IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 17 March 2025, accepted 23 April 2025, date of publication 30 April 2025, date of current version 9 May 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3565798

RESEARCH ARTICLE

Performance Evaluation of Activation Functions in Deep Residual Networks for Short-Term Load Forecasting

JUNCHEN LIU[1], FAISUL ARIF AHMAD[1], (Member, IEEE), KHAIRULMIZAM SAMSUDIN[1], FAZIRULHISYAM HASHIM[1], (Member, IEEE), AND MOHD ZAINAL ABIDIN AB KADIR[2], (Senior Member, IEEE)

[1]Department of Computer and Communication Systems Engineering, Faculty of Engineering, Universiti Putra Malaysia (UPM), Seri Kembangan, Selangor 43400, Malaysia
[2]Advanced Lightning, Power and Energy Research Centre (ALPER), Faculty of Engineering, Universiti Putra Malaysia (UPM), Seri Kembangan, Selangor 43400, Malaysia

Corresponding author: Faisul Arif Ahmad (faisul@upm.edu.my)

ABSTRACT Short-Term Load Forecasting (STLF) is essential for ensuring efficient and reliable power system operations, requiring accurate predictions of electricity demand. Deep Residual Networks (DRNs), with their ability to mitigate gradient vanishing and model complex nonlinear relationships in load data, have emerged as a powerful tool for STLF. This study evaluates the performance of various activation functions within DRN models, focusing on their impact on predictive precision and generalization. Experiments were conducted using the DRN architecture for STLF on two distinct datasets: ISO-NE and Malaysia. The findings demonstrate that activation functions significantly influence the predictive performance of DRN-based STLF models. Specifically, the DRN model using Swish achieved the best results on the ISO-NE dataset (Mean Absolute Percentage Error, MAPE = 1.3806%), while the DRN model with Hyperbolic Tangent (Tanh) excelled on the Malaysia dataset (MAPE = 4.9809%). These results underscore the importance of aligning activation function selection with dataset characteristics to optimize the performance of DRN models in STLF. This study provides valuable insights for advancing STLF research and guiding practical applications in load forecasting.

INDEX TERMS STLF, DRN, activation function.

I. INTRODUCTION

In modern power systems, load forecasting (LF) is a vital tool for ensuring efficient and reliable grid operation. By accurately predicting future electricity demand, LF helps power companies optimize grid planning, operation, and management. This not only enhances energy utilization efficiency, reduces operational costs, and ensures supply stability but also contributes to sustainable energy development. Specifically, improving the precision of load forecasting enables better resource allocation, supports the integration of renewable energy sources, and minimizes environmental impact. As electricity demand grows and consumption patterns become more diverse, the importance and complexity of load forecasting continue to increase [1].

The four primary forms of load forecasting are Very Short-Term Load Forecasting (VSTLF), Short-Term Load Forecasting (STLF), Medium-Term Load Forecasting (MTLF), and Long-Term Load Forecasting (LTLF), as seen in Fig. 1 and categorized based on the forecasting time range [2], [3]. VSTLF is used for real-time control and concentrates on urgent operational demands. Up to an hour beforehand, it can predict load. Real-time power system scheduling and operation depend heavily on STLF, which can last anywhere from one hour to a week. MTLF assists with operational planning, such as scheduling maintenance and managing the power supply to meet expected medium-term demand. MTLF typically spans one week to one year.

The associate editor coordinating the review of this manuscript and approving it for publication was Davide Patti.

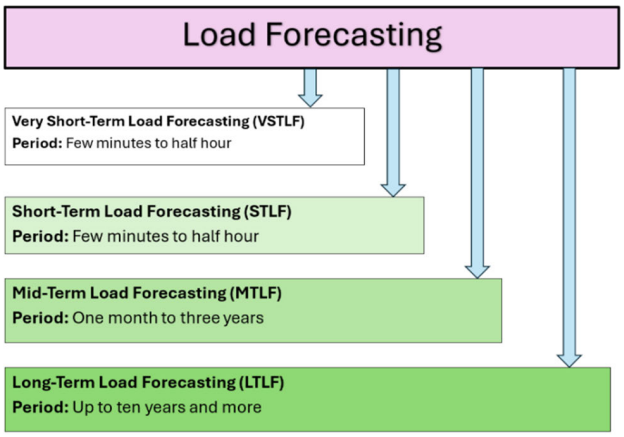

FIGURE 1. Classification of load forecasting.

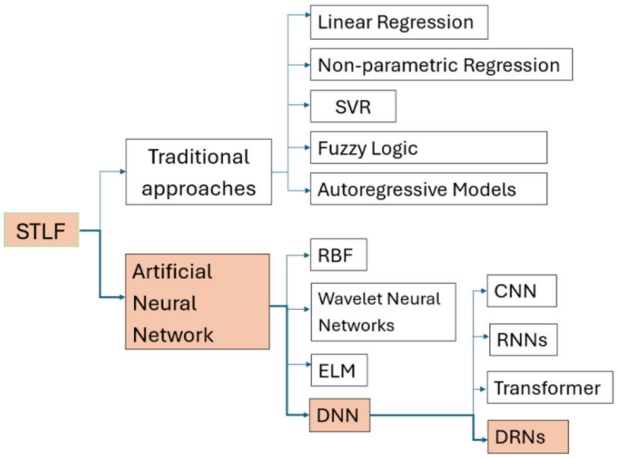

FIGURE 2. Technological Evolution of STLF.

As it helps predict future power demands and the need for additional generating facilities, LTLF can range from one year to many years. This makes it an important tool for strategic planning, policy-making, and investment choices in power system infrastructure. In particular, STLF is essential to the power system's real-time operation and dispatching. The weekly load forecasting for the upcoming seven days, the daily load forecasting for the upcoming 24 hours, and the forecasting many hours in advance are all included in these projections [4].

Future power system operation and management will need for quicker decision-making and the capacity to deal with unpredictability. Load forecasting is essential in many application domains, such as energy trading, unit commitment choices, system security evaluation, economical power production distribution, and performance monitoring. Therefore, the calibration and verification of estimated precision become increasingly important [5]. Grid scheduling, load flow analysis, daily operations, and performance are only a few of the issues covered by load forecasting. Precise forecasting is crucial to the system's regular functioning as imprecise load forecasting can result in unforeseen expenses [6].

In recent years, researchers have proposed many methods to address STLF problems, broadly categorized into traditional and modern forecasting methods. Common traditional forecasting methods include linear regression [7] or non-parametric methods (e.g., non-parametric regression [8], exponential smoothing [9], [10]), support vector regression (SVR) [1], [11], [12], autoregressive models [13], and fuzzy logic methods [14], [15]. However, these methods often exhibit limitations in practical applications, such as overly simplistic models, difficulty in capturing complex load patterns, and tendencies toward overfitting and model expansion as input variables increase [1], [16].

To overcome these limitations, modern forecasting methods mainly leverage artificial intelligence and machine learning technologies, particularly artificial neural networks (ANNs), which have become the mainstream solution for constructing STLF systems. Compared with traditional methods, modern ANN methods use deep learning and advanced network structures to better capture complex load patterns, improve prediction precision, and reduce overfitting risks [17], [18]. However, despite increasing the number of input parameters, hidden nodes, or layers, another criticism is that networks are prone to "overfitting" [19]. Nevertheless, other types and subtypes of ANNs, such as radial basis function (RBF) neural networks [20], wavelet-based neural networks [21], and extreme learning machines (ELM) [22], have been proposed and applied to STLF.

As a modern forecasting method, various techniques based on artificial neural networks (ANN), particularly deep neural network (DNN) structures, have been widely applied to load forecasting, especially STLF. DNNs, characterized by multiple hidden layers, effectively extract and learn complex load patterns through hierarchical feature representation. In recent years, load forecasting research has transitioned from traditional shallow neural network structures to more complex and specialized deep learning architectures. These architectures integrate information from diverse sources to capture intricate temporal and spatial dependencies in load data. This shift has been driven by advancements in deep learning techniques and their demonstrated efficacy in handling complex prediction tasks [23], [24], [25]. Figure 2 summarizes the technological evolution of STLF from traditional methods to DNNs, clearly illustrating the methodological development trends in the field of load forecasting.

In recent years, load forecasting research has gradually shifted away from predefined shallow network structures, instead integrating diverse information from various applications into neural network topologies. Convolutional Neural Networks (CNN) [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36] have shown excellent performance in image processing and local feature extraction, effectively capturing local temporal features in load data. However, CNNs are inherently limited to local feature extraction, making

it challenging to model long-term dependencies. Furthermore, as the network depth increases, CNNs often encounter gradient vanishing problems, which make training deeper architectures difficult and limit their practicality in complex tasks like load forecasting.

Recurrent Neural Networks (RNN) [37], [38], [39], [40], [41], [42], [43], [44], [45], [46], [47], [48], [49], [50], [51], [52], [53], particularly variants like Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRU), are widely used in load forecasting due to their ability to model temporal dependencies. LSTM and GRU introduce memory cells and gating mechanisms to alleviate gradient vanishing issues, enabling them to capture both short-term and long-term dependencies in sequential data. However, these models process sequences step-by-step, which not only increases computational complexity but also makes them less efficient for very long time series. For deeper networks, both LSTM and GRU remain susceptible to vanishing or exploding gradients, which can limit their scalability. Similarly, while advanced RNN variants such as Bidirectional LSTM (BiLSTM) and Bidirectional GRU (BiGRU) improve sequence modeling by considering both past and future information, their dual-directional processing further increases computational cost and complexity.

Transformer models [54], [55], [56], [57], [58], [59], [60], [61], [62], [63], [64], [65], [66], [67], [68], [69], [70], [71], leveraging self-attention mechanisms, have recently emerged as a powerful tool for time series forecasting. Transformers excel in capturing long-range dependencies and can efficiently handle sequences of varying lengths. However, their computational complexity grows quadratically with sequence length, making them resource-intensive for ultra-long sequences. Additionally, as with CNNs and RNNs, training stability decreases as the model depth increases, often necessitating additional architectural adjustments to address these challenges.

As model depth increases, the training difficulties associated with CNNs, RNNs, and Transformers often restrict the number of hidden layers, limiting their ability to effectively learn complex patterns. In this context, Chen et al. [72] proposed a DRN for STLF. Unlike other network structures, DRN alleviates the gradient vanishing problem in deep networks by incorporating residual connections, making the model stable even with deep architectures. DRN can directly use historical load, temperature, and time data as input, minimizing the need for extensive feature engineering while automatically extracting complex features from the data. Compared to CNNs, which are effective at capturing local patterns but struggle with long-term dependencies, and RNN-based models, which can model temporal sequences but suffer from vanishing gradients and high computational cost, DRNs offer a more robust and scalable solution for STLF. Moreover, although Transformers can capture long-range dependencies efficiently, their high resource consumption and unstable performance in deeper architectures present challenges in real-time forecasting scenarios. DRNs strike a practical balance between depth, training stability, and representational power, making them particularly suitable for STLF tasks that require both precision and efficiency.

Although the DRN structure demonstrates significant advantages in STLF, its performance is still highly dependent on the choice of activation function. In existing DRN studies, the Scaled Exponential Linear Unit (SELU) activation function, which has self-normalizing properties, has been widely adopted (Chen et al. [72]; Xu et al. [73]; Tian et al. [74]; Kondaiah et al. [75]; Chen et al. [76]; Kondaiah et al. [77]; Sheng et al. [78]). These studies indicate that SELU can significantly enhance prediction precision in various complex short-term load scenarios. Its self-normalizing properties effectively alleviate the vanishing gradient problem and improve the training stability of deep networks. Apart from SELU, some studies have also explored the potential of other activation functions. For instance, Ding et al. [79] and Li et al. [80] utilized the Rectified Linear Unit (ReLU) activation function in their DRN model to take advantage of its computational efficiency and simplicity, which are particularly beneficial for training deep networks on large-scale datasets. By employing ReLU, their model demonstrated competitive prediction precision while maintaining robustness in addressing specific STLF scenarios. On the other hand, Sheng et al. [81] conducted a simple comparison of Sigmoid, ReLU, and Leaky Rectified Linear Unit (Leaky ReLU) activation functions in their study. By analyzing the Mean Absolute Percentage Error (MAPE) of the model under different activation functions, they ultimately selected Leaky ReLU as the activation function for their improved convolutional residual network, significantly enhancing the prediction precision for STLF. This result further demonstrates that different activation functions have unique advantages in addressing nonlinear load patterns and complex data characteristics.

Although previous studies have explored the role of activation functions in Deep Residual Networks (DRN) for Short-Term Load Forecasting (STLF), no comprehensive evaluation has been conducted comparing multiple activation functions within the DRN framework specifically for STLF. The selection of an appropriate activation function remains a critical but understudied factor in optimizing DRN-based load forecasting models. This study addresses this gap by systematically analyzing and comparing the predictive performance of eight commonly used activation functions in DRN for STLF. A thorough review of the literature from the Web of Science database [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44], [45], [46], [47], [48], [49], [50], [51], [52], [53], [54], [55], [56], [57], [58], [59], [60], [61], [62], [63], [64], [65], [66], [67], [68], [69], [70], [71], [72], [73], [74], [75], [76], [77], [78], [79], [80], [81] identified key advancements in CNN, RNN, Transformer, and DRN models applied to STLF, as well as the activation functions most frequently used in these models. Figure 3 provides an overview of activation

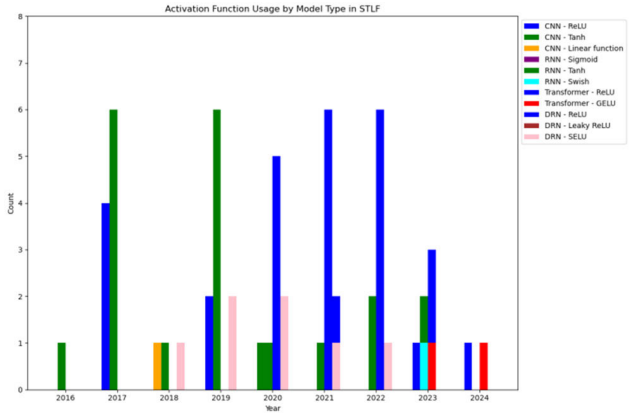

FIGURE 3. Activation Function Usage in STLF Models (CNN, RNN, Transformer, DRN) from 2016 to 2024 in web of science [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44], [45], [46], [47], [48], [49], [50], [51], [52], [53], [54], [55], [56], [57], [58], [59], [60], [61], [62], [63], [64], [65], [66], [67], [68], [69], [70], [71], [72], [73], [74], [75], [76], [77], [78], [79], [80], [81].

function trends across different architectures, underscoring the need for a structured evaluation of their effectiveness in DRN models. To bridge this knowledge gap, a rigorous evaluation was conducted on the impact of different activation functions on DRN performance using two distinct datasets—ISO-NE (temperate climate with significant seasonal variations) and Malaysia (tropical climate with relatively stable load patterns). The experimental results highlight the influence of activation function selection on predictive precision and propose a structured approach for aligning activation function choices with dataset characteristics. The findings offer valuable theoretical insights for enhancing DRN-based STLF models and provide practical idea for improving load forecasting precision in real-world applications.

This study aims to systematically evaluate the performance of various activation functions within the DRN framework to optimize its predictive capabilities in STLF tasks. The remainder of this paper is organized as follows: Section II reviews the application of DRN in STLF; Section III outlines the research methodology, including data preprocessing and experimental setup; Section IV presents the experimental results and discusses the performance of different activation functions on the ISO-NE and Malaysia datasets; finally, Section V concludes the study with key findings and recommendations for future research.

II. DRN FOR STLF

A. BASIC DRN STRUCTURE

The DRN is employed to unravel the intricate nonlinear interplay between input data and the resulting output. Generally, a neural network's learning potential escalates with increased model depth. Yet, paradoxically, this depth might,

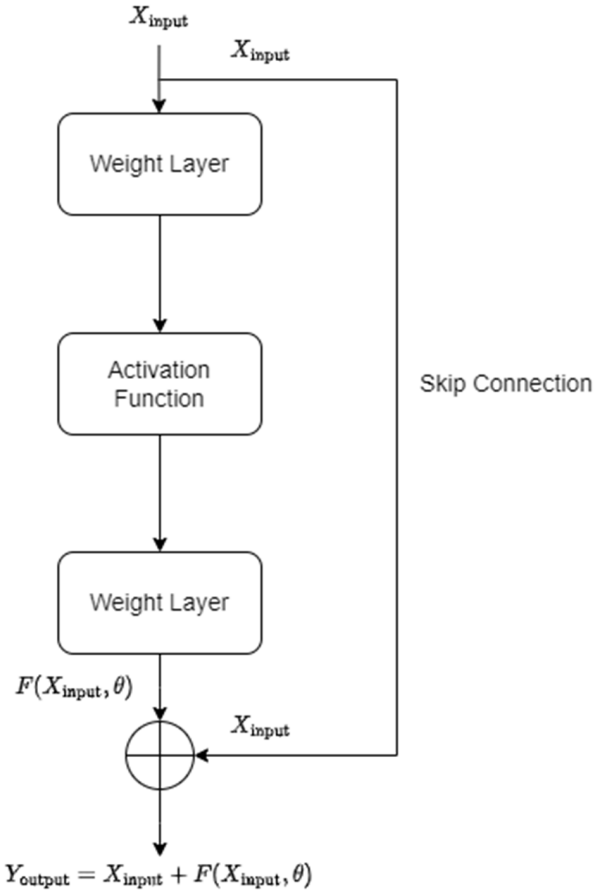

FIGURE 4. The basic component of the deep residual network [72].

in reality, impede the deep learning model's efficacy. This decline in performance could stem from either the intrinsic complexities of the data or the sophisticated nature of the model itself. To address this challenge, residual blocks are incorporated into the architecture. In these blocks, the learning process isn't about mapping directly from input to output but rather about mapping from input to a residual function. This approach facilitates the effective training of deeper networks by optimizing the learning process through residual connections, ensuring better gradient flow and reducing the risk of vanishing gradients [82].

As depicted in Figure 4, a residual network (ResNet) features two sequential levels bridged by a skip connection.

A skip connection typically operates as an identical mapping when the dimensions of its input and output align. Under these conditions, the corresponding ResNet's output is as follows Equation (1):

$$y_{output} = x_{Input} + \mathcal{F}\left(x_{Input}, \Theta\right) \tag{1}$$

However, when input and output dimensions differ, the skip connection assumes the role of a linear projection. In such instances, the associated ResNet yields an output

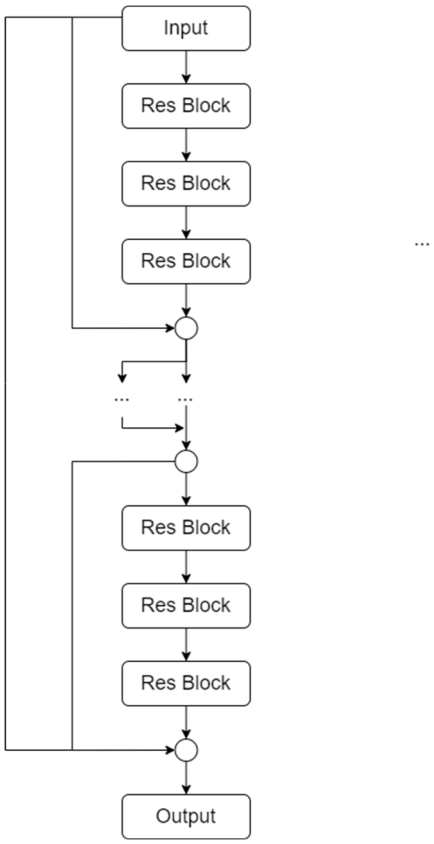

FIGURE 5. The structure of a deep residual network [72].

that integrates this linear projection (L_p), as outlined in Equation (2):

$$y_{output} = L_{p*}x_{Input} + \mathcal{F}\left(x_{Input}, \Theta\right) \tag{2}$$

Stacking multiple residual blocks allows for the straightforward construction of a deep residual network. Figure 5 depicts the structure of a deep residual network.

This skip connection indicates that the learning capacity of a residual block (ResBlock) is, at minimum, on par with that of an equivalently layered stack. When n residual blocks are sequentially arranged, the Equation (3) for forward-propagation is as follows:

$$y(x) = x_0 + \sum_{j=1}^{n} F\left(x_j{-}1, \Theta_j{-}1\right) \tag{3}$$

where x_0 is the input of the residual network, x_n the output of the residual network, and

$$\Theta_j = \left\{\theta_{j,z}\big|_{1\le z\le Z}\right\}$$

the set of weights associated with the jth residual block, Z being the number of layers within the block.

The back propagation of the overall loss of the neural network to x_0 can then be calculated as Equation (4):

$$\frac{\partial loss}{\partial x_0} = \frac{\partial loss}{\partial x_n}\left(1 + \frac{\partial}{\partial x_0}\sum_{j=1}^{n} F\left(x_j{-}1, \Theta_j{-}1\right)\right) \tag{4}$$

In the given Equation, loss represents the total loss of the neural network. The presence of "1" signifies that gradients from the network's output can be directly propagated backward to its input. This direct back-propagation reduces the likelihood of gradient vanishing, a common issue when gradients must traverse multiple layers before reaching the input, thus enhancing the network's learning efficiency.

B. DRN STRUCTURE FOR STLF

The DRN for STLF is based on the basic DRN architecture detailed in [72], primarily composed of a fundamental structure and modified ResNet (ResNetPlus). ResNetPlus, an enhanced version of ResNet designed to improve 24-hour load forecasting performance, retains the block structure consistent with ResNet.

To begin with, a neural network featuring densely connected layers, commonly referred to as the 'fundamental structure,' is utilized. This foundational architecture is responsible for generating an initial load forecast for the upcoming 24 hours. The visual depiction of the model employing the fundamental structure is presented in Figure 6.

Secondly, T_h^{month}, T_h^{week} and T_h^{day} are the temperature readings concurrent with L_h^{month}, L_h^{week} and L_h^{day}, respectively. T_h is the actual temperature forecasted for the next day. S, W, and H are one-hot encoded variables representing the season, weekday, and holiday status, respectively. The output from this fundamental structure, denoted as L_h, serves as the input for the second segment of the model.

Within this architecture, every fully connected layer (FC) corresponding to $[L_h^{day}, T_h^{day}]$, $[L_h^{week}, T_h^{week}]$, $[L_h^{month}, T_h^{month}]$ and L_h^{hour} comprises 10 hidden nodes. On the other hand, the fully-connected layers associated with [S, W] are equipped with 5 hidden nodes. Additionally, both fully connected layer 1, FC2, and the fully-connected layer preceding L_h contain 10 hidden nodes. It's worth noting that all layers, except for the output layer, employ an activation function.

The fundamental structure of the model shown in Figure 6 is used for preliminary forecasting of the next 24 hours. In this framework, L_h^{month} denotes the load values for the corresponding hour from the days 1, 2, and 3 months before the predicted day. L_h^{week} signifies the load values for the same hour from 1 to 8 weeks prior, and L_h^{day} corresponds to the loads of the same hour for each day of the preceding week. L_h^{hour} represents the load values for the same hour from the previous 24 hours.

The ResNetPlus model represents an advanced evolution in neural network design, deriving its foundational concept from the classic ResNet architecture but with notable enhancements. This innovative variant integrates residual blocks, each containing two hidden layers with 20 nodes apiece, utilizing identical activation functions. By sequentially arranging four of these blocks, each with its unique connections, and

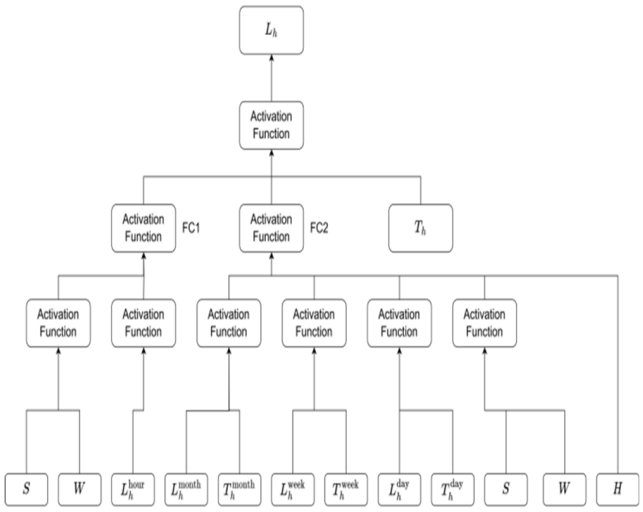

FIGURE 6. The structure of a deep residual network [72].

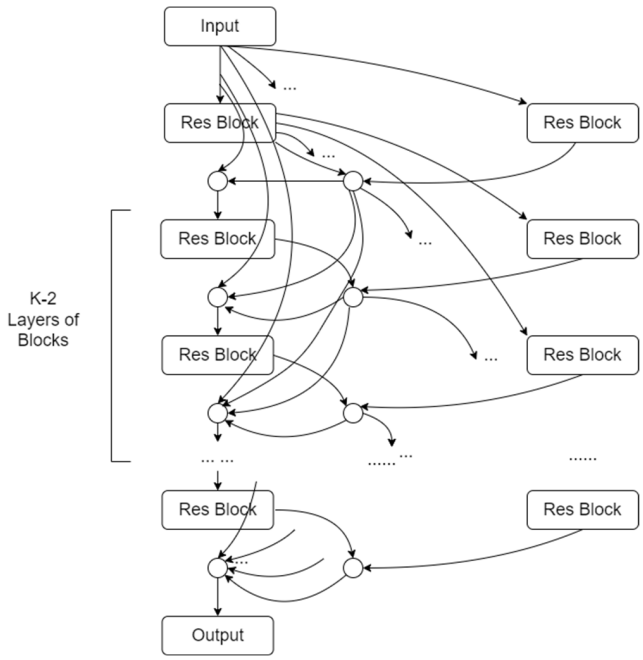

FIGURE 7. A depiction of the modified structure of the ResNetPlus [72].

repeating the process across 10 layers, ResNetPlus achieves significant depth and complexity. The design includes a novel shortcut connection extending from the output of the final block directly to the network's entry point, culminating in the model's output. Such a structure not only simplifies the formation of a deep residual network but also optimizes its efficiency. As depicted in Figure 7, while ResNetPlus preserves the hyperparameters found in its ResNet predecessors within these blocks, it augments the original framework to exploit the full potential of the residual design.

To effectively train the models, the model's loss, denoted as Loss, is defined as the cumulative result of two distinct components Equation (5):

$$\text{Loss} = \text{Loss}_\text{E} + \text{Loss}_\text{R} \tag{5}$$

where Loss_E quantifies the discrepancy in predictions, and Loss_R serves as a penalty term for out-of-range values, designed to expedite the training phase. Particularly, Loss_E is articulated as Equation (6):

$$\text{Loss}_\text{E} = \frac{1}{\text{NumH}} \sum_{\text{j}=1}^{\text{N}} \sum_{\text{h}=1}^{\text{H}} \frac{\left|\hat{\text{y}}_{(\text{j,h})} - \text{y}_{(\text{j,h})}\right|}{\text{y}_{(\text{j,h})}} \tag{6}$$

where $\hat{\text{y}}_{(\text{j,h})}$ represents the model's output and $\text{y}_{(\text{j,h})}$ denotes the actual normalized load for the hth hour of the jth day. Here, Num symbolizes the number of data samples, while H indicates the number of hourly loads within a day (notably, H = 24 in this scenario). This metric, commonly recognized as the MAPE, is employed both as a measure of error and as a criterion for assessing the forecast results of the models. The second term, Loss_R, is computed as Equation (7):

$$\begin{aligned}\text{Loss}_\text{R} = \frac{1}{2\text{Num}} \sum_{\text{j}=1}^{\text{Num}} &\max\left(0, \max_\text{h} \hat{\text{y}}_{(\text{j,h})} - \max_\text{h} \text{y}_{(\text{j,h})}\right) \\ &+ \max\left(0, \min_\text{h} \text{y}_{(\text{j,h})} - \min_\text{h} \hat{\text{y}}_{(\text{j,h})}\right)\end{aligned} \tag{7}$$

This term imposes a penalty on the model if the predicted daily load curves deviate beyond the actual load ranges, thereby hastening the initial phase of the training process. As the model begins to generate forecasts with greater precision, this term increasingly underscores the expense of overestimating the peaks and underestimating the troughs of the load curves.

III. RESEARCH METHODOLOGY

A. RESEARCH DATA

1) DATA PRE-PROCESSING

The datasets collected often present multiple irregularities, including missing values, incomplete information, noise, and raw formats [83]. Unrefined data can harbor flaws and inconsistencies, potentially leading to misunderstandings and undermining the integrity of data analysis. Consequently, the pre-processing stage, an integral part of the data refinement process, is especially critical for real-world datasets. It ensures the system's performance and reliability in extracting meaningful information from actual data scenarios.

Typically, the data pre-processing stage encompasses several essential steps or phases, applied to raw data to enhance its quality before it's considered refined. These steps include: (1) Data Cleaning, (2) Data Transformation, (3) Data Reduction, and (4) Data Discretization. These stages are systematically implemented during the pre-processing phase to polish and assess the data. This approach enables more efficient and accurate predictions. Consequently, depending on the nature of the data, the strategy employed, and the specific input requirements for the proposed methodology, various sub-phases can be effectively utilized to optimize the data for further analysis.

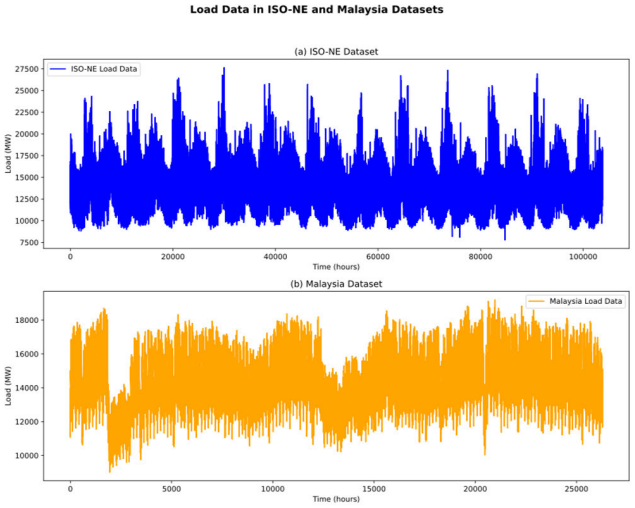

FIGURE 8. Load data in ISO-NE and Malaysia datasets.

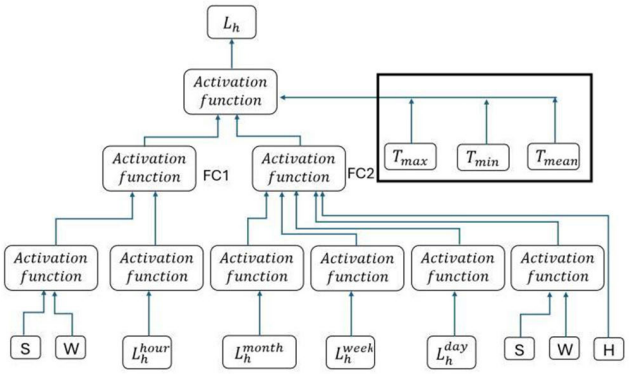

FIGURE 9. Adjusted one-hour input infrastructure.

2) DESCRIPTION OF THE DATA

This study employs two publicly available datasets: the New England's Independent System Operator (ISO-NE) dataset and the Malaysia dataset, which provide diverse perspectives for analyzing load forecasting under different climatic and consumption conditions (data sources are provided in Appendix -A). The ISO-NE dataset includes hourly load and weather data (temperature) recorded from March 2003 to December 2014, reflecting typical consumption patterns in a temperate climate with significant seasonal and annual variations. In contrast, the Malaysia dataset covers hourly load data for the Petaling Jaya area from January 2020 to December 2022, as well as providing information on daily temperatures (mean temperature, minimum temperature, maximum temperature). Representing a tropical climate, the Malaysia dataset shows relatively steady demand patterns with moderate fluctuations, while the ISO-NE dataset exhibits pronounced periodic trends driven by seasonal effects, with load ranging from 7,500 megawatt (MW) to 27,500 MW compared to 10,000 MW to 18,000 MW in the Malaysia dataset. Figure 8 visualizes the load data from both datasets, highlighting their contrasting characteristics and making them suitable for evaluating the performance of load forecasting models across diverse scenarios.

3) DATA FEATURE PROCESSING

In this study, the ISO-NE and Malaysia datasets differ significantly in terms of temporal granularity, requiring different feature processing strategies. For the ISO-NE dataset, hourly load L_h^{hour} and temperature data T_h^{month}, T_h^{week}, T_h^{day} are directly fed into the model. The original fundamental structure extracts load features L_h^{month}, L_h^{week}, L_h^{day} based on different temporal ranges, combined with seasonal S, W, and H information to generate model inputs. S is divided into four seasons: spring, summer, autumn and winter. S is divided into four seasons: spring, summer, autumn and winter, and H includes the United States Independence Day, Christmas, etc.

In contrast, the Malaysia dataset provides only daily temperature data, including T_{mean}, T_{max}, T_{min}, which differs from the hourly temperature data in the ISO-NE dataset. To address this difference, the fundamental structure was adjusted to directly accept daily temperature data as input, and the modified fundamental structure is shown in Figure 9. In the adjusted model, daily temperature data T_{mean}, T_{max}, T_{min} are concatenated as a single feature input, without temporal segmentation. Meanwhile, the processing of load features L_h^{month}, L_h^{week}, L_h^{day} remains unchanged, continuing to extract information from the past 24 hours, 8 weeks, and 3 months. Date-related features, such as S, W, and H, are combined with load and temperature features to form the final input to the model. S is divided into two seasons: rainy season and dry season, and H includes Malaysia Independence Day, Eid al-Fitr, etc.

This adjustment enables the Malaysia dataset's daily temperature features to be utilized directly, avoiding redundancy from expanding daily data into hourly data, while simplifying the preprocessing steps. The adjusted model processes load features and date information consistently for both datasets, with the main difference being in the handling of temperature features based on temporal granularity. This ensures the model's adaptability and generalization across datasets with varying temporal resolutions.

B. EXPERIMENTAL DESIGN

To ensure a controlled and consistent experimental environment, this study adopts a fixed DRN architecture. The primary objective is to evaluate the performance of different activation functions while controlling for architectural variability. By keeping the model structure consistent, the comparative analysis focuses solely on the impact of activation functions, enhancing the clarity and validity of the findings. This study adopts a systematic experimental design to evaluate the performance of eight activation functions in a DRN, specifically in the DRN model. The activation functions selected for this study—Linear [84], Sigmoid [85],

Tanh [86], ReLU [87], Leaky ReLU [88], SELU [89], GELU [90], and Swish [91]—were chosen based on their widespread usage in recent STLF literature, as previously shown in Figure 3. By selecting both classical and modern activation functions, this study ensures a balanced and representative evaluation. The mathematical definitions and formulas for each function are provided in Appendix -B.

These functions were chosen to represent a diverse range of traditional and modern activation functions, enabling a comprehensive comparison of their impact on model performance. The experimental setup includes model architecture, optimizer settings, training process, and data preprocessing. Except for the activation functions and their corresponding initialization mechanisms, all other aspects are consistent with Reference [72]. The main focus of this study is to evaluate the performance of these activation functions in improving predictive precision.

Random weight initialization was applied using the default initialization methods corresponding to each activation function. Linear, Tanh, and Sigmoid activation functions use Glorot Uniform initialization [92], [93]; ReLU and Leaky ReLU activation functions use He Normal initialization [94]; SELU activation function uses LeCun Normal initialization [89]; and GELU and Swish activation functions use Glorot Normal initialization [90], [91].

Snapshot ensemble is a model ensemble technique that periodically saves model weights, referred to as snapshots, at different learning rate phases during training [72], [95]. By averaging the predictions from these snapshots, this method reduces training instability and mitigates the risk of overfitting associated with relying on a single model. In this study, each model underwent 700 training epochs, comprising 600 initial epochs followed by two additional rounds of 50 short-term training epochs. Model weights were saved at the end of each short-term round, resulting in three snapshots per training repetition. This entire training process was repeated five times for each model configuration, producing a total of 15 snapshots. For each individual model, the total training time—including all repetitions and snapshot phases—was observed to remain within eight hours. It is worth noting that exceeding this duration does not adversely affect model performance, as the eight-hour benchmark reflects practical computational efficiency rather than a limiting factor. This snapshot ensemble strategy effectively enhanced prediction stability and reduced overfitting risk across repeated experiments.

The Adam optimizer was selected, and its adaptive learning rate adjustment mechanism was employed. The initial learning rate was set to 0.001, which is the default value for the Adam optimizer and ensures effective model training [96]. This study utilized two datasets: the ISO-NE dataset and the Malaysia dataset. The ISO-NE dataset and the Malaysia dataset represent temperate and tropical regions, respectively, encompassing diverse load patterns and climatic characteristics, providing a variety of scenarios for model performance evaluation. The ISO-NE dataset used the most recent three years of data for model training, while the Malaysia dataset used the most recent two years.

All experiments were conducted in a Python 3.8 environment using Keras 2.10.0 and TensorFlow 2.10.0 as the backend, on a Lenovo laptop equipped with an AMD Ryzen 7 6800H processor and Radeon Graphics.

This experimental design provides a reliable basis for evaluating the performance of activation functions in the DRN framework and lays the groundwork for subsequent research and model optimization.

C. EVALUATION METRICS

When comparing the performance of various DRN models in STLF, different studies have employed several evaluation metrics to measure the predictive precision of the models [72], [73], [74], [75], [76], [77], [78], [79], [80], [81]. These metrics include Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Square Error (MSE), Normalized Mean Square Error (NMSE), Correlation Coefficient (R), and Coefficient of Determination (R^2). The specific evaluation metrics used vary across studies, and their corresponding Equations are presented below, listed in the order of Equation (8) to Equation (14):

$$\mathrm{MAPE} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}}\left|\frac{\mathrm{y_i}-\hat{\mathrm{y}}_\mathrm{i}}{\mathrm{y_i}}\right| \times 100 \tag{8}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{\mathrm{N}}\sum_{\mathrm{i}-1}^{\mathrm{N}}\left(\mathrm{y_i}-\hat{\mathrm{y}}_\mathrm{i}\right)^2} \tag{9}$$

$$\mathrm{MAE} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}-1}^{\mathrm{N}}\left|\mathrm{y_i}-\hat{\mathrm{y}}_\mathrm{i}\right| \tag{10}$$

$$\mathrm{MSE} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}-1}^{\mathrm{N}}\left(\mathrm{y_i}-\hat{\mathrm{y}}_\mathrm{i}\right)^2 \tag{11}$$

$$\mathrm{NMSE} = \frac{\sum_{\mathrm{i}-1}^{\mathrm{N}}\left(\mathrm{y_i}-\hat{\mathrm{y}}_\mathrm{i}\right)^2}{\mathrm{N}\cdot\sigma_\mathrm{y}^2} \tag{12}$$

$$\mathrm{R} = \frac{\sum_{\mathrm{i}-1}^{\mathrm{N}}\left(\mathrm{y_i}-\bar{\mathrm{y}}\right)\left(\hat{\mathrm{y}}_\mathrm{i}-\bar{\hat{\mathrm{y}}}\right)}{\sqrt{\sum_{\mathrm{i}-1}^{\mathrm{N}}\left(\mathrm{y_i}-\bar{\mathrm{y}}\right)^2\sum_{\mathrm{i}-1}^{\mathrm{N}}\left(\hat{\mathrm{y}}_\mathrm{i}-\bar{\hat{\mathrm{y}}}\right)^2}} \tag{13}$$

$$\mathrm{R}^2 = 1-\frac{\sum_{\mathrm{i}-1}^{\mathrm{N}}\left(\mathrm{y_i}-\hat{\mathrm{y}}_\mathrm{i}\right)^2}{\sum_{\mathrm{i}-1}^{\mathrm{N}}\left(\mathrm{y_i}-\bar{\mathrm{y}}\right)^2} \tag{14}$$

The parameters used in these metrics are as follows. N represents the total number of input values used in the calculation, $\mathrm{y_i}$ is the actual value for the i−th sample, and $\hat{\mathrm{y}}_\mathrm{i}$ is the predicted value for the i−th sample. $\bar{\mathrm{y}}$ and $\bar{\hat{\mathrm{y}}}$ are the mean values of all actual and predicted values, respectively, while σ_y^2 denotes the variance of the actual values, which is used to normalize MSE in NMSE. These parameters ensure that the metrics comprehensively evaluate model performance by reflecting prediction precision, error magnitude, and the relationship between actual and predicted values.

In general, the smaller the MAPE, RMSE, MAE, MSE, and NMSE, the better, indicating that the model has a smaller prediction error and stronger generalization ability; and the

TABLE 1. Performance metrics of activation functions on ISO-NE dataset.

Activation Function	MAPE	RMSE	MAE	MSE	NMSE	R	R^2
Linear	0.024130	0.025864	0.015987	0.000669	0.044387	0.977569	0.955613
Tanh	0.014929	0.015898	0.009721	0.000253	0.016077	0.991591	0.983230
Sigmoid	0.025298	0.026031	0.016561	0.000678	0.044962	0.977297	0.955038
ReLU	0.015142	0.015831	0.009710	0.000251	0.016629	0.991668	0.983371
Leaky ReLU	0.014939	0.015620	0.009614	0.000244	0.016190	0.991882	0.983810
SELU	0.014651	0.015790	0.009484	0.000249	0.016543	0.991760	0.983457
GELU	0.013977	0.015190	0.008968	0.000231	0.015310	0.992366	0.984690
Swish	0.013806	0.015444	0.008909	0.000239	0.015826	0.992095	0.984174

closer R and R^2 are to 1, the better, indicating that the model has stronger prediction precision and fitting ability.

IV. RESULTS AND DISCUSSION

A. ISO-NE DATASET

The model was trained on ISO-NE dataset spanning from March 2003 to December 2005, and predictions were generated for the entirety of 2006, with the experimental results summarized in Table 1 and visually presented in Figure 10, showcasing the performance of various activation functions across key metrics. Figure 11 illustrates the comparison between the actual load and the predicted load using the DRN model with Swish as the activation function.

The Swish activation function, when applied in the DRN model for STLF, demonstrated the best performance, achieving MAPE, RMSE, MAE, and MSE values of 0.013806, 0.015444, 0.008909, and 0.000239, respectively, significantly outperforming other activation functions within the same framework. Furthermore, its R and R^2 values reached 0.992095 and 0.984174, respectively, indicating superior prediction precision and fitting capability in the context of STLF. The GELU activation function ranked second, showing excellent performance across most metrics, with a MAPE of 0.013977, slightly higher than Swish but still considerably lower than other activation functions, demonstrating its strong predictive capability and stability within the DRN model.

The Tanh and ReLU activation functions also performed well in the DRN model, with MAPEs of 0.014929 and 0.015142, respectively. These results suggest that they retain certain advantages in capturing nonlinear features but fall short of the performance exhibited by Swish and GELU. Leaky ReLU and SELU delivered strong results as well, albeit slightly inferior to Swish and GELU. In contrast, the DRN model using Sigmoid and Linear activation functions performed poorly, with MAPE values of 0.025298 and 0.02413, respectively, highlighting the limited capacity of these activation functions to model complex load patterns in the STLF task.

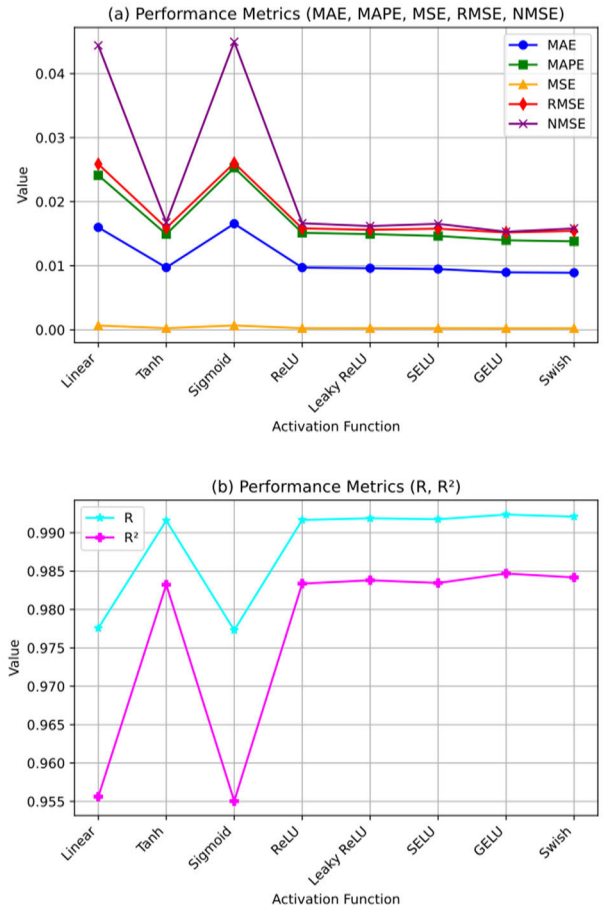

FIGURE 10. Performance Metrics of Activation Functions on ISO-NE Dataset.

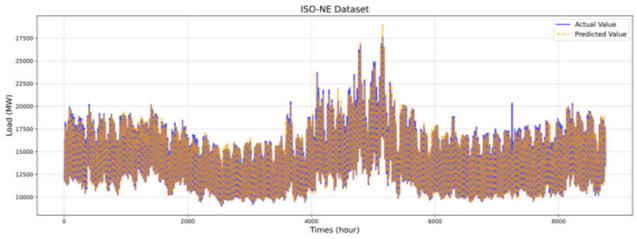

FIGURE 11. The comparison of DRN (Swish) prediction with the actual load on the ISO-NE dataset.

These experimental findings underscore the substantial impact of activation function selection on the performance of DRN models in STLF. Swish and GELU enabled the DRN model to effectively capture the nonlinear characteristics of the data, thereby enhancing its predictive ability. Conversely, traditional activation functions, such as Sigmoid and Linear, were less effective due to their inherent limitations in modeling complex features. This study provides valuable insights for optimizing DRN-based STLF models, suggesting that future research should prioritize refining and exploring the application of advanced activation functions like Swish and GELU to further improve model performance.

TABLE 2. Performance metrics of activation functions on Malaysia dataset.

Activation Function	MAPE	RMSE	MAE	MSE	NMSE	R	R^2
Linear	0.061178	0.053909	0.031909	0.002906	0.102076	0.949055	0.897924
Tanh	0.049809	0.044492	0.025493	0.001980	0.069529	0.965210	0.930471
Sigmoid	0.060240	0.053210	0.031015	0.002831	0.099446	0.949616	0.900554
ReLU	0.058765	0.049890	0.031045	0.002489	0.087423	0.957696	0.912577
Leaky ReLU	0.059134	0.052834	0.031674	0.002791	0.098045	0.952008	0.901955
SELU	0.050429	0.044696	0.026160	0.001998	0.070170	0.965325	0.929830
GELU	0.051454	0.045799	0.026354	0.002098	0.073675	0.963039	0.926325
Swish	0.050570	0.045162	0.025360	0.002040	0.071639	0.963689	0.928361

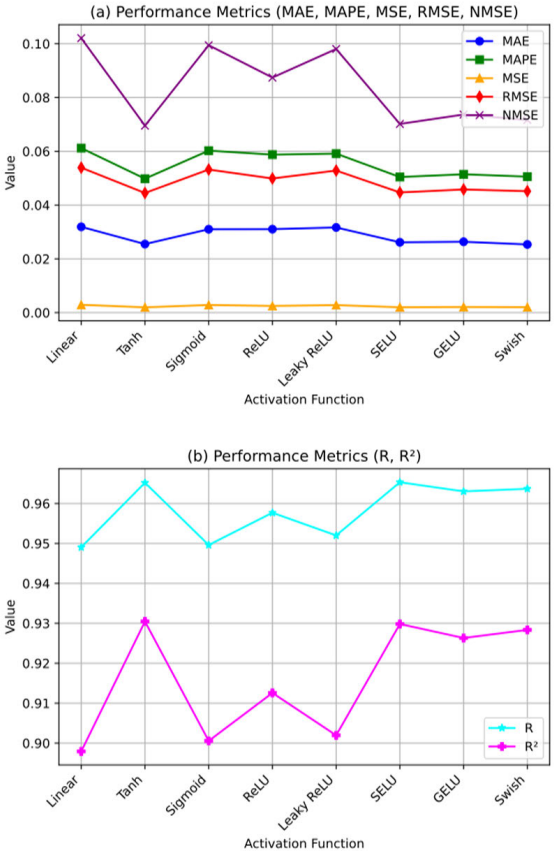

FIGURE 12. Performance Metrics of Activation Functions on Malaysia Dataset.

B. MALAYSIA DATASET

The model is trained on the Malaysia dataset from 2020 to 2021 and used to predict the whole year of 2022. The experimental results on the Malaysia datasetset are summarized in Table 2 and presented visually in Figure 12, showing the performance of various activation functions in terms of key metrics. Figure 13 illustrates the comparison between the actual load and the predicted load using the DRN model with Tanh as the activation function.

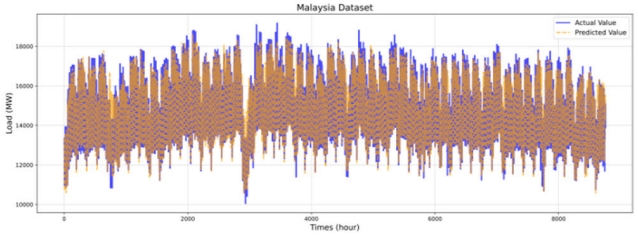

FIGURE 13. The comparison of DRN (Tanh) prediction with the actual load on the Malaysia dataset.

Among all activation functions applied in the DRN model for STLF on the Malaysia dataset, Tanh demonstrated the best performance, achieving a MAPE of 0.049809, RMSE of 0.044492, and R^2 of 0.930471. These results highlight Tanh's strong ability to model the nonlinear characteristics of Malaysian load patterns.

The SELU activation function closely followed, with a MAPE of 0.050429 and RMSE of 0.044696, indicating its ability to stabilize training through its self-normalizing property. The Swish activation function also performed well, achieving a MAPE of 0.050570, showcasing its smooth gradient behavior and flexibility.

While the GELU activation function exhibited competitive performance with a MAPE of 0.051454, it fell short compared to Tanh and SELU. The ReLU and Leaky ReLU activation functions demonstrated moderate performance, with MAPEs of 0.058765 and 0.059134, respectively.

In contrast, the Sigmoid and Linear activation functions performed poorly, with MAPEs of 0.060240 and 0.061178, respectively. These results suggest that their limited nonlinear capacity and saturation effects hindered the model's ability to capture the complex load patterns present in the Malaysia dataset.

Overall, Tanh outperformed other activation functions in modeling the Malaysia dataset, while SELU and Swish also showcased strong predictive capabilities.

C. SUMMARY

The experimental results for the DRN model in STLF on the ISO-NE and Malaysia datasets highlight the significant role of activation function selection in determining model performance. These results emphasize the importance of aligning activation functions with dataset characteristics, such as variability, stability, and complexity of load patterns.

On the ISO-NE dataset, where load patterns exhibit high variability and complex nonlinear relationships, the Swish activation function achieved the best performance with a MAPE of 0.013806 and an R^2 of 0.984174. Swish's smooth and self-gating property allows it to effectively model subtle changes in the data, enabling improved gradient flow during backpropagation and ensuring better convergence and generalization. GELU, ranking second with a MAPE of 0.013977, demonstrated excellent stability and predictive precision, attributed to its probabilistic nature that balances gradient flow and model complexity.

Traditional activation functions such as Tanh and ReLU also performed competitively on the ISO-NE dataset, achieving MAPEs of 0.014929 and 0.015142, respectively. Tanh's symmetric output range effectively handles nonlinear features, while ReLU's computational efficiency ensures robustness. However, their performance was still slightly inferior to advanced activation functions like Swish and GELU.

In contrast, the Sigmoid and Linear activation functions performed poorly, with MAPEs of 0.025298 and 0.024130, respectively. Sigmoid's vanishing gradient problem limits its performance in deeper networks, while Linear's lack of nonlinearity restricts its ability to capture complex load relationships.

On the Malaysia dataset, where load patterns are relatively stable and less volatile, the Tanh activation function emerged as the top performer, achieving a MAPE of 0.049809 and an R^2 of 0.930471. Tanh's ability to model nonlinear transformations effectively aligns with the dataset's characteristics, ensuring strong predictive precision.

The SELU activation function, which ranked second with a MAPE of 0.050429, demonstrated its strength in stabilizing the training process through its self-normalizing property, making it well-suited for deep networks on stable datasets. The Swish activation function also exhibited strong performance, with a MAPE of 0.050570, reflecting its robustness and ability to model subtle patterns.

The GELU activation function, with a MAPE of 0.051454, delivered competitive results but fell slightly short compared to Tanh and SELU. ReLU and Leaky ReLU exhibited moderate performance, with MAPEs of 0.058765 and 0.059134, respectively, showing robustness but lacking the precision offered by advanced activation functions. Once again, Sigmoid and Linear functions recorded the poorest performance, with MAPEs of 0.060240 and 0.061178, respectively, reflecting their limitations in capturing even the relatively simpler patterns of the Malaysia dataset.

These findings demonstrate that the performance of activation functions is highly dataset-dependent. On the ISO-NE dataset, advanced activation functions like Swish and GELU excelled due to their ability to handle high variability and complex features. Conversely, on the Malaysia dataset, the Tanh and SELU activation functions performed best, owing to their suitability for stable, less volatile patterns.

While the findings of this study offer valuable insights into the role of activation functions in DRN-based STLF models, it is important to acknowledge certain experimental constraints. All training processes were conducted on a standard laptop with limited computational capacity, and the total training time for each individual model—including all repetitions and snapshot phases—was observed to remain within eight hours. These practical constraints ensured consistency and reproducibility across experiments but may have limited the exploration of deeper network architectures or more extensive hyperparameter tuning.

In conclusion, Swish and GELU are particularly effective for datasets with high variability, while Tanh and SELU are better suited for stable datasets. The consistently poor performance of Sigmoid and Linear activation functions across both datasets highlights their limitations in STLF tasks. Future research should focus on further optimizing advanced activation functions and exploring their adaptability to various dataset characteristics, ensuring improved precision and robustness in DRN-based STLF models.

V. CONCLUSION

This study systematically evaluated the performance of various activation functions within DRN models specifically designed for STLF, employing the DRN framework on two distinct datasets: ISO-NE and Malaysia. The findings highlight the pivotal role of activation function selection in enhancing the performance of DRN-based STLF models and demonstrate significant variations influenced by the characteristics of each dataset.

The key insights from this study are as follows:

Dataset-Specific Performance: The effectiveness of activation functions depends significantly on dataset characteristics when applied within DRN models. On the ISO-NE dataset, where load patterns are highly variable and nonlinear, the Swish activation function demonstrated the best performance, achieving the lowest MAPE of 0.013806 and the highest correlation metrics ($R = 0.992095$, $R^2 = 0.984174$). This result highlights Swish's superior ability to capture complex and fluctuating demand patterns due to its smooth self-gating property and ability to maintain effective gradient flow. Conversely, on the Malaysia dataset, where load patterns are relatively stable and less volatile, the Tanh activation function performed the best, achieving a MAPE of 0.049809. Its symmetric output range effectively modeled the dataset's temporal consistency. The SELU activation function, with its self-normalizing property, also exhibited strong performance on the Malaysia dataset, achieving a MAPE of 0.050429, RMSE of 0.044696, and R^2 of 0.929830, making it particularly well-suited for maintaining stable activation distributions.

Advantages of Modern Activation Functions: Modern activation functions such as Swish, GELU, and SELU emerged as top performers in this study. Swish excelled on the ISO-NE dataset due to its smooth nonlinear expressive capability, allowing the DRN model to converge efficiently and capture intricate load patterns. GELU, with its probabilistic nature, provided competitive results, ranking second on the ISO-NE dataset with a MAPE of 0.013977, RMSE of 0.015190, and R^2 of 0.984690, demonstrating its ability to balance gradient flow and model stability. On the Malaysia dataset, SELU's self-normalizing properties aligned well with the dataset's steady demand patterns, enabling the DRN model to stabilize activations and achieve robust performance. Swish and GELU also delivered strong results, with MAPEs of 0.050570 and 0.051454, respectively, showcasing their adaptability and generalization ability across diverse data characteristics.

Continued Relevance of Traditional Functions: While modern activation functions delivered superior performance traditional activation functions like Tanh and ReLU exhibited robust and consistent results across both datasets. On the ISO-NE dataset, Tanh and ReLU achieved MAPEs of 0.014929 and 0.015142, respectively, demonstrating their effectiveness in capturing nonlinear features. On the Malaysia dataset, Tanh achieved the lowest error (MAPE = 0.049809), reaffirming its suitability for datasets with relatively stable demand patterns. However, traditional activation functions fell short when compared to modern alternatives like Swish and SELU, which offer enhanced gradient flow, stability, and expressive power. This suggests that while classical functions remain viable options, they may not fully leverage the capabilities of modern DRN models for complex forecasting scenarios.

Guidance for DRN Optimization: The results reinforce the importance of aligning activation function selection with dataset characteristics. For datasets with high variability and complex load patterns, advanced activation functions like Swish and GELU should be prioritized for their ability to model intricate relationships. For datasets with stable and consistent patterns, Tanh and SELU offer reliable and precise solutions. These findings provide a strong foundation for optimizing DRN-based STLF models, emphasizing the need for a data-driven approach to activation function selection.

Future Research Directions: This study highlights the need for further exploration of activation functions tailored to DRN-based STLF and other time-series forecasting tasks. Developing new activation functions with enhanced self-adaptive properties and nonlinear expressive capabilities could further optimize model performance. Expanding the scope to include additional datasets, hybrid DRN models (e.g., integrating BiLSTM layers), and activation function variants will deepen the understanding of activation function impacts and guide the design of more effective forecasting systems. In addition, while the training processes in this study were conducted under practical hardware conditions using a standard laptop, and the total training time for each individual model—including all repetitions and snapshot phases—was observed to remain within eight hours, these constraints may have limited the depth of network exploration and restricted more extensive hyperparameter tuning. Future work could benefit from the use of high-performance computing environments and extended training time to assess the scalability and optimization potential of deeper or more complex DRN architectures. Furthermore, although the evaluation was based on two climatically distinct datasets—ISO-NE and Malaysia—the seasonal bias within datasets (e.g., summer vs. winter) was not explicitly analyzed. Future research may consider conducting seasonal segmentation to investigate whether certain activation functions perform better under specific seasonal load conditions. Lastly, expanding the analysis to include additional datasets with diverse regional, climatic, and consumption characteristics will further validate the applicability and robustness of the findings across broader real-world STLF scenarios. Such extensions would not only strengthen the generalizability of the proposed activation function selection strategy but also improve DRN adaptability across varied energy systems.

In conclusion, this study provides the first systematic evaluation of activation functions in DRN models for STLF, demonstrating that activation function selection plays a crucial role in optimizing predictive performance. The results reveal that Swish and GELU are particularly effective for datasets with high variability and complex load patterns, such as ISO-NE, whereas Tanh and SELU yield better performance on more stable datasets like Malaysia. These findings emphasize the necessity of tailoring activation function choices to the specific characteristics of the dataset to maximize forecasting accuracy. By integrating advanced activation functions that enhance gradient flow and nonlinearity, DRN models can achieve greater precision, robustness, and adaptability across diverse load forecasting scenarios. The findings of this study offer clear suggestion on selecting activation functions based on dataset characteristics. For datasets with high variability and complex load patterns, such as ISO-NE, activation functions like Swish and GELU are recommended due to their ability to enhance gradient flow and capture nonlinear dependencies effectively. Conversely, for datasets with more stable load patterns, such as Malaysia, Tanh and SELU are preferable as they provide improved stability and convergence. These insights serve as a reference for model developers aiming to optimize DRN-based STLF models in different power system contexts.

ABBREVIATIONS

The following abbreviations are used in this manuscript:

Abbreviation	Full name
ANN	Artificial Neural Network.
CNN	Convolutional Neural Network.
DNN	Deep Neural Network.
DRN	Deep Residual Network.
ELM	Extreme Learning Machine.
FC	Fully Connected Layer.
GELU	Gaussian Error Linear Unit.
GRU	Gated Recurrent Unit.
ISO-NE	Independent System Operator of New England.
LF	Load Forecasting.
LSTM	Long Short-Term Memory.
LTLF	Long-Term Load Forecasting.
MAE	Mean Absolute Error.
MAPE	Mean Absolute Percentage Error.
MSE	Mean Square Error.
MTLF	Medium-Term Load Forecasting.
MW	Megawatt.
NMSE	Normalized Mean Square Error.
PCA	Principal Component Analysis.
R	Correlation Coefficient.
R^2	Coefficient of Determination.

RBF	Radial Basis Function.
ReLU	Rectified Linear Unit.
ResBlock	Residual Block.
ResNet	Residual Network.
ResNetPlus	Modified Residual Network.
RMSE	Root Mean Square Error.
RNN	Recurrent Neural Network.
SELU	Scaled Exponential Linear Unit.
STLF	Short-Term Load Forecasting.
SVR	Support Vector Regression.
Tanh	Hyperbolic Tangent.
VSTLF	Very Short-Term Load Forecasting

A. DATASET SOURCE

1) ISO-NE DATASET

https://www.iso-ne.com/isoexpress/web/reports/load-and-demand

2) MALAYSIA DATASET

https://www.gso.org.my/SystemData/SystemDemand.aspx

B. ACTIVATION FUNCTION FORMULA

1) LINEAR

The Linear activation function is mathematically defined in Equation (15):

$$f(x) = x \tag{15}$$

where x is the input value. This function outputs the input directly, making it ideal for regression tasks where the relationship between variables is linear. However, it lacks nonlinearity, which limits its ability to capture complex patterns in data, making it unsuitable for deep networks.

2) SIGMOID

The Sigmoid function is defined in Equation (16):

$$f(x) = \frac{1}{1 + e^{-x}} \tag{16}$$

where x is the input value. It maps the input to the range (0, 1), making it useful for binary classification tasks. The main advantage of Sigmoid is its probabilistic output. However, it suffers from gradient vanishing for large input magnitudes and outputs that are not zerocentered, complicating optimization.

3) TANH

The Tanh function is described by Equation (17):

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{17}$$

where x is the input value. It maps inputs to the range $[-1, 1]$, providing zero-centered outputs that improve convergence during training. However, like Sigmoid, Tanh is prone to gradient vanishing for large input magnitudes, limiting its effectiveness in deeper networks.

4) RELU

The ReLU is defined as in Equation (18):

$$f(x) = \max(0, x) \tag{18}$$

where x is the input value. ReLU is computationally efficient and effectively mitigates the vanishing gradient problem, making it suitable for training deep networks. Its limitation lies in the "dying neurons" problem, where negative inputs result in zero output, rendering some neurons inactive during training.

5) LEAKY RELU

Leaky ReLU modifies ReLU to allow small gradients for negative inputs, as shown in Equation (19):

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{if } x \leq 0 \end{cases} \tag{19}$$

where x is the input value and α is a small constant (e.g., 0.01). This prevents neurons from becoming inactive. However, it introduces an additional hyperparameter α that requires tuning.

6) SELU

The SELU is expressed in Equation (20):

$$f(x) = \begin{cases} \lambda x & \text{if } x > 0 \\ \lambda\alpha\left(e^{x} - 1\right) & \text{if } x \leq 0 \end{cases} \tag{20}$$

where x is the input value, $\lambda \approx 1.05$ is a scaling factor for normalization, and $\alpha \approx 1.67$ adjusts the output for negative inputs. SELU ensures self-normalization, stabilizing mean and variance across layers. However, its reliance on specific initialization and dropout configurations can complicate implementation.

7) GELU

The GELU is defined in Equation (21):

$$f(x) = x \cdot \Phi(x) \tag{21}$$

where x is the input value, and $\Phi(x)$ represents the cumulative distribution function of the standard normal distribution. The cumulative distribution function is further expressed as Equation (22):

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2}} dt \tag{22}$$

Here, x is the input to the activation function, and $\Phi(x)$ determines the probability that a random variable with a standard normal distribution is less than or equal to x. The integral uses t as a dummy variable, while e and π are fundamental mathematical constants, representing Euler's number (≈ 2.718) and the ratio of a circle's circumference to its diameter (≈ 3.1416), respectively.

GELU smoothly combines linear and nonlinear behavior by weighting the input x with the probability derived from the standard normal distribution. This design facilitates smooth transitions between activated and non-activated

states, enhancing gradient flow and convergence. Although GELU's computation involves the Gaussian cumulative distribution function, which is computationally more complex than ReLU, its benefits in training stability and precision make it a preferred choice in many deep learning tasks.

8) SWISH

The Swish function uses a self-gating mechanism, as defined in Equation (23):

$$f(x) = x \cdot \sigma(x), \sigma(x) = \frac{1}{1 + e^{-x}} \tag{23}$$

where x is the input value and $\sigma(x)$ is the Sigmoid function. Swish improves gradient flow and learning dynamics but involves slightly higher computational cost due to the Sigmoid calculation.

ACKNOWLEDGMENT

AUTHOR CONTRIBUTIONS

Conceptualization: Junchen Liu and Faisul Arif Ahmad; Methodology: Junchen Liu; Investigation: Junchen Liu and Faisul Arif Ahmad; Writing—Original Draft Preparation: Junchen Liu; Writing—Review and Editing: Junchen Liu, Faisul Arif Ahmad, Khairulmizam Samsudin, Fazirulhisyam Hashim, and Mohd Zainal Abidin Ab Kadir; and Supervision: Faisul Arif Ahmad, Khairulmizam Samsudin, Fazirulhisyam Hashim, and Mohd Zainal Abidin Ab Kadir. All authors have read and agreed to the published version of the manuscript.

REFERENCES

[1] E. Ceperic, V. Ceperic, and A. Baric, "A strategy for short-term load forecasting by support vector regression machines," *IEEE Trans. Power Syst.*, vol. 28, no. 4, pp. 4356–4364, Nov. 2013, doi: 10.1109/TPWRS.2013.2269803.

[2] M. Akhtaruzzaman, M. K. Hasan, S. R. Kabir, S. N. H. S. Abdullah, M. J. Sadeq, and E. Hossain, "HSIC bottleneck based distributed deep learning model for load forecasting in smart grid with a comprehensive survey," *IEEE Access*, vol. 8, pp. 222977–223008, 2020, doi: 10.1109/ACCESS.2020.3040083.

[3] H. Zheng, J. Yuan, and L. Chen, "Short-term load forecasting using EMD-LSTM neural networks with a XGBoost algorithm for feature importance evaluation," *Energies*, vol. 10, no. 8, p. 1168, Aug. 2017, doi: 10.3390/en10081168.

[4] V. Y. Kondaiah, B. Saravanan, P. Sanjeevikumar, and B. Khan, "A review on short-term load forecasting models for micro-grid application," *J. Eng.*, vol. 2022, no. 7, pp. 665–689, Jul. 2022, doi: 10.1049/tje2.12151.

[5] S. R. U. Hassnain and A. Khan, "Short term load forecasting using particle swarm optimization based ANN approach," in *Proc. Int. Joint Conf. Neural Netw.*, Aug. 2007, pp. 1476–1481, doi: 10.1109/ijcnn.2007.4371176.

[6] S. Ruzic, A. Vuckovic, and N. Nikolic, "Weather sensitive method for short term load forecasting in electric power utility of Serbia," *IEEE Trans. Power Syst.*, vol. 18, no. 4, pp. 1581–1586, Nov. 2003, doi: 10.1109/TPWRS.2003.811172.

[7] K.-B. Song, Y.-S. Baek, D. H. Hong, and G. Jang, "Short-term load forecasting for the holidays using fuzzy linear regression method," *IEEE Trans. Power Syst.*, vol. 20, no. 1, pp. 96–101, Feb. 2005, doi: 10.1109/TPWRS.2004.835632.

[8] W. Charytoniuk, M. S. Chen, and P. Van Olinda, "Nonparametric regression based short-term load forecasting," *IEEE Trans. Power Syst.*, vol. 13, no. 3, pp. 725–730, Mar. 1998, doi: 10.1109/59.708572.

[9] J. W. Taylor, "Short-term electricity demand forecasting using double seasonal exponential smoothing," *J. Oper. Res. Soc.*, vol. 54, no. 8, pp. 799–805, Aug. 2003, doi: 10.1057/palgrave.jors.2601589.

[10] J. W. Taylor, "Triple seasonal methods for short-term electricity demand forecasting," *Eur. J. Oper. Res.*, vol. 204, no. 1, pp. 139–152, Jul. 2010, doi: 10.1016/j.ejor.2009.10.003.

[11] E. E. Elattar, J. Goulermas, and Q. H. Wu, "Electric load forecasting based on locally weighted support vector regression," *IEEE Trans. Syst., Man, Cybern., C, Appl. Rev.*, vol. 40, no. 4, pp. 438–447, Jul. 2010, doi: 10.1109/TSMCC.2010.2040176.

[12] G. Zhang and J. Guo, "A novel method for hourly electricity demand forecasting," *IEEE Trans. Power Syst.*, vol. 35, no. 2, pp. 1351–1363, Mar. 2020, doi: 10.1109/TPWRS.2019.2941277.

[13] D. Alberg and M. Last, "Short-term load forecasting in smart meters with sliding window-based ARIMA algorithms," *Vietnam J. Comput. Sci.*, vol. 5, nos. 3–4, pp. 241–249, Sep. 2018, doi: 10.1007/s40595-018-0119-7.

[14] M. Rejc and M. Pantos, "Short-term transmission-loss forecast for the Slovenian transmission power system based on a fuzzy-logic decision approach," *IEEE Trans. Power Syst.*, vol. 26, no. 3, pp. 1511–1521, Aug. 2011, doi: 10.1109/TPWRS.2010.2096829.

[15] M. Ali, M. Adnan, M. Tariq, and H. V. Poor, "Load forecasting through estimated parametrized based fuzzy inference system in smart grids," *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 1, pp. 156–165, Jan. 2021, doi: 10.1109/TFUZZ.2020.2986982.

[16] H. S. Hippert, C. E. Pedreira, and R. C. Souza, "Neural networks for short-term load forecasting: A review and evaluation," *IEEE Trans. Power Syst.*, vol. 16, no. 1, pp. 44–55, Jan. 2001, doi: 10.1109/59.910780.

[17] C. Kuster, Y. Rezgui, and M. Mourshed, "Electrical load forecasting models: A critical systematic review," *Sustain. Cities Soc.*, vol. 35, pp. 257–270, Nov. 2017, doi: 10.1016/j.scs.2017.08.009.

[18] L. Hernández, C. Baladrón, J. M. Aguiar, B. Carro, A. Sánchez-Esguevillas, and J. Lloret, "Artificial neural networks for short-term load forecasting in microgrids environment," *Energy*, vol. 75, pp. 252–264, Oct. 2014, doi: 10.1016/j.energy.2014.07.065.

[19] L. Clark, D. Lou, D. Michelle, G. T. Alegata, and C. Gabrielle, "Day-ahead load forecasting using support vector regression machines," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 3, pp. 1–6, 2018, doi: 10.14569/ijacsa.2018.090305.

[20] C. Cecati, J. Kolbusz, P. Rózycki, P. Siano, and B. M. Wilamowski, "A novel RBF training algorithm for short-term electric load forecasting and comparative studies," *IEEE Trans. Ind. Electron.*, vol. 62, no. 10, pp. 6519–6529, Oct. 2015, doi: 10.1109/TIE.2015.2424399.

[21] Y. Chen, P. B. Luh, C. Guan, Y. Zhao, L. D. Michel, M. A. Coolbeth, P. B. Friedland, and S. J. Rourke, "Short-term load forecasting: Similar day-based wavelet neural networks," *IEEE Trans. Power Syst.*, vol. 25, no. 1, pp. 322–330, Feb. 2010, doi: 10.1109/TPWRS.2009.2030426.

[22] Y. Zhao, P. B. Luh, C. Bomgardner, and G. H. Beerel, "Short-term load forecasting: Multi-level wavelet neural networks with holiday corrections," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Jul. 2009, pp. 1–7, doi: 10.1109/PES.2009.5275304.

[23] I. Goodfellow, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[24] Y. Eren and İ. Küçükdemiral, "A comprehensive review on deep learning approaches for short-term load forecasting," *Renew. Sustain. Energy Rev.*, vol. 189, Jan. 2024, Art. no. 114031, doi: 10.1016/j.rser.2023.114031.

[25] S. Fan and R. J. Hyndman, "Short-term load forecasting based on a semi-parametric additive model," *IEEE Trans. Power Syst.*, vol. 27, no. 1, pp. 134–141, Feb. 2012, doi: 10.1109/TPWRS.2011.2162082.

[26] L. Li, K. Ota, and M. Dong, "Everything is image: CNN-based short-term electrical load forecasting for smart grid," in *Proc. 14th Int. Symp. Pervasive Syst., Algorithms Netw. 11th Int. Conf. Frontier Comput. Sci. Technol. 3rd Int. Symp. Creative Comput. (ISPAN-FCST-ISCC)*, Jun. 2017, pp. 344–351, doi: 10.1109/ISPAN-FCST-ISCC.2017.78.

[27] X. Dong, L. Qian, and L. Huang, "Short-term load forecasting in smart grid: A combined CNN and K-means clustering approach," in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Feb. 2017, pp. 119–125, doi: 10.1109/BIGCOMP.2017.7881726.

[28] Z. Deng, B. Wang, Y. Xu, T. Xu, C. Liu, and Z. Zhu, "Multi-scale convolutional neural network with time-cognition for multi-step short-term load forecasting," *IEEE Access*, vol. 7, pp. 88058–88071, 2019, doi: 10.1109/ACCESS.2019.2926137.

[29] W. He, "Load forecasting via deep neural networks," *Proc. Comput. Sci.*, vol. 122, pp. 308–314, Jun. 2017, doi: 10.1016/j.procs.2017.11.374.

[30] L. Han, Y. Peng, Y. Li, B. Yong, Q. Zhou, and L. Shu, "Enhanced deep networks for short-term and medium-term load forecasting," *IEEE Access*, vol. 7, pp. 4045–4055, 2019, doi: 10.1109/ACCESS.2018.2888978.

[31] C. Tian, J. Ma, C. Zhang, and P. Zhan, "A deep neural network model for short-term load forecast based on long short-term memory network and convolutional neural network," *Energies*, vol. 11, no. 12, p. 3493, Dec. 2018, doi: 10.3390/en11123493.
[32] M. Cai, M. Pipattanasomporn, and S. Rahman, "Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques," *Appl. Energy*, vol. 236, pp. 1078–1088, Feb. 2019, doi: 10.1016/j.apenergy.2018.12.042.
[33] Z. Kong, C. Zhang, H. Lv, F. Xiong, and Z. Fu, "Multimodal feature extraction and fusion deep neural networks for short-term load forecasting," *IEEE Access*, vol. 8, pp. 185373–185383, 2020, doi: 10.1109/ACCESS.2020.3029828.
[34] Y. Shi, S. Wang, L. Zhang, F. Yang, and X. Luo, "A novel short-term power-load forecasting method based on high-dimensional meteorological data dimensionality reduction and hybrid deep neural network," *J. Energy Eng.*, vol. 149, no. 6, Dec. 2023, Art. no. 04023049, doi: 10.1061/jleed9.eyeng-5009.
[35] M. Jurado, M. Samper, and R. Rosés, "An improved encoder–decoder-based CNN model for probabilistic short-term load and PV forecasting," *Electr. Power Syst. Res.*, vol. 217, Apr. 2023, Art. no. 109153, doi: 10.1016/j.epsr.2023.109153.
[36] A. Baul, G. C. Sarker, P. Sikder, U. Mozumder, and A. Abdelgawad, "Data-driven short-term load forecasting for multiple locations: An integrated approach," *Big Data Cognit. Comput.*, vol. 8, no. 2, p. 12, Jan. 2024, doi: 10.3390/bdcc8020012.
[37] Ö. F. Ertugrul, "Forecasting electricity load by a novel recurrent extreme learning machines approach," *Int. J. Electr. Power Energy Syst.*, vol. 78, pp. 429–435, Jun. 2016, doi: 10.1016/j.ijepes.2015.12.006.
[38] A. Narayan and K. W. Hipel, "Long short term memory networks for short-term electric load forecasting," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2017, pp. 2573–2578, doi: 10.1109/SMC.2017.8123012.
[39] D. L. Marino, K. Amarasinghe, and M. Manic, "Building energy load forecasting using deep neural networks," in *Proc. 42nd Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Oct. 2016, pp. 7046–7051, doi: 10.1109/IECON.2016.7793413.
[40] D. Gan, Y. Wang, N. Zhang, and W. Zhu, "Enhancing short-term probabilistic residential load forecasting with quantile long–short-term memory," *J. Eng.*, vol. 2017, no. 14, pp. 2622–2627, Jan. 2017, doi: 10.1049/joe.2017.0833.
[41] Y. Wang, D. Gan, M. Sun, N. Zhang, Z. Lu, and C. Kang, "Probabilistic individual load forecasting using pinball loss guided LSTM," *Appl. Energy*, vol. 235, pp. 10–20, Feb. 2019, doi: 10.1016/j.apenergy.2018.10.078.
[42] P. Bento, J. Pombo, S. J. P. S. Mariano, and M. R. A. Calado, "Short-term load forecasting using optimized LSTM networks via improved bat algorithm," in *Proc. Int. Conf. Intell. Syst. (IS)*, Sep. 2018, pp. 351–357, doi: 10.1109/IS.2018.8710498.
[43] W. Kong, Z. Y. Dong, D. J. Hill, F. Luo, and Y. Xu, "Short-term residential load forecasting based on resident behaviour learning," *IEEE Trans. Power Syst.*, vol. 33, no. 1, pp. 1087–1088, Jan. 2018, doi: 10.1109/TPWRS.2017.2688178.
[44] S. Muzaffar and A. Afshari, "Short-term load forecasts using LSTM networks," *Energy Proc.*, vol. 158, pp. 2922–2927, Feb. 2019, doi: 10.1016/j.egypro.2019.01.952.
[45] S. Wang, X. Wang, S. Wang, and D. Wang, "Bi-directional long short-term memory method based on attention mechanism and rolling update for short-term load forecasting," *Int. J. Electr. Power Energy Syst.*, vol. 109, pp. 470–479, Jul. 2019, doi: 10.1016/j.ijepes.2019.02.022.
[46] W. Wu, W. Liao, J. Miao, and G. Du, "Using gated recurrent unit network to forecast short-term load considering impact of electricity price," *Energy Proc.*, vol. 158, pp. 3369–3374, Feb. 2019, doi: 10.1016/j.egypro.2019.01.950.
[47] X. Tang, Y. Dai, Q. Liu, X. Dang, and J. Xu, "Application of bidirectional recurrent neural network combined with deep belief network in short-term load forecasting," *IEEE Access*, vol. 7, pp. 160660–160670, 2019, doi: 10.1109/ACCESS.2019.2950957.
[48] B.-S. Kwon, R.-J. Park, and K.-B. Song, "Short-term load forecasting based on deep neural networks using LSTM layer," *J. Electr. Eng. Technol.*, vol. 15, no. 4, pp. 1501–1509, Jul. 2020, doi: 10.1007/s42835-020-00424-7.
[49] C. Cai, Y. Tao, T. Zhu, and Z. Deng, "Short-term load forecasting based on deep learning bidirectional LSTM neural network," *Appl. Sci.*, vol. 11, no. 17, p. 8129, Sep. 2021, doi: 10.3390/app11178129.
[50] S. Atef, K. Nakata, and A. B. Eltawil, "A deep bi-directional long-short term memory neural network-based methodology to enhance short-term electricity load forecasting for residential applications," *Comput. Ind. Eng.*, vol. 170, Aug. 2022, Art. no. 108364, doi: 10.1016/j.cie.2022.108364.
[51] A. Haque and S. Rahman, "Short-term electrical load forecasting through heuristic configuration of regularized deep neural network," *Appl. Soft Comput.*, vol. 122, Jun. 2022, Art. no. 108877, doi: 10.1016/j.asoc.2022.108877.
[52] Q. Liu, J. Cao, J. Zhang, Y. Zhong, T. Ba, and Y. Zhang, "Short-term power load forecasting in FGSM-Bi-LSTM networks based on empirical wavelet transform," *IEEE Access*, vol. 11, pp. 105057–105068, 2023, doi: 10.1109/access.2023.3316516.
[53] M. Liu, Y. Li, J. Hu, X. Wu, S. Deng, and H. Li, "A new hybrid model based on SCINet and LSTM for short-term power load forecasting," *Energies*, vol. 17, no. 1, p. 95, Dec. 2023, doi: 10.3390/en17010095.
[54] Z. Zhao, C. Xia, L. Chi, X. Chang, W. Li, T. Yang, and A. Y. Zomaya, "Short-term load forecasting based on the transformer model," *Information*, vol. 12, no. 12, p. 516, Dec. 2021, doi: 10.3390/info12120516.
[55] S. Huang, J. Zhang, Y. He, X. Fu, L. Fan, G. Yao, and Y. Wen, "Short-term load forecasting based on the CEEMDAN-sample entropy-BPNN-transformer," *Energies*, vol. 15, no. 10, p. 3659, May 2022, doi: 10.3390/en15103659.
[56] V. A. N. Valencia and J. E. Sanchez-Galan, "Use of attention-based neural networks to short-term load forecasting in the republic of Panama," in *Proc. IEEE 40th Central Amer. Panama Conv. (CONCAPAN)*, Aug. 2022, pp. 1–6, doi: 10.1109/CONCAPAN48024.2022.9997752.
[57] B. Jiang, Y. Liu, H. Geng, H. Zeng, and J. Ding, "A transformer-based method with wide attention range for enhanced short-term load forecasting," in *Proc. 4th Int. Conf. Smart Power Internet Energy Syst. (SPIES)*, May 2022, pp. 1684–1690, doi: 10.1109/SPIES55999.2022.10082249.
[58] M. López Santos, S. Díaz García, X. García-Santiago, A. Ogando-Martínez, F. Echevarría Camarero, G. Blázquez Gil, and P. Carrasco Ortega, "Deep learning and transfer learning techniques applied to short-term load forecasting of data-poor buildings in local energy communities," *Energy Buildings*, vol. 292, Aug. 2023, Art. no. 113164, doi: 10.1016/j.enbuild.2023.113164.
[59] S. Li, W. Zhang, and P. Wang, "TS2ARCformer: A multi-dimensional time series forecasting framework for short-term load prediction," *Energies*, vol. 16, no. 15, p. 5825, Aug. 2023, doi: 10.3390/en16155825.
[60] A. Upadhyay, D. Garg, and M. Singh, "Short term load forecasting for smart grids using apache spark and a modified transformer model," *Comput. Informat.*, vol. 42, no. 1, pp. 75–97, 2023, doi: 10.31577/cai_2023_1_75.
[61] G. Zhang, C. Wei, C. Jing, and Y. Wang, "Short-term electrical load forecasting based on time augmented transformer," *Int. J. Comput. Intell. Syst.*, vol. 15, no. 1, p. 67, Aug. 2022, doi: 10.1007/s44196-022-00128-y.
[62] H. Liao and K. K. Radhakrishnan, "Short-term load forecasting with temporal fusion transformers for power distribution networks," in *Proc. IEEE Sustain. Power Energy Conf. (iSPEC)*, Dec. 2022, pp. 1–5, doi: 10.1109/iSPEC54162.2022.10033079.
[63] E. Giacomazzi, F. Haag, and K. Hopf, "Short-term electricity load forecasting using the temporal fusion transformer: Effect of grid hierarchies and data sources," in *Proc. 14th ACM Int. Conf. Future Energy Syst.*, Jun. 2023, pp. 353–360, doi: 10.1145/3575813.3597345.
[64] Q. Zhang, J. Chen, G. Xiao, S. He, and K. Deng, "TransformGraph: A novel short-term electricity net load forecasting model," *Energy Rep.*, vol. 9, pp. 2705–2717, Dec. 2023, doi: 10.1016/j.egyr.2023.01.050.
[65] P. Ran, K. Dong, X. Liu, and J. Wang, "Short-term load forecasting based on CEEMDAN and transformer," *Electric Power Syst. Res.*, vol. 214, Jan. 2023, Art. no. 108885, doi: 10.1016/j.epsr.2022.108885.
[66] P. C. Huy, N. Q. Minh, N. D. Tien, and T. T. Q. Anh, "Short-term electricity load forecasting based on temporal fusion transformer model," *IEEE Access*, vol. 10, pp. 106296–106304, 2022, doi: 10.1109/ACCESS.2022.3211941.
[67] C. Xu and G. Chen, "Interpretable transformer-based model for probabilistic short-term forecasting of residential net load," *Int. J. Electr. Power Energy Syst.*, vol. 155, Jan. 2024, Art. no. 109515, doi: 10.1016/j.ijepes.2023.109515.
[68] S. Cen and C. G. Lim, "Multi-task learning of the PatchTCN-TST model for short-term multi-load energy forecasting considering indoor environments in a smart building," *IEEE Access*, vol. 12, pp. 19553–19568, 2024, doi: 10.1109/ACCESS.2024.3355448.

[69] W. Lu and X. Chen, "Short-term load forecasting for power systems with high-penetration renewables based on multivariate data slicing transformer neural network," *Frontiers Energy Res.*, vol. 12, Jan. 2024, Art. no. 1355222, doi: 10.3389/fenrg.2024.1355222.

[70] Z. Lin, L. Xie, and S. Zhang, "A compound framework for short-term gas load forecasting combining time-enhanced perception transformer and two-stage feature extraction," *Energy*, vol. 298, Jul. 2024, Art. no. 131365, doi: 10.1016/j.energy.2024.131365.

[71] B. Fang, L. Xu, Y. Luo, Z. Luo, and W. Li, "A method for short-term electric load forecasting based on the FMLP-iTransformer model," *Energy Rep.*, vol. 12, pp. 3405–3411, Dec. 2024, doi: 10.1016/j.egyr.2024.09.023.

[72] K. Chen, K. Chen, Q. Wang, Z. He, J. Hu, and J. He, "Short-term load forecasting with deep residual networks," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 3943–3952, Jul. 2019, doi: 10.1109/TSG.2018.2844307.

[73] Q. Xu, X. Yang, and X. Huang, "Ensemble residual networks for short-term load forecasting," *IEEE Access*, vol. 8, pp. 64750–64759, 2020, doi: 10.1109/ACCESS.2020.2984722.

[74] Y. Tian, S. Yu, M. Wen, K. Zhang, and Y. Chen, "Short-term load forecasting scheme based on improved deep residual network and LSTM," in *Proc. CIRED Berlin Workshop (CIRED)*, Sep. 2020, pp. 117–120, doi: 10.1049/OAP-CIRED.2021.0257.

[75] V. Y. Kondaiah and B. Saravanan, "Short-term load forecasting with deep learning," in *Proc. Innov. Power Adv. Comput. Technol. (i-PACT)*, Nov. 2021, pp. 1–5, doi: 10.1109/i-PACT52855.2021.9696634.

[76] W. Chen, G. Han, H. Zhu, L. Liao, and W. Zhao, "Deep ResNet-based ensemble model for short-term load forecasting in protection system of smart grid," *Sustainability*, vol. 14, no. 24, p. 16894, Dec. 2022, doi: 10.3390/su142416894.

[77] V. Y. Kondaiah and B. Saravanan, "A modified deep residual network for short-term load forecasting," *Frontiers Energy Res.*, vol. 10, Oct. 2022, Art. no. 1038819, doi: 10.3389/fenrg.2022.1038819.

[78] Z. Sheng, Z. An, H. Wang, G. Chen, and K. Tian, "Residual LSTM based short-term load forecasting," *Appl. Soft Comput.*, vol. 144, Sep. 2023, Art. no. 110461, doi: 10.1016/j.asoc.2023.110461.

[79] A. Ding, T. Liu, and X. Zou, "Integration of ensemble GoogLeNet and modified deep residual networks for short-term load forecasting," *Electronics*, vol. 10, no. 20, p. 2455, Oct. 2021, doi: 10.3390/electronics10202455.

[80] H. Li, P. Zhang, and C. Li, "Short-term load forecasting for distribution substations based on residual neutral networks and long short-term memory neutral networks with attention mechanism," *J. Phys., Conf. Ser.*, vol. 2030, no. 1, Sep. 2021, Art. no. 012087, doi: 10.1088/1742-6596/2030/1/012087.

[81] Z. Sheng, H. Wang, G. Chen, B. Zhou, and J. Sun, "Convolutional residual network to short-term load forecasting," *Appl. Intell.*, vol. 51, no. 4, pp. 2485–2499, Apr. 2021, doi: 10.1007/s10489-020-01932-9.

[82] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[83] H. Shi, M. Xu, and R. Li, "Deep learning for household load forecasting—A novel pooling deep RNN," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 5271–5280, Sep. 2018, doi: 10.1109/TSG.2017.2686012.

[84] S. Sharma, S. Sharma, and A. Athaiya, "Activation functions in neural networks," *Towards Data Sci.*, vol. 6, no. 12, pp. 310–316, 2017, doi: 10.33564/ijeast.2020.v04i12.054.

[85] B. L. Kalman and S. C. Kwasny, "Why tanh: Choosing a sigmoidal function," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 4, Apr. 1992, pp. 578–581, doi: 10.1109/IJCNN.1992.227257.

[86] K.-I. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Netw.*, vol. 2, no. 3, pp. 183–192, Jan. 1989.

[87] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML-10)*, Jun. 2010, pp. 807–814.

[88] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, 2013, vol. 30, no. 1, p. 3.

[89] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, Jan. 2017, pp. 1–11.

[90] D. Hendrycks and K. Gimpel, "Gaussian error linear units (GELUs)," 2016, *arXiv:1606.08415*.

[91] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," 2017, *arXiv:1710.05941*.

[92] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, Mar. 2010, pp. 249–256.

[93] Y. LeCun, L. Bottou, G. B. Orr, and K. R. Müller, "Efficient backprop," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 2002, pp. 9–50, doi: 10.1007/3-540-49430-8_2.

[94] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2015, pp. 1026–1034, doi: 10.1109/ICCV.2015.123.

[95] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger, "Snapshot ensembles: Train 1, get M for free," 2017, *arXiv:1704.00109*.

[96] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.