scientific reports

Check for updates

OPEN

A novel parametric scaled exponential linear unit activation function for deep residual networks in short-term load forecasting

Junchen Liu[1], Faisul Arif Ahmad[1]✉, Khairulmizam Samsudin[1], Fazirulhisyam Hashim[1] & Mohd Zainal Abidin Ab Kadir[2]

Short-Term Load Forecasting (STLF) is essential for ensuring the stability and operational efficiency of modern power systems. Deep Residual Networks (DRNs) have recently demonstrated promising performance in this domain, enabling more effective training of deeper architectures. However, existing activation functions such as the Scaled Exponential Linear Unit (SELU) rely on fixed parameters and strict initialization, which may limit their adaptability to seasonally varying load–weather conditions. This study introduces a modified activation function, the Parametric Scaled Exponential Linear Unit (PSELU), which incorporates a small tunable parameter γ in the negative region, extending the formulation of SELU while preserving its self-normalizing characteristics. Experiments conducted within the DRN framework on two benchmark datasets—ISO-NE (temperate climate) and Malaysia (tropical climate)—demonstrate that the DRN model employing the proposed PSELU (γ = 0.02) achieves modest yet consistent improvements in forecasting accuracy compared with the DRN model using SELU. Specifically, the Mean Absolute Percentage Error (MAPE) decreased from 1.718% to 1.662% for ISO-NE and from 5.251% to 5.012% for Malaysia. Although the improvements are moderate, they were statistically validated through 10,000-iteration Bootstrap resampling at the 95% confidence level. These results suggest that the limited parameterization of SELU enhances consistency and adaptability in forecasting performance across different climatic and seasonal conditions. Future work will expand the evaluation to a wider range of datasets and model architectures to further examine the generalizability and practical applicability of PSELU in diverse forecasting contexts.

Keywords Activation function, DRNs, Parametric, SELU, STLF

Abbreviations

ANN	Artificial Neural Network
APSELU	Adaptive Parametric Scaled Exponential Linear Unit
Adam	Adaptive Moment Estimation
BiGRU	Bidirectional Gated Recurrent Unit
BiLSTM	Bidirectional Long Short-Term Memory
CI	Confidence Interval
CNN	Convolutional Neural Network
CNN-LSTM-MMA	CNN-LSTM Networks Based on a Multi-Modal Attention Mechanism
Conv1D	One-Dimensional Convolution
CRN	Convolutional Residual Network
DNN	Deep Neural Network
DRN	Deep Residual Network
ELM	Extreme Learning Machine
FC	Fully Connected Layer

[1]Department of Computer and Communication Systems Engineering, Faculty of Engineering, Universiti Putra Malaysia (UPM), Serdang 43400, Selangor, Malaysia. [2]Advanced Lightning, Power and Energy Research Centre (ALPER), Faculty of Engineering, Universiti Putra Malaysia (UPM), Serdang 43400, Selangor, Malaysia. ✉email: faisul@upm.edu.my

FLOPs	Floating-Point Operations
GRU	Gated Recurrent Unit
LSTM	Long Short-Term Memory
LTLF	Long-Term Load Forecasting
Leaky ReLU	Leaky Rectified Linear Unit
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MARE	Mean Absolute Relative Error
MB	Megabytes
MSE	Mean Square Error
MSRE	Mean Squared Relative Error
MTLF	Medium-Term Load Forecasting
MW	Megawatt
NMSE	Normalized Mean Square Error
PSELU	Parametric Scaled Exponential Linear Unit
R	Correlation Coefficient
RBF	Radial Basis Function
RMSPE	Root Mean Squared Percentage Error
RMSRE	Root Mean Squared Relative Error
RNN	Recurrent Neural Network
ReLU	Rectified Linear Unit
ResBlock	Residual Block
ResNet	Residual Network
ResNetPlus	Modified ResNet
R^2	Coefficient of Determination
SD	Standard Deviation
SELU	Scaled Exponential Linear Unit
STLF	Short-Term Load Forecasting
SVR	Support Vector Regression
VSTLF	Very Short-Term Load Forecasting

Load forecasting (LF) is essential for optimizing grid performance and ensuring reliability in modern power systems. By predicting future electricity demand, it supports utilities in system planning, operation, and management, thereby improving energy efficiency, reducing operational costs, and maintaining supply stability. With growing demand and increasingly diverse consumption patterns, LF has become more critical and complex[1]. To address these challenges, faster decision-making and better uncertainty management are required for effective power system operation. LF underpins numerous applications such as energy trading, system security analysis, unit commitment, economic dispatch, and performance monitoring, where forecasting accuracy directly influences grid reliability and operational costs. Inaccurate forecasts can lead to significant financial losses, emphasizing the need for dependable LF models. According to the forecasting horizon, LF is commonly divided into four categories[2]: Very Short-Term Load Forecasting (VSTLF, minutes to 1 h), Short-Term Load Forecasting (STLF, 1 h to 1 week), Medium-Term Load Forecasting (MTLF, 1 week to several months), and Long-Term Load Forecasting (LTLF, several months to years). Among them, STLF is particularly crucial for daily system scheduling and dispatch, as even a 1% reduction in its forecasting error can yield millions of dollars in annual savings for large utilities[3,4].

To address the challenges of STLF, numerous approaches have been developed, broadly categorized into classical and modern techniques. Conventional methods such as linear or non-parametric regression, support vector regression (SVR), autoregressive models, and fuzzy logic often suffer from oversimplification, difficulty in capturing nonlinear load patterns, and overfitting as the number of input variables increases[5,6]. Modern approaches, particularly artificial neural networks (ANNs), have emerged as effective alternatives, improving prediction accuracy and representing complex load behaviors through deep learning[7]. However, even with increased inputs or hidden layers, ANNs may still face overfitting, motivating the development of enhanced variants such as radial basis function (RBF) networks[8], wavelet-based networks[9], and extreme learning machines (ELM)[10].

More recently, deep neural networks (DNNs) with multiple hidden layers have gained prominence for their ability to hierarchically model complex temporal dynamics, marking a shift from shallow to deep architectures in STLF research[11]. Specialized deep models such as convolutional neural networks (CNNs) have been applied to extract local temporal features[12,13], yet their limited capacity to model long-term dependencies constrains their performance in complex scenarios. Recurrent neural networks (RNNs) and their variants, including long short-term memory (LSTM) and gated recurrent unit (GRU) networks, address this limitation by incorporating memory cells and gating mechanisms to capture both short- and long-term dependencies[14,15]. Nonetheless, their inherently sequential computations result in higher training costs and inefficiency for long sequences[16]. Bidirectional extensions such as BiLSTM and BiGRU further enhance sequential modeling by processing data in both directions but introduce additional computational overhead[17]. More recently, transformer-based models leveraging self-attention mechanisms have demonstrated strong potential for capturing long-range dependencies in STLF while enabling parallel sequence processing[18]. However, their quadratic computational complexity with respect to sequence length and sensitivity to training stability remain significant drawbacks[19,20]. To exploit complementary strengths across architectures, hybrid deep learning models have been proposed, such as Transformer–LSTM[21] and CNN–LSTM networks with multi-modal attention mechanisms (CNN–LSTM–

MMA)[22], which effectively combine local feature extraction, temporal dependency modeling, and attention-driven feature weighting. Although these hybrids achieve high forecasting precision, they often suffer from increased model complexity, computational cost, and limited interpretability, underscoring the need for efficient and robust deep learning frameworks for STLF.

As neural network models become deeper, training difficulties in architectures such as CNNs, RNNs, and Transformers often restrict their scalability and limit their ability to capture intricate patterns. To address these issues, He et al.[23] introduced the residual network (ResNet), which employs identity shortcut connections to facilitate gradient flow and effectively mitigate the vanishing gradient problem during backpropagation. Building upon this concept, Chen et al.[24] proposed a deep residual network (DRN) specifically tailored for STLF, incorporating architectural refinements to enhance stability and predictive accuracy. By maintaining efficient gradient propagation through residual connections, DRNs can consistently learn complex temporal and nonlinear relationships between load and weather variables, offering a balanced combination of depth, robustness, and reliability for STLF tasks.

Currently, DRN-based STLF models primarily utilize three widely adopted activation functions: Rectified Linear Unit (ReLU)[24,25], Leaky Rectified Linear Unit (Leaky ReLU)[26], and Scaled Exponential Linear Unit (SELU)[23,27–32]. ReLU and Leaky ReLU are known for their simplicity and computational efficiency but lack normalization capabilities, limiting their effectiveness in capturing complex nonlinear load behaviors. In contrast, SELU introduces self-normalizing properties that stabilize the mean and variance across network layers, facilitating the training of deeper architectures. However, SELU's reliance on fixed parameters and strict initialization conditions reduces its adaptability when facing seasonally fluctuating or heterogeneous input distributions.

In practical power systems, short-term load data exhibit strong seasonal non-stationarity, where statistical characteristics such as mean, variance, and load–temperature correlations vary dynamically across seasons and climatic regimes. These shifts often lead to degraded model performance because fixed activations cannot adjust their response to changing input distributions. As a result, even self-normalizing activations like SELU may suffer from reduced flexibility and unstable gradient propagation under rapidly changing load conditions.

To address this issue, this study proposes a Parametric Scaled Exponential Linear Unit (PSELU), a lightweight yet effective variant of SELU that enhances DRN adaptability under seasonally non-stationary load environments. By introducing a small linear adjustment term γ in the negative activation region—conceptually inspired by the design of Leaky ReLU—PSELU maintains the self-normalizing property of SELU while improving its responsiveness to varying input distributions. This design enables DRN models to more effectively capture nonlinear load–weather dependencies and temporal fluctuations across diverse climatic conditions without altering the network architecture or introducing additional trainable parameters.

The main contributions of this study can be summarized as follows: (1) A parametric activation function is proposed to specifically address the instability of DRN training under seasonally non-stationary load conditions, improving flexibility while preserving the self-normalizing behavior of SELU; (2) a comprehensive evaluation across two representative systems—ISO-NE (temperate) and Malaysia (tropical)—demonstrates that PSELU consistently enhances forecasting accuracy compared with conventional activation functions; and (3) the statistical significance of these improvements is rigorously verified through 10,000-iteration nonparametric Bootstrap testing, confirming that the observed gains are substantial and not due to random variation.

To guide the reader through the proposed methodology and its validation, the paper is organized as follows. Section 2 describes the architecture of DRNs and their application in STLF. Section 3 details the proposed PSELU, along with the experimental framework and evaluation metrics. Section 4 presents and analyzes the experimental results, including comparative assessments against mainstream activation functions. Section 5 concludes the study with key findings, discusses challenges and limitations, and outlines future research directions.

Short-term load forecasting using deep residual networks

The stlf's DRN structure

The DRN model's framework in STLF is shown in Fig. 1. A basic structure and the modified ResNet (ResNetPlus) make up the majority of the DRN for STLF, which is based on the structure described in[23]. ResNetPlus, an improved ResNet variant created to boost 24-hour STLF efficiency, keeps ResNet's block structure while adding improvements for more accurate predictions.

Each fully connected layer (FC), which corresponds to [L_h^{day},T_h^{day}], [L_h^{week}, T_h^{week}], [L_h^{month}, T_h^{month}] and L_h^{hour} in this design, has ten hidden nodes. The completely connected layers linked to [S, W], however, include five hidden nodes. There are ten hidden nodes in the fully-connected layer before L_h as well as in FC1 and FC2. An activation function is used by every layer except the output layer, which is worth mentioning. The load levels for the corresponding hour from days 1, 2, and 3 months before to the anticipated day are indicated by the letter L_h^{month} in this basic structure. L_h^{day} represents the loads of the same hour for each day of the previous week, whereas L_h^{week} represents the load values for the same hour from 1 to 8 weeks ago. L_h^{hour} stands for the load levels from the preceding 24 h for the same hour.

With significant improvements over the original ResNet architecture, the ResNetPlus model is a sophisticated development in neural network design. Each residual block (ResBlock) in this model consists of one hidden layer with 20 nodes followed by the same activation function as in the basic structure, while retaining the residual connection structure. ResNetPlus generates a great deal of depth and complexity by successively constructing four of these blocks, each with its own unique connections, and repeating the process throughout ten levels. In order to reach the model's output, the design incorporates a unique shortcut link that runs straight from the last block's output to the network's entrance point. Such a configuration maximizes the effectiveness of a DRN while also making its development simpler. Such a configuration maximizes the effectiveness of a DRN

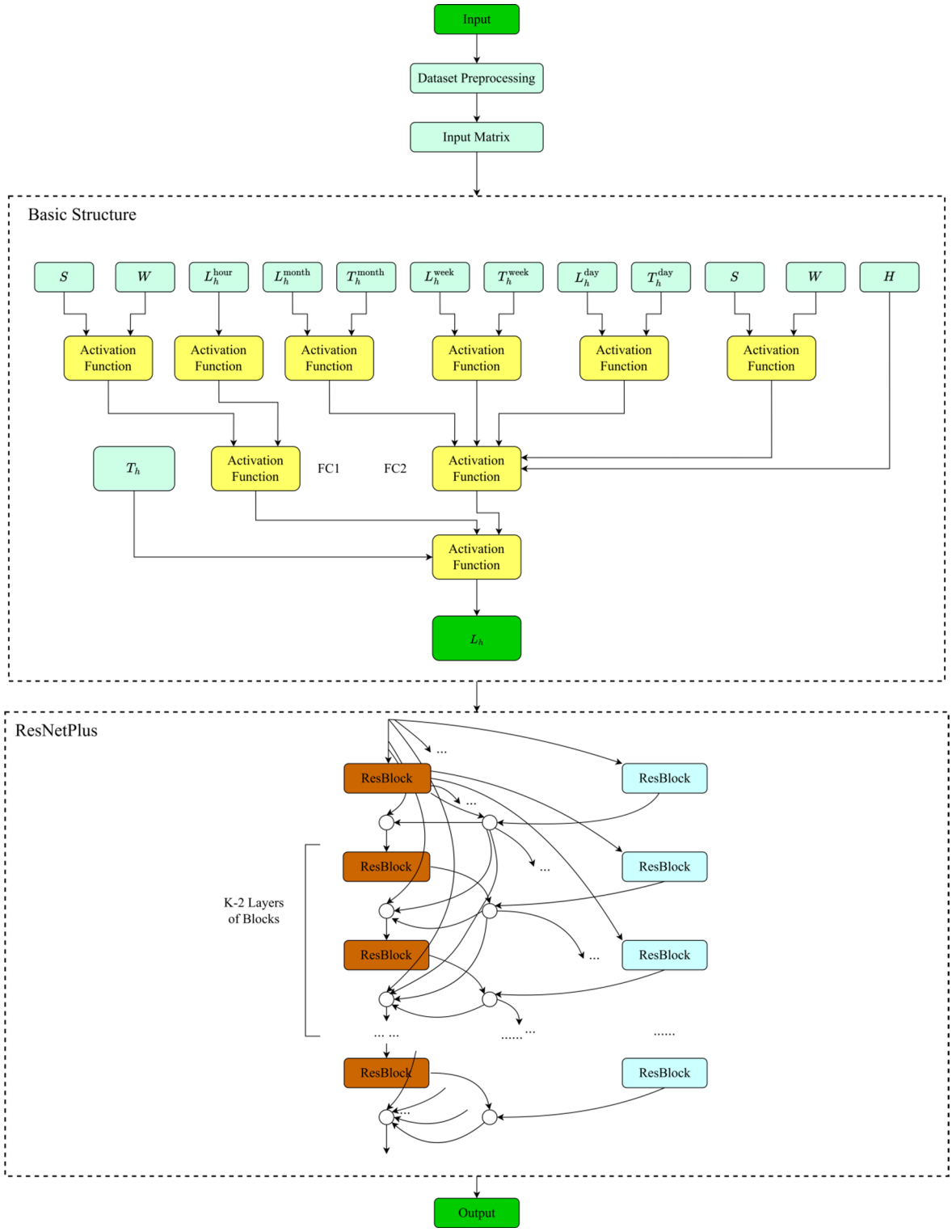

Fig. 1. DRN framework for STLF[23].

while also making its development simpler. The ResNetPlus architecture fully leverages the residual design while maintaining the hyperparameters inherited from its ResNet predecessors within these blocks.

First, a neural network with layers that are closely coupled, known as the 'basic structure,' is used. The initial load projection for the next 24 h is produced using this core architecture. Second, temperature values T_h^{month}, T_h^{week} and T_h^{day} and load values L_h^{month}, L_h^{week} and L_h^{day} are taken simultaneously. The actual temperature predicted for the next day is T_h. Season, weekday, and holiday status are represented by the one-hot encoded

variables S, W, and H, accordingly. The input for the second part of the model is the result of this basic structure, represented by the letter L_h.

The overall model Loss, denoted as Loss, is composed of two distinct components, combined to facilitate efficient model training. This relationship is expressed in Eq. (1):

$$\mathrm{Loss}=\mathrm{Loss_E}+\mathrm{Loss_R} \tag{1}$$

In order to speed up the training process, $\mathrm{Loss_R}$ acts as a penalty term for values that are outside of the range, whereas $\mathrm{Loss_E}$ measures the difference in predictions. Specifically, $\mathrm{Loss_E}$ is expressed using Eq. (2):

$$\mathrm{Loss_E} = \frac{1}{\mathrm{NumH}} \sum_{\mathrm{N}}^{\mathrm{N}} \sum_{\mathrm{h}=1}^{\mathrm{H}} \frac{\left|\hat{y}_{(\mathrm{j,h})} - \mathrm{y}_{(\mathrm{j,h})}\right|}{\mathrm{y}_{(\mathrm{j,h})}} \tag{2}$$

Where $\mathrm{y}_{(\mathrm{j,h})}$ stands for the actual normalized load for the hth hour of the jth day, and $\hat{y}'_{(\mathrm{j,h})}$ is the model's output. represents the number of data samples in this case, and H is the number of hourly loads in a day (in this case, H = 24). Known as the Mean Absolute Percentage Error (MAPE), this metric is used as a criterion for evaluating the models' predicted outcomes as well as a measure of inaccuracy. Equation (3) is used to calculate the second term $\mathrm{Loss_R}$.

$$\begin{aligned} Loss_R = & \frac{1}{2Num} \sum_{j=1}^{Num} \max\left(0,\ \max_h \hat{y}_{(j,h)} - \max_h y_{(j,h)}\right) \\ & + \max\left(0,\ \min_h y_{(j,h)} - \min_h \hat{y}_{(j,h)}\right) \end{aligned} \tag{3}$$

If the projected daily load curves diverge from the actual load ranges, this term penalizes the model, speeding up the first round of training. This phrase emphasizes the cost of overestimating the load curves' peaks and underestimating their troughs with increasing precision when the model starts to produce projections.

Application status of activation functions in DRN models

In DRNs, activation functions not only determine the network's nonlinear modeling capacity but also directly influence training stability and generalization performance. In STLF, where load series typically exhibit strong nonlinearity, periodicity, and abrupt changes, the choice of activation function is particularly critical. At present, DRN-based STLF models mainly adopt three typical activation functions: ReLU[24,25], Leaky ReLU[26], and SELU[23,27–32]. Notably, SELU has been adopted most frequently due to its self-normalizing properties, which help stabilize training in deep architectures. However, each of these functions presents distinct advantages and limitations across different studies, influencing model learning, convergence speed, and forecasting accuracy. Moreover, when applied to real-world STLF problems—such as those involving diverse climatic conditions, irregular consumption patterns, and complex external influences—these commonly used activation functions still exhibit notable shortcomings, limiting their ability to generalize across varying scenarios. To provide a clearer picture, representative DRN-based studies, their adopted activation functions, and corresponding performance metrics are summarized in Table 1.

The ReLU is mathematically defined in Eq. (4):

$$\mathrm{f(x)}=\max(0,\mathrm{x}) \tag{4}$$

x is the input value in this case. ReLU is appropriate for deep network training as it efficiently addresses the vanishing gradient issue and is computationally efficient.

Algorithm	Performance Metric	Activation Function	References
DRN(ResNetPlus)	MAPE	SELU	23
ERN	MAE, MAPE, NMSE, R, RMSE		28
ResNetPlus-LSTM	MAE, MAPE, MSE, RMSE		27
DRN(ResNetPlus)	MAPE		29
GoogleNet-ResNetPlus	MAE, MAPE, R^2, RMSE	ReLU	24
CRN	MAPE	Leaky ReLU	26
ResNet-LSTM-Attention	MAPE	ReLU	25
ResNet-Based Ensemble Model	MAE, MAPE, RMSE	SELU	30
deep-ResNet	MAPE, RMSE, MAE	SELU	31
Residual LSTM Plus	MAPE, R^2	SELU	32

Table 1. Summary of activation functions in DRN-based STLF Models.

Ding et al.[24] employed ReLU as the primary activation function in their hybrid model integrating ensemble GoogLeNet with a modified DRN for STLF, emphasizing its computational efficiency and ability to mitigate vanishing gradients, which enhanced feature extraction and maintained stable training in deep architectures. Similarly, Li et al.[25] adopted ReLU in their ResNet–LSTM model with an attention mechanism, highlighting that its non-saturating linear property accelerates the convergence of DRNs and improves the capture of complex temporal load features, thereby enhancing forecasting accuracy. However, both studies also faced inherent drawbacks of ReLU, such as the "dying neurons" problem that suppresses negative signals and its lack of normalization capability, which may limit performance in scenarios with highly variable load and weather patterns.

To alleviate this issue, the Leaky ReLU introduces a small slope for negative inputs, as shown in Eq. (5):

$$f(x) = \begin{cases} x & \text{if} x > 0 \\ \alpha x & \text{if } x \leq 0 \end{cases} \tag{5}$$

where α is a tiny constant (e.g., 0.01) and x is the input value. This keeps neurons from going dormant. But it adds another hyperparameter that needs to be adjusted.

Sheng et al.[26] adopted Leaky ReLU in their convolutional residual network (CRN) for STLF, noting that it effectively mitigates the "dying neurons" issue inherent to ReLU by introducing a small non-zero slope for negative inputs. This adjustment improved gradient flow in deep ResBlocks, enhanced feature extraction for nonlinear load patterns, and provided better overall forecasting performance compared to traditional ReLU-based models. However, the study also pointed out that Leaky ReLU introduces an additional hyperparameter—the negative slope—that requires careful tuning, and it lacks intrinsic normalization capabilities, which may lead to performance instability when dealing with highly variable load or weather conditions.

The SELU offers a different approach by incorporating self-normalizing properties. Its mathematical formulation is provided in Eq. (6):

$$f(x) = \begin{cases} \lambda x & if\ x > 0 \\ \lambda\alpha\left(e^{x} - 1\right) & if\ x \leq 0 \end{cases} \tag{6}$$

where x is the input value, $\lambda = 1.0507$ is a normalization scaling factor, and $\alpha = 1.6733$ modifies the output to account for negative inputs. Self-normalization is guaranteed by SELU, which stabilizes the mean and variance between layers. Its need on certain initialization and dropout conditions, however, may make implementation more difficult. SELU is designed to maintain mean and variance stability across layers, which is advantageous in deep architectures.

Chen et al.[23] first introduced SELU into a DRN for STLF, emphasizing its self-normalizing property that stabilized the mean and variance across layers, which facilitated the training of deeper networks and reduced the risk of gradient vanishing. Tian et al.[27] extended this approach by integrating SELU into a ResNetPlus–LSTM hybrid structure, where SELU improved feature extraction and accelerated convergence, contributing to higher prediction accuracy for complex load series. Xu et al.[28] employed SELU in an ensemble residual network framework to enhance model generalization across diverse datasets, demonstrating its effectiveness in improving robustness. Kondaiah and Saravanan[29] utilized SELU in a modified ResNetPlus model, reporting that it helped maintain network stability and reduced overfitting in deeper architectures. Chen et al.[30] further applied SELU in a ResNet-based ensemble model, where it supported multi-scale feature learning and mitigated gradient issues in deep structures. Similarly, Kondaiah and Saravanan[31] incorporated SELU into their modified deep ResNet, confirming its role in enhancing generalization for datasets. Most recently, Sheng et al.[32] embedded SELU in a hybrid residual LSTM model with an attention mechanism, leveraging its normalization capabilities to stabilize gradient flow in stacked residual–recurrent layers. These studies collectively demonstrate that SELU enhances the stability of deep residual networks by maintaining consistent activation variance and preventing gradient explosion or vanishing during backpropagation. This self-normalizing behavior enables DRN models to converge more smoothly and generalize more reliably under diverse climatic and load conditions. However, SELU's dependence on specific initialization and dropout settings, together with its fixed activation parameters, constrains adaptability when handling highly irregular or heterogeneous load patterns. These limitations highlight the necessity for more flexible and data-adaptive activation functions to further improve the robustness of DRN-based STLF models.

Each of the three activation functions exhibits distinct strengths and weaknesses in the context of STLF. ReLU and Leaky ReLU are simple in structure and efficient in training, but they fall short in modeling complex nonlinear load patterns and do not support output normalization. SELU, in contrast, provides more stable training dynamics and is better suited for deep network construction due to its self-normalizing properties, yet it lacks adaptability and depends on strict initialization and regularization settings. As DRN models become increasingly deep and the input features more diverse—incorporating complex temporal structures and heterogeneous weather conditions—the limitations of existing activation functions become more pronounced.

To address these limitations, recent research has explored parameterized and adaptive activation functions that introduce trainable shape flexibility while retaining desirable properties such as stability and normalization. Bingham et al.[33] combined evolutionary search with gradient-based refinement to automatically discover and optimize parametric activation functions, achieving consistent improvements over static activations like ReLU and SELU across various architectures and datasets. Similarly, Nader and Azar[34] conducted extensive empirical evaluations of evolved parametric activations, confirming their superior generalization ability across diverse tasks. These studies suggest that incorporating parameterization into activation functions is a promising direction for enhancing flexibility and robustness in complex forecasting applications.

Beyond STLF, recent studies have increasingly investigated adaptive or parametric activation functions in broader time-series and scientific forecasting tasks. Pourkamali-Anaraki et al.[35] explored parametric variants of traditional activation functions for sparse experimental data, showing that adaptive parameterization can improve predictive accuracy and model confidence. Vaca-Rubio et al.[36] introduced Kolmogorov–Arnold Networks, where spline-parameterized functions replace fixed activations, achieving superior performance in real-world time-series applications. More recently, Zhang et al.[37] proposed simple yet effective adaptive activation functions to accelerate convergence, while Zhang et al.[38] developed the RoSwish function by integrating multiple nonlinear properties into a flexible design. Collectively, these works highlight the potential of adaptive and parameterized activation functions to enhance flexibility and generalization in complex forecasting tasks. However, they also reveal important trade-offs: the introduction of trainable parameters may increase the risk of overfitting in small-sample or noisy scenarios, while sensitivity to initialization and additional computational cost may limit training stability. Hence, adaptive activation functions should be regarded as promising but not universally superior, and their deployment in DRN-based STLF requires careful balance between adaptability and robustness.

Overall, ReLU, Leaky ReLU, and SELU each exhibit distinct advantages in DRN-based STLF; however, their adaptability remains limited when dealing with multi-seasonal, heterogeneous, and highly nonlinear load characteristics. To overcome these limitations, researchers have recently begun exploring parametric activation functions, which introduce trainable parameters into the negative region, slope, or scaling factors of the function. Such parameterization enables dynamic shape adjustment based on data distribution, thereby enhancing generalization in complex forecasting tasks while preserving key properties such as stability and self-normalization. This trend offers a promising avenue for further improving the performance and robustness of DRN models in STLF.

Research methodology

Dataset configurations

In order to analyze STLF under various climatic and consumption situations, this study uses two datasets: the New England Independent System Operator (ISO-NE) dataset and the Malaysia dataset (data sources are provided in Appendix). The use of datasets from regions with distinct climatic conditions enables the evaluation of model robustness and generalization under varying weather–load relationships[39]. Hourly load and weather (temperature) data from March 2003 to December 2014 are included in the ISO-NE dataset. These data show normal consumption patterns in a temperate environment with notable seasonal and yearly fluctuations. On the other hand, the Malaysia dataset includes daily temperature information (mean temperature, minimum temperature, maximum temperature) and hourly load data for the Petaling Jaya region from January 2020 to December 2022. With load ranging from around 7500 MW to 27,500 MW, compared to roughly 9000 MW to 19,000 MW in the Malaysia dataset, the ISO-NE dataset exhibits pronounced periodic trends driven by seasonal effects, whereas the Malaysia dataset, representing a tropical climate, maintains relatively steady demand with moderate fluctuations. The load statistics from both datasets are shown in Figure 2, which draws attention to their disparate features and qualifies them for assessing how well STLF algorithms perform in a variety of circumstances.

Dataset pre-processing

Data pre-processing is a crucial step that enhances the quality, consistency, and reliability of raw data before model training and evaluation[40]. Since the ISO-NE dataset was used as a standard benchmark dataset in this work and had already been preprocessed by the data supplier, no further cleaning was necessary. On the other hand, the load and temperature records in the Malaysia dataset included many missing values. To solve this and guarantee data continuity, linear interpolation was used to fill in the missing entries. In this stage, the completeness and suitability of the input characteristics for model training and evaluation were guaranteed.

In addition, pearson correlation coefficients[41] were calculated to preliminarily examine the relationships between load and temperature variables in both datasets. The results are visualized as heatmaps in Figs. 3 and 4. For the ISO-NE dataset (Figure 6), load shows a moderate positive correlation with hourly temperature, reflecting the clear seasonal influence in a temperate climate. For the Malaysia dataset (Figure 7), mean temperature exhibits the strongest correlation with load, followed by maximum temperature and minimum temperature, though all correlations remain relatively weak. These results indicate that temperature plays a more prominent role in load variation under temperate conditions, while in tropical regions its linear impact is comparatively limited.

A proposed parametric SELU activation function for DRN in STLF

Activation functions are essential components of DNNs, directly influencing the model's nonlinear representational capacity, gradient flow, and training stability. In this paper, a novel parametric variant of the SELU activation function is proposed to enhance the flexibility and modeling capability of these tasks.

The conventional SELU possesses self-normalizing properties that help alleviate the vanishing gradient problem and accelerate convergence in deep architectures. However, its activation behavior is governed by fixed parameters, limiting its adaptability to varying input distributions. To address this limitation, the proposed function introduces an additional parameter γ into the negative region of SELU. Inspired by the structure of Leaky ReLU—where a small slope is introduced in the negative domain to mitigate the dead neuron problem—γ acts as a linear adjustment term that enhances the flexibility of the negative activation response while preserving SELU's self-normalizing characteristics.

The mathematical formulation of PSELU is as follows, as shown in Eq. (7):

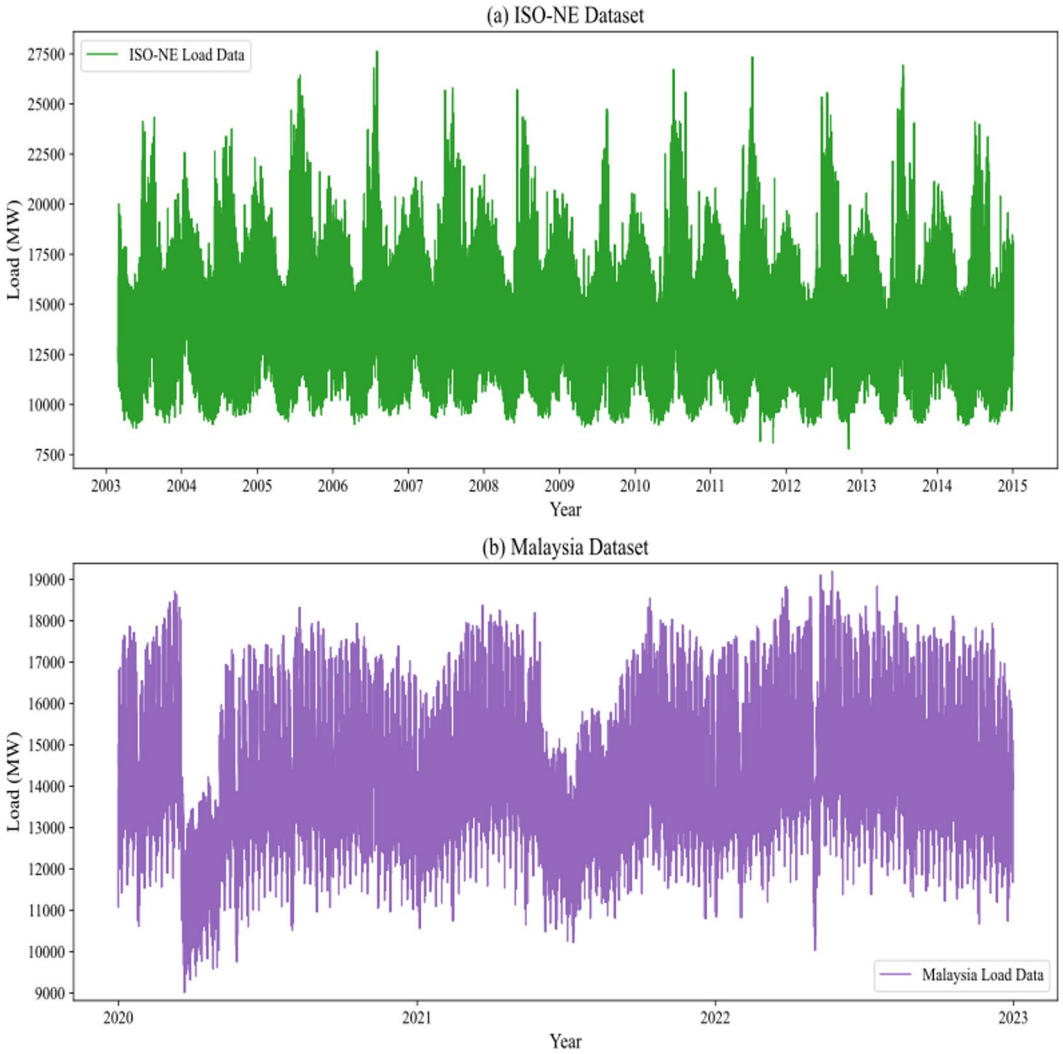

Fig. 2. Load records from the (**a**) ISO-NE dataset and (**b**) Malaysia dataset.

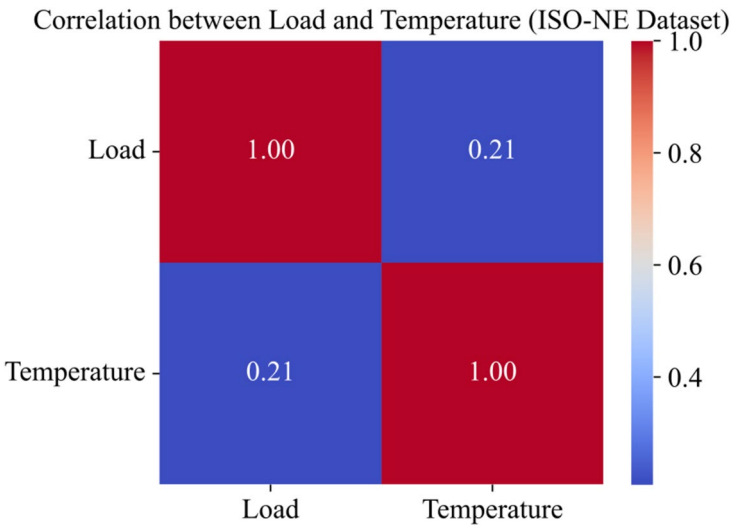

Fig. 3. Correlation between Load and Temperature (ISO-NE Dataset). This heatmap was generated using Python 3.8 with TensorFlow 2.10.0 (https://www.tensorflow.org/).

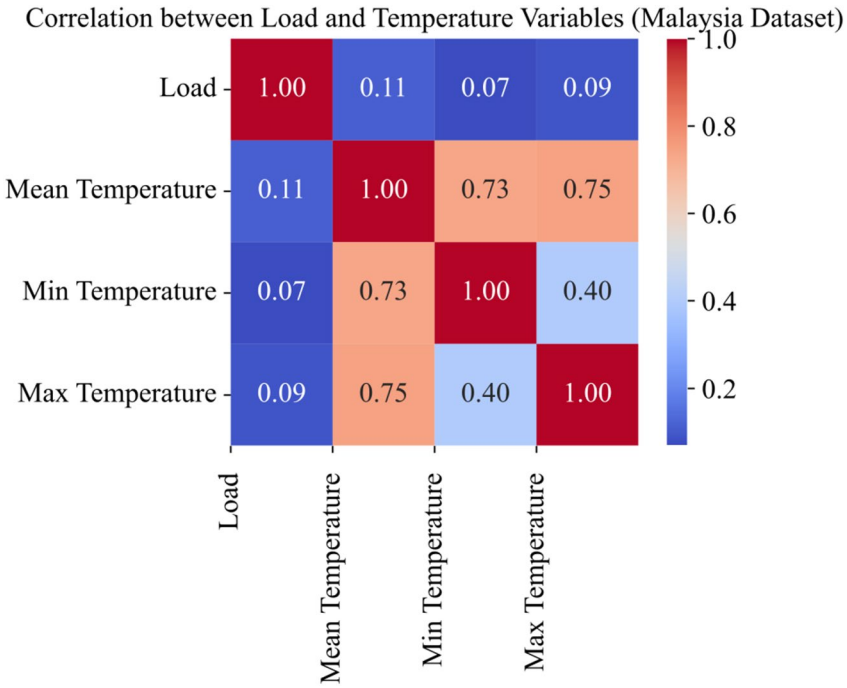

Fig. 4. Correlation between Load and Temperature Variables (Malaysia Dataset). This heatmap was generated using Python 3.8 with TensorFlow 2.10.0 (https://www.tensorflow.org/).

$$PSELU(x) = \begin{cases} \lambda(x), & x \geq 0 \\ \lambda\left[\alpha\left(e^{x}-1\right)+\gamma x\right], & x < 0 \end{cases} \tag{7}$$

where α = 1.6733 and λ = 1.0507 are constants inherited from the original SELU[42] to maintain stable mean and variance of the activation outputs. In this study, the parameter γ is fixed at a value of 0.01, following the design principle of Leaky ReLU, which introduces a small slope in the negative region to preserve gradient flow. This choice improves the model's robustness and generalization by enhancing responsiveness to negative input values.

To theoretically analyze the effect of γ on self-normalization, consider an input variable xxx that follows a normal (Gaussian) distribution, denoted as $x \sim N(\mu, \sigma^2)$,, where N represents the normal distribution, and μ and σ^2 denote the mean and variance of the input, respectively. This assumption facilitates the derivation of the expected mean and variance of the activation outputs, which are essential for examining whether the PSELU function preserves the self-normalizing property. The expected output of PSELU can then be expressed as Eq. (8):

$$E[f(x)] = \lambda\left(\int_{0}^{\infty} xp(x)dx + \int_{-\infty}^{0}\left[\alpha\left(e^{x}-1\right)+\gamma x\right]p(x)dx\right) \tag{8}$$

and the variance is obtained from Eq. (9):

$$Var[f(x)] = E\left[f(x)^2\right] - \left(E[f(x)]\right)^2 \tag{9}$$

Compared with SELU[42], the introduction of γ adds an additional contribution in the negative domain, expressed in Eq. (10):

$$\Delta_{\gamma} = \lambda\gamma\int_{-\infty}^{0} xp(x)dx \tag{10}$$

which slightly modifies the mean and variance propagation across layers. When $\gamma = 0$, the formulation reduces to SELU, where $\left(\mu\sigma^2\right) = (0)$ is a fixed point that ensures convergence across layers. For small γ values (e.g., 0.01), the perturbation is minor and does not destroy the fixed-point stability, thereby preserving the self-normalizing property while enhancing gradient flow in the negative domain.

The theoretical derivation of Eqs. (8)-(10) follows the analytical framework established in the original SELU study by Klambauer et al.[42], which demonstrated that a fixed point of mean and variance $\left(\mu = 0, \sigma^2 = 1\right)$ ensures the self-normalizing property across layers. The proposed PSELU function introduces a small linear perturbation γ in the negative activation region while retaining the exponential structure of SELU. This modification can be viewed as a first-order linear adjustment that slightly alters the mean and variance

propagation by $\Delta\gamma$, as described in Eq. (10), without violating the fixed-point convergence condition. When $|\gamma|$ is sufficiently small (e.g., 0.01), the variance curve remains within SELU's stability basin, thereby preserving the self-normalizing property while improving gradient flow in the negative domain. This analytical consistency aligns with the perturbation stability principle in nonlinear systems, where small bounded perturbations do not destabilize fixed equilibria.

In addition to the statistical integral analysis presented earlier, the inter-layer propagation behavior can be further examined. As expressed in Eq. (11), the propagation of activations through consecutive layers follows:

$$z_{l+1} = \begin{cases} \lambda z_l, & z_l \geq 0 \\ \lambda\left(\alpha e^{z_l} - \alpha + \gamma z_l\right), & z_l < 0 \end{cases} \tag{11}$$

where λ and α are the same scaling constants as in SELU, and γ introduces a small linear perturbation in the negative region.

Assuming that the pre-activation z_l follows a zero-mean distribution with variance σ_1^2, the mean and variance of activations at the subsequent layer can be formulated as shown in Eq. (12):

$$\mu_{l+1} = \lambda E\left[f\left(z_l\right)\right],\ \sigma_{l+1}^2 = \lambda^2 Var\left[f\left(z_l\right)\right] \tag{12}$$

By substituting $f(z_l)$ with the PSELU function, the generalized variance propagation can be derived, as presented in Eq. (13):

$$\sigma_{l+1}^2 = \lambda^2\left[pVar\left(z_l|z_l \geq 0\right) + (1-p)Var\left(\alpha e^{z_l} - \alpha + \gamma z_l|z_l < 0\right)\right] \tag{13}$$

where $p=P(z_l \geq 0)$ represents the probability of a positive activation.

Equation (13) extends the variance formulation of the original SELU, in which the second term captures the contribution from the negative region modified by the parameter γ.

The additional linear term γz_1 introduces a small correction to the negative variance component. When $|\gamma| << 1$, this correction term can be approximated by Eq. (14):

$$\Delta\sigma^2 \approx (1-p)\lambda^2\gamma^2 Var\left(z_l|z_l < 0\right) \tag{14}$$

which remains bounded and small compared with the overall variance.

Consequently, the fixed-point condition ($\mu = 0, \sigma^2$), which guarantees self-normalizing behavior, remains approximately valid under small perturbations of γ. Therefore, the PSELU retains the theoretical stability of SELU while enhancing flexibility in the negative activation region.

Nevertheless, the modified design also introduces certain trade-offs. The incorporation of additional exponential and multiplication operations, along with the linear term in the negative region, slightly increases computational complexity compared to simpler activation functions such as ReLU, Leaky ReLU, and SELU. This additional cost may influence deployment efficiency in highly resource-constrained environments.

Regarding parameterization, γ plays a critical role in balancing representational flexibility and training stability. Previous research on parametric activation functions has demonstrated that adjusting the negative-region slope can significantly influence model behavior and that empirical analysis is necessary to identify the most suitable parameter configuration[33]. Although a fixed γ was adopted in this work, inappropriate values could potentially slow convergence or induce overfitting. To ensure robustness, γ was initially set to 0.01, consistent with the convention of Leaky ReLU[43], which provides a small slope in the negative domain to maintain gradient flow. A sensitivity analysis was subsequently performed across $\gamma \in \{0.001, 0.005, 0.01, 0.02, 0.03, 0.04, 0.05\}$ to investigate whether small perturbations within this range could preserve self-normalizing stability while influencing representation flexibility. The comparative performance of different γ values will be analyzed in the experimental section to identify the most effective configuration.

In addition, an adaptive variant—Adaptive Parametric Scaled Exponential Linear Unit (APSELU)—was introduced, where γ was initialized at 0.01 and treated as a learnable parameter during training. This adaptive formulation aims to enhance flexibility by allowing the network to automatically adjust the slope of the negative activation region according to the learning dynamics. Nevertheless, such adaptability may also introduce additional gradient fluctuations or convergence instability, depending on the dataset characteristics and optimization conditions.

Proposed method

In this paper, the original DRN architecture is retained and extended into a unified DRN (PSELU) framework, which consists of two main components: the basic structure (PSELU) and the ResNetPlus (PSELU) module. Both components consistently employ the proposed PSELU activation function, ensuring architectural coherence and enabling its performance to be evaluated throughout the entire framework. The overall architecture of DRN (PSELU) is illustrated in Fig. 5.

Since the temporal granularity of the ISO-NE and Malaysia datasets in this work varies greatly, distinct feature processing techniques are required in the basic structure (PSELU) to adapt the model inputs accordingly. Hourly load L_h^{hour} and temperature data T_h^{month}, T_h^{week}, T_h^{day} for the ISO-NE dataset are supplied into the model directly. To create model inputs, the same basic structure described earlier is used, which combines S, W, and H information with load characteristics L_h^{month}, L_h^{week}, L_h^{day} that are extracted based on various temporal ranges. The seasons of S include spring, summer, autumn and winter, whereas H encompasses holidays like

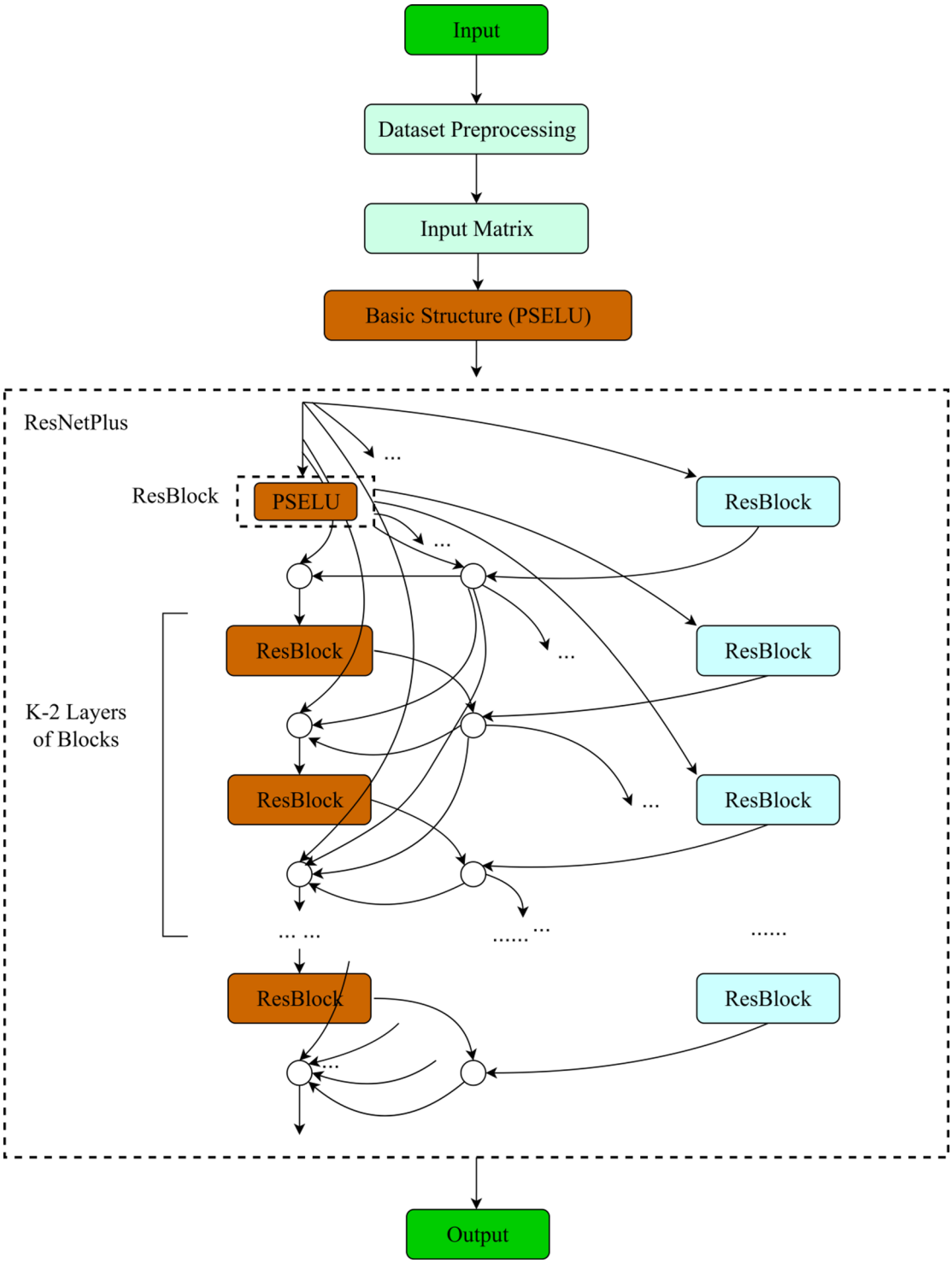

Fig. 5. Proposed DRN (PSELU) framework for STLF.

Christmas and Independence Day in the United States. An illustration of the basic structure within the DRN architecture using the PSELU activation function, as applied to the ISO-NE dataset, is provided in Fig. 6.

As opposed to the hourly temperature data in the ISO-NE dataset, the Malaysia dataset only offers daily temperature data, including $T_{mean}, T_{max}, T_{min}$. To address this discrepancy in temporal granularity, this study adopts a basic structure capable of directly accepting daily temperature data. The basic structure for the Malaysia dataset, as illustrated in Fig. 7, is adapted from the original architecture in previous research to accommodate daily temperature inputs by removing the need for hourly-level segmentation, which is not applicable to daily data[2]. The architecture does not use temporal segmentation; instead, daily temperature data $T_{mean}, T_{max}, T_{min}$ is concatenated as a single feature input. Load feature $L_h^{month}, L_h^{week}, L_h^{day}$ processing, meanwhile, is unaltered

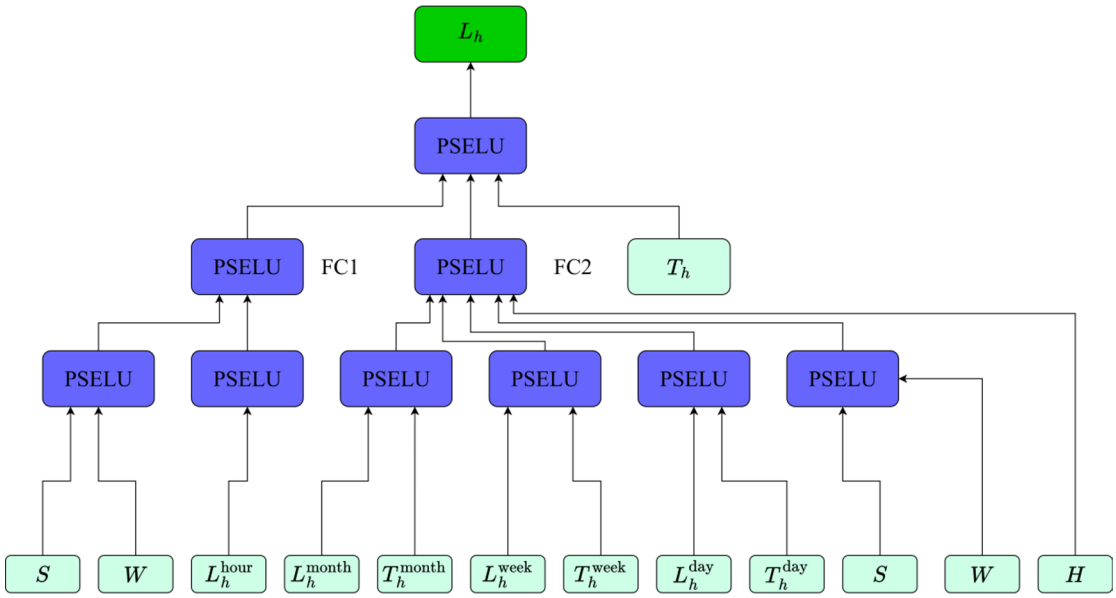

Fig. 6. Basic structure in DRN (PSELU) for ISO-NE dataset.

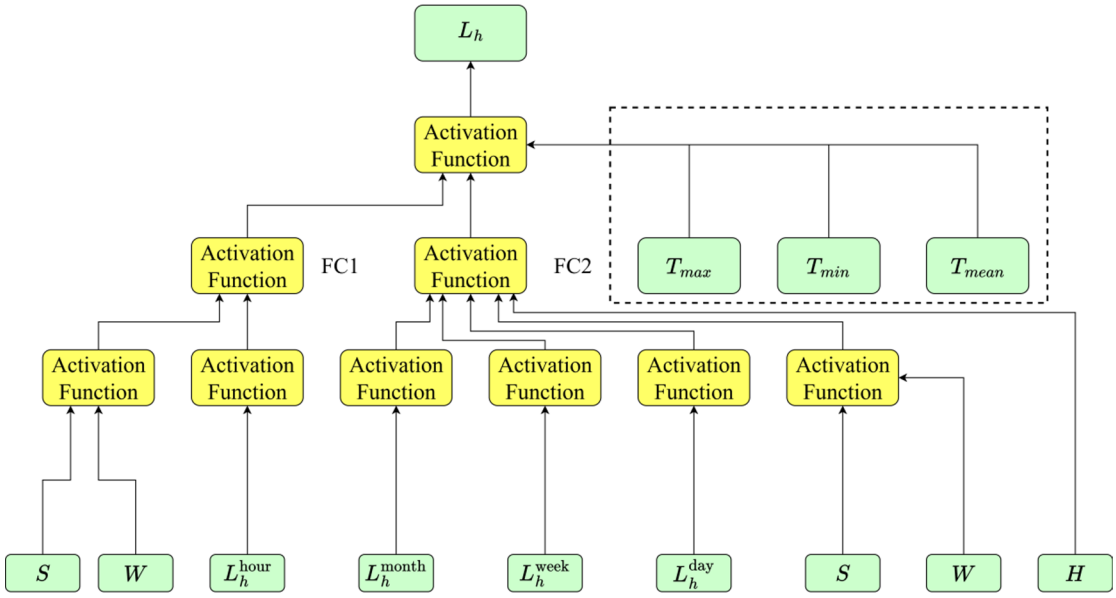

Fig. 7. Basic structure in DRN for Malaysia dataset[2].

and keeps pulling data from the previous 24 h, 8 weeks, and 3 months. To create the final input to the model, date-related information (such S, W, and H) are mixed with load and temperature features. H contains events like Eid al-Fitr and Malaysia Independence Day, whereas S is separated into two seasons: the rainy season and the dry season. Building upon this, Fig. 8 presents the basic structure in PSELU for the Malaysia dataset, in which only the activation function is replaced with PSELU. All other architectural components—including the temporal feature construction, input concatenation strategy, and load processing layers—are kept unchanged to ensure structural consistency with the previously adopted framework for daily temperature inputs.

In addition to the basic structure (PSELU), the ResNetPlus (PSELU) component of the proposed framework is also modified to employ the PSELU activation function. While the basic structure (PSELU) adapts to dataset-specific temporal granularities, the ResNetPlus (PSELU) architecture remains consistent with the original design, consisting of ten levels of stacked residual blocks, each containing a hidden layer with 20 nodes followed by the same PSELU activation function. Skip connections are preserved across all blocks to facilitate gradient propagation and ensure training stability. The only modification introduced in this study is the uniform

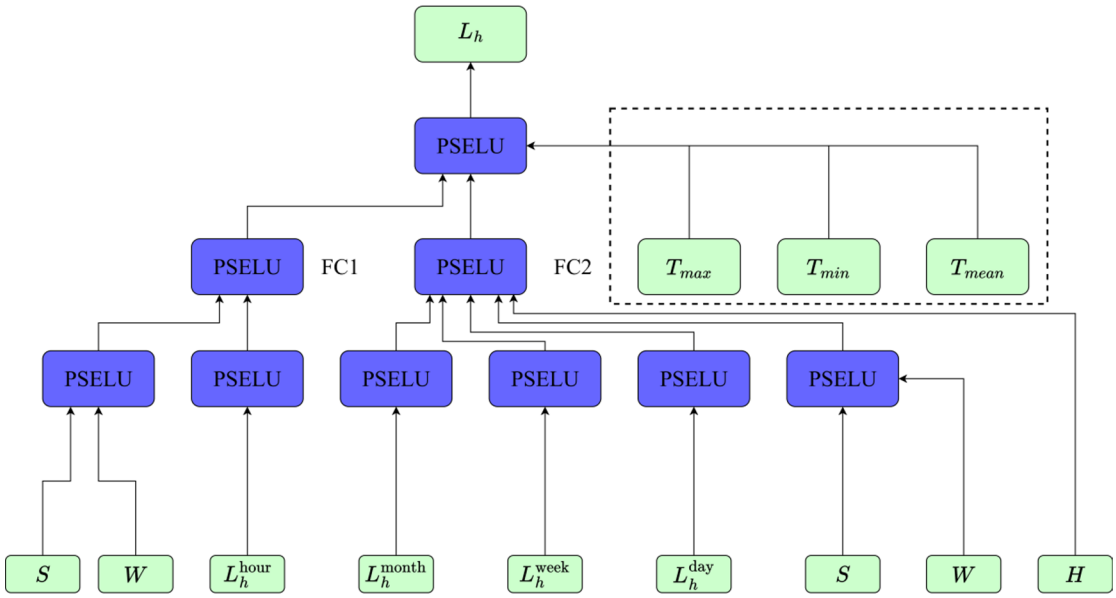

Fig. 8. Basic structure (PSELU) in DRN (PSELU) for Malaysia dataset.

replacement of SELU with PSELU in both the basic structure and the ResNetPlus modules. This unified design ensures that the observed performance improvements are solely attributable to the proposed activation function, while maintaining strict structural comparability with previously validated DRN-based STLF frameworks.

Experimental design

Two real-world datasets are employed in this study: the ISO-NE dataset and the Malaysia dataset. Specifically, the ISO-NE dataset uses data from March 2003 to December 2005 for training and the year 2006 for testing, while the Malaysia dataset uses data from January 2020 to December 2021 for training and the year 2022 for testing. These datasets represent two distinct climatic regions—temperate and tropical—and exhibit diverse load patterns, providing a broad experimental setting for model evaluation.

To maintain a controlled and consistent experimental environment, this study adopts a fixed DRN architecture. The primary objective is to evaluate the impact of different activation functions while keeping the model structure unchanged. In particular, the experiments investigate ReLU, Leaky ReLU, and SELU, as well as the proposed PSELU, where the parameter γ is systematically varied to assess sensitivity and robustness. This setup allows the influence of activation functions to be isolated, thereby enhancing the clarity and reliability of the experimental results. Furthermore, the DRN models are compared against several widely adopted deep learning architectures, including CNN, LSTM, BiLSTM, GRU, BiGRU, Transformer, Transformer-LSTM[21], and CNN-LSTM-MMA[22], to provide a comprehensive performance benchmark.

The CNN was implemented as a one-dimensional convolution (Conv1D) architecture consisting of 64 filters, a kernel size of 3, ReLU activation, and 'same' padding. The RNN variants (LSTM, BiLSTM, GRU, BiGRU) employed 64 recurrent units with default activation functions. For the Transformer, a standard configuration was adopted, comprising a single encoder layer, eight attention heads with 64 dimensions each, a 64-dimensional embedding, a 2048-dimensional feed-forward network, positional encoding, and a dropout rate of 0.1. The Transformer-LSTM and CNN-LSTM-MMA followed the configurations reported in their original studies. All models were trained and evaluated under identical experimental setups, using the same input features and preprocessing strategies, thereby ensuring a fair and unbiased comparison of their performance.

A snapshot ensemble approach is utilized during the training process, where model weights (snapshots) are periodically saved[23,44]. Averaging predictions across multiple snapshots reduces the risk of overfitting associated with a single model and improves prediction stability and generalization capability. This approach provides a computationally efficient alternative to performing multiple independent training trials while achieving comparable robustness and stability[45]. The training consists of 600 initial epochs, followed by two additional rounds of 50 short-term epochs, with three snapshots stored at the end of each round. The final prediction is obtained by averaging the outputs of the three saved snapshots, ensuring a more robust and stable forecasting performance.

Additionally, a nonparametric Bootstrap resampling approach with 10,000 iterations was used in the experimental design to thoroughly evaluate whether the performance increase of the enhanced model over the original model is statistically significant. The Bootstrap approach provides a more robust framework for comparing model performance than the paired Student's t-test, which assumes that the paired differences are normally distributed[46]. Two criteria were used to establish statistical significance. First, the improvement is regarded as significant at the 95% confidence level if the 95% confidence interval (CI) of the mean performance difference between the two models is completely above zero; if the interval contains zero, the difference is regarded as inconsequential. Second, a p-value below 0.05 also denotes a statistically significant difference at

the 95% confidence level according to the Bootstrap framework. Note that a stated $p \approx 0$ in practice does not represent an actual zero, but rather a very small probability (less than 0.0001). Due to its widespread usage in STLF research and its ability to give an interpretable, scale-independent measure of relative prediction error, MAPE was selected as the assessment metric.

Apart from forecasting accuracy, this study also evaluates computational complexity to provide a more comprehensive assessment of model practicality. In particular, the number of trainable parameters, floating-point operations (FLOPs), and execution time (training and inference) are reported for the DRN models using SELU and PSELU. Such complexity metrics are widely adopted in the literature as supplementary indicators of model efficiency and scalability, which are critical for real-world deployment in power system applications[47,48].

Default hyperparameters are adopted based on previous research. The optimizer used is adaptive moment estimation (Adam), equipped with an adaptive learning rate mechanism and initialized at 0.001[49]. All experiments are conducted using Python 3.8, with Keras 2.10.0 and TensorFlow 2.10.0 as the backend. Training is executed on a Lenovo laptop powered by an AMD Ryzen 7 6800 H processor and integrated Radeon Graphics, equipped with an NVIDIA GeForce RTX 3050 Ti Laptop GPU (4 GB VRAM) and 16 GB of Samsung DDR5 4800 MHz memory.

Evaluation metrics

To assess the effectiveness of various DRN models in STLF, researchers employ a variety of metrics to gauge prediction precision, as summarized in Table 1. Because of its easy interpretation and extensive application in these studies, MAPE is the most widely utilized and representative measure among them. 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) are further often used metrics. In addition, Mean Squared Relative Error (MSRE), Root Mean Squared Relative Error (RMSRE), Mean Absolute Relative Error (MARE), and Root Mean Squared Percentage Error (RMSPE) can also be employed to provide deeper insights into relative error distributions and percentage-based deviations. Depending on particular goals and dataset properties, many research may use distinct assessment measures. The related formulae are shown in Equations (15) through (25).

$$\mathrm{MAPE} = \frac{1}{\mathrm{N}} \sum_{\mathrm{i}\,=\,1}^{\mathrm{N}} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times\ 100 \tag{15}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{\mathrm{N}} \sum_{\mathrm{i}\ -\ 1}^{\mathrm{N}} (y_i - \hat{y}_i)^2} \tag{16}$$

$$MAE = \frac{1}{N} \sum_{i-1}^{N} |y_i - \hat{y}_i| \tag{17}$$

$$\mathrm{MSE} = \frac{1}{\mathrm{N}} \sum_{\mathrm{i}\ -\ 1}^{\mathrm{N}} (y_i - \hat{y}_i)^2 \tag{18}$$

$$NMSE = \frac{\sum_{i-1}^{N} (y_i - \hat{y}_i)^2}{N \cdot \sigma_y^2} \tag{19}$$

$$R = \frac{\sum_{i-1}^{N} (y_i - \bar{y}) \left(\hat{y}_i - \bar{\hat{y}}\right)}{\sqrt{\sum_{i-1}^{N} (y_i - \bar{y})^2 \sum_{i-1}^{N} \left(\hat{y}_i - \bar{\hat{y}}\right)^2}} \tag{20}$$

$$R^2 = 1 - \frac{\sum_{i-1}^{N} (y_i - \hat{y}_i)^2}{\sum_{i-1}^{N} (y_i - \bar{y}_i)^2} \tag{21}$$

$$\mathrm{MSRE} = \frac{1}{\mathrm{N}} \sum_{\mathrm{i}\,=\,1}^{\mathrm{N}} \left(\frac{y_i - \hat{y}_i}{y_i} \right)^2 \tag{22}$$

$$\mathrm{RMSRE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2} \tag{23}$$

$$\mathrm{MARE} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}}\left|\frac{y_i - \hat{y}_i}{\hat{y}_i}\right| \tag{24}$$

$$\mathrm{RMSPE} = 1 - \frac{\sum_{i-1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i-1}^{N}(y_i - \bar{y}_i)^2} \tag{25}$$

The parameters of these measures are as follows: NMSE computations employ the variance of the actual values, denoted by σ_{y}^2; the mean values of the actual and expected data, denoted by $\bar{y}$ and $\bar{y}'$; the total number of samples, denoted by N; the actual value for the i-th sample, denoted by $\mathrm{y_i}$; and the predicted value for the same sample, denoted by $\mathrm{y'_i}$. With the help of these characteristics, the metrics may evaluate performance in a more thorough way by taking into consideration factors like error magnitude, prediction precision, and the correlation between actual and anticipated values. While lower values for MAPE, RMSE, MAE, MSE, NMSE, MSRE, RMSRE, MARE, and RMSPE frequently imply fewer prediction errors and better generalization, higher R and R^2 values indicate better accuracy and model fitting.

Results and discussion

Forecasting performance on the ISO-NE dataset

The ISO-NE dataset was employed to examine the sensitivity of the proposed PSELU activation function to different γ values. As presented in Table 2, the model obtained the lowest Mean Absolute Percentage Error (MAPE = 1.662%) and the highest coefficient of determination (R^2 = 98.119%) when γ = 0.02. Across γ values ranging from 0.001 to 0.05, smaller γ values (0.01–0.02) consistently yielded lower MAPE and RMSE, whereas larger γ values resulted in a gradual increase in all error metrics. The adaptive variant (APSELU) produced slightly higher errors than the fixed-γ configurations, indicating that the fixed small negative-region adjustment was more stable under the tested conditions.

For model benchmarking, the DRN model was compared with several established deep-learning architectures, including CNN, LSTM, BiLSTM, GRU, BiGRU, Transformer, Transformer–LSTM, and CNN–LSTM–MMA. The comparative results in Table 3 show that the DRN model using SELU achieved the lowest MAPE (1.718%) and highest R^2 (97.957%) among these models, indicating that the residual-network framework remained competitive across different deep-learning baselines under identical experimental settings. Additionally, a previously reported CRN[26] attained a MAPE of 1.73% on a comparable dataset, whereas the DRN model using

γ	RMSE	MAE	MSE	NMSE	*R*	R^2
0.001	1.722	1.114	0.030	1.968	99.045	98.032
0.005	1.740	1.102	0.030	2.009	98.994	97.991
0.01	1.716	1.078	0.030	1.954	99.019	98.046
0.02	1.684	1.072	0.028	1.881	99.057	98.119
0.03	1.701	1.101	0.029	1.921	99.041	98.079
0.04	1.718	1.100	0.030	1.958	99.038	98.042
0.05	1.716	1.075	0.030	1.955	99.022	98.045
Adaptive	1.715	1.083	0.029	1.952	99.019	98.048
γ	**MAPE**	**MSRE**	**RMSRE**	**MARE**	**RMSPE**	
0.001	1.736	0.067	2.584	1.749	2.584	
0.005	1.705	0.069	2.634	1.703	2.634	
0.01	1.668	0.066	2.572	1.670	2.572	
0.02	1.662	0.064	2.535	1.659	2.535	
0.03	1.703	0.064	2.529	1.695	2.529	
0.04	1.671	0.065	2.549	1.677	2.549	
0.05	1.665	0.066	2.573	1.666	2.573	
Adaptive	1.681	0.069	2.622	1.672	2.622	

Table 2. Performance evaluation of DRN (PSELU) with different γ values on the ISO-NE dataset (All indicators ×100, expressed as percentages).

Model	RMSE	MAE	MSE	NMSE	*R*	R^2
CNN	1.930	1.234	0.037	2.471	98.764	97.529
LSTM	2.040	1.333	0.042	2.761	98.617	97.240
BiLSTM	1.968	1.242	0.039	2.569	98.744	97.431
GRU	2.040	1.333	0.042	2.761	98.617	97.240
BiGRU	1.768	1.122	0.031	2.075	98.981	97.925
Transformer	2.275	1.435	0.052	3.434	98.276	96.566
Transformer-LSTM	1.762	1.119	0.031	2.061	98.971	97.939
CNN-LSTM-MMA	1.865	1.216	0.035	2.307	98.852	97.693
SELU)	1.755	1.114	0.031	2.043	98.977	97.957
PSELU	1.684	1.072	0.028	1.881	99.057	98.119
Model	**MAPE**	**MSRE**	**RMSRE**	**MARE**	**RMSPE**	
CNN	1.903	0.083	2.878	1.889	2.878	
LSTM	2.075	0.094	3.059	2.074	3.059	
BiLSTM	1.914	0.086	2.933	1.895	2.933	
GRU	2.075	0.094	3.059	2.074	3.059	
BiGRU	1.737	0.070	2.642	1.743	2.642	
Transformer	2.193	0.107	3.269	2.209	3.269	
Transformer-LSTM	1.728	0.071	2.662	1.726	2.662	
CNN-LSTM-MMA	1.909	0.081	2.841	1.917	2.841	
SELU	1.718	0.068	2.616	1.713	2.616	
PSELU	1.662	0.064	2.535	1.659	2.535	

Table 3. Evaluation of performance measures in various models on the ISO-NE dataset (All indicators ×100, expressed as percentages).

Model	RMSE	MAE	MSE	NMSE	*R*	R^2
ReLU	1.891	1.193	0.036	2.372	98.820	97.628
Leaky ReLU	1.736	1.113	0.030	1.999	98.996	98.001
SELU	1.755	1.114	0.031	2.043	98.977	97.957
PSELU	1.684	1.072	0.028	1.881	99.057	98.119
Model	**MAPE**	**MSRE**	**RMSRE**	**MARE**	**RMSPE**	
ReLU	1.849	0.080	2.823	1.858	2.823	
Leaky ReLU	1.736	0.069	2.629	1.732	2.629	
SELU	1.718	0.068	2.616	1.713	2.616	
PSELU	1.662	0.064	2.535	1.659	2.535	

Table 4. Evaluation of performance measures of various activation Functions, including the proposed PSELU, in DRN on the ISO-NE dataset (All indicators ×100, expressed as percentages).

Model	Parameters	Peak Memory (MB)	FLOPs	Training Time (s)	Inference Time (s)	Inference Time / Sample (ms)
SELU	131,704	4.355	256,984	2983.179	3.219	8.820
PSELU	131,704	5.977	301,624	4918.109	6.959	19.066

Table 5. Computational complexity comparison between DRN (SELU) and DRN (PSELU) in ISO-NE dataset.

PSELU ($\gamma = 0.02$) further reduced the error to 1.662%, demonstrating a consistent performance improvement relative to both internal baselines and external references.

After replacing SELU with PSELU ($\gamma = 0.02$), the model further reduced MAPE from 1.718 to 1.662% and increased R^2 from 97.957 to 98.119%, as summarized in Table 4. The improvement is consistently observed across RMSE, MAE, and other percentage-based error metrics, suggesting a uniform performance gain rather than metric-specific fluctuations.

Table 5 summarizes the computational analysis of the DRN model using SELU and PSELU activations. Both configurations contained the same number of trainable parameters (131704). However, the model employing PSELU required higher computational resources. Specifically, peak memory usage increased from 4.355 MB

to 5.977 MB, the number of FLOPs rose from 256,984 to 301,624, and the average training runtime per full run increased from 2983.179 s to 4918.109 s. Similarly, inference time increased from 3.219 s to 6.959 s, corresponding to an average of 19.066 ms per sample compared with 8.820 ms per sample for the model using SELU. These findings indicate that introducing PSELU incurred a moderate computational overhead, reflected in higher memory usage, FLOPs, and training time, while maintaining the same parameter count.

Figure 9 illustrates the seasonal and hourly distribution of forecasting errors (MAPE). Higher errors occurred during summer afternoons (13:00–20:00) and winter evenings (16:00–19:00), corresponding to load peaks caused by cooling and heating demands, while spring and autumn exhibited smaller deviations. Figure 10 compares weekly load forecasts across four seasons, showing that the predicted curves followed the observed patterns closely, with slight deviations during peak-demand periods. These results collectively indicate that the proposed PSELU ($\gamma = 0.02$) activation function achieved lower overall errors across different seasonal conditions while maintaining consistent behavior throughout the study period.

Forecasting performance on the Malaysia dataset

The Malaysia dataset, representing a tropical climate with relatively stable yet weather-sensitive load variations, was used to further evaluate the proposed activation function. As summarized in Table 6, the DRN model using PSELU achieved its lowest MAPE (5.012%) and highest coefficient of determination ($R^2 = 92.920\%$) when $\gamma = 0.02$. Across γ values ranging from 0.001 to 0.05, smaller values (0.01–0.02) consistently produced lower MAPE and RMSE, whereas larger values led to performance degradation. The adaptive variant (APSELU) yielded slightly higher errors, indicating that a fixed small γ offered more stable behavior under the tested configuration.

To benchmark performance, the baseline DRN model using SELU was compared with several widely used deep-learning architectures, including CNN, LSTM, BiLSTM, GRU, BiGRU, Transformer, Transformer-LSTM, and CNN-LSTM-MMA. The comparative results in Table 7 show that the DRN model using SELU achieved the lowest MAPE (5.251%) and highest R^2 (92.799%) among these baselines, confirming its strong forecasting capability under consistent experimental conditions. Additionally, after substituting SELU with the proposed

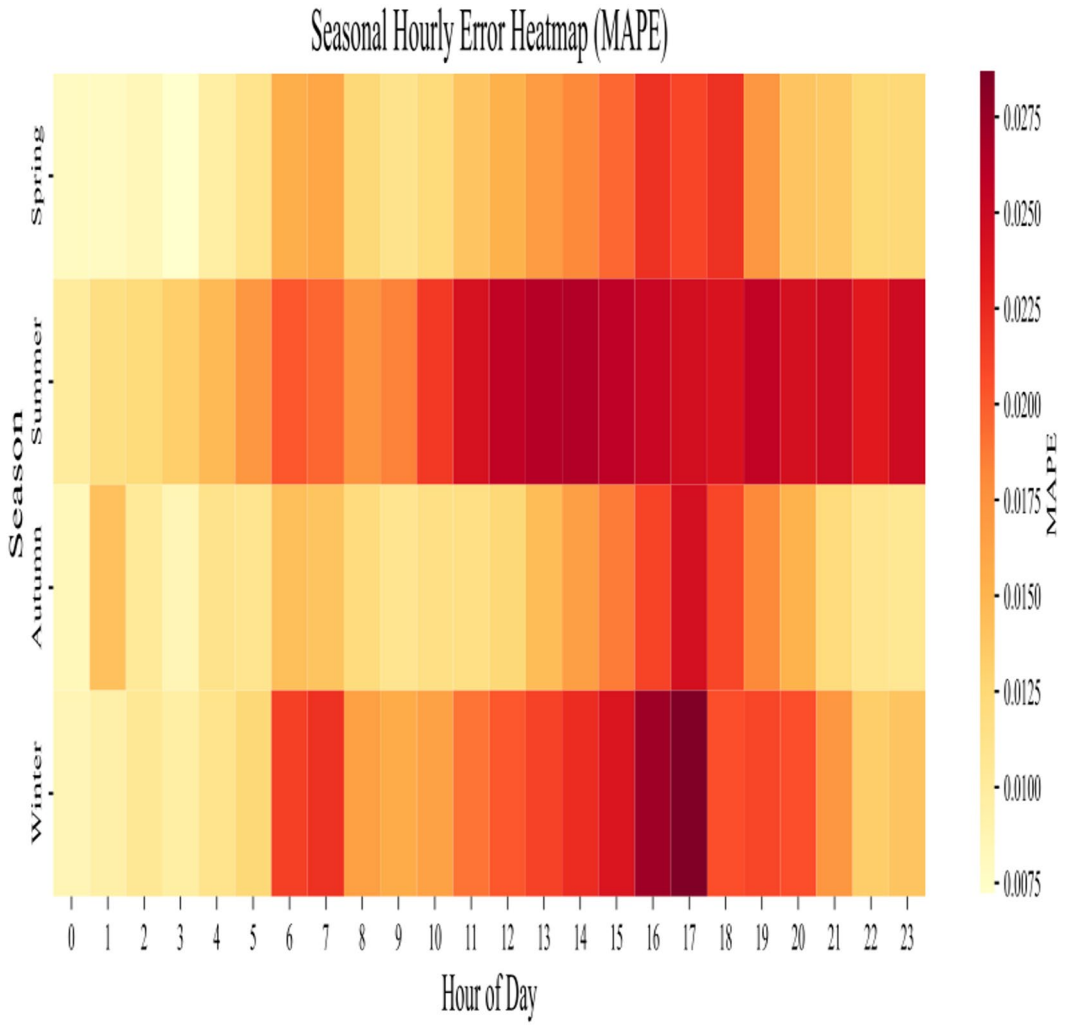

Fig. 9. Seasonal Hourly Error Heatmap (MAPE) in the ISO-NE Dataset. This heatmap was generated using Python 3.8 with TensorFlow 2.10.0 (https://www.tensorflow.org/).

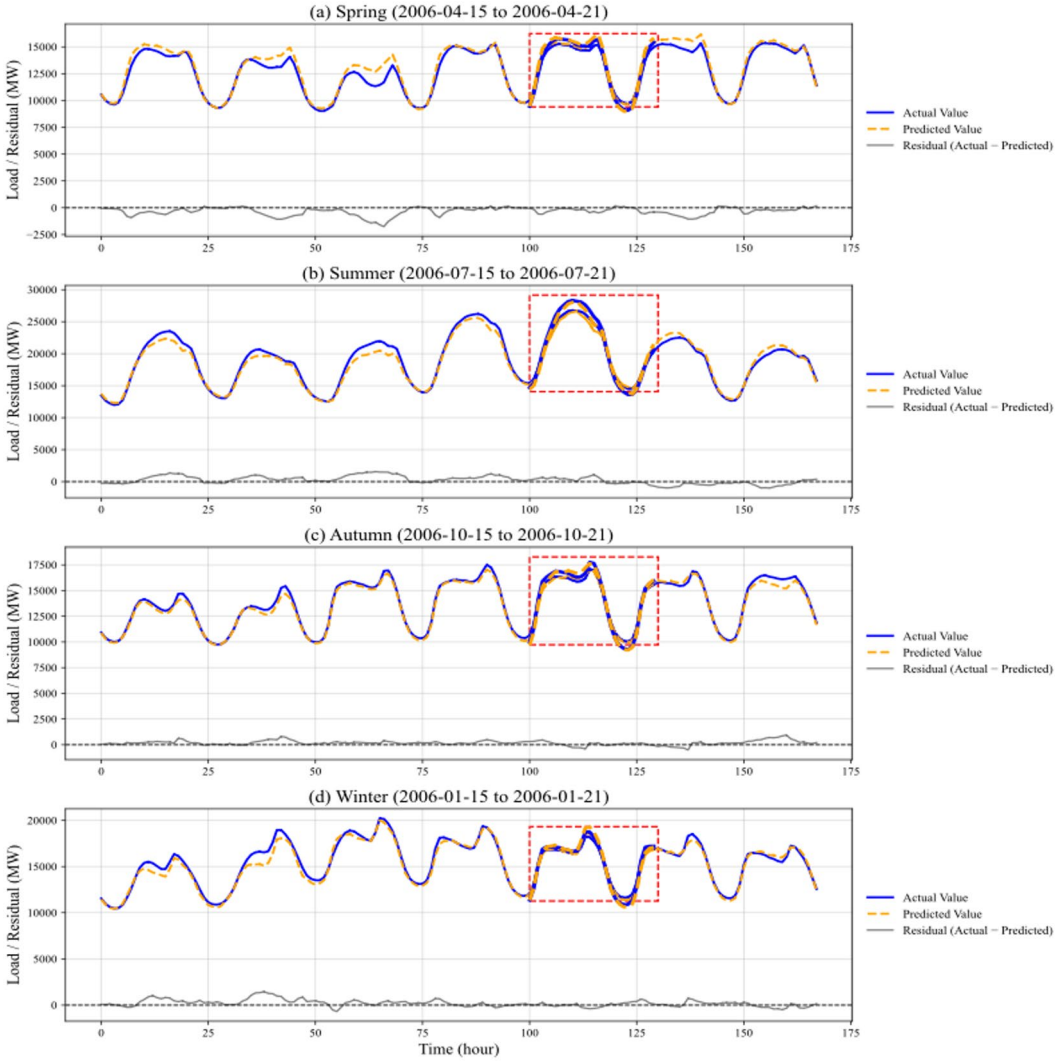

Fig. 10. Seasonal weekly STLF results in ISO-NE dataset: (**a**) Spring, (**b**) Summer, (**c**) Autumn, and (**d**) Winter.

PSELU ($\gamma = 0.02$), the model further reduced MAPE from 5.251% to 5.012% and slightly increased R^2 from 92.799% to 92.920%, as reported in Table 8. These improvements were accompanied by corresponding decreases in RMSE, MAE, and other percentage-based error metrics, indicating uniform performance enhancement across all evaluation indicators.

Table 9 summarizes the computational complexity comparison between the DRN model using SELU and the DRN model using PSELU. Both configurations contained the same number of trainable parameters (111544). However, the model employing PSELU required higher computational resources. Specifically, peak memory usage increased from 3.482 MB to 4.347 MB, the number of floating-point operations (FLOPs) rose from 218,104 to 249,784, and the average training runtime per full run increased from 1425.773 s to 2785.769 s. Similarly, inference time increased from 2.172 s to 4.414 s, corresponding to an average of 12.092 ms per sample compared with 5.952 ms per sample for the model using SELU. These results indicate a moderate rise in computational demand for the PSELU-based configuration, primarily evident in memory usage and processing time, while the overall network size remained unchanged.

Figure 11 shows the seasonal and hourly distribution of forecasting errors (MAPE). Larger deviations occurred during daytime hours, particularly from 08:00 to 17:00 in the dry season, reflecting higher commercial and cooling demand. Errors were lower during nighttime and early-morning periods. Figure 12 presents weekly forecasts for the dry and rainy seasons, indicating that the predicted curves closely followed the actual load profiles, with minor deviations during peak hours. These results demonstrate that the proposed activation function achieved lower forecasting errors across seasonal conditions while maintaining consistent predictive behavior under tropical load patterns.

Overall, the results on the Malaysia dataset indicate that the DRN incorporating the proposed PSELU ($\gamma = 0.02$) consistently achieved lower forecasting errors and maintained stable performance across both dry and rainy seasonal conditions, demonstrating reliability under tropical load characteristics.

γ	RMSE	MAE	MSE	NMSE	*R*	R^2
0.001	4.991	2.998	0.249	8.748	95.744	91.253
0.005	4.721	2.761	0.223	7.830	96.075	92.170
0.01	4.595	2.636	0.211	7.416	96.250	92.584
0.02	4.490	2.572	0.202	7.080	96.438	92.920
0.03	5.075	3.116	0.258	9.046	95.665	90.954
0.04	4.622	2.822	0.214	7.503	96.389	92.497
0.05	4.750	2.794	0.226	7.925	96.082	92.075
Adaptive	4.838	2.920	0.234	8.222	95.986	91.778
γ	**MAPE**	**MSRE**	**RMSRE**	**MARE**	**RMSPE**	
0.001	5.676	1.516	12.313	5.192	12.313	
0.005	5.297	1.351	11.623	4.779	11.623	
0.01	5.193	1.439	11.996	4.606	11.996	
0.02	5.012	1.271	11.272	4.526	11.272	
0.03	5.920	1.707	13.066	5.396	13.066	
0.04	5.360	1.305	11.425	4.909	11.425	
0.05	5.373	1.423	11.929	4.831	11.929	
Adaptive	5.609	1.504	12.262	5.091	12.262	

Table 6. Performance evaluation of DRN (PSELU) with different γ values on the Malaysia dataset (All indicators ×100, expressed as percentages).

Model	RMSE	MAE	MSE	NMSE	*R*	R^2
CNN	4.503	2.691	0.203	7.123	96.457	92.877
LSTM	5.047	2.921	0.255	8.947	95.493	91.053
BiLSTM	4.982	2.825	0.248	8.717	95.622	91.283
GRU	4.654	2.817	0.217	7.608	96.239	92.392
BiGRU	5.176	3.006	0.268	9.410	95.291	90.590
Transformer	4.625	2.674	0.214	7.512	96.171	92.488
Transformer-LSTM	4.534	2.646	0.206	7.222	96.372	92.779
CNN-LSTM-MMA	4.958	2.862	0.246	8.633	95.697	91.367
SELU	4.528	2.647	0.205	7.201	96.403	92.799
PSELU	4.490	2.572	0.202	7.080	96.438	92.920
Model	**MAPE**	**MSRE**	**RMSRE**	**MARE**	**RMSPE**	
CNN	5.310	1.309	11.442	4.786	11.442	
LSTM	5.698	1.854	13.618	5.035	13.618	
BiLSTM	5.437	1.516	12.313	5.046	12.313	
GRU	5.317	1.187	10.893	4.937	10.893	
BiGRU	5.720	1.572	12.539	5.272	12.539	
Transformer	5.402	1.574	12.545	4.814	12.545	
Transformer-LSTM	5.177	1.326	11.515	4.601	11.515	
CNN-LSTM-MMA	5.627	1.775	13.322	4.977	13.322	
SELU	5.251	1.400	11.832	4.644	11.832	
PSELU	5.012	1.271	11.272	4.526	11.272	

Table 7. Evaluation of performance measures in various models on the Malaysia dataset (All indicators ×100, expressed as percentages).

Statistical significance testing

To rigorously verify whether the observed differences among activation functions are statistically significant, a nonparametric Bootstrap resampling procedure with 10,000 iterations was performed. This method allows the estimation of confidence intervals and p-values without assuming normally distributed paired errors. For each comparison, the MAPE is reported together with its standard deviation (SD). The mean difference represents the average MAPE reduction between two models, and a 95% CI is obtained based on the Bootstrap samples. The corresponding p-values are also calculated within the same framework.

Model	RMSE	MAE	MSE	NMSE	R	R^2
ReLU	5.706	3.414	0.326	11.435	94.263	88.566
Leaky ReLU	5.397	3.114	0.291	10.231	94.798	89.769
SELU	4.528	2.647	0.205	7.201	96.403	92.799
PSELU	4.490	2.572	0.202	7.080	96.438	92.920
Model	**MAPE**	**MSRE**	**RMSRE**	**MARE**	**RMSPE**	
ReLU	6.446	1.966	14.022	6.040	14.022	
Leaky ReLU	6.128	2.251	15.005	5.358	15.005	
SELU	5.251	1.400	11.832	4.644	11.832	
PSELU	5.012	1.271	11.272	4.526	11.272	

Table 8. Evaluation of performance measures of various activation Functions, including the proposed PSELU, in DRN on the Malaysia dataset (All indicators ×100, expressed as percentages).

Model	Parameters	Peak Memory (MB)	FLOPs	Training Time (s)	Inference Time (s)	Inference Time / Sample (ms)
SELU	111,544	3.482	218,104	1425.773	2.172	5.952
PSELU	111,544	4.347	249,784	2785.769	4.414	12.092

Table 9. Computational complexity comparison between DRN (SELU) and DRN (PSELU) in Malaysia dataset.

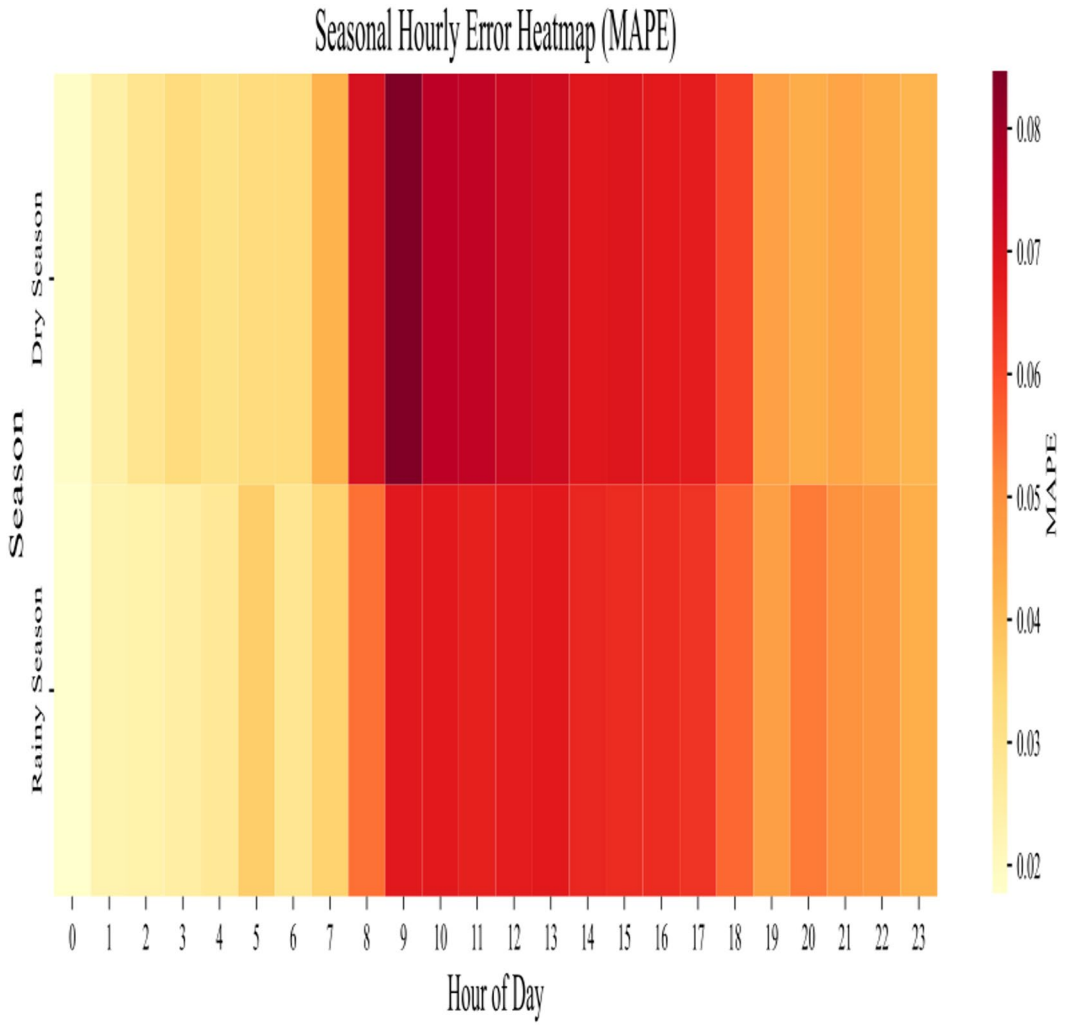

Fig. 11. Seasonal Hourly Error Heatmap (MAPE) in the Malaysia dataset. This heatmap was generated using Python 3.8 with TensorFlow 2.10.0 (https://www.tensorflow.org/).

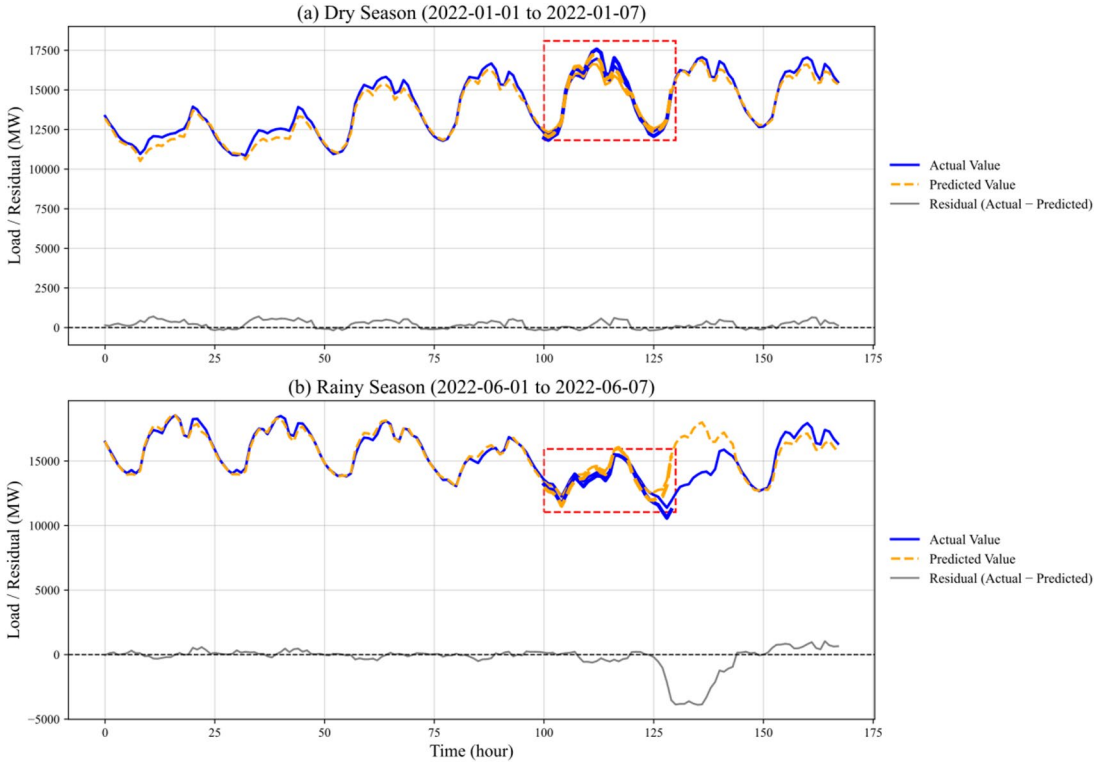

Fig. 12. Weekly STLF results in Malaysia dataset: (**a**) Dry season and (**b**) Rainy season.

1st Model	2nd Model	MAPE ± SD (1st Model)	MAPE ± SD (2nd Model)	Mean Difference	CI (95%)	*p*-value
ReLU	PSELU	1.849 ± 2.133	1.662 ± 1.915	0.187	[-0.032, 0.032]	≈ 0
Leaky ReLU	PSELU	1.736 ± 1.975	1.662 ± 1.915	0.074	[-0.029, 0.028]	≈ 0
SELU	PSELU	1.718 ± 1.973	1.662 ± 1.915	0.056	[-0.026, 0.026]	≈ 0

Table 10. Bootstrap test results for DRN models with different activation functions on the ISO-NE dataset (All indicators ×100, expressed as percentages).

1st Model	2nd Model	MAPE ± SD (1st Model)	MAPE ± SD (2nd Model)	Mean Difference	CI (95%)	*p*-value
ReLU	PSELU	6.446 ± 12.453	5.012 ± 10.097	1.434	[-0.127, 0.132]	≈ 0
Leaky ReLU	PSELU	6.128 ± 13.696	5.012 ± 10.097	1.116	[-0.142, 0.145]	≈ 0
SELU	SELU	5.251 ± 10.603	5.012 ± 10.097	0.239	[-0.068, 0.068]	≈ 0

Table 11. Bootstrap test results for DRN models with different activation functions on the Malaysia dataset (All indicators ×100, expressed as percentages).

The results for the ISO-NE dataset are summarized in Table 10. The DRN model using PSELU ($\gamma = 0.02$) achieved lower MAPE values compared with those using ReLU, Leaky ReLU, and SELU. The mean MAPE differences were 0.187 (95% CI: [− 0.032, 0.032]) compared with ReLU, 0.074 (95% CI: [− 0.029, 0.028]) compared with Leaky ReLU, and 0.056 (95% CI: [− 0.026, 0.026]) compared with SELU. In all cases, the reported p-values were approximately zero ($p < 0.0001$), indicating that these differences are statistically significant.

Similarly, the results for the Malaysia dataset, presented in Table 11, show that the DRN model using PSELU ($\gamma = 0.02$) produced lower MAPE values than those employing the other activation functions. The mean differences were 1.434 (95% CI: [− 0.127, 0.132]) compared with ReLU, 1.116 (95% CI: [− 0.142, 0.145]) compared with Leaky ReLU, and 0.239 (95% CI: [− 0.068, 0.068]) compared with SELU. The associated p-values were again approximately zero ($p < 0.0001$), confirming that the observed performance improvements are statistically significant.

The Bootstrap test results therefore provide consistent statistical evidence that the DRN model using PSELU ($\gamma = 0.02$) achieves significantly lower forecasting errors than the DRN models using ReLU, Leaky ReLU, or SELU across both datasets.

Summary

The experimental results across both datasets indicate that the DRN framework consistently achieves higher forecasting accuracy than other deep-learning architectures. In both the ISO-NE and Malaysia datasets, the DRN model using SELU obtained lower errors than CNN, LSTM, GRU, BiLSTM, BiGRU, Transformer, Transformer–LSTM, and CNN–LSTM–MMA, consistent with the stabilizing effects commonly associated with residual connections and self-normalizing activations.

Building on this baseline, the introduction of the proposed PSELU produced additional quantitative improvements. With the optimal parameter $\gamma = 0.02$, the DRN model using PSELU achieved lower forecasting errors than the model using SELU in both datasets. The sensitivity analysis further showed that small fixed γ values yielded the most stable and accurate results, whereas larger or adaptive values led to higher variance. Bootstrap significance testing confirmed that the observed improvements were statistically significant ($p < 0.0001$).

While PSELU slightly increases training and inference time compared with SELU, the additional computational cost remains proportionate to the achieved accuracy gains. In addition to numerical improvements, PSELU demonstrates stable training behavior across different seasonal conditions. This characteristic is associated with consistent forecasting performance when the load–temperature relationship changes across seasons, indicating reliable operation under seasonally non-stationary environments.

Analyses of seasonal error distributions show that PSELU reduces peak-hour forecasting deviations and improves prediction consistency in both temperate (ISO-NE) and tropical (Malaysia) systems. These findings demonstrate that the proposed activation function achieves higher forecasting accuracy and stability across varying operating conditions. Accordingly, PSELU provides an efficient and practical activation design for DRN-based STLF, achieving a balanced combination of adaptability, robustness, and accuracy.

Beyond forecasting accuracy, PSELU also shows stable feature–load relationships. The behavior in the negative activation region was associated with more consistent feature-sensitivity and partial-dependence evaluations, resulting in reliable attributions for temperature- and time-related variables. Error analyses further show that PSELU effectively captures seasonal variations such as summer/winter peaks in ISO-NE and rainy-season peaks in Malaysia. Nevertheless, limitations remain in dataset coverage, computational requirements, and model interpretability. Although the proposed approach has been validated on two representative datasets, further assessment using recent or renewable-rich datasets is necessary to confirm broader applicability.

In the broader landscape of time series forecasting, models often struggle with unstable convergence and limited adaptability to non-stationary data distributions. The PSELU activation function was therefore developed not merely to improve predictive accuracy, but to overcome inherent limitations of fixed-form activations such as SELU[42]—specifically, their reduced sensitivity to input variance and constrained gradient dynamics in DRNs. These limitations arise because SELU employs fixed scaling parameters that ensure self-normalization but restrict adaptability when input statistics vary, thereby constraining gradient propagation and limiting the model's responsiveness under shifting data distributions. By introducing a learnable scaling parameter, PSELU enhances self-normalization flexibility, stabilizes training under complex temporal conditions, and ultimately yields consistent performance improvements. Unlike the fixed scaling in SELU, the tunable parameter in PSELU allows the activation function to dynamically adjust its response to input variance, preserving self-normalization while promoting smoother gradient flow and more effective variance propagation in deep architectures.

Conclusion

This study introduced the PSELU activation function to enhance the performance of DRNs in STLF. The proposed function incorporates a small linear adjustment parameter γ in the negative activation region, inspired by Leaky ReLU. This design modifies the SELU formulation to improve representational flexibility while preserving stable network training observed in experiments.

Across two benchmark datasets with distinct climatic conditions—ISO-NE in a temperate region and Malaysia in a tropical region—the DRN model using PSELU ($\gamma = 0.02$) achieved lower forecasting errors than the DRN model using SELU and conventional activation functions such as ReLU and Leaky ReLU. Although the improvements were moderate, they were statistically validated through nonparametric Bootstrap testing, with all results showing $p < 0.0001$. Error-distribution analyses further revealed reduced peak-hour misestimations in both datasets. While computational requirements increased slightly, the increase remained moderate relative to the achieved improvement in forecasting accuracy.

Beyond technical performance, the results have practical relevance for large-scale power systems. Even small reductions in forecasting error are generally associated with measurable improvements in generation scheduling, reserve allocation, and demand-side management. Previous studies have shown that a 1% reduction in forecasting error can lead to cost savings for utilities operating at multi-gigawatt scales. The reductions in error observed in this study indicate potential operational benefits for grid reliability and efficiency under different climatic conditions.

Nevertheless, this study has certain limitations. The proposed PSELU focuses on functional-level modification within the activation domain rather than architectural innovation. As a result, it does not directly address forecasting challenges that may be better handled through model-level structural improvements[50], which represent a different line of advancement in time-series forecasting. Moreover, the evaluation is limited to DRN-based frameworks and two representative datasets, suggesting that broader validation across different network structures, data domains, and temporal spans would further substantiate its general applicability.

Future research could further explore combining activation-level refinements with architectural innovations to achieve complementary strengths and enhance model robustness under complex, real-world load variations. Expanding evaluation to more recent and diverse datasets that reflect modern grid conditions, investigating approaches to reduce computational overhead while maintaining accuracy, and applying the PSELU-enhanced DRN framework to other forecasting domains—such as renewable generation, electricity prices, and building energy consumption—are also promising directions.

In conclusion, the findings of this study demonstrate that activation functions remain an important component influencing the stability and adaptability of deep learning-based STLF models. The proposed PSELU provides an empirically supported modification to the DRN framework, achieving consistent improvements in forecasting accuracy and robustness. With continued validation and refinement, PSELU could serve as a practical activation design for developing reliable and adaptive STLF models suited to modern power system environments.

Data availability

The datasets generated and/or analysed during the current study are not publicly available due to licensing and institutional restrictions, but are available from the corresponding author upon reasonable request.

Appendix

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.

Received: 31 July 2025; Accepted: 10 December 2025
Published online: 07 January 2026

References

1. Ahmad, F. A., Liu, J., Hashim, F. & Samsudin, K. Short-term load forecasting utilizing a combination model: A brief review. *Int. J. Technol.* **15**, 121–129. https://doi.org/10.14716/ijtech.v15i1.5543 (2024).
2. Liu, J., Ahmad, F. A., Samsudin, K. & Hashim, F. Ab Kadir, M. Z. A. Performance evaluation of activation functions in deep residual networks for short-term load forecasting. *IEEE Access*. https://doi.org/10.1109/ACCESS.2025.3565798 (2025).
3. Hobbs, B. F. et al. Analysis of the value for unit commitment of improved load forecasts. *IEEE Trans. Power Syst.* **14**, 1342–1348. https://doi.org/10.1109/59.801894 (1999).
4. Wang, Y. & Wu, L. Improving economic values of day-ahead load forecasts to real-time power system operations. *IET Gener Transm Distrib.* **11**, 4238–4247. https://doi.org/10.1049/iet-gtd.2017.0517 (2017).
5. Ceperic, E., Ceperic, V. & Baric, A. A strategy for short-term load forecasting by support vector regression machines. *IEEE Trans. Power Syst.* **28**, 4356–4364. https://doi.org/10.1109/TPWRS.2013.2269803 (2013).
6. Hippert, H. S., Pedreira, C. E. & Souza, R. C. Neural networks for short-term load forecasting: A review and evaluation. *IEEE Trans. Power Syst.* **16**, 44–55. https://doi.org/10.1109/59.910780 (2002).
7. Kuster, C., Rezgui, Y. & Mourshed, M. Electrical load forecasting models: A critical systematic review. *Sustain. Cities Soc.* **35**, 257–270. https://doi.org/10.1016/j.scs.2017.08.009 (2017).
8. Cecati, C., Kolbusz, J., Różycki, P., Siano, P. & Wilamowski, B. M. A novel RBF training algorithm for short-term electric load forecasting and comparative studies. *IEEE Trans. Ind. Electron.* **62**, 6519–6529. https://doi.org/10.1109/TIE.2015.2424399 (2015).
9. Chen, Y. et al. Short-term load forecasting: similar day-based wavelet neural networks. *IEEE Trans. Power Syst.* **25**, 322–330. https://doi.org/10.1109/TPWRS.2009.2030426 (2009).
10. Zhao, Y., Luh, P. B., Bomgardner, C. & Beerel, G. H. Short-term load forecasting: Multi-level wavelet neural networks with holiday corrections. In Proc. IEEE Power & Energy Soc. Gen. Meet. 1–7IEEE, (2009). https://doi.org/10.1109/PES.2009.5275304
11. Eren, Y. & Küçükdemiral, İ. A comprehensive review on deep learning approaches for short-term load forecasting. *Renew. Sustain. Energy Rev.* **189**, 114031. https://doi.org/10.1016/j.rser.2023.114031 (2024).
12. Li, L., Ota, K. & Dong, M. Everything is image: CNN-based short-term electrical load forecasting for smart grid. In Proc. 14th Int. Symp. Pervasive Syst. Algorithms Netw. (ISPAN-FCST-ISCC) 344–351 (IEEE, 2017). https://doi.org/10.1109/ISPAN-FCST-ISCC.2017.78
13. Jurado, M., Samper, M. & Roses, R. An improved encoder–decoder-based CNN model for probabilistic short-term load and PV forecasting. *Electr. Power Syst. Res.* **217**, 109153. https://doi.org/10.1016/j.epsr.2023.109153 (2023).
14. Narayan, A. & Hipel, K. W. Long short-term memory networks for short-term electric load forecasting. In Proc. IEEE Int. Conf. Syst. Man Cybern. (SMC) 2573–2578 (IEEE, 2017). https://doi.org/10.1109/SMC.2017.8123012
15. Bento, P., Pombo, J. & Mariano, S. & do Rosário Calado, M. Short-term load forecasting using optimized LSTM networks via improved bat algorithm. In Proc. Int. Conf. Intell. Syst. (IS) 351–357 (IEEE, 2018). https://doi.org/10.1109/IS.2018.8710498
16. Kwon, B. S., Park, R. J. & Song, K. B. Short-term load forecasting based on deep neural networks using LSTM layer. *J. Electr. Eng. Technol.* **15**, 1501–1509. https://doi.org/10.1007/s42835-020-00424-7 (2020).
17. Tang, X., Dai, Y., Liu, Q., Dang, X. & Xu, J. Application of bidirectional recurrent neural network combined with deep belief network in short-term load forecasting. *IEEE Access.* **7**, 160660–160670. https://doi.org/10.1109/ACCESS.2019.2950957 (2019).
18. Ran, P., Dong, K., Liu, X. & Wang, J. Short-term load forecasting based on CEEMDAN and transformer. *Electr. Power Syst. Res.* **214**, 108885. https://doi.org/10.1016/j.epsr.2022.108885 (2023).
19. Jiang, B., Liu, Y., Geng, H., Zeng, H. & Ding, J. A transformer-based method with wide attention range for enhanced short-term load forecasting. In Proc. Int. Conf. Smart Power & Internet Energy Syst. (SPIES) 1684–1690 (IEEE, 2022). https://doi.org/10.1109/SPIES55999.2022.10082249
20. Li, S., Zhang, W. & Wang, P. TS2ARCformer: A multi-dimensional time series forecasting framework for short-term load prediction. *Energies* **16**, 5825. https://doi.org/10.3390/en16155825 (2023).
21. Chen, Y., Wang, Y., Liu, X. & Huang, J. Short-term load forecasting for industrial users based on Transformer–LSTM hybrid model. In Proc. IEEE 5th Int. Electr. Energy Conf. (CIEEC) 2470–2475 (IEEE, 2022). https://doi.org/10.1109/CIEEC54735.2022.9846658

22. Guo, W., Liu, S., Weng, L. & Liang, X. Power grid load forecasting using a CNN–LSTM network based on a multi-modal attention mechanism. *Appl. Sci.* **15**, 2435. https://doi.org/10.3390/app15052435 (2025).
23. Chen, K. et al. Short-term load forecasting with deep residual networks. *IEEE Trans. Smart Grid*. **10**, 3943–3952. https://doi.org/10.1109/TSG.2018.2844307 (2018).
24. Ding, A., Liu, T. & Zou, X. Integration of ensemble GoogLeNet and modified deep residual networks for short-term load forecasting. *Electronics* **10**, 2455. https://doi.org/10.3390/electronics10202455 (2021).
25. Li, H., Zhang, P. & Li, C. Short-term load forecasting for distribution substations based on residual neural networks and long short-term memory neural networks with attention mechanism. *J. Phys. Conf. Ser.* **2030**, 012087. https://doi.org/10.1088/1742-6596/2030/1/012087 (2021).
26. Sheng, Z., Wang, H., Chen, G., Zhou, B. & Sun, J. Convolutional residual network to short-term load forecasting. *Appl. Intell.* **51**, 2485–2499. https://doi.org/10.1007/s10489-020-01932-9 (2021).
27. Tian, Y., Yu, S., Wen, M., Zhang, K. & Chen, Y. Short-term load forecasting scheme based on improved deep residual network and LSTM. *CIRED CP767*. **2020**, 117–120. https://doi.org/10.1049/oap-cired.2021.0257 (2020).
28. Xu, Q., Yang, X. & Huang, X. Ensemble residual networks for short-term load forecasting. *IEEE Access.* **8**, 64750–64759. https://doi.org/10.1109/ACCESS.2020.2984722 (2020).
29. Kondaiah, V. Y. & Saravanan, B. Short-term load forecasting with deep learning. In Proc. Innovations Power Adv. Comput. Technol. (i-PACT) 1–5 (IEEE, 2021). https://doi.org/10.1109/i-PACT52855.2021.9696634
30. Chen, W., Han, G., Zhu, H., Liao, L. & Zhao, W. Deep ResNet-based ensemble model for short-term load forecasting in protection system of smart grid. *Sustainability* **14**, 16894. https://doi.org/10.3390/su142416894 (2022).
31. Kondaiah, V. Y. & Saravanan, B. A modified deep residual network for short-term load forecasting. *Front. Energy Res.* **10**, 1038819. https://doi.org/10.3389/fenrg.2022.1038819 (2022).
32. Sheng, Z., An, Z., Wang, H., Chen, G. & Tian, K. Residual LSTM based short-term load forecasting. *Appl. Soft Comput.* **144**, 110461. https://doi.org/10.1016/j.asoc.2023.110461 (2023).
33. Bingham, G. & Miikkulainen, R. Discovering parametric activation functions. *Neural Netw.* **148**, 48–65. https://doi.org/10.1016/j.neunet.2022.01.001 (2022).
34. Nader, A. & Azar, D. Evolution of activation functions: an empirical investigation. *ACM Trans. Evol. Learn. Optim.* **1**, 1–36. https://doi.org/10.1145/3464384 (2021).
35. Pourkamali-Anaraki, F., Nasrin, T., Jensen, R. E., Peterson, A. M. & Hansen, C. J. Adaptive activation functions for predictive modeling with sparse experimental data. *Neural Comput. Appl.* **36**, 18297–18311. https://doi.org/10.1007/s00521-024-10156-8 (2024).
36. Vaca-Rubio, C. J., Blanco, L., Pereira, R. & Caus, M. Kolmogorov-Arnold networks (KANs) for time series analysis. *ArXiv Preprint arXiv*. https://doi.org/10.48550/arXiv.2405.08790 (2024). :2405.08790.
37. Zhang, J. & Ding, C. Simple yet effective adaptive activation functions for physics-informed neural networks. *Comput. Phys. Commun.* **307**, 109428. https://doi.org/10.1016/j.cpc.2024.109428 (2025).
38. Zhang, S., Ren, G. & RoSwish A novel rotating swish activation function with adaptive rotation around zero. *Neural Netw.* **107892** https://doi.org/10.1016/j.neunet.2025.107892 (2025).
39. Neumann, O., Turowski, M., Mikut, R., Hagenmeyer, V. & Ludwig, N. Using weather data in energy time series forecasting: the benefit of input data transformations. *Energy Inf.* **6**, 44. https://doi.org/10.1186/s42162-023-00299-8 (2023).
40. Shi, H., Xu, M. & Li, R. Deep learning for household load forecasting—a novel pooling deep RNN. *IEEE Trans. Smart Grid*. **9**, 5271–5280. https://doi.org/10.1109/TSG.2017.2686012 (2017).
41. Wang, Y. Y. et al. Short-term probability density function forecasting of industrial loads based on ConvLSTM-MDN. *Front. Energy Res.* **10**, 891680. https://doi.org/10.3389/fenrg.2022.891680 (2022).
42. Klambauer, G., Unterthiner, T., Mayr, A. & Hochreiter, S. Self-normalizing neural networks. *Adv. Neural Inf. Process. Syst.* **30** https://doi.org/10.48550/arXiv.1706.02515 (2017).
43. Maas, A. L., Hannun, A. Y. & Ng, A. Y. Rectifier nonlinearities improve neural network acoustic models. In Proc. ICML 30, 3 (2013).
44. Huang, G. et al. Snapshot ensembles: train 1, get m for free. *ArXiv Preprint arXiv*. https://doi.org/10.48550/arXiv.1704.00109 (2017). :1704.00109.
45. Khoshkangini, R., Tajgardan, M., Lundström, J., Rabbani, M. & Tegnered, D. A snapshot-stacked ensemble and optimization approach for vehicle breakdown prediction. *Sensors* **23**, 5621. https://doi.org/10.3390/s23125621 (2023).
46. Johnston, M. G. & Faulkner, C. A bootstrap approach is a superior statistical method for the comparison of non-normal data with differing variances. *New. Phytol.* **230**, 23–26. https://doi.org/10.1111/nph.17159 (2021).
47. Fan, C. et al. A multi-stage ensemble model for power load forecasting based on decomposition, error factors, and multi-objective optimization algorithm. *Int. J. Electr. Power Energy Syst.* **155**, 109620. https://doi.org/10.1016/j.ijepes.2023.109620 (2024).
48. Liu, M., Xia, C., Xia, Y., Deng, S. & Wang, Y. TDCN: A novel Temporal depthwise convolutional network for short-term load forecasting. *Int. J. Electr. Power Energy Syst.* **165**, 110512. https://doi.org/10.1016/j.ijepes.2025.110512 (2025).
49. Adam, K. D. B. J. A method for stochastic optimization. ArXiv Preprint ArXiv:1412.6980 (2014). https://doi.org/10.48550/arXiv.1412.6980
50. Yadav, H., Thakkar, A. & TXtreme Transformer-based extreme value prediction framework for time series forecasting. *Discov Appl. Sci.* **7**, 98. https://doi.org/10.1007/s42452-025-06478-4 (2025).

Author contributions

J.L. and F.A.A. contributed to the conceptualization of the study. J.L. developed the methodology and conducted the investigation with F.A.A. J.L. prepared the original draft. J.L., F.A.A., K.S., F.H., and M.Z.A.A.K. contributed to the review and editing of the manuscript. F.A.A., K.S., F.H., and M.Z.A.A.K. provided supervision. All authors have read and agreed to the published version of the manuscript.

Funding

Declaration.

The authors declare that no external funding was received for this study.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to F.A.A.