasting. The primary task focuses on predicting future load values, while the secondary task reconstructs historical (lagged) loads. This design, inspired by the Autoencoder structure, employs a shared encoder and two decoders for reconstruction and forecasting. Furthermore, to strengthen temporal dependency learning, the hidden and cell states of the forecasting decoder are initialized with those of the encoder at the beginning of each forward pass. This state transfer occurs for every batch during training, ensuring that each decoder starts from a temporally informed representation rather than a randomly initialized state. This approach enhances the model's ability to maintain consistency across consecutive forecasting steps and improves its robustness in 24 hour ahead predictions.

The remainder of this paper is organized as follows: Section II describes the proposed methodology, Section III presents and discusses the results on a publicly available dataset, and Section IV concludes the study.

II. Methodology

A. Data preprocessing

Preprocessing begins by converting the time column into a datetime format and sorting all records chronologically. Calendar features, including hour, day of the month, month, and day of the week, are extracted from the timestamp to capture temporal variations in the load profile. Redundant index columns, if present, are removed to ensure data consistency. In addition, typographical inconsistencies in city names are corrected to maintain uniformity across regional temperature columns.

B. Proposed Framework

1) Model Design and Architecture: The proposed model employs an MTL framework to achieve two objectives simultaneously: forecasting future load values (prediction task) and reconstructing historical load patterns (reconstruction task). An MTL model consists of shared layers that support all tasks and some layers dedicated to individual objectives. This configuration facilitates knowledge sharing among tasks and tends to improve accuracy compared with single-task models [12]. In the literature, two main strategies are commonly used when implementing MTL. The first aims to forecast multiple primary targets simultaneously, such as predicting both cooling or heating loads and electrical loads [12]. The second defines an auxiliary task to enhance the performance of the main forecasting objective [13], which is the approach adopted in this study.

Based on this formulation, the proposed architecture integrates a shared temporal representation that supports both forecasting and reconstruction objectives, while task-specific layers are used to specialize each output. This design enables the reconstruction task to act as a regularizing mechanism, improving the robustness and generalization capability of the forecasting task.

The architecture of the proposed model is illustrated in Fig. 1, and its components are described in the following subsections.

Encoder: At the core of the framework lies an LSTM network. Each LSTM unit maintains two internal states: a hidden state (h_i) that captures short-term temporal dependencies, and a cell state (c_i) that retains longer-term contextual information across time steps. Together, these states enable the model to learn complex sequential patterns in the load data. Instead of revisiting the internal gate mechanisms of a conventional LSTM, which are well documented in prior studies, the focus here is on how these states interact within the proposed architecture.

As one of our objectives is to examine the effect of reconstructing historical load sequences on forecasting performance, the overall design adopts an encoder–decoder structure. The encoder processes an input sequence over a defined historical window of 24 hours and generates final states (h_i, c_i) that summarize the temporal dynamics of the observed data. These encoded states are simultaneously transferred to two decoders: one dedicated to reconstruction and the other to forecasting. Sharing a common temporal representation enables both decoders to operate within the same learned context, thereby improving temporal coherence and overall generalization.

Reconstruction Branch: The reconstruction path begins with the *Reconstruction Repeat* block, which expands the encoder's final hidden representation to match the original sequence length of 24. This repeated context is then processed by the *Reconstruction Decoder* block, initialized with the encoder's final hidden and cell states. The decoder output is passed through the *Reconstruction Output* block to reproduce the original input sequence. By learning to reconstruct historical data, this branch enhances the encoder's ability to capture stable temporal representations and promotes generalization.

Forecasting Branch: The forecasting path starts with the *Forecasting Decoder* block, which also receives the encoder's final hidden and cell states as initial conditions. The resulting

979-8-3315-8138-1

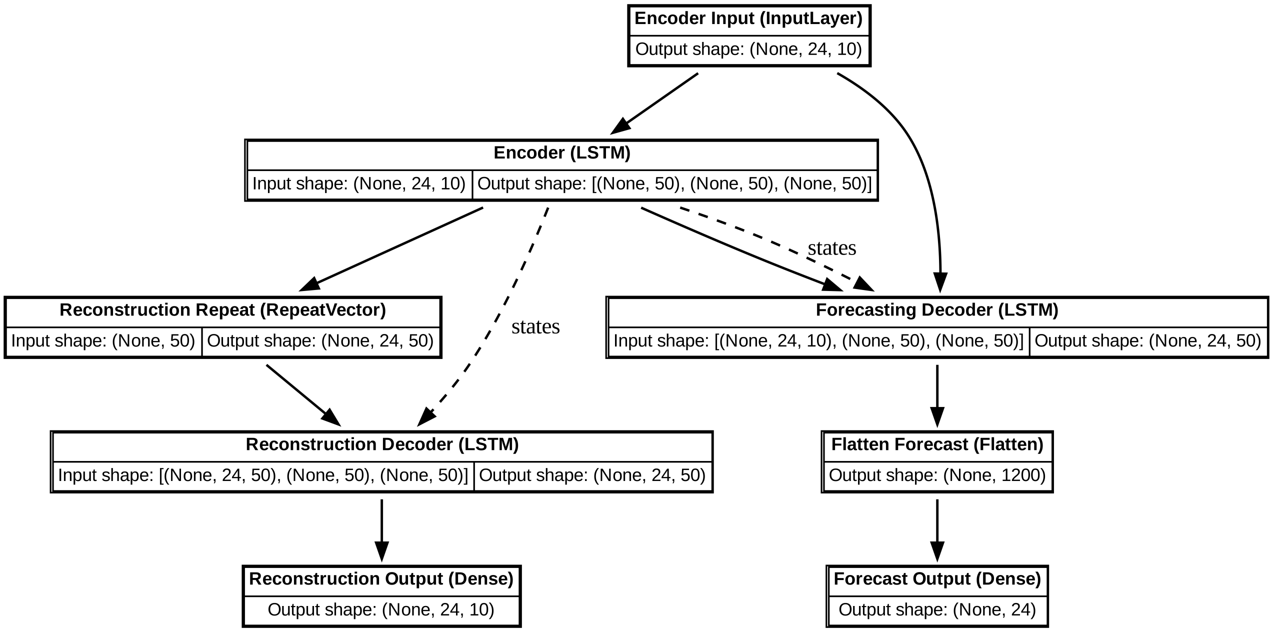

Fig. 1: The proposed model.

sequence output is flattened through the *Flatten Forecast* block, transforming it into a feature vector. Finally, the *Forecast Output* block maps this representation to the 24-hour ahead load forecast. This state transfer enables the forecasting decoder to leverage temporal knowledge encoded by the encoder, improving both convergence speed and forecasting accuracy.

2) Model compilation: To balance the two learning objectives of the model, appropriate loss weights were assigned to each output. The reconstruction output was given a weight of 1.0 to maintain accurate reproduction of historical load patterns, while the forecasting output was assigned a higher weight of 2.0 to emphasize the model's primary objective of predicting future load values. The model was then compiled using the Adam optimizer due to its adaptive learning rate mechanism, which supports efficient and stable convergence. The custom loss function is applied to both outputs, with the predefined weights controlling the contribution of each component to the total loss.

To further improve training efficiency and prevent overfitting, several callback functions were applied. The early stopping function monitored the validation forecasting loss and stopped the training process when no significant improvement was observed for consecutive epochs, preventing overfitting and unnecessary computation. The model checkpoint function saved the model weights whenever an improvement occurred in the validation forecasting loss, ensuring that the best-performing version of the model was retained. In addition, the learning rate reduction on plateau callback decreased the learning rate when validation forecasting loss stagnated over several epochs, allowing finer optimization during later training stages and promoting smoother convergence toward better solution. For consistency and fair comparison, these callbacks are applied to all benchmark models discussed in the following section.

III. Results and Discussion

A. Settings and Case Definitions

The hourly electrical load demand dataset [14] from PJM, a regional transmission organization in the United States, is used in this study. The dataset exhibits an average hourly load of approximately 89,300 MW. The dataset spans from April 2020 to August 2023 and provides hourly load measurements. PJM operates the wholesale electricity market and manages power flow across multiple states, including Delaware, Maryland, Pennsylvania, and New Jersey. In addition, hourly temperature data for the corresponding regions are incorporated as explanatory variables due to their strong influence on load fluctuations. It should be noted that the dataset contains three additional demand-related columns namely *fiveMinDemand*, *fiveMinDemandAVG*, and *AnticipatedDemand*, which should be excluded to avoid look-ahead bias and data leakage during training.

Data from April 14, 2020, to December 31, 2022 are used for training, and January 1, 2023, to August 12, 2023 for testing. Within the training period, a small portion of the samples are reserved for validation to support model selection and early stopping while preventing data leakage.

To evaluate the performance of the proposed model, four commonly used metrics in load forecasting were adopted: Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and the coefficient of determination (R^2). These metrics are computed for both the overall forecast and each time step within the 24-hour

979-8-3315-8138-1

prediction horizon. Candidate hyperparameters are selected based on well-established papers in this domain and refined through empirical experimentation. Specifically, parameters such as the number of LSTM units, number of epochs, and optimizer type were explored through a systematic trial-and-error process using a validation subset comprising 20% of the training data. Through this procedure, the final configuration was determined to include a single LSTM layer with 50 units. A dense layer was then added following the LSTM block, with 24 neurons corresponding to the 24-hour forecasting horizon.

For benchmarking, two reference models from the literature are implemented: (1) an Autoencoder LSTM (AE-LSTM) [15] without the reconstruction component but identical in design, and (2) a conventional single-task LSTM model [4], [10] configured with the same hyperparameters as the proposed approach. These baselines enable a fair and consistent comparison to assess the contribution of MTL and state transfer in the proposed framework.

B. Results and Discussion

Table I summarizes the overall forecasting performance of the proposed model compared with the AE-LSTM and the single-task LSTM across the 24-hour forecasting horizon. The proposed framework achieves the best performance in all evaluation metrics, recording a MAPE of 3.587, an MAE of 3215.20, an RMSE of 4184.61, and an R^2 score of 0.915. These results indicate that the proposed MTL framework with state transfer effectively captures temporal dependencies and nonlinear load variations, resulting in higher forecasting accuracy and improved generalization compared with the baseline models.

TABLE I: Overall error metrics for the 24-hour forecasting horizon.

Model	MAPE (%)	MAE (MW)	RMSE (MW)	R^2
Proposed Model	3.587	3215.20	4184.61	0.915
AE-LSTM	3.878	3455.15	4510.88	0.902
LSTM	3.664	3291.49	4323.03	0.910

Fig. 2 illustrates the forecasting performance of the proposed MTL model compared with the single-task LSTM and AE-LSTM across the 24-hour horizon. The figure shows the variation of key performance metrics, including RMSE, MAE, MAPE, and R^2, across all forecast steps. The proposed framework consistently achieves the lowest RMSE, MAE, and MAPE values and the highest R^2 scores for most forecast hours, indicating superior overall accuracy and robustness.

On average, the proposed MTL model reduces RMSE by approximately 8% compared with the single-task LSTM and by nearly 30% relative to the AE-LSTM. Similarly, MAE decreases by around 7% and 30% compared with the single-task and AE-LSTM models, respectively. The MAPE improvement is more moderate, with reductions of about 6% compared with AE-LSTM and consistent superiority across all forecast steps.

The largest relative improvements are observed during the early forecast hours, where the MTL model achieves up to 30% lower RMSE than the AE-LSTM and around 6% lower than the single-task LSTM at the first forecast hour. At the 12th forecast step, corresponding to mid-range horizons, the RMSE reduction remains substantial at approximately 25% compared with the AE-LSTM. Even in the final forecast hour, the MTL framework maintains an advantage of around 7% over the single-task LSTM and 8% over the AE-LSTM, confirming its ability to preserve accuracy over extended horizons.

The proposed model also exhibits remarkable stability across the 24-hour horizon. Its MAPE increases gradually from approximately 2.5% at the first forecast hour to 4.2% at the 24th hour, reflecting its ability to manage the accumulation of uncertainty over longer lead times. In contrast, both the single-task and AE-LSTM models show larger and more irregular fluctuations in their error metrics, particularly during the later hours.

Although the numerical improvements in aggregate error metrics are marginal, the proposed framework demonstrates consistent performance gains across the entire 24 hour forecasting horizon. As shown in Fig. 2, the proposed MTL model exhibits slower accumulation of forecasting error as the horizon increases compared with both the single task LSTM and the AE LSTM benchmarks. In addition, the degradation of performance with respect to the forecasting step is smoother and more monotonic, indicating improved temporal stability rather than isolated improvements at selected hours. This behavior suggests that the proposed framework enhances the ability to preserve temporal information over multiple steps, which is a critical requirement in multi step short term load forecasting.

IV. Conclusion

This paper presented a study on enhancing the performance of LSTM-based models for STLF. The proposed framework, inspired by MTL and Autoencoder structures, simultaneously forecasts future loads and reconstructs historical patterns through the transfer of the encoder's hidden and cell states to both decoders. This state transfer mechanism enriches the forecasting decoder with temporal information, thereby strengthening its capacity to capture sequential dependencies and improve long-term consistency. Experimental results demonstrated that the proposed model consistently outperforms the benchmark AE-LSTM and single-task LSTM models across all evaluation metrics and most forecasting horizons. Overall, the proposed framework provided a reliable and consistent solution for STLF. As future work, the proposed methodology can be extended to other power system time series, such as wind and solar generation, which play an important role in modern grid operations.

Acknowledgment

The authors acknowledge the use of an AI-based language model for editorial assistance and language polishing. The AI system did not contribute to the technical content, analysis, or conclusions of this paper.

979-8-3315-8138-1

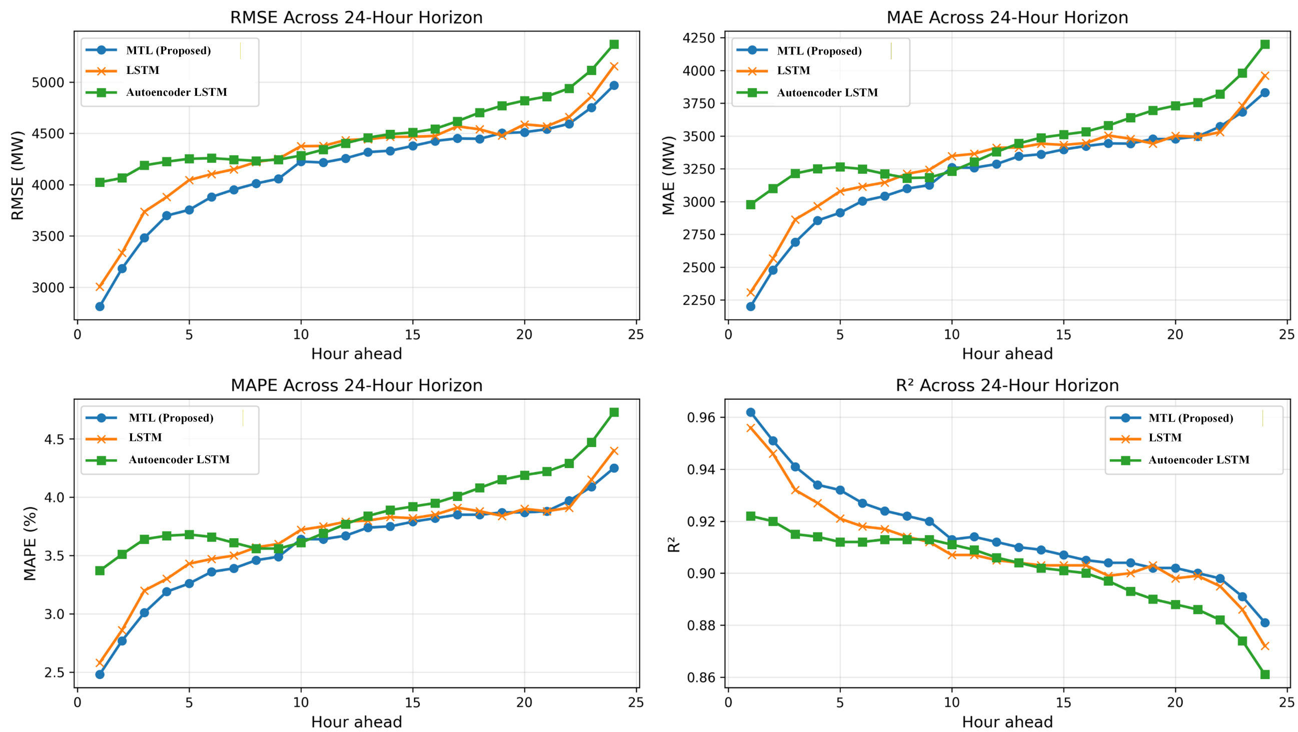

Fig. 2: Comparison of performance across the 24-hour horizon for the proposed model, Autoencoder-LSTM, and single-task LSTM

REFERENCES

[1] A. Ahmadi, H. Zareipour, and H. Leung, "Globalization for scalable short-term load forecasting," *arXiv preprint arXiv:2507.11729*, 2025.

[2] S. Schreck, I. P. de La Comble, S. Thiem, and S. Niessen, "A methodological framework to support load forecast error assessment in local energy markets," *IEEE Transactions on Smart Grid*, vol. 11, no. 4, pp. 3212–3220, 2020.

[3] A. Forootani, M. Rastegar, and A. Sami, "Short-term individual residential load forecasting using an enhanced machine learning-based approach based on a feature engineering framework: A comparative study with deep learning methods," *Electric Power Systems Research*, vol. 210, p. 108119, 2022.

[4] 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 transactions on smart grid*, vol. 10, no. 1, pp. 841–851, 2017.

[5] X. Cao, S. Dong, Z. Wu, and Y. Jing, "A data-driven hybrid optimization model for short-term residential load forecasting." IEEE, 2015, pp. 283–287.

[6] K. G. Boroojeni, M. H. Amini, S. Bahrami, S. S. Iyengar, A. I. Sarwat, and O. Karabasoglu, "A novel multi-time-scale modeling for electric power demand forecasting: From short-term to medium-term horizon," *Electric Power Systems Research*, vol. 142, pp. 58–73, 2017.

[7] G. Zhang and J. Guo, "A novel method for hourly electricity demand forecasting," *IEEE Transactions on Power Systems*, vol. 35, no. 2, pp. 1351–1363, 2019.

[8] R. Zhang, Y. Xu, Z. Y. Dong, W. Kong, and K. P. Wong, "A composite k-nearest neighbor model for day-ahead load forecasting with limited temperature forecasts," in *2016 IEEE Power and Energy Society General Meeting (PESGM)*. IEEE, 2016, pp. 1–5.

[9] F. H. Al-Qahtani and S. F. Crone, "Multivariate k-nearest neighbour regression for time series data—a novel algorithm for forecasting uk electricity demand," in *The 2013 international joint conference on neural networks (IJCNN)*. IEEE, 2013, pp. 1–8.

[10] 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 Transactions on Smart Grid*, vol. 10, no. 1, pp. 841–851, 2019.

[11] L. Han, H.-J. Ye, and D.-C. Zhan, "The capacity and robustness trade-off: Revisiting the channel independent strategy for multivariate time series forecasting," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 11, pp. 7129–7142, 2024.

[12] Y. Guo, Y. Li, X. Qiao, Z. Zhang, W. Zhou, Y. Mei, J. Lin, Y. Zhou, and Y. Nakanishi, "Bilstm multitask learning-based combined load forecasting considering the loads coupling relationship for multienergy system," *IEEE Transactions on Smart Grid*, vol. 13, no. 5, pp. 3481–3492, 2022.

[13] Y. Wang, J. Yan, W. Zhang, Z. Yang, J. Wang, Y. Geng, and D. Srinivasan, "Mutitask learning network for partial discharge condition assessment in gas-insulated switchgear," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 10, pp. 11 998–12 009, 2024.

[14] RobikScube, "Hourly energy consumption," https://www.kaggle.com/datasets/robikscube/hourly-energy-consumption/data, accessed: Month Year.

[15] X. Tong, J. Wang, C. Zhang, T. Wu, H. Wang, and Y. Wang, "Ls-lstm-ae: Power load forecasting via long-short series features and lstm-autoencoder," *Energy Reports*, vol. 8, pp. 596–603, 2022.

979-8-3315-8138-1