Scientific Reports

https://doi.org/10.1038/s41598-026-35410-y

Article in Press

# Deep residual networks with convolutional feature extraction for short-term load forecasting

Received: 4 July 2025

Accepted: 6 January 2026

Published online: 27 January 2026

Cite this article as: Liu J., Ahmad F.A., Samsudin K. *et al.* Deep residual networks with convolutional feature extraction for short-term load forecasting. *Sci Rep* (2026). https://doi.org/10.1038/s41598-026-35410-y

**Junchen Liu, Faisul Arif Ahmad, Khairulmizam Samsudin, Fazirulhisyam Hashim & Mohd Zainal Abidin Ab Kadir**

# Deep Residual Networks with Convolutional Feature Extraction for Short-Term Load Forecasting

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

[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

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

**Abstract.** Conventional deep learning models struggle with balancing feature extraction and long-term temporal representation in Short-Term Load Forecasting (STLF). This study proposes a Convolutional Neural Network–Embedded Deep Residual Network (CNN-Embedded DRN) designed to enhance early-stage feature extraction and generalization capability across diverse climatic conditions. The objectives of this study are to integrate Convolutional Neural Network (CNN)-based local feature extraction into the DRN framework for capturing fine-grained temporal and spatial load patterns, to employ residual learning for mitigating gradient degradation and improving network stability, to evaluate the model's predictive performance against baseline and ablation models across two datasets representing temperate (ISO-NE) and tropical (Malaysia) climates, and to validate its statistical significance and seasonal robustness through bootstrap analysis and multi-seasonal evaluation. The results demonstrate that the proposed CNN-Embedded DRN consistently outperforms all comparative models, achieving the lowest Mean Absolute Percentage Error (MAPE) values of 1.5303% and 5.0566% on the ISO-NE and Malaysia datasets, respectively. The inclusion of residual network (ResNet) and CNN-Embedded ResNet as ablation experiments confirms that CNN-based local feature extraction effectively complements residual learning, while bootstrap analysis verifies that the observed improvements are statistically significant. The proposed model provides a reliable and generalizable framework for STLF, offering improved accuracy, robustness, and adaptability under varying climatic and demand conditions. Future research will focus on extending this framework toward multi-regional and multi-scale forecasting, incorporating attention mechanisms for enhanced long-term dependency modeling, and exploring adaptive hybrid residual architectures for real-time energy management applications.

**Keywords:** CNN, DRN, DNN, STLF.

## 1. Introduction

In order to provide reliable and efficient grid operation, load forecasting (LF) is a crucial part of modern power networks [1]. LF forecasts future energy usage to help power firms optimize grid operations, planning, and administration.

Maintaining supply consistency, cutting operational costs, and improving energy efficiency all depend on it. With rising energy consumption and shifting usage patterns, LF is becoming more and more sophisticated and significant.

LF may be divided into four categories: Very Short-Term Load Forecasting (VSTLF), Short-Term Load Forecasting (STLF), Medium-Term Load Forecasting (MTLF), and Long-Term Load Forecasting (LTLF) [2]. These groupings are defined by their temporal bounds. To fulfill crucial operating requirements, VSTLF makes preparations up to an hour in advance. STLF, which can last anywhere from an hour to a week, is necessary for dispatch and system operation. Mid-range planning, which includes supply management and maintenance scheduling, is the aim of MTLF. Its time frame ranges from a week to a year. Long-term infrastructure planning and strategic decision-making across a variety of years are made easier by LTLF. What distinguishes STLF from the others is its role in daily and weekly grid management, which includes forecasts for the next day or week.

Future power system management calls for more adaptability and speedier decision-making in the face of unpredictability. Applications that largely rely on STLF include energy trading, unit commitment, economic dispatch, and system reliability evaluation. Because precise prediction directly affects grid performance, its significance has increased. The importance of STLF reliability for daily operations and load flow planning is highlighted by the fact that forecasting mistakes can result in large unanticipated costs. For example, a 1% reduction in forecast error might result in an annual savings of up to $1.6 million for a 10000 Megawatt (MW) utility. Similarly, a 1% decrease in prediction inaccuracy can save hundreds of thousands or even millions of dollars for utilities with annual fuel expenses in the billions [3].

Traditional and current STLF methods are the two main categories. Conventional methods that frequently fail in real-world applications include linear, non-parametric (e.g., non-parametric regression, exponential smoothing, support vector regression (SVR), autoregressive models, and fuzzy logic. They may have poor generalization, overfit, or oversimplify complicated load dynamics as the number of input variables rises [4, 5].

To address these issues, artificial neural networks (ANNs) in particular have emerged as a potent alternative to STLF. By employing deep learning, ANN-based models may enhance prediction accuracy, decrease overfitting risks, and more accurately capture intricate load patterns. However, if a network gets more complex by adding more inputs, nodes, or layers, overfitting problems can still occur [6]. To improve model performance in STLF, ANN variants such as radial basis function (RBF) networks [7], wavelet-based networks [8], and extreme learning machines (ELM) [9] have been created.

In recent years, deep neural networks (DNNs), which are characterized by their layered architecture, have gained popularity because to their ability to learn hierarchical representations of complex load data. LF research has advanced from traditional shallow designs to intricate deep learning structures that employ several variables to represent intricate temporal and spatial relationships. This shift reflects the growing use of deep learning techniques to difficult forecasting issues [10].

Pre-made shallow network designs have been replaced in recent years by neural network topologies that integrate various inputs. Convolutional Neural Networks (CNNs), which are well known for their capacity to extract local characteristics, have been effectively used to detect temporal load patterns in STLF [11, 12]. However, their difficulty in training deeper systems and their inability to replicate long-term interactions restrict their utility in complex LF scenarios.

By using memory cells and gating methods, two representative forms of Recurrent Neural Networks (RNNs) that are excellent at simulating sequential data are Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) [13]. These structures mitigate the vanishing gradient issue and successfully capture both short-term and long-term dependencies in STLF [14]. However, when working with very lengthy sequences, LSTM and GRU are less effective because to their intrinsically sequential computation, which increases computational complexity [15]. By processing input in both forward and backward directions, more sophisticated variations like Bidirectional Long Short-Term Memory (BiLSTM) and Bidirectional Gated Recurrent Unit (BiGRU) improve sequence modeling even more [16], but at the expense of greater computational complexity.



Recently, transformer-based models have drawn interest in STLF because of their capacity to use self-attention processes to capture long-range dependencies [17]. Because these models enable parallel processing and flexible sequence management, they show good performance in time series forecasting applications. Transformers are less appropriate for very long sequences, though, since their computing cost rises quadratically with sequence length [18]. Furthermore, in order to solve training stability and convergence problems, deeper Transformer topologies frequently need structural improvements [19].

To address gradient-related issues in deep networks, Chen et al. [20] proposed a deep residual network (DRN) for STLF, drawing inspiration from the residual network (ResNet), which introduces identity shortcut connections to effectively mitigate vanishing gradients and enhance training stability and representational capacity. The DRN model consists of two main components: a basic structure responsible for early-stage feature extraction, and a prediction layer based on a

modified ResNet structure (ResNetPlus) that further refines the output through deep residual learning. By leveraging historical load, temperature, and time features, this architecture enables robust and scalable deep learning without relying on extensive manual feature engineering. In contrast, traditional models such as CNN, RNN, and Transformer face specific challenges when scaled deeper: CNN struggles with capturing long-term dependencies, RNN is prone to gradient vanishing and high computational cost, and Transformer often requires substantial resources and may become unstable in deeper layers. DRN, however, achieves a better balance among representational power, training stability, and network depth, making it an effective and widely adopted approach for building high-performance STLF models.

In recent years, DRN-based STLF models have increasingly integrated deep learning modules to enhance performance. For example, Tian et al. [21] introduced LSTM after the ResNetPlus model to strengthen temporal modeling capabilities; Li et al. [22] adopted a similar architecture by placing an LSTM layer after the ResNet model and further incorporating an attention mechanism, which improved the model's ability to capture key information and enhanced the final forecasting performance. In addition, Sheng et al. [23] and Sheng et al. [24] respectively embedded CNN and LSTM modules into the prediction layer of DRN to reinforce local feature extraction and temporal dependency modeling. However, these approaches mainly focus on optimization at the prediction stage, overlooking the modeling potential of DRN during the early stages of feature extraction, thus limiting the model's expressive power under complex load dynamics.

To improve the feature extraction capability of DRN, Ding et al. [25] proposed the GoogLeNet-ResNetPlus model, incorporating the Inception convolutional structure of GoogLeNet into the DRN feature extraction layer to enhance multi-scale load pattern recognition. This study demonstrated the significant potential of convolutional structures in strengthening DRN's local feature modeling. Other research also confirmed the effectiveness of CNN for local feature extraction in STLF tasks. For instance, Cui et al. [26] validated the ability of CNN to extract spatial features under diverse climatic conditions, significantly improving the model's generalization performance; Hua et al. [27] utilized CNN to extract local features from load and weather variables, and integrated them with temporal modeling techniques, effectively reducing prediction errors. Despite these findings highlighting the strength of CNN in feature extraction for STLF, current DRN models still deploy CNN modules only at the prediction stage, failing to fully engage them in the early feature extraction process [23]. This decoupled design restricts the expressive power of CNN in deep modeling and hinders optimal synergy from local feature learning to global prediction.

Therefore, this study proposes an innovative approach that embeds CNN directly

into the foundational structure of DRN. By integrating CNN at the early feature extraction stage, the model's ability to capture local patterns and short-term fluctuations is significantly improved, enhancing prediction accuracy and generalization. While previous studies have typically embedded deep learning modules within the prediction layer of DRN or appended them as sequential components after the DRN output, this design achieves, for the first time, deep integration of CNN within the DRN basic structure, offering a new direction for optimizing shallow-layer modeling. The proposed CNN-Embedded DRN architecture enhances feature representation in the early stages, improves robustness, and maintains training stability. Empirical evaluations on two benchmark datasets confirm that the method outperforms traditional DRN and mainstream models in both accuracy and generalization, demonstrating strong adaptability and practical value.

To clearly define the focus of this study, the main objectives are fourfold: (1) to integrate CNN-based local feature extraction into the foundational structure of DRN for enhancing early-stage representation learning; (2) to employ residual connections to mitigate gradient degradation and ensure stable training in deep forecasting networks; (3) to evaluate the proposed model’s predictive performance against multiple baseline and ablation models—including CNN, LSTM, GRU, Transformer, ResNet, and DRN—using datasets from distinct climatic regions (temperate and tropical); and (4) to validate the model’s robustness and generalization capability through bootstrap-based statistical analysis and seasonal evaluation. By achieving these objectives, the proposed CNN-Embedded DRN aims to provide a more accurate, stable, and generalizable framework for STLF across varying climatic and demand conditions.

The remainder of this paper is organized as follows: Section 2 reviews representative deep learning paradigms for STLF, analyzes the foundational architecture and limitations of DRN–based models, and introduces the basic principles of CNN to motivate their integration into the DRN framework. Section 3 introduces the proposed CNN-Embedded DRN model, including data preprocessing, feature design, and architectural details. Section 4 presents experimental results and comparative analyses on the New England Independent System Operator (ISO-NE) and Malaysia datasets, examining performance across different configurations, baseline models, and seasonal conditions. Section 5 concludes the paper and discusses potential directions for future work.

## 2. Related Work

This section reviews existing studies related to deep learning–based STLF, with a particular focus on DRN architectures. It first summarizes commonly used deep learning approaches for STLF, including convolution-based, recurrent-based, and attention-based models. Subsequently, the fundamental principles and architectural characteristics of DRN are introduced. The limitations of existing

DRN-based frameworks are then analyzed, highlighting their insufficient exploitation of feature extraction at the foundational level. Finally, the foundational architecture of CNN is reviewed to motivate their integration into the DRN framework.

### 2.1 Deep Learning-Based Methods for Short-Term Load Forecasting

With the rapid growth of data availability and computational resources, deep learning techniques have become increasingly prominent in STLF due to their strong capability in modeling nonlinear relationships and complex temporal patterns. Over the past decade, deep learning–based approaches for STLF have been widely adopted, and the commonly used methods mainly include convolution-based models, recurrent-based models, and attention-based models, each emphasizing different aspects of feature representation and temporal dependency learning.

CNNs have been widely applied in STLF for extracting local temporal patterns and short-range dependencies. Li et al. [11] proposed a CNN-based forecasting approach that transforms load time series into image-like representations to enable spatial feature extraction through convolution operations. The model effectively improved forecasting accuracy across most time points; however, the reliance on image preprocessing and a dual-branch architecture increases system complexity and limits scalability for large-scale or real-time applications. Jurado et al. [12] further developed an encoder–decoder CNN framework combined with Monte Carlo Dropout and probabilistic density estimation to enhance uncertainty modeling in STLF. While the approach demonstrated notable improvements over conventional recurrent baselines, its forecasting performance deteriorated during peak demand periods, indicating limitations in capturing extreme load variations.

RNNs, particularly LSTM networks, have been extensively employed in STLF owing to their ability to capture sequential dependencies. Narayan and Hipel [13] developed a deep LSTM-based framework for regional hourly load forecasting, achieving improved performance compared with traditional statistical and shallow neural models across multiple seasons. Nevertheless, the absence of exogenous variables such as meteorological factors may restrict the model’s adaptability in dynamic operating environments. To further enhance LSTM performance, Bento et al. [14] introduced an optimized LSTM architecture using metaheuristic-based hyperparameter tuning, resulting in improved forecasting accuracy. However, the iterative optimization process introduces additional computational overhead, which may limit its applicability in large-scale forecasting systems.

Bidirectional recurrent architectures have also been explored to strengthen temporal feature learning. Kwon et al. [15] proposed a stacked BiLSTM model

with feedback mechanisms, demonstrating strong accuracy in day-ahead forecasting scenarios. Despite its effectiveness under typical conditions, the robustness of the model under irregular load patterns, such as holidays and abnormal events, was not comprehensively evaluated. From a hybrid modeling perspective, Tang et al. [16] proposed a complex architecture combining deep belief networks, Bidirectional RNNs, and ensemble empirical mode decomposition. Although this model exhibited strong capability in capturing peak load behavior, its multi-stage training process and high structural complexity pose challenges for real-time deployment.

In recent years, attention-based and Transformer architectures have attracted increasing interest in STLF due to their ability to model long-range dependencies through self-attention mechanisms. Ran et al. [17] integrated empirical mode decomposition techniques with a Transformer framework to enhance temporal feature representation, demonstrating improved forecasting performance. However, the reliance on fixed decomposition parameters and prolonged training time may limit adaptability to new datasets. Jiang et al. [18] proposed a Transformer-based STLF model with an expanded attention range, which improved prediction accuracy compared with conventional attention mechanisms but incurred higher memory consumption. To further enhance multidimensional temporal representation, Li et al. [19] proposed TS2ARCformer, a Transformer-based forecasting framework that integrates contextual encoding, cross-dimensional attention, and autoregressive components. While the model demonstrated superior performance on public datasets, its hierarchical attention structure and autoregressive integration increased architectural complexity and tuning difficulty for general STLF applications.

Taken together, convolution-based, recurrent-based, and attention-based models have each contributed to improving STLF by focusing on local feature extraction, sequential dependency modeling, and long-range dependency learning, respectively. However, these approaches are often developed to emphasize specific modeling capabilities and may encounter challenges when deeper architectures are required to jointly capture complex temporal patterns and nonlinear relationships. In particular, as network depth increases, training stability and performance degradation can become critical issues. These observations suggest the need for a more stable and scalable learning framework that can support deep model construction. In this context, DRNs, by introducing residual connections, provide an effective mechanism for alleviating gradient-related issues and enabling deeper architectures, thereby offering a solid foundation for advanced STLF modeling.

### 2.2 Deep Residual Network for Short-Term Load Forecasting

Motivated by the training instability and performance degradation observed in deep learning models for STLF, residual learning has been introduced as an

effective strategy to facilitate the construction of deeper neural networks. By reformulating the learning objective through identity-based shortcut connections, residual learning alleviates gradient-related issues and enables stable optimization of deep models. This concept, originally introduced in ResNet, was subsequently adapted and extended to deep network frameworks, leading to the development of DRNs.

### 2.2.1 Foundational Architecture

The DRN is designed to capture the complex nonlinear relationships between input components and the expected outcome [28]. Increasing the depth of a neural network usually improves its learning capabilities, but paradoxically, this can also cause performance to decline. This decline in effectiveness might be due to the complexity of the input data or the intricate structure of the model. The architecture incorporates residual blocks to mitigate these challenges. Instead of focusing on a simple input-to-output translation, these blocks focus on learning the residual function. By employing residual connections, this design improves gradient flow, lessens the likelihood of disappearing gradients, and makes it easier to train deeper networks effectively. As seen in Fig 1, a ResNet is made up of two subsequent layers connected by a skip link.

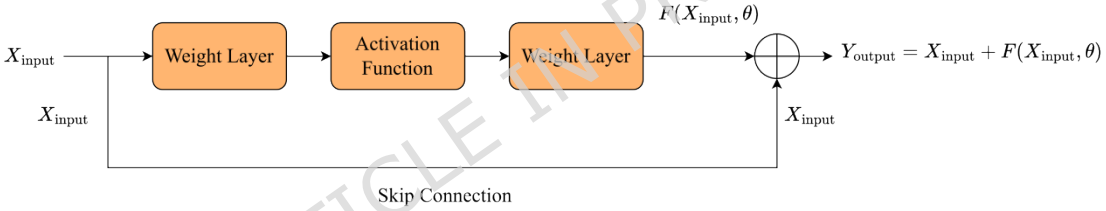



Fig 1 The foundation block of the ResNet [28].

Equation (1) demonstrates how the skip connection, which functions as an identity mapping, generates the ResNet output when the input and output dimensions coincide, where $x_{Input}$ stands for the ResNet's input, $y_{output}$ for the block output, $F$ for the residual mapping function, and $\Theta$ for the function's learnable parameters.

$$y_{output} = x_{Input} + F(x_{Input}, \Theta) \quad (1)$$

When the dimensions of the input and output are different, the skip connection makes a linear projection. This linear projection ($L_p$) is sometimes included in the ResNet output, as seen in Equation (2):

$$y_{output} = L_p * x_{Input} + F(x_{Input}, \Theta) \quad (2)$$

### 2.2.2 Model Framework

The model comprises two main components: a basic structure and an enhanced ResNet variant called ResNetPlus, which together strengthen the model's feature extraction and predictive capability as shown in Fig. 2 [20]. The core architecture is the fundamental framework that uses several linked layers to

extract fundamental information in order to provide the first 24-hour load projections. The predictions are then further refined by integrating ResNetPlus, which maintains the original ResNet's block structure while including enhancements to boost accuracy and computing efficiency. The Scaled Exponential Linear Unit (SELU) is employed as the activation function in both the basic structure and the ResNetPlus layers, promoting self-normalizing properties and stabilizing training in deeper networks, hence promoting robust learning. This combination ensures accuracy and scalability in STLF jobs by effectively managing both short-term and long-term dependencies. The SELU may be expressed using Equation (3).

$$f(x)=\begin{cases}\lambda x & \text{if } x>0\\ \lambda\alpha(e^{x}-1) & \text{if } x\leq 0\end{cases} \quad (3)$$

where $\lambda\approx 1.05$ is a normalization scaling factor, $\alpha\approx 1.67$ corrects the output for negative inputs, and $x$ is the input value. SELU guarantees self-normalization and stabilizes the mean and variance between layers.

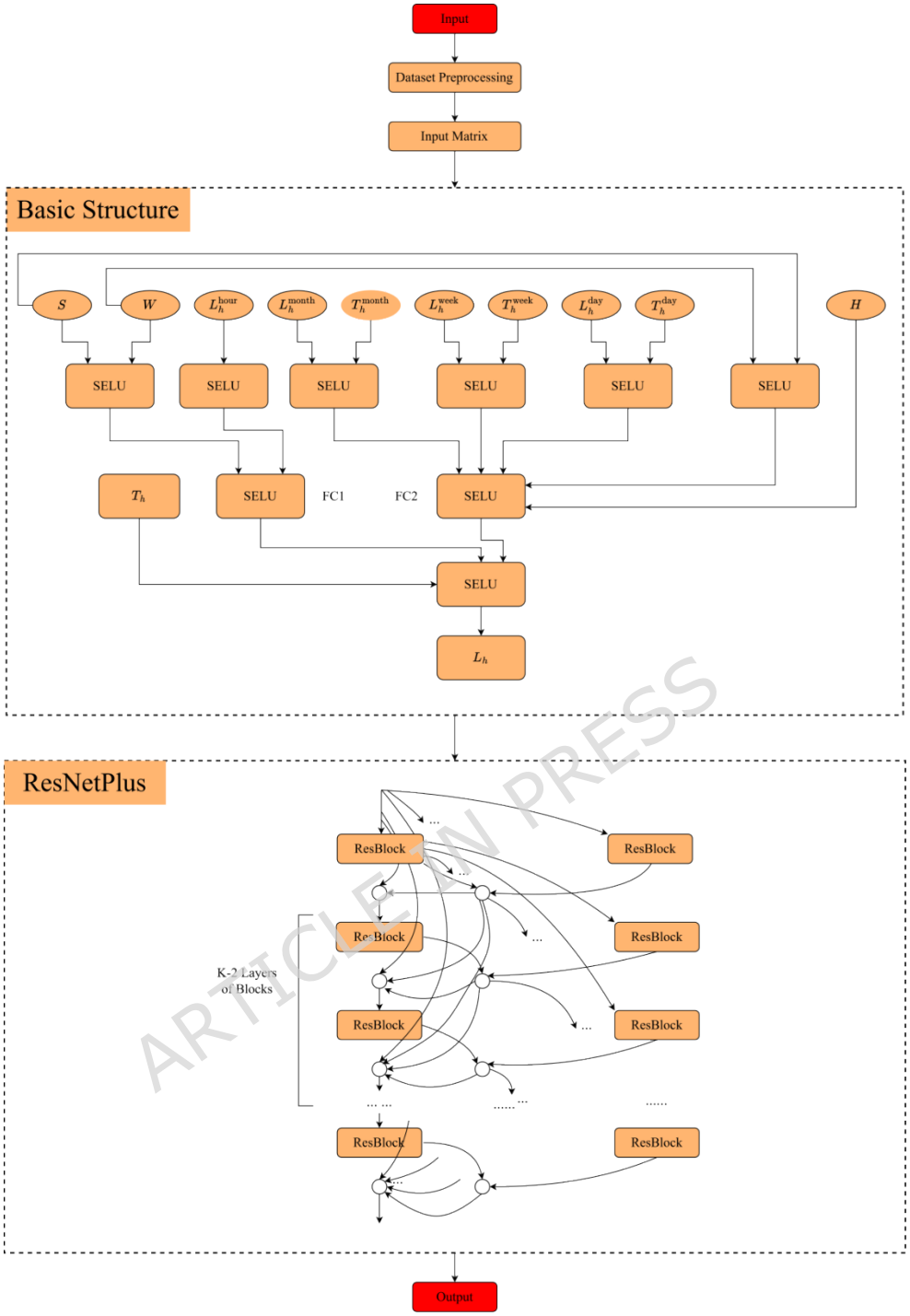



Fig.2 The original framework [20].

The "basic structure" of the model is a neural network, which is made up of several interconnected layers. With this simple design, the initial load prediction for the next 24 hours is generated. Ten hidden nodes are present in each fully connected (FC) layer, denoted by $[L_h^{day},T_h^{day}]$, $[L_h^{week},T_h^{week}]$, $[L_h^{month},T_h^{month}]$ and $L_h^{hour}$ in this topology. Five hidden nodes linked to weekdays and season [W, S] are present in each FC layer in the interim. The fully linked layers FC1, FC2, and the layer before $L_h$ also include ten hidden nodes. It's interesting to note that all

but the output layer have activation functions. The load values for the corresponding hour from 1, 2, and 3 months before to the target day are represented by $L_h^{week}$ in this basic structure, while the load values for the same hour from 1 to 8 weeks earlier are represented by $L_h^{day}$. Furthermore, while $L_h^{hour}$ logs the load values for the same hour for the previous 24 hours, $L_h^{day}$ displays the load values for the same hour on each day of the previous week. Additionally, temperature values in $T_h^{month}$, $T_h^{week}$ and $T_h^{day}$ match those in $L_h^{month}$, $L_h^{week}$ and $L_h^{day}$, respectively. The letter $T_h$ stands for the actual expected temperature for the next day. The one-hot encoded inputs $S$, $W$, and $H$ stand for the season, weekday, and holiday status, respectively. The output of this fundamental structure is used as input in the second phase of the model to increase the forecast's accuracy.

The ResNetPlus model is an advanced advancement in neural network architecture that maintains the core concepts of the original ResNet while introducing notable improvements. This enhanced version includes residual blocks, each consisting of two layers: a hidden layer with 20 neurons activated by the same nonlinear function (SELU) as used in the basic structure, and a linear transformation layer employed to match the feature dimensions required for residual addition. The model regularly replicates this structure across 10 levels, building four of these blocks in succession, each with its own connections, to provide a substantial amount of depth and intricacy. A shortcut connection, which connects the output of the preceding block directly to the network's input, is one of ResNetPlus's special features. This idea simplifies the building of a DRN while increasing its overall efficiency. ResNetPlus maintains the hyperparameters used in the original ResNet blocks while optimizing the architecture to maximize the ResNet design's capabilities.

Equation (4) shows how the two components are added to determine the total loss, or $Loss$, in order to effectively train the models:

$$Loss = Loss_E + Loss_R \quad (4)$$

To enable a more effective training procedure, $Loss_E$ computes the prediction error and $Loss_R$ serves as a penalty for values that are outside the range. is accurately explained by Equation (5):

$$Loss_E = \frac{1}{NumH} \sum_{j=1}^{N} \sum_{h=1}^{H} \frac{\left| \grave{y}_{(j,h)} - y_{(j,h)} \right|}{y_{(j,h)}} \quad (5)$$

The expected production is represented by $\grave{y}_{(j,h)}$ in this equation, whereas the actual normalized load for the $h$th hour of the $j$th day is represented by $y_{(j,h)}$. While $H$ (in this example set to 24) denotes the number of hourly load values

each day, the variable refers to the total number of data samples. Furthermore, Equation (6) defines $\mathrm{Loss_R}$:

$$\mathrm{Loss_R} = \frac{1}{2\mathrm{Num}} \sum_{j=1}^{\mathrm{Num}} \max\left(0, \max_h \grave{y}_{(j,h)} - \max_h y_{(j,h)}\right) + \max\left(0, \min_h y_{(j,h)} - \min_h \grave{y}_{(j,h)}\right) \quad (6)$$

This term speeds up early training and emphasizes the penalty of overestimating peaks and underestimating troughs in the load curves as the model's predictions get more accurate by penalizing the model when the predicted daily load curve deviates outside the actual load range.

### 2.2.3 Current Restrictions

Residual learning has been widely adopted in deep time-series forecasting models and is generally regarded as an effective strategy for stabilizing the training of deep architectures while enhancing feature representation capability. In recent years, within the domain of general time-series forecasting, Challu et al. [29] proposed neural hierarchical interpolation for time series forecasting (N-HiTS), which extends residual-based forecasting by introducing a hierarchical multi-scale residual structure. By progressively decomposing time series into different temporal resolutions and refining forecasts through stacked residual blocks, N-HiTS achieves improved accuracy while maintaining stable optimization behavior. Despite its strong performance on benchmark datasets, the model primarily focuses on univariate or general-purpose time-series forecasting and does not explicitly consider domain-specific characteristics or exogenous variables, which may limit its applicability to complex real-world forecasting tasks.

In spatio-temporal modeling tasks such as urban demand and traffic flow prediction, residual learning has also demonstrated notable advantages. Zhang et al. [30] developed a spatio-temporal residual graph attention network, in which residual connections are embedded within a graph attention framework to jointly model temporal dynamics and spatial correlations. Although this approach enhances the ability to capture complex spatio-temporal dependencies, it typically relies on sophisticated graph construction and attention mechanisms, which may limit scalability and computational efficiency. Based on a similar residual learning paradigm, Bao and Yang [31] proposed a global–local spatio-temporal residual correlation network for traffic state prediction. By leveraging multi-scale residual structures, this model effectively integrates global evolution trends with local dynamic variations. However, its network design is primarily tailored to specific traffic scenarios, and its generalization capability to other types of time-series forecasting tasks remains to be further validated.

Residual learning has subsequently been extended to energy-related forecasting tasks. Ashebir and Kim [32] combined residual blocks with variational modeling and recurrent neural networks to develop a temporal variational residual

framework for energy demand forecasting, significantly enhancing the model's ability to capture multi-scale fluctuations, uncertainty, and nonlinear relationships. Nevertheless, the introduction of additional probabilistic modeling components increases architectural complexity, making the training process more sensitive to parameter initialization and convergence stability. Similarly, in the context of urban public transportation systems, Zhang et al. [33] proposed a deep residual learning-based framework for short-term passenger flow forecasting. By incorporating ResNet-style skip connections, the model alleviates gradient degradation in deep architectures and improves training stability. However, it still largely relies on conventional spatio-temporal feature modeling strategies and exhibits limited capability in capturing long-term temporal dependencies and cross-scale feature interactions.

Taken together, existing studies have convincingly demonstrated the effectiveness of residual learning in time-series forecasting, traffic systems, and energy prediction tasks, particularly in stabilizing deep model training and enhancing feature representation. However, most of these approaches are designed for general time-series or domain-specific applications and do not explicitly address the distinctive characteristics of STLF, such as strong periodicity, multi-scale temporal dependencies, and complex nonlinear interactions between load demand and exogenous variables. In contrast, DRN, through their hierarchical residual structures, provide a more systematic architectural foundation for constructing deeper and more stable models that are better aligned with the intrinsic properties of this task. Consequently, further exploration of how to enhance temporal feature extraction and long-term dependency modeling within a DRN-based framework remains a critical research direction in this domain.



In current research on STLF based on DRN, many hybrid models have incorporated deep learning components to enhance prediction accuracy. However, most of these methods integrate such modules only at the model's output or prediction stage, failing to fully exploit the modeling potential of the foundational structure within DRN during the feature extraction process. This design limits the model's capacity to deeply learn local patterns and dynamic variations in the early stages of feature extraction, thereby hindering further improvements in overall forecasting performance.

For instance, Tian et al. [21] proposed the ResNetPlus-LSTM model by placing the LSTM module directly after ResNetPlus to enhance temporal modeling capability. Building on this, Li et al. [22] introduced the ResNet-LSTM-Attention model, which incorporates LSTM and attention mechanisms after ResNet to improve attention to critical information and final forecasting performance.

Some studies have attempted to embed deep learning modules into the internal

structure of DRN. Sheng et al. [23] proposed the convolutional residual network (CRN) model, which integrates CNN modules into the prediction layer of the ResNet framework to strengthen local feature extraction. Sheng et al. [24] further proposed the Residual LSTM Plus model, embedding LSTM into the prediction layer of DRN to enhance temporal modeling. Although these studies achieved internal integration of the modules, they remain primarily focused on the prediction phase, failing to fully exploit the feature extraction potential of the shallow layers within DRN.

In addition, Ding et al. [25] incorporated the GoogLeNet structure into the GoogLeNet-ResNetPlus model to improve the recognition of complex load patterns at multiple scales. GoogLeNet, belonging to the Inception network family, features parallel multi-scale convolution paths for feature extraction, enabling the integration of hierarchical information while maintaining computational efficiency. This study demonstrates that convolutional structures have significant potential in enhancing DRN's feature extraction capabilities, particularly in spatial and local pattern modeling.

Further research has validated the effectiveness of CNN in feature extraction for STLF tasks. Cui et al. [26] demonstrated CNN's strength in capturing spatial features, significantly improving model generalization and robustness under diverse climate conditions. Hua et al. [27] applied CNN to extract local features from load and meteorological variables and combined it with temporal modeling structures, effectively reducing prediction error and enhancing responsiveness to dynamic load changes. These studies indicate that CNN performs well in mining local and spatial patterns in input data and supports power load modeling in complex environments. However, despite its demonstrated importance in feature extraction, most existing methods still adopt a single structural integration strategy: CNN modules are typically deployed as independent pre-processing structures, lacking deep integration with the backbone network. This decoupled design limits the CNN's expressive power in deep feature modeling and its capacity for global optimization, making it difficult to fully leverage its potential.

In order to overcome the design limitations of previous studies where CNN modules were embedded only at the prediction layer, this study introduces CNN into the foundational structure of the DRN, achieving deep integration between CNN and the basic structure of DRN during the feature extraction stage. Existing research typically connects deep learning modules after the DRN or embeds them into the prediction layer (such as the current study that embeds CNN into the prediction layer of DRN [23]), which fails to effectively participate in the feature extraction process and limits the model's ability to learn local features. In contrast, the fusion strategy proposed in this study enhances the model's perception of local patterns, not only improving the expressive power of

shallow-layer modeling, but also strengthening the network's nonlinear modeling capability and generalization performance. It expands the design space of DRN structures in STLF tasks and provides a new structural design perspective and development direction for optimizing the feature extraction mechanism at the foundational level.

In contrast to prior DRN-based hybrid architectures that integrate CNN or LSTM modules only at the prediction layer, the proposed model introduces a fundamentally different design by embedding CNN directly into the DRN’s basic structure. This deep integration allows CNN to participate in the early-stage feature extraction process rather than acting as a post-processing module. As a result, the model captures localized temporal dependencies and hierarchical load features more effectively, establishing a clear structural distinction from existing DRN-based hybrid frameworks.

### 2.3 CNN Foundational Architecture

Time series data modeling frequently uses CNNs, which are powerful tools for extracting features from sequential data [34]. One-dimensional convolutional neural network (1D CNN) efficiently capture local temporal patterns and enhance prediction accuracy in STLF by applying convolution operations along the time axis. A typical 1D CNN architecture consists of a one-dimensional convolutional layer (Conv1D), a one-dimensional pooling layer (Pooling1D), and a FC layer [35]. Each Conv1D layer applies multiple filters, with each filter consisting of one or more kernels that slide over the input to extract local features. After each convolution operation, a non-linear activation function—commonly the Rectified Linear Unit (ReLU)—is applied to introduce non-linearity and enhance the model's capacity to learn complex temporal relationships. The ReLU activation function is mathematically defined as Equation (7):

$$f(x)=\max(0,x) \quad (7)$$

In this function, $x$ represents the input to the activation function, typically the weighted sum of a neuron's inputs. When the $x$ is greater than zero, the output is equal to the input; otherwise, the output is zero. This simple yet effective formulation enables ReLU to accelerate training convergence and avoid vanishing gradient problems.

As illustrated in Fig.3, the input signal passes through stacked convolutional layers, where the receptive fields grow progressively to capture more abstract temporal features. These convolutional layers can process multiple time steps in parallel, thereby improving computational efficiency. Following the convolution, pooling layers—such as max pooling, average pooling, or global average pooling (GAP)—are typically applied to reduce the dimensionality of feature maps, suppress noise, and enhance generalization. GAP performs an average operation over the entire receptive field of each feature map, effectively compressing the

output and significantly reducing the number of parameters.

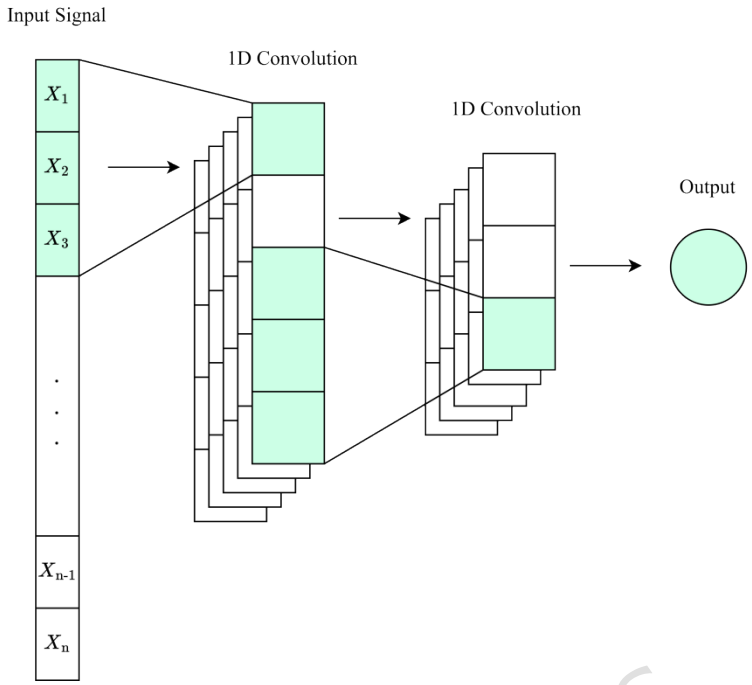



Fig.3 Architecture of 1D CNN [35].

In the context of one-dimensional time-series modeling, one-dimensional global average pooling (GAP1D) is used to average across the entire temporal dimension of each feature map. This not only helps mitigate overfitting but also improves model robustness and interpretability, especially when followed by fully connected layers. The final high-level features are then passed to one or more fully connected layers, which integrate the information and generate the prediction output.

Due to its limited receptive field, the 1D CNN is particularly effective in capturing short-term temporal dependencies, such as intra-day or inter-day fluctuations, thus demonstrating strong feature extraction capabilities in STLF. Through local convolution operations, CNNs can efficiently extract local temporal features while significantly enhancing computational efficiency and accelerating model convergence. However, when used alone, CNNs face certain limitations in modeling long-term dependencies, primarily because their small receptive fields make it difficult to comprehensively capture dynamic patterns over extended time horizons.

### 2.4 Summary

This section reviewed representative deep learning approaches for STLF, including convolution-based, recurrent-based, and attention-based models, and discussed their respective strengths and limitations. The principles and architectural characteristics of DRNs were then introduced, followed by a detailed analysis of the limitations of existing DRN-based frameworks,

particularly their insufficient exploitation of feature extraction at the foundational level. In addition, the basic architecture and feature extraction capability of CNN were reviewed to highlight their effectiveness in modeling local temporal patterns. Based on these analyses, the motivation for embedding CNN directly into the basic structure of DRN is clearly established. The next chapter presents the research methodology, datasets, and experimental setup used to evaluate the proposed model.

## 3. Methodology of Research

This section presents the research methodology adopted in this study, including data selection, preprocessing, model design, and experimental setup. It first describes the characteristics of the ISO-NE and Malaysia datasets, which represent temperate and tropical climatic conditions, respectively. Then, the section details the architecture of the proposed model, its input features, and the training configuration. Finally, it outlines the evaluation indicators and experimental framework used to assess forecasting performance and model generalization capability.

### 3.1 Research Data

Irregular formats, noise, incomplete entries, and missing values are common issues in real-world datasets [36], making data preprocessing an essential step to ensure the reliability and robustness of forecasting models. This study employed two actual datasets—ISO-NE and Malaysia—which offer contrasting insights into STLF under distinct climatic and demand conditions. The ISO-NE dataset provides hourly load and temperature records from March 2003 to December 2014, representing a temperate climate with strong annual and seasonal variations. It covers six states of the United States of America (Connecticut, Maine, Massachusetts, New Hampshire, Rhode Island, and Vermont), which together constitute the New England power grid. For the ISO-NE dataset, the regional hourly temperature data were used as observed inputs rather than forecasted data. In contrast, the Malaysia dataset comprises nationwide hourly load data combined with the regional daily mean, maximum, and minimum temperature data of Petaling Jaya from January 2020 to December 2022, representing a tropical climate characterized by relatively stable consumption patterns. No forecasted meteorological information was used in this study; all weather variables were obtained from historical observations to ensure that the forecasting model relies solely on data available up to the prediction point. The ISO-NE data, preprocessed by its provider, were directly adopted as a benchmark, whereas gaps in the Malaysia dataset were filled using linear interpolation to maintain chronological continuity. Fig. 4 shows that most ISO-NE load values range from approximately 7500 MW to 27500 MW with clear seasonal fluctuations, while Malaysia’s values mainly lie between 10000 MW and 18000 MW, reflecting steadier tropical demand. Finally, both datasets were normalized to ensure that all input features operated on a consistent scale.

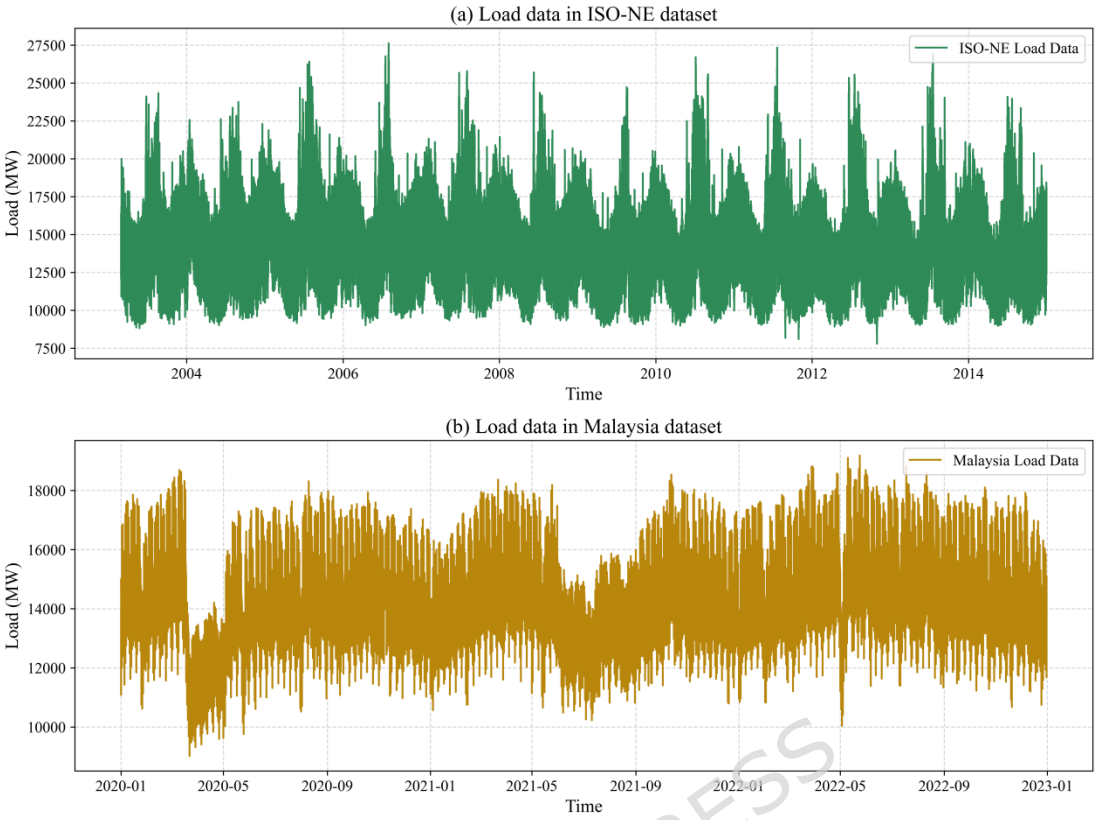



Fig.4 (a) Load data in ISO-NE dataset; (b) Load data in Malaysia dataset.

### 3.2 Proposed CNN-Embedded DRN for STLF

#### 3.2.1 The Proposed Model

The proposed model consists of two key components: the CNN-Embedded basic structure and the ResNetPlus network is depicted in Fig.5. The first component is a modified version of the original basic structure, where CNN blocks are integrated to process input data and produce an initial 24-hour load forecast. The second component is the ResNetPlus network, which refines the initial output to generate the final prediction, thereby improving forecasting accuracy and overall model performance.

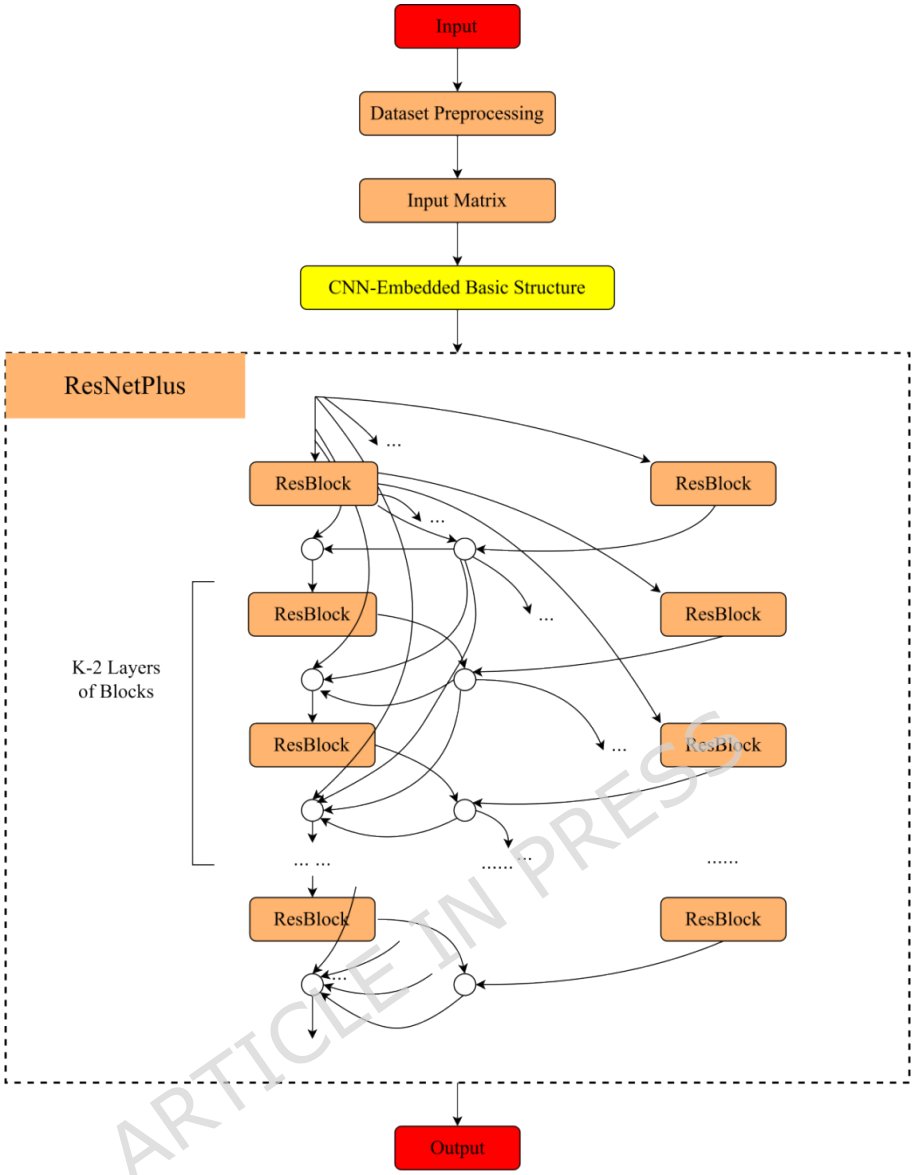



Fig.5 Proposed model's framework.

In the first component of the model, CNN blocks are inserted into the original basic structure. It is important to note that the input variables used in the modified model still include ($L_h^{hour}$, $L_h^{day}$, $L_h^{week}$, $L_h^{month}$, $T_h^{day}$, $T_h^{week}$, $T_h^{month}$, S, W and H.). Specifically, the long-term load variables (such as $L_h^{day}$, $L_h^{week}$, $L_h^{month}$) and long-term temperature variables (such as $T_h^{day}$, $T_h^{week}$, $T_h^{month}$) are processed through CNN blocks for feature extraction. The Conv1D configurations in the CNN block, including the number of filters and kernel sizes, are treated as hyperparameters to be optimized. Multiple configurations are tested during the experimental phase to identify the most effective setting, as detailed in the Experimental Setup section. After Conv1D in the CNN block, GAP1D is applied to compress each feature map into a single representative value. This operation

reduces the number of parameters and helps mitigate overfitting, while preserving the most relevant temporal features for each input. Embedding CNN at the early feature-extraction stage provides unique advantages over conventional hybrid designs. By allowing convolutional operations to process raw input features before residual refinement, the model captures localized load and temperature patterns at multiple temporal scales. This enhances the discriminative representation of short-term fluctuations while preserving the continuity of residual learning in later stages. In contrast, previous DRN-based hybrids that introduce CNN after the residual blocks primarily enhance post-hoc refinement, offering limited improvement to initial feature representation.

In the CNN-Embedded basic structure, each sub-network independently predicts the load for a specific hour in the future. By combining the prediction results of 24 sub-networks, the model generates an initial forecast for the entire next day's load, as shown in Fig. 6. It is worth noting that SELU is used as the default activation function for all layers in the model, except for the CNN blocks, which employ the ReLU activation, and the output layer. In this stage, the CNN-Embedded basic structure receives historical load, temperature, and temporal variables and applies convolutional operations to extract localized temporal dependencies. The Conv1D layers capture short-term load fluctuations as well as correlations between recent demand patterns and temperature variations. Subsequently, a global average pooling layer aggregates these localized features into compact representations, which are used to generate the initial forecast. Each of the 24 sub-networks focuses on one specific hour of the next day, and their outputs are concatenated to form a complete 24-hour-ahead load forecast sequence.

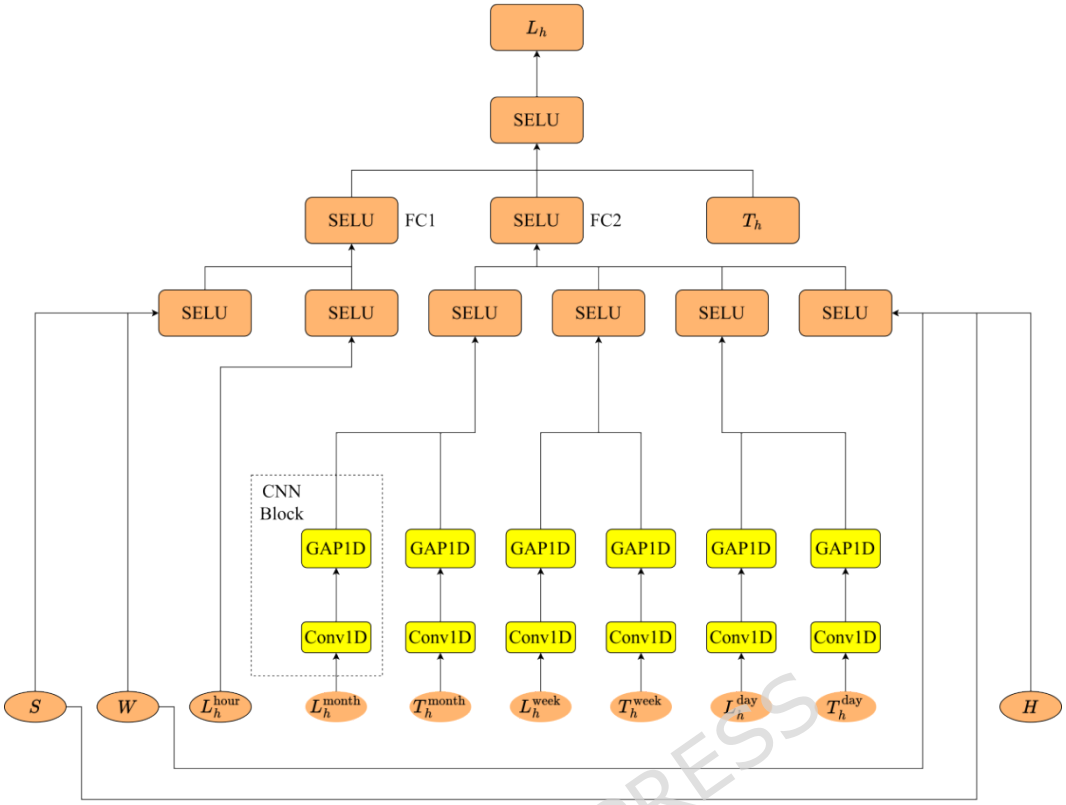



Fig.6 The proposed model basic structure in ISO-NE dataset.

In the second stage, the ResNetPlus network is employed to further refine the initial 24-hour load forecast generated by the CNN-Embedded basic structure. The number of layers and structural components of the ResNetPlus network remain unchanged, ensuring consistency with the original DRN framework. Through residual blocks with identity shortcut connections, ResNetPlus captures longer-term dependencies while maintaining stable gradient flow during deep training. These residual connections allow information extracted in earlier layers to be preserved and reused, thereby enhancing continuity between short-term and long-term feature representations.

The model is trained in an end-to-end manner using the same loss function as the original DRN, which combines the mean squared error with an additional penalty term to constrain predictions within a realistic demand range. By refining the preliminary predictions produced in the first stage, the ResNetPlus network improves forecasting accuracy and robustness without introducing additional structural complexity. This two-stage learning strategy enables the proposed CNN-Embedded DRN to effectively integrate localized feature extraction and deep residual learning for stable and accurate STLF.

### 3.2.2 Input Features in the Proposed Model

The ISO-NE and Malaysia datasets used in this study exhibit significantly

different temporal granularities, necessitating distinct feature processing methods. The ISO-NE dataset's hourly load $L_h^{hour}$ and temperature $T_h^{month}$, $T_h^{week}$, $T_h^{day}$ values are supplied straight into the model. The original CNN-Embedded basic structure combines S, W, and H information with load characteristics $L_h^{month}$, $L_h^{week}$, $L_h^{day}$ depending on different temporal ranges to provide model inputs. S is made up of the seasons spring, summer, autumn, and winter; H is made up of Christmas, Independence Day, and other holidays.

Whereas the Malaysia dataset provides just daily temperature data, including $T_{mean}$, $T_{max}$, $T_{min}$, the ISO-NE dataset provides hourly temperature data. The altered CNN-Embedded basic structure is shown in Fig.7. To address this disparity, the basic structure was modified to accept daily temperature data as input directly. The daily temperature data $T_{mean}$, $T_{max}$, $T_{min}$ are concatenated as a single feature input in the updated model without temporal segmentation. While this is going on, load feature $L_h^{month}$, $L_h^{week}$, $L_h^{day}$ processing continues to retrieve data from the last 24 hours, 8 weeks, and 3 months. Temperature and load data, along with date-related information like S, W, and H, make up the model's final input.While S is divided into two seasons—the rainy season and the dry season—H includes occasions like Eid al-Fitr and Malaysia Independence Day.



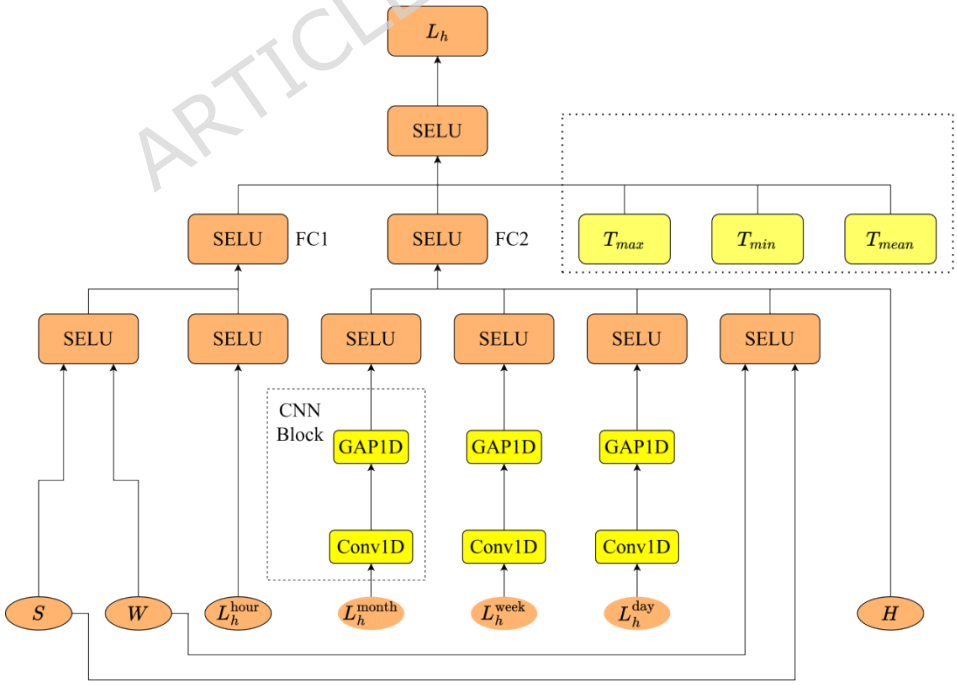



Fig.7 The proposed model basic structure in Malaysia dataset.

This change simplifies the preparation procedures and enables the direct use of

the daily temperature parameters from the Malaysia dataset by minimizing the repetition caused by converting daily data into hourly data. With the exception of how temperature variables that rely on temporal granularity are handled, the new model handles load characteristics and date information uniformly for both datasets. By doing this, the model's adaptability and generalizability to datasets with different temporal resolutions are guaranteed.

### 3.3 Experimental Setup

This study evaluates the performance of the proposed model through experiments and compares it with several benchmark and variant models commonly used for STLF. These include the conventional CNN model, RNN-based models (LSTM, GRU, BiLSTM, BiGRU), and Transformer. In addition, four residual-based architectures—ResNet, CNN-Embedded ResNet, the original DRN, and the proposed CNN-Embedded DRN—were designed as complementary ablation variants to systematically analyze the effects of the CNN block and the deeper residual architecture. Two real-world datasets were used: the ISO-NE dataset and the Malaysia dataset. The details of the training and testing partitions for both datasets are summarized in Table 1. These datasets represent two distinct climatic regions—temperate (ISO-NE) and tropical (Malaysia)—providing diverse scenarios for model evaluation.

**Table 1 Summary of dataset partitions used for training and testing.**

| Dataset | Training Period | Training Samples | Testing Period | Testing Samples |
|---|---|---|---|---|
| ISO-NE | 2003.03-2005.12 | 24888 | 2006.01-2006.12 | 8760 |
| Malaysia | 2020.01-2021.12 | 17544 | 2022.01-2022.12 | 8760 |

The CNN block in the proposed CNN-Embedded DRN model consists of Conv1D layers followed by a GAP1D layer. To explore the optimal configuration of the CNN block, hyperparameter tuning experiments were conducted on the ISO-NE dataset. The experiments systematically varied two key hyperparameters: the number of filters in the Conv1D layers and the kernel size. Specifically, five settings were considered for the number of filters: 16, 32, 64, 128, and 256; and four settings for the kernel size: 1, 3, 5, and 7. This resulted in a total of 20 configurations. All other parameters were kept constant to isolate the effects of these variations. The Conv1D layers used ReLU activation with He normal initialization, ‘same' padding, and a stride of 1. The GAP1D layer was applied after the convolution to compress each feature map into a single representative value, reducing parameters and helping mitigate overfitting.

The CNN baseline model used in this study adopted the same CNN block configuration as that in the proposed CNN-Embedded DRN model to ensure fair comparison. Similarly, for the RNN-based baseline models, the number of units in each recurrent layer (LSTM, GRU, BiLSTM, BiGRU) was set equal to the number of filters used in the CNN configurations, enabling consistent capacity across models for comparative analysis. For the Transformer baseline model, the embedding dimension was set equal to the number of filters in the CNN configurations. The Transformer adopted a standard encoder-only structure with one encoder layer, eight attention heads (each with 64 dimensions when the embedding dimension allowed), and a feed-forward network dimension of 2048. Other components, such as positional encoding and dropout (0.1), were kept at default settings to ensure consistency. The DRN model was implemented using its default parameter settings as defined in the original framework, while the ResNet architecture, similar to the DRN, also employed the SELU activation function and consisted of ten stacked layers, each being a SELU-activated layer with 20 neurons, to maintain structural consistency.

The model was trained for over 700 epochs in total, including an initial training phase of 600 epochs followed by two rounds of short-term training, each consisting of 50 epochs [20]. During the later stages, three model snapshots were saved at the end of each 50-epoch segment. This technique, known as snapshot ensemble [37], mitigates overfitting and enhances training stability by averaging predictions from multiple model checkpoints. By combining these snapshots, the method reduces the risk of overfitting associated with a single model and improves both prediction stability and generalization capability. Furthermore, this ensemble approach provides a computationally efficient alternative to performing multiple independent training trials while maintaining comparable robustness and stability in forecasting performance [38]. The training process adopted commonly used default parameter settings from previous studies, with the loss function defined as the MAPE to evaluate forecasting accuracy. The Adaptive Moment Estimation (Adam) [39] optimizer was employed with an initial learning rate of 0.001 to achieve efficient adaptive learning.

A nonparametric Bootstrap resampling method with 10000 iterations was used to thoroughly evaluate if the improved model's performance increase was statistically significant. Unlike the paired Student’s t-test, which assumes that paired differences follow a normal distribution, the Bootstrap method is free from such distributional constraints, offering a more robust basis for performance comparison [40]. Statistical significance was evaluated using two criteria. First, if the 95% confidence interval (CI) of the mean performance difference lies entirely above zero, the improvement is considered significant at the 95% confidence level; if zero is contained within the interval, the difference

is deemed insignificant. Second, within the Bootstrap framework, a p-value below 0.05 similarly indicates significance at the 95% level. It should be noted that $p \approx 0$ represents an extremely small probability (typically $< 0.0001$) rather than an actual zero. MAPE was employed as the evaluation metric owing to its widespread use in STLF research and its interpretable, scale-independent representation of relative prediction error.

All experiments were conducted in a Python 3.8 environment using TensorFlow 2.10.0 and Keras 2.10.0 as the deep learning backends. For the tests, a Lenovo laptop equipped with an AMD Ryzen 7 6800H CPU, 16GB DDR5 4800MHz RAM, and an NVIDIA GeForce RTX 3050 Ti Laptop GPU (4GB) was utilized.

### 3.4 Evaluation Indicators

To compare the performance of different DRN models in STLF, researchers have employed a range of criteria to assess prediction precision [20-25]. MAPE is the most often used statistic among them due to its interpretability and effectiveness in evaluating relative forecasting precision across different datasets and scales. Mean Absolute Error (MAE), Mean Square Error (MSE), Normalized Mean Square Error (NMSE), Root Mean Square Error (RMSE), Correlation Coefficient (R), and Coefficient of Determination ($R^2$) are additional metrics that have been used in previous studies in addition to MAPE. Different studies employ different evaluation criteria, depending on the specific forecasting objectives and dataset characteristics. Each of these measurements has a formula in equations (8) through (14).

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i - \grave{y}_i}{y_i}\right| \times 100 \quad (8)$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i-1}^{N}\left(y_i - \grave{y}_i\right)^2} \quad (9)$$

$$MAE = \frac{1}{N}\sum_{i-1}^{N}\left|y_i - \grave{y}_i\right| \quad (10)$$

$$MSE = \frac{1}{N}\sum_{i-1}^{N}\left(y_i - \grave{y}_i\right)^2 \quad (11)$$

$$NMSE = \frac{\sum_{i-1}^{N}\left(y_i - \grave{y}_i\right)^2}{N\,\sigma_y^2} \quad (12)$$

$$R = \frac{\sum_{i-1}^{N}\left(y_i - \bar{y}\right)\left(\grave{y}_i - \bar{\grave{y}}\right)}{\sqrt{\sum_{i-1}^{N}\left(y_i - \bar{y}\right)^2 \sum_{i-1}^{N}\left(\grave{y}_i - \bar{\grave{y}}\right)^2}} \quad (13)$$

$$R^2 = 1 - \frac{\sum_{i-1}^{N}\left(y_i - \grave{y}_i\right)^2}{\sum_{i-1}^{N}\left(y_i - \bar{y}\right)^2} \quad (14)$$

A description of the parameters utilized in these measures is provided below: $y_i$ represents the actual value of the i-th sample, $\grave{y}_i$ represents the predicted value

of the $i$-th sample, and $N$ represents the total number of input samples. Additionally, $\sigma_y^2$ stands for the variance of the actual values, which is used to normalize MSE in the calculation of NMSE, while $\bar{y}$ and $\bar{\grave{y}}$ indicate the mean values of all actual and forecast values, respectively. The metrics are able to assess model performance comprehensively by considering the correlation between actual and expected values, prediction accuracy, and error magnitude. Reduced prediction errors and improved generalization capacity are frequently shown by smaller values for MAPE, RMSE, MAE, MSE, and NMSE. On the other hand, R and $R^2$ values nearer 1 suggest stronger model fitting ability and higher forecast precision.

### 3.5 Summary

This section described the overall research methodology, including dataset selection, preprocessing procedures, model architecture, and experimental configurations. The proposed CNN-Embedded DRN framework integrates convolutional feature extraction and deep residual learning to improve STLF accuracy and robustness across different climatic conditions. The subsequent chapter presents the experimental results, comparative analyses, and discussions to validate the model's effectiveness.

## 4. Results and Discussion of The Experiment

This section discusses the experimental results and comparative analyses of the proposed model. It first evaluates the performance of the model under different CNN configurations, followed by comparisons with various baseline and ablation models. The section further examines the model's robustness across different seasons and climatic conditions and validates the statistical significance of its improvements using Bootstrap analysis. These comprehensive evaluations collectively demonstrate the effectiveness and generalization ability of the proposed approach for STLF.

### 4.1 Performance of the Model with Different Settings

To evaluate the effectiveness of the proposed CNN-Embedded DRN model, a series of experiments were conducted on the ISO-NE dataset by varying the number of filters and kernel sizes within the CNN module. The primary objective was to determine the optimal configuration of CNN hyperparameters that minimizes forecasting error while maintaining computational efficiency. Table 2 presents the MAPE values obtained under different CNN parameter settings, followed by an in-depth analysis of the results.

As shown in Table 2, both the number of filters and kernel size substantially influence forecasting accuracy. Moderate configurations generally yield more stable results, whereas excessively small or large kernels tend to cause performance degradation. Among all tested settings, the combination of 32

filters and a kernel size of 1 achieved the lowest MAPE of 0.015303, indicating that a relatively shallow convolutional layer with a narrow receptive field can effectively capture short-term temporal dependencies in the ISO-NE dataset. This finding aligns with the dataset's intrinsic characteristics, where local load and temperature fluctuations exhibit strong short-range periodicity and recurring patterns. The use of a small kernel minimizes feature smoothing, while 32 filters provide sufficient representation capacity without introducing overfitting.

**Table 2 Comparison of MAPE for CNN-Embedded DRN Using Different CNN Hyperparameter Settings**

| Filters | Kernel Size | MAPE |
|---|---|---|
| 16 | 1 | 0.016965 |
| 16 | 3 | 0.016120 |
| 16 | 5 | 0.015818 |
| 16 | 7 | 0.017018 |
| **32** | **1** | **0.015303** |
| 32 | 3 | 0.015788 |
| 32 | 5 | 0.017202 |
| 32 | 7 | 0.017036 |
| 64 | 1 | 0.016640 |
| 64 | 3 | 0.016380 |
| 64 | 5 | 0.016796 |
| 64 | 7 | 0.015582 |
| 128 | 1 | 0.016635 |
| 128 | 3 | 0.016542 |
| 128 | 5 | 0.015687 |
| 128 | 7 | 0.016929 |
| 256 | 1 | 0.015975 |
| 256 | 3 | 0.016505 |
| 256 | 5 | 0.016547 |
| 256 | 7 | 0.016179 |



Interestingly, configurations such as 64 filters with a kernel size of 7 and 128 filters with a kernel size of 5 also achieved competitive MAPE values of 0.015582

and 0.015687, respectively. This observation suggests that larger kernels can occasionally improve performance when accompanied by an adequate number of filters, as the wider receptive field enables the model to capture slightly longer-term temporal dependencies. However, this advantage only appears when the network has sufficient capacity to preserve diverse feature representations. When the filter count is low (e.g., 16 or 32), large kernels tend to oversmooth local patterns, leading to performance degradation—as seen in the 16-filter configuration with kernel size 7 (MAPE = 0.017018) and the 32-filter configuration with kernel size 5 (MAPE = 0.017202).

Furthermore, increasing the number of filters beyond 128 fails to produce consistent performance gains. Although deeper configurations possess greater representational power, their increased complexity introduces redundancy and the risk of overfitting, leading to unstable or suboptimal generalization. For example, the 256-filter setting produced MAPE values above 0.0159 in all kernel size combinations, providing no tangible improvement over lighter models. These results confirm that simply enlarging network capacity does not necessarily enhance predictive accuracy and that a balance between model complexity and dataset characteristics must be maintained.

In general, kernel sizes of 1 and 5 yielded the most favorable and stable outcomes across different filter settings. While kernel 1 focuses on fine-grained short-term variations, kernel 5 offers a balance between short-term and slightly extended temporal feature capture. In contrast, kernel 7, which enlarges the receptive field excessively, may introduce redundant or smoothed representations that obscure critical load fluctuations.



Taken together, these results demonstrate that CNN hyperparameters have a considerable impact on forecasting accuracy and should be tuned carefully to achieve an optimal trade-off between accuracy, generalization, and computational cost. Based on the empirical findings, the configuration of 32 filters and kernel size 1 is selected as the optimal CNN setting in the proposed model. This configuration achieves the best performance (MAPE = 0.015303) while maintaining model efficiency and interpretability. Moreover, it surpasses the accuracy of advanced DRN-based models reported in prior research—reducing MAPE by approximately 11.56% compared to the CRN model (MAPE = 0.0173) [23] and by about 1.90% compared to the Residual LSTM Plus model (MAPE = 0.0156) [24]. These results further validate the effectiveness of embedding CNN layers into the foundational DRN structure to enhance early-stage feature extraction and overall forecasting performance.

### 4.2 Contrast with Base Models

#### 4.2.1 ISO-NE Dataset

**Table 3 Comparison of the proposed model with base models in ISO-NE**

**dataset**

| Model | MAPE | RMSE | MAE | MSE | NMSE | R | $R^2$ |
|---|---|---|---|---|---|---|---|
| CNN | 0.024224 | 0.024393 | 0.015815 | 0.000595 | 0.039482 | 0.980696 | 0.960518 |
| LSTM | 0.023277 | 0.023283 | 0.015163 | 0.000542 | 0.035968 | 0.982602 | 0.964032 |
| GRU | 0.018612 | 0.018986 | 0.01207 | 0.00036 | 0.023918 | 0.988042 | 0.976082 |
| BiLSTM | 0.02166 | 0.022404 | 0.014175 | 0.000502 | 0.033306 | 0.983255 | 0.966694 |
| BiGRU | 0.019921 | 0.019651 | 0.012803 | 0.000386 | 0.025624 | 0.987392 | 0.974376 |
| Transformer | 0.021925 | 0.022749 | 0.014352 | 0.000517 | 0.034337 | 0.982763 | 0.965663 |
| ResNet | 0.018286 | 0.017879 | 0.011663 | 0.00032 | 0.02121 | 0.990105 | 0.97879 |
| CNN-Embedded ResNet | 0.017346 | 0.018193 | 0.011191 | 0.000331 | 0.021962 | 0.989057 | 0.978038 |
| DRN | 0.017182 | 0.017548 | 0.011138 | 0.000308 | 0.020432 | 0.989767 | 0.979568 |
| **CNN-Embedded DRN** | **0.015303** | **0.016277** | **0.009840** | **0.000265** | **0.017580** | **0.991237** | **0.982420** |

The experimental results for the ISO-NE dataset are summarized in Table 3 and visualized in Fig.8. Among the baseline models, the DRN achieved the best performance, with a MAPE of 0.017182, RMSE of 0.017548, MAE of 0.011138, MSE of 0.000308, and NMSE of 0.020432. It also attained high correlation metrics, with R = 0.989767 and $R^2$ = 0.979568. These results highlight the effectiveness of residual learning in mitigating gradient degradation and improving model stability, which enabled the DRN to outperform conventional architectures.

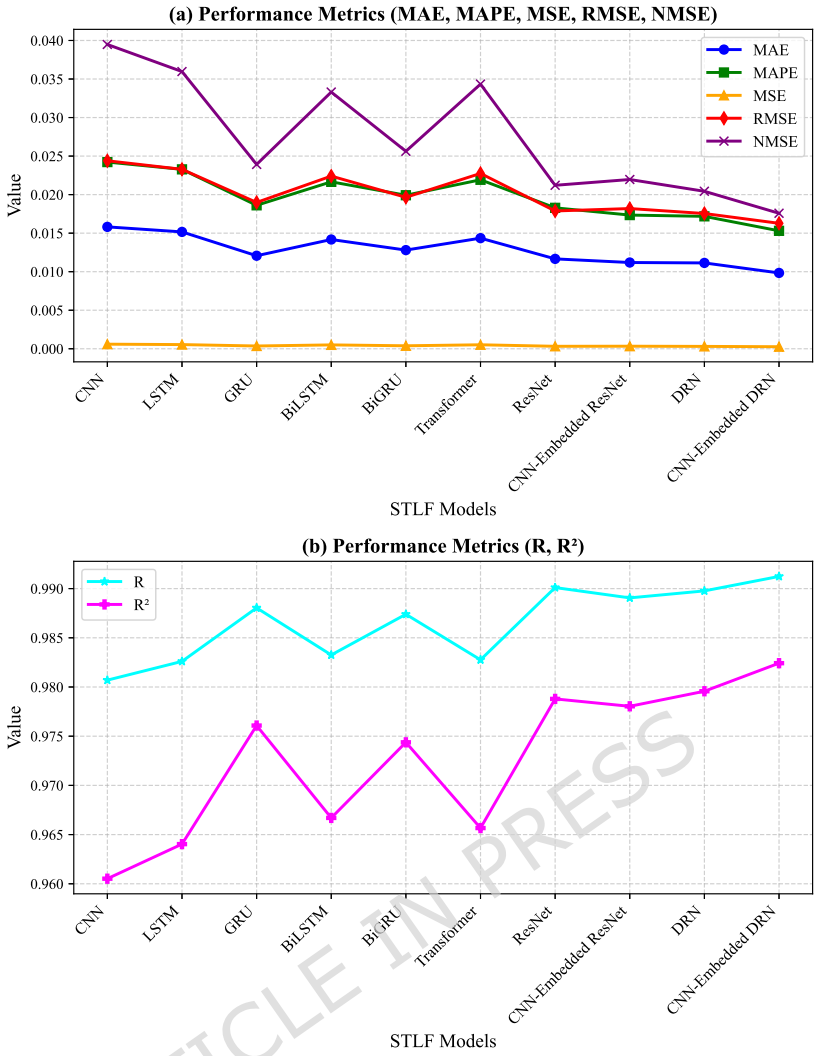



Fig.8 A comparison between the proposed model and the baseline models in ISO-NE Dataset: (a) MAPE, RMSE, MAE, MSE, NMSE; (b) R and $R^2$.

To further investigate the effect of residual learning and convolutional feature extraction, two additional ablation models—ResNet and CNN-Embedded ResNet—were introduced. The ResNet, representing a classical residual framework, achieved a MAPE of 0.018286, confirming the advantage of skip-connection mechanisms but still performing slightly worse than the DRN. This difference suggests that the deeper structure and refined residual mapping in DRN provide stronger feature propagation and learning stability. When convolutional layers were incorporated into the residual framework, the CNN-Embedded ResNet improved the MAPE to 0.017346 and achieved consistent gains across all other metrics compared with the standard ResNet. These results verify that embedding convolutional layers within residual blocks enhances local pattern extraction and complements residual learning for more effective spatiotemporal representation.

The GRU and BiGRU models also demonstrated relatively strong performance, with GRU recording a MAPE of 0.018612 and BiGRU achieving 0.019921. Their

RMSE and MAE values were similarly competitive, showing that gated recurrent units can effectively capture temporal dependencies in the ISO-NE dataset. However, their performance lagged slightly behind the DRN-based models, suggesting that residual learning adds significant value beyond recurrent temporal modeling alone. In contrast, standard CNN and LSTM architectures, such as CNN (MAPE 0.024224) and LSTM (MAPE 0.023277), exhibited higher error rates because they struggled to fully capture the complex seasonal and daily patterns present in the dataset. The Transformer model achieved moderate results (MAPE 0.021925), benefiting from its self-attention mechanism but still not matching the residual-based architectures. This may be due to the relatively short input sequences and the dataset's strong periodicity, which favor models with explicit residual connections.

When compared to these baselines and ablation variants, the proposed CNN-Embedded DRN demonstrated a clear performance advantage. It achieved the lowest MAPE of 0.015303, improving upon the original DRN by over 10% in relative error reduction. Similarly, its RMSE (0.016277) and MAE (0.009840) were the smallest among all models, indicating better absolute accuracy. The MSE (0.000265) and NMSE (0.017580) were also the lowest, showing enhanced robustness in minimizing both raw and normalized prediction errors. The correlation metrics were the highest observed ($R = 0.991237$, $R^2 = 0.982420$), confirming the model's superior fit to actual load values. These results demonstrate that integrating CNN-based local feature extraction into the residual learning framework enables the model to more effectively capture both fine-grained temporal patterns and long-term seasonal trends. The CNN module enhances the DRN's ability to identify local load fluctuations, while the residual connections ensure stable deep learning and mitigate degradation. Together, these components contribute to superior forecasting accuracy and generalization performance on the ISO-NE dataset.



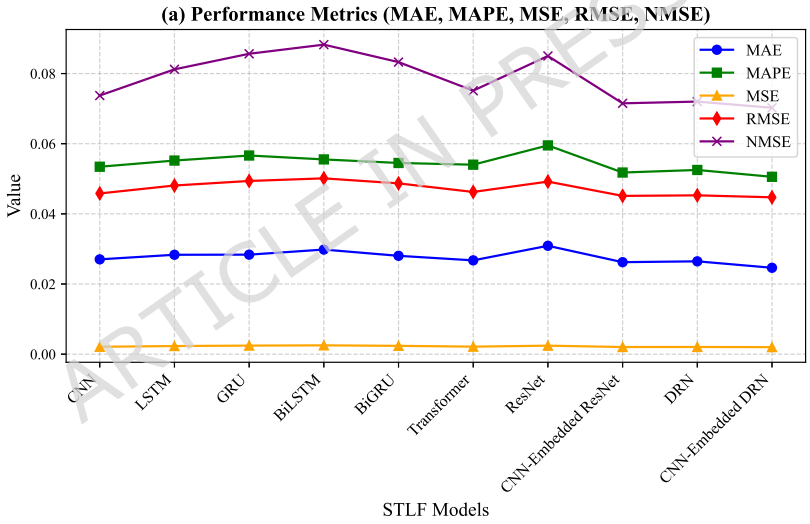
### 4.2.2 Malaysia Dataset

The experimental results for the Malaysia dataset are presented in Table 4 and Fig.9. Among the baseline models, the DRN once again achieved the best performance, with a MAPE of 0.052514, RMSE of 0.045278, MAE of 0.026467, MSE of 0.002050, and NMSE of 0.072007. Its correlation metrics ($R = 0.964032$, $R^2 = 0.927993$) were the highest among the baseline models. These results confirm the advantage of residual learning even in a tropical dataset where load patterns are relatively stable and less affected by seasonal fluctuations.

**Table 4 A comparison between the proposed model and the baseline models in the Malaysia dataset**

| Model | MAPE | RMSE | MAE | MSE | NMSE | R | $R^2$ |
|---|---|---|---|---|---|---|---|
| CNN | 0.053434 | 0.045817 | 0.027030 | 0.002099 | 0.073731 | 0.962746 | 0.926269 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| LSTM | 0.055195 | 0.048080 | 0.028335 | 0.002312 | 0.081197 | 0.959842 | 0.918803 |
| GRU | 0.056635 | 0.049378 | 0.028368 | 0.002438 | 0.085638 | 0.957068 | 0.914362 |
| BiLSTM | 0.055521 | 0.050124 | 0.029821 | 0.002512 | 0.088245 | 0.956569 | 0.911755 |
| BiGRU | 0.054511 | 0.048690 | 0.028037 | 0.002371 | 0.083268 | 0.958281 | 0.916732 |
| Transformer | 0.054016 | 0.046247 | 0.026741 | 0.002139 | 0.075124 | 0.961710 | 0.924876 |
| ResNet | 0.059517 | 0.049196 | 0.030896 | 0.002420 | 0.085010 | 0.959000 | 0.914990 |
| CNN-Embedded ResNet | 0.051784 | 0.045122 | 0.026223 | 0.002036 | 0.071514 | 0.964538 | 0.928486 |
| DRN | 0.052514 | 0.045278 | 0.026467 | 0.002050 | 0.072007 | 0.964032 | 0.927993 |
| **CNN-Embedded DRN** | **0.050566** | **0.044719** | **0.024629** | **0.001999** | **0.070241** | **0.965433** | **0.929759** |



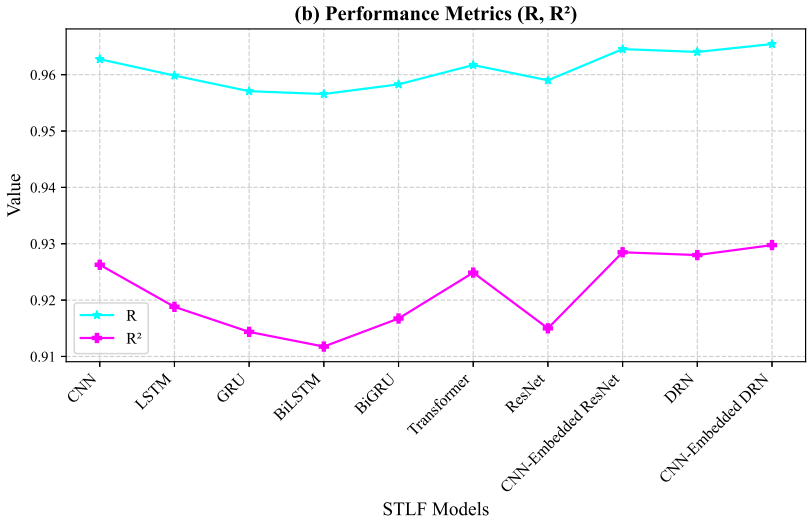



Fig.9 A comparison between the proposed model and the baseline models in Malaysia Dataset: (a) MAPE, RMSE, MAE, MSE, NMSE; (b) R and $R^2$.

To evaluate the contribution of convolutional embedding in residual frameworks, two ablation models—ResNet and CNN-Embedded ResNet—were again considered. The ResNet model recorded a MAPE of 0.059517, slightly higher than that of the DRN, indicating that the deep residual mapping and enhanced skip connections in DRN offer superior learning capability. When CNN layers were introduced into the residual blocks, the CNN-Embedded ResNet achieved a MAPE of 0.051784, outperforming both ResNet and DRN in this dataset. This demonstrates that CNN-based local feature extraction can effectively complement residual learning by capturing subtle load fluctuations that would otherwise be smoothed out in a relatively stable tropical environment.

The performance of other baseline models followed a similar trend as observed in the ISO-NE dataset. CNN and Transformer models provided competitive results, with CNN achieving a MAPE of 0.053434 and Transformer achieving 0.054016. These models were able to capture general load patterns but fell short in precision compared to DRN-based models. The recurrent models, including LSTM (0.055195), GRU (0.056635), BiLSTM (0.055521), and BiGRU (0.054511), exhibited slightly higher error rates. The smaller performance gap between models suggests that the Malaysia dataset's more stable and less volatile load profile reduces the relative advantage of architectures specialized in handling temporal dependencies.

The CNN-Embedded DRN delivered the best performance across all evaluation metrics. It achieved the lowest MAPE of 0.050566, representing a noticeable improvement over the original DRN (0.052514) and other baselines. Its RMSE (0.044719) and MAE (0.024629) were also the smallest, demonstrating enhanced absolute accuracy. The MSE (0.001999) and NMSE (0.070241) further confirmed its advantage in minimizing prediction errors, while the highest R (0.965433) and $R^2$ (0.929759) values indicated a stronger correlation between the predicted and actual load values than any other model tested. These findings highlight the CNN-Embedded DRN's ability to capture subtle variations in the Malaysia dataset despite its smoother load patterns. The CNN layers contribute to improved local feature extraction, which helps the model identify minor load fluctuations, while the residual connections continue to play a key role in ensuring stable and efficient learning. Collectively, the inclusion of ResNet and CNN-Embedded ResNet as ablation experiments further validates the superiority and robustness of the proposed CNN-Embedded DRN architecture across datasets with different climatic characteristics.

### 4.3 Seasonal Variations in the Proposed Model's Performance

At last, this work uses test data from several seasons in the ISO-NE and Malaysia datasets to assess and compare the CNN-Embedded DRN model's performance.

The model's prediction accuracy and capacity for generalization across seasons are further confirmed by include DRN as a benchmark for comparison study. The first week (168 hours) of data from various seasons is chosen as the test set for both the ISO-NE and Malaysia datasets in order to guarantee the experiment's fairness and the accuracy of the findings. This enables an evaluation of the model's flexibility and predictive capabilities over a range of time periods.

A comparison between the actual load curves in the ISO-NE dataset over the four seasons (spring, summer, autumn, and winter) and the load forecast curves of several models is shown in Fig.10. It is evident that, throughout all seasons, the CNN-Embedded DRN's prediction curve closely resembles the real load, showing low error and strong trend alignment. This suggests that the model can retain high forecast accuracy, efficiently capture patterns of load changes throughout seasons, and adjust to seasonal variations in load demand.

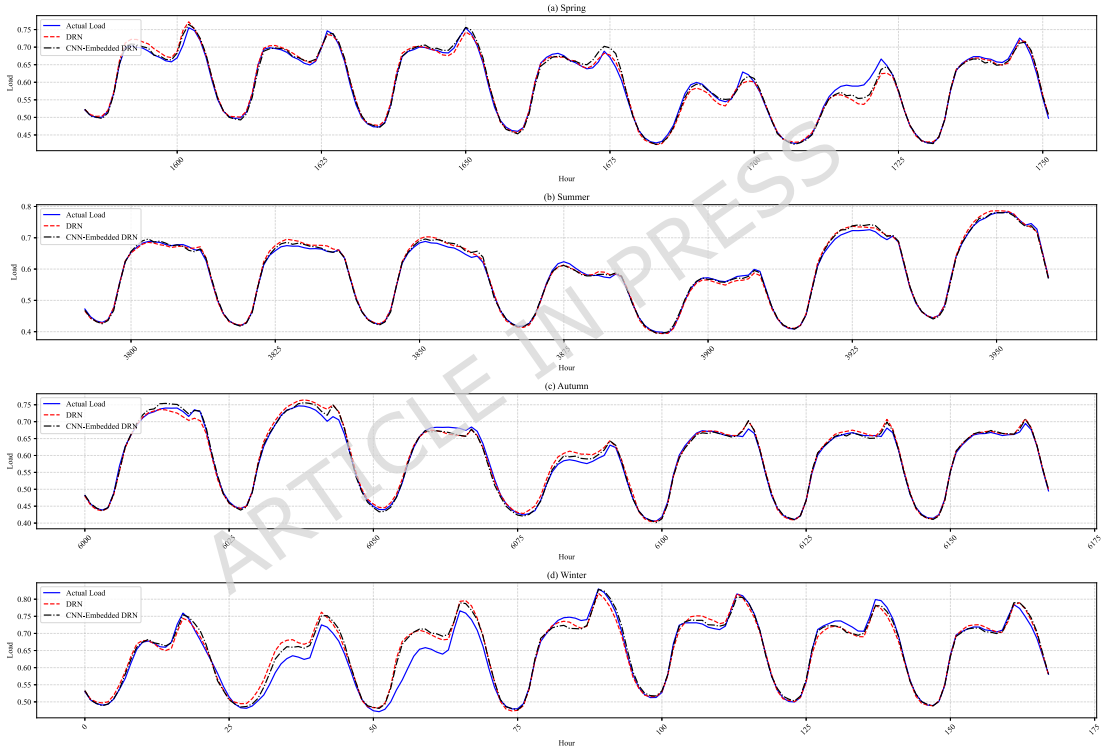



Fig.10 The model's performance on the ISO-NE dataset for different seasons: (a) Spring, (b) Summer, (c) Autumn, and (d) Winter (forecast values are normalized).

Fig.11 further illustrates the comparison between the actual and predicted load curves of several models on the test set of the Malaysia dataset during both the wet and dry seasons. Given that the forecast curve of the CNN-Embedded DRN closely follows the actual load, the results highlight its strong predictive capability under both climatic conditions. This indicates that the model not only adapts well to seasonal variations in load demand but also maintains stable performance across different weather patterns, ensuring reliable forecasts.

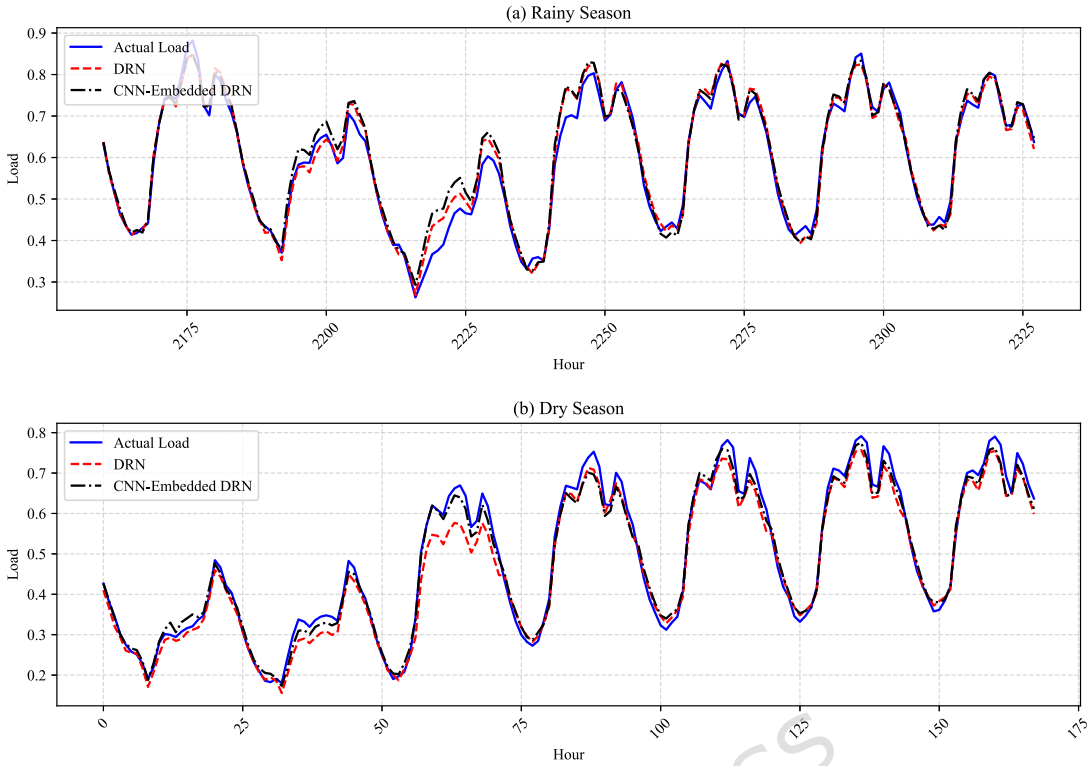



Fig.11 The model's performance on the Malaysia dataset for different seasons: (a) rainy, (b) dry (forecast values are normalized).

As a whole, the results further validate the CNN-Embedded DRN's resilience and usefulness in STLF tasks by validating the model using test data from several seasons. In particular, the model continuously maintains good prediction accuracy, even when load shows notable seasonal fluctuations. As a result, the model offers dependable support for real-world applications and exhibits a strong generalization potential for STLF.

### 4.4 Statistical Significance Analysis Based on Bootstrap

Tables 5 and 6 present the Bootstrap evaluation results for both datasets, where ResNet is compared with CNN-Embedded ResNet and the original DRN is compared with CNN-Embedded DRN based on MAPE. In this case, the mean difference denotes the average performance disparity between the two examined models, while the standard deviation (SD) quantifies the variation in MAPE across all resampling iterations.

For the ISO-NE dataset, the CNN-Embedded ResNet and CNN-Embedded DRN achieved lower MAPE values than their respective baseline models, indicating enhanced forecasting accuracy. The mean differences between the compared models were 0.00094 for ResNet versus CNN-Embedded ResNet and 0.00188 for DRN versus CNN-Embedded DRN. Moreover, the 95% CIs of the mean differences did not include zero, and the corresponding p-values were approximately zero, demonstrating statistically significant improvements in model performance.

Similarly, for the Malaysia dataset, the CNN-Embedded ResNet and CNN-Embedded DRN again outperformed the baseline architectures. The mean differences were 0.00773 and 0.00195, respectively, both supported by extremely small p-values (≈0), confirming the robustness of the improvements across distinct climatic and consumption conditions. The narrow CIs and small SDs further emphasize the consistency of the Bootstrap resampling results.

Table 5 Bootstrap Results on the ISO-NE Dataset

| 1st Model | 2nd Model | MAPE ± SD (Model 1) | MAPE ± SD (Model 2) | Mean Difference | CI (95%) | p-value |
|---|---|---|---|---|---|---|
| ResNet | CNN-Embedded ResNet | 0.018286 ± 0.020353 | 0.017346 ± 0.021283 | 0.000940 | [-0.000299, 0.000297] | ≈ 0 |
| DRN | CNN-Embedded DRN | 0.017182 ± 0.019726 | 0.015303 ± 0.019655 | 0.001880 | [-0.000283, 0.000286] | ≈ 0 |

Table 6 Bootstrap Results on the Malaysia Dataset

| 1st Model | 2nd Model | MAPE ± SD (Model 1) | MAPE ± SD (Model 2) | Mean Difference | CI (95%) | p-value |
|---|---|---|---|---|---|---|
| ResNet | CNN-Embedded ResNet | 0.059517 ± 0.111446 | 0.051784 ± 0.114594 | 0.007733 | [-0.000771, 0.000784] | ≈ 0 |
| DRN | CNN-Embedded DRN | 0.052514 ± 0.106031 | 0.050566 ± 0.109999 | 0.001948 | [-0.000588, 0.000580] | ≈ 0 |

Taken together, these results provide strong statistical evidence that integrating CNN into the ResNet framework significantly enhances the model's predictive performance. The consistent improvements observed across two geographically and climatically distinct datasets suggest that the CNN-Embedded architectures are capable of effectively capturing localized spatial patterns in load and temperature variations, thereby enhancing generalization capability and forecasting stability.

### 4.5 Summary

This section comprehensively evaluated the performance of the proposed CNN-Embedded DRN model through extensive experiments on two benchmark datasets, ISO-NE and Malaysia. The results demonstrated that the proposed architecture consistently outperformed conventional deep learning models—including CNN, LSTM, GRU, BiLSTM, BiGRU, and Transformer—as well as advanced residual-based networks such as ResNet and DRN.

By embedding convolutional layers into the foundational structure of DRN, the model enhanced early-stage feature extraction and significantly improved forecasting precision, particularly in capturing short-term load fluctuations and local temporal patterns. The inclusion of ResNet and CNN-Embedded ResNet as ablation experiments further verified that CNN-based local feature extraction complements residual learning, yielding consistent performance gains across different climatic conditions.

In addition, the Bootstrap significance analysis confirmed that the observed improvements of CNN-Embedded ResNet and CNN-Embedded DRN over their corresponding baseline models were statistically significant. The narrow confidence intervals and extremely small p-values obtained across both datasets provide strong evidence that integrating CNN modules within residual frameworks not only enhances predictive accuracy but also ensures model robustness and stability.



Seasonal evaluations further validated the adaptability of the CNN-Embedded DRN, showing that it maintains high accuracy across various seasonal and climatic scenarios. Overall, these findings demonstrate that embedding CNN-based local feature extraction within ResNets offers a clear and statistically verified advantage for STLF, providing a reliable and generalizable framework for future research on hybrid deep residual models and long-term dependency modeling.

## 5. Conclusion

This study proposed a CNN-Embedded DRN architecture for STLF, uniquely integrating CNN modules into the foundational structure of DRNs. This design effectively balances the extraction of fine-grained local fluctuations with robust long-term feature representation. Extensive experiments on ISO-NE (temperate) and Malaysia (tropical) datasets demonstrated the model's superior generalizability, achieving significant MAPE reductions of 10.94% and 3.71%, respectively, compared to the standard DRN. Experimental results demonstrated that the proposed model consistently outperforms baseline architectures, including CNN-, RNN-, Transformer-, and ResNet-based variants, while

bootstrap-based statistical analysis further confirmed the significance of the observed improvements.

Practically, the model offers a reliable framework for daily scheduling and energy trading by handling diverse climatic conditions with high accuracy and computational efficiency. However, limitations regarding fixed kernel configurations and reliance on deterministic weather data remain. Future research will address these by exploring adaptive convolutional kernels, attention mechanisms, and probabilistic forecasting to support real-time, multi-scale energy management systems.

**Abbreviations**

| Abbreviation | Full name |
|---|---|
| 1D CNN | One-Dimensional Convolutional Neural Network |
| Adam | Adaptive Moment Estimation |
| ANN | Artificial Neural Network |
| BiGRU | Bidirectional Gated Recurrent Unit |
| BiLSTM | Bidirectional Long Short-Term Memory |
| CNN | Convolutional Neural Network |
| Conv1D | One-Dimensional Convolutional Layer |
| CRN | Convolutional Residual Network |
| DNN | Deep Neural Network |
| DRN | Deep Residual Network |
| ELM | Extreme Learning Machines |
| FC | Fully Connected |
| GAP | Global Average Pooling |
| GAP1D | One-Dimensional Global Average Pooling |
| GRU | Gated Recurrent Unit |
| ISO-NE | New England Independent System Operator |
| LF | Load Forecasting |
| LSTM | Long Short-Term Memory |
| MAE | Mean Absolute Error |
| MAPE | Mean Absolute Percentage Error |
| MSE | Mean Square Error |
| MW | Megawatt |
| MTLF | Medium-Term Load Forecasting |
| NMSE | Normalized Mean Square Error |
| N-HiTS | Neural Hierarchical Interpolation for Time Series Forecasting |
| Pooling1D | One-Dimensional Pooling Layer |
| R | Correlation Coefficient |

| | |
|---|---|
| $R^2$ | Coefficient of Determination |
| RBF | Radial Basis Function |
| ReLU | Rectified Linear Unit |
| ResNet | Residual Network |
| ResNetPlus | Modified ResNet Structure |
| RNN | Recurrent Neural Network |
| SELU | Scaled Exponential Linear Unit |
| STLF | Short-Term Load Forecasting |
| SVR | Support Vector Regression |
| VSTLF | Very Short-Term Load Forecasting |

**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

**Data availability statement**

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.

**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.

**Competing Interests Statement**

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

**Funding Declaration**

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

**Reference**

[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 (2024). https://doi.org/10.14716/ijtech.v15i1.5543

[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 13, 78618–78633 (2025). https://doi.org/10.1109/ACCESS.2025.3565798

[3] Koponen P, Ikäheimo J, Koskela J, Brester C & Niska H. Assessing and comparing short term load forecasting performance. Energies 13, 2054 (2020). https://doi.org/10.3390/en13082054

[4] 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 (2013). https://doi.org/10.1109/TPWRS.2013.2269803

[5] 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 (2001). https://doi.org/10.1109/59.910780

[6] Kuster C, Rezgui Y & Mourshed M. Electrical load forecasting models: a critical systematic review. Sustain. Cities Soc. 35, 257–270 (2017). https://doi.org/10.1016/j.scs.2017.08.009

[7] 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 (2015). https://doi.org/10.1109/TIE.2015.2424399

[8] Chen Y et al. Short-term load forecasting: similar day-based wavelet neural networks. IEEE Trans. Power Syst. 25, 322–330 (2009). https://doi.org/10.1109/TPWRS.2009.2030426

[9] Zhao Y, Luh P B, Bomgardner C & Beerel G H. Short-term load forecasting: multi-level wavelet neural networks with holiday corrections. Proc. IEEE Power Energy Soc. Gen. Meet. 1–7 (2009). https://doi.org/10.1109/PES.2009.5275304

[10] Eren Y & Küçükdemiral İ. A comprehensive review on deep learning approaches for short-term load forecasting. Renew. Sustain. Energy Rev. 189, 114031 (2024). https://doi.org/10.1016/j.rser.2023.114031

[11] Li L, Ota K & Dong M. Everything is image: CNN-based short-term electrical load forecasting for smart grid. Proc. 14th Int. Symp. Pervasive Syst. Algorithms Netw. 344–351 (2017). https://doi.org/10.1109/ISPAN-FCST-ISCC.2017.78

[12] Jurado M, Samper M & Rosés R. An improved encoder-decoder-based CNN model for probabilistic short-term load and PV forecasting. Electr. Power Syst. Res. 217, 109153 (2023). https://doi.org/10.1016/j.epsr.2023.109153

[13] Narayan A & Hipel K W. Long short term memory networks for short-term electric load forecasting. Proc. IEEE Int. Conf. Syst. Man Cybern. 2573–2578 (2017). https://doi.org/10.1109/SMC.2017.8123012

[14] Bento P, Pombo J, Mariano S & Calado M R. Short-term load forecasting using optimized LSTM networks via improved bat algorithm. Proc. Int. Conf. Intell. Syst. 351–357 (2018). https://doi.org/10.1109/IS.2018.8710498

[15] 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 (2020). https://doi.org/10.1007/s42835-020-00424-7

[16] 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 (2019). https://doi.org/10.1109/ACCESS.2019.2950957

[17] Ran P, Dong K, Liu X & Wang J. Short-term load forecasting based on CEEMDAN and Transformer. Electr. Power Syst. Res. 214, 108885 (2023). https://doi.org/10.1016/j.epsr.2022.108885

[18] Jiang B, Liu Y, Geng H, Zeng H & Ding J. A transformer based method with wide attention range for enhanced short-term load forecasting. Proc. 4th Int. Conf. Smart Power Internet Energy Syst. 1684–1690 (2022).

[19] Li S, Zhang W & Wang P. TS2ARCformer: a multi-dimensional time series forecasting framework for short-term load prediction. Energies 16, 5825 (2023). https://doi.org/10.3390/en16155825

[20] Chen K et al. Short-term load forecasting with deep residual networks. IEEE Trans. Smart Grid 10, 3943–3952 (2018). https://doi.org/10.1109/TSG.2018.2844307

[21] Tian Y, Yu S, Wen M, Zhang K & Chen Y. Short-term load forecasting scheme based on improved deep residual network and LSTM. Proc. CIRED Berlin Workshop CIRED 2020 117–120 (2020). https://doi.org/10.1049/oap-cired.2021.0257

[22] Li H, Zhang P & Li C. 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. 2030, 012087 (2021). https://doi.org/10.1088/1742-6596/2030/1/012087

[23] Sheng Z, Wang H, Chen G, Zhou B & Sun J. Convolutional residual network to short-term load forecasting. Appl. Intell. 51, 2485–2499 (2021). https://doi.org/10.1007/s10489-020-01932-9

[24] Sheng Z, An Z, Wang H, Chen G & Tian K. Residual LSTM based short-term load forecasting. Appl. Soft Comput. 144, 110461 (2023). https://doi.org/10.1016/j.asoc.2023.110461

[25] Ding A, Liu T & Zou X. Integration of ensemble GoogLeNet and modified deep residual networks for short-term load forecasting. Electronics 10, 2455 (2021). https://doi.org/10.3390/electronics10202455

[26] Ullah K et al. Short-term load forecasting: a comprehensive review and simulation study with CNN-LSTM hybrids approach. IEEE Access (2024). https://doi.org/10.1109/ACCESS.2024.3440631

[27] Hua Q et al. A short-term power load forecasting method using CNN-GRU with an attention mechanism. Energies 18, 106 (2024). https://doi.org/10.3390/en18010106

[28] He K, Zhang X, Ren S & Sun J. Deep residual learning for image recognition. Proc. IEEE Conf. Comput. Vis. Pattern Recognit. 770–778 (2016). https://doi.org/10.48550/arXiv.1512.03385

[29] Challu, C., Olivares, K. G., Oreshkin, B. N., Ramirez, F. G., Canseco, M. M. & Dubrawski, A. N-HiTS: Neural hierarchical interpolation for time series forecasting. Proc. AAAI Conf. Artif. Intell. 37, 6989–6997 (2023). https://doi.org/10.1609/aaai.v37i6.25854

[30] Zhang, Q., Li, C., Su, F. & Li, Y. Spatiotemporal residual graph attention network for traffic flow forecasting. IEEE Internet Things J. 10, 11518–11532 (2023). https://doi.org/10.1109/JIOT.2023.3243122

[31] Bao, Y. X., Cao, Y., Shen, Q. Q. & Shi, Q. Global–local spatial–temporal residual correlation network for urban traffic status prediction. Comput. Intell. Neurosci. 2022, 7344522 (2022). https://doi.org/10.1155/2022/7344522

[32] Ashebir, S. & Kim, S. Energy demand forecasting using temporal variational residual network. Forecasting 7(3), 42 (2025). https://doi.org/10.3390/forecast7030042

[33] Zhang, J., Chen, F., Cui, Z., Guo, Y. & Zhu, Y. Deep learning architecture for short-term passenger flow forecasting in urban rail transit. IEEE Trans. Intell. Transp. Syst. 22(11), 7004–7014 (2020). https://doi.org/10.1109/TITS.2020.3000761

[34] Yamashita R, Nishio M, Do R K G & Togashi K. Convolutional neural networks: an overview and application in radiology. Insights Imaging 9, 611–629 (2018). https://doi.org/10.1007/s13244-018-0639-9

[35] Kiranyaz S et al. 1D convolutional neural networks and applications: a survey. Mech. Syst. Signal Process. 151, 107398 (2021). https://doi.org/10.1016/j.ymssp.2020.107398

[36] Shi H, Xu M & Li R. Deep learning for household load forecasting—a novel pooling deep RNN. IEEE Trans. Smart Grid 9, 5271–5280 (2017). https://doi.org/10.1109/TSG.2017.2686012

[37] Huang G et al. Snapshot ensembles: train 1, get m for free. Preprint at https://arxiv.org/abs/1704.00109 (2017). https://doi.org/10.48550/arXiv.1704.00109

[38] Khoshkangini, R., Tajgardan, M., Lundström, J., Rabbani, M. & Tegnered, D. A snapshot-stacked ensemble and optimization approach for vehicle breakdown prediction. Sensors 23, 5621 (2023). https://doi.org/10.3390/s23125621

[39] Kingma D P & Ba J. Adam: a method for stochastic optimization. Preprint at https://arxiv.org/abs/1412.6980 (2014). https://doi.org/10.48550/arXiv.1412.6980

[40] 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 (2021). https://doi.org/10.1111/nph.17159