IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 21 December 2023, accepted 14 January 2024, date of publication 18 January 2024, date of current version 8 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3355448

RESEARCH ARTICLE

Multi-Task Learning of the PatchTCN-TST Model for Short-Term Multi-Load Energy Forecasting Considering Indoor Environments in a Smart Building

SENFENG CEN AND CHANG GYOON LIM, (Member, IEEE)
Department of Computer Engineering, Chonnam National University, Yeosu, Jeonnam 59626, South Korea

Corresponding author: Chang Gyoon Lim (cglim@jnu.ac.kr)

This work was supported by the Regional Innovation Strategy (RIS) through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (MOE) under Grant 2021RIS-002.

ABSTRACT Energy consumption in buildings contributes to over a third of global energy consumption and 28% of greenhouse gas emissions. With urbanization and population growth, rising building energy demand can lead to environmental degradation. While significant renewable resources are used to generate electricity to mitigate environmental problems, demand-side management remains crucial for achieving net-zero emissions and enhancing energy efficiency. Accurate building load forecasting is pivotal in devising optimal demand response schemes to shift or reduce the demand on power grids. Recent studies have achieved progressive breakthroughs in building energy forecasting through machine learning algorithms. However, most studies focused on building-level energy forecasting rather than individual load forecasting, which cannot support controlled demand response programs. In this study, we propose a multi-task learning model incorporating Patch, Temporal Convolutional Network and Time-Series Transformer (PatchTCN-TST) based on the channel-independent strategy for floor-level multiple electricity loads and indoor environmental forecasting. The PatchTCN-TST model is implemented to predict future data ranging from one-step ahead to three-step ahead on a real-world office building in Bangkok, Thailand. The experiment results indicate that the prediction performance of our model outperforms the prevalent methods, including LSTM, GRU, TCN, Transformer, Informer and Autoformer. The PatchTCN-TST model demonstrates superior accuracy in three forecasting scenarios, significantly reducing MAE, MSE, RMSE, and aSMAPE by 34%, 23%, 12%, and 36.4%, respectively, compared to the best baseline model.

INDEX TERMS Deep learning, demand response, time-series forecasting, smart buildings.

I. INTRODUCTION

Over the past two decades, global annual energy consumption has risen by 41%, attaining 178,889 TWh due to accelerated industrial, infrastructure, and economic growth [1]. The predominant reliance on non-renewable energy sources, coupled with the rising energy demand, has resulted in an exponential increase in greenhouse gas emissions. Such emissions contribute to anthropogenic environmental issues such as global warming, sea-level rise, and ocean acidification [2], [3], [4]. Considering the finite reserves of non-renewable resources and heightened environmental concerns, there has been an augmented integration of Renewable Energy Sources (RESs) into the power grid [5].

While RESs present an eco-friendly, abundant, secure, and scalable solution, their power generation can be fluctuating, intermittent, and non-dispatchable. The high penetration of RESs can cause disparities between power generation and consumption, threatening power system stability [6], especially with the large-scale installation of wind turbines and

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

photovoltaic (PV) panels. To overcome the limitations of RESs, an array of Energy Storage Systems (ESSs) have been employed in the microgrid. ESSs enable energy storage and subsequent discharge when required, which meet energy demand, improve power quality, and augment grid flexibility [7]. Additionally, Demand-Side Management (DSM) is also crucial in ensuring power grid stability and improving energy efficiency [8], [9].

The International Energy Agency (IEA) indicates that the building sector accounts for 30% and 28% of global energy consumption and greenhouse gas emissions in 2022 [10]. Consequently, building energy management is a critical component of DSM for global energy-saving and carbon neutrality [11]. Generally, building types can be broadly categorized into residential, commercial, office, and industrial. It should be noted that energy usage patterns in buildings vary due to the building types and operational schedules. Accurate forecasting of building energy demand is crucial within DSM, facilitating the implementation of optimal control strategies and promoting energy efficiency. Building energy consumption forecasting can be categorized into ultra-short-term, short-term, medium, and long-term, with forecasting horizons ranging from hours to years [12].

In recent years, numerous methods based on Machine Learning (ML) models have been used for building energy forecasting [13]. However, most scholars and pundits have concentrated on building-level energy consumption forecasting, which predicts the entire building's energy consumption. Although the forecasting performance improved with advanced algorithms, achieving optimal energy efficiency control is still challenging due to the limited knowledge of individual load consumption forecasting. To fill this gap, we propose a PatchTCN-TST model for floor-level multiple loads consumption and indoor environmental forecasting in smart buildings. Compared with previous studies, our proposed model forecasts multiple loads consumption rather than total building energy consumption for energy management and optimal scheduling. Among previous multivariate forecasting, our proposed model is based on the channel-independent strategy, assuming each load pattern is unrelated. The main contribution of this study can be summarized as follows:

1) We applied TCN residual block for scalar projection rather than traditional CNN projection in input representation. The main difference between TCN and CNN is that TCN uses casual convolutions to ensure the model extracts features from previous and current information, which is more reasonable in sequential modeling.
2) Our multiple loads consumption and indoor environmental forecasting method is based on multivariate data, which belongs to multi-channel input and multi-channel output. Most models that extract features through adopt the channel-dependent strategy from all time-series data. However, it is proven that the performance of cross-channel fusion in time-series forecasting is worse than channel-independent [14], [15]. Hence, the channel-independent strategy is adopted to extract features from single-channel information but share the same parameters.
3) In order to understand the relationship between sub-series, patching is applied before feeding into the attention mechanisms, which split the input sequence into several sub-series. Unlike most previous models, patching allows the model to extract locality and comprehensive information from sub-series rather than point-wise dependency from entire series like Transformer. Moreover, the patching operation could significantly reduce the computational complexity of the attention mechanism.

The remainder of this paper is organized as follows. Section II introduces the recent literature on building energy forecasting. Section III presents the PatchTCN-TST model for multi-task learning in multiple loads and indoor environmental forecasting. In Section V, the case study and hyperparameter configurations are presented. Section VI analyzes the forecasting performance of our model and other benchmark models, and Section VI summarizes our study.

II. LITERATURE REVIEW

Traditional building load forecasting methods rely on physical-based models, which simulate the building energy consumption based on physical properties and environmental parameters without historical energy consumption data. Various simulation software tools, such as EnergyPlus, eQUEST, and ESP-r, have been developed for building energy consumption forecasting [16], [17]. However, constructing physical-based models is time-consuming, requires numerous variables, and the forecasting performance relies on domain expertise, which constrains real-time DSM. With the advocacy of smart buildings, advanced sensors and energy monitoring units are deployed to harvest indoor environmental and energy consumption data [18], [19]. In light of the abundant data availability and the rapid advancement of ML and Deep Learning (DL) techniques, data-driven models have recently attracted more attention for building energy forecasting [20]. Data-driven models forecast future energy consumption using historical time-series energy consumption and exogenous variables without any intricate building specifics. The data-driven methods of building energy consumption forecasting can be divided into univariate and multivariate based on the number of input features in the model [21]. Univariate methods predict energy consumption only using energy consumption data, while multivariate methods forecast energy consumption by incorporating additional energy-related features, such as humidity, temperature, and wind ambient light.

Early data-driven models utilize statistical methods for building energy consumption forecasting, which primarily include Autoregression (AR), exponential smoothing,

Autoregressive Integrated Moving Average (ARIMA), and Seasonal Autoregressive Integrated Moving Average (SARIMA). Vu et al. introduced an AR model with time-varying components for short-term energy demand forecasting, which gained the best performance compared with five benchmark models [22]. Sen et al. employed the ARIMA model to forecast the energy consumption in India's iron sector [23]. Fang and Lahdelma applied SARIMA combined with linear regression to predict the heat demand based on multivariate data [24].

Although statistical models perform well in stationarity and linear time-series forecasting scenarios, they cannot accurately predict building energy consumption in nonlinear and intricate temporal patterns. To discover the nonlinear energy consumption patterns, ML-based models have been widely adopted to forecast building energy consumption, such as Random Forest (RF) [25], Artificial Neural Networks (ANNs) [26], and Support Vector Regression (SVR) [27]. Wang et al. used RF to predict the hourly building energy consumption on two institutional buildings at the University of Florida [28]. It showed superior forecasting performance compared with the regression tree model and SVR. Lahouar and Ben Hadj Slama forecast day-ahead load consumption in Tunisian Power Company through refined inputs and RF [29]. The experiment results showed that the expert feature selection method improved the RF forecast performance. Ahmad et al. developed an ANN model for hourly HVAC energy consumption forecasting in a hotel using multivariate time-series data, including outdoor air temperature, humidity, and wind speed [30]. Yang et al. combined the SVR with k-shape clustering to improve the forecasting accuracy in different types of ten institutional buildings [31].

Furthermore, advanced algorithms with more sophisticated structures have been proposed to enhance building energy consumption forecasting accuracy, such as Convolutional Neural Network (CNN) [32], Recurrent Neural Network (RNN), Long short-term memory (LSTM), Gated Recurrent Unit (GRU), Temporal Convolutional Network (TCN). Aurangzeb et al. grouped energy customers via the DBSCAN clustering algorithm and built a Pyramid-CNN model to forecast the power load [33]. The proposed Pyramid-CNN obtained the best score on MAPE compared with other ML algorithms. Tan et al. carried out Multi-task Learning via the LSTM (MTL-LSTM) model for predicting total load and electricity, heat and cooling loads [34]. Kim and Cho proposed a CNN-LSTM model based on multivariate time-series data to predict residential energy consumption [35]. Compared with other methods, the evaluation results showed the best performance in CNN-LSTM under different time resolutions. Sajjad et al. proposed a CNN-GRU framework for short-term building energy prediction and evaluated the model in two datasets [36]. Lemos et al. applied the TCN model for monthly energy consumption forecasting in eight different types of buildings [37]. TCN uses dilated causal convolutions to extract features from past information and

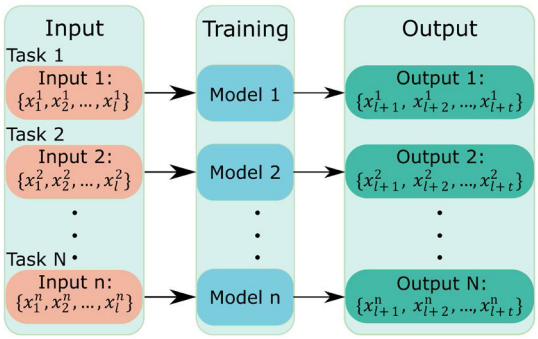


FIGURE 1. The framework of single-task learning.

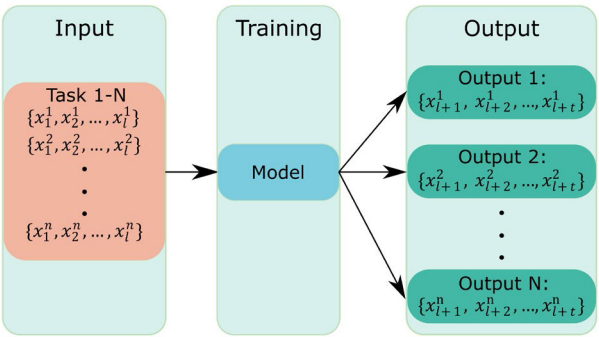


FIGURE 2. The framework of multi-task learning.

a broader range, which are more reasonable for time-series tasks [38].

Moreover, various Transformer-based models have been proposed for sequential tasks in the past five years [39], [40], [41], [42], [43], which performed better than previous algorithms. In the building energy forecasting area, Zhao et al. employed a Transformer model combined with K-Means and Light-GBM for day-ahead load forecasting [44]. In [45], authors proposed a Multiple-Decoder Transformer (MultiDeT) model for day-ahead multienergy load forecasting, which comprises one encoder and multiple decoders. Jiang et al. proposed a Deep-Autoformer that decomposed the series into seasonal and trend parts and designed an auto-correlation mechanism that enables the model to discover the series-wise dependencies for day-ahead residential load forecasting [46]. Their proposed method achieved the best results compared to the five basic models. Given the superior performance of Transformer-based models in building energy consumption forecasting, this paper proposes a PatchTCN-TST framework, providing State-Of-The-Art (SOTA) results for a baseline model in smart building multi-load and indoor environments forecasting.

III. PROPOSED METHOD

This section proposes a novel PatchTCN-TST model for multiple loads and indoor environment forecasting. Our proposed model comprises three main parts: patching operation, TCN embedding, and a TST-based stacked encoder structure. For

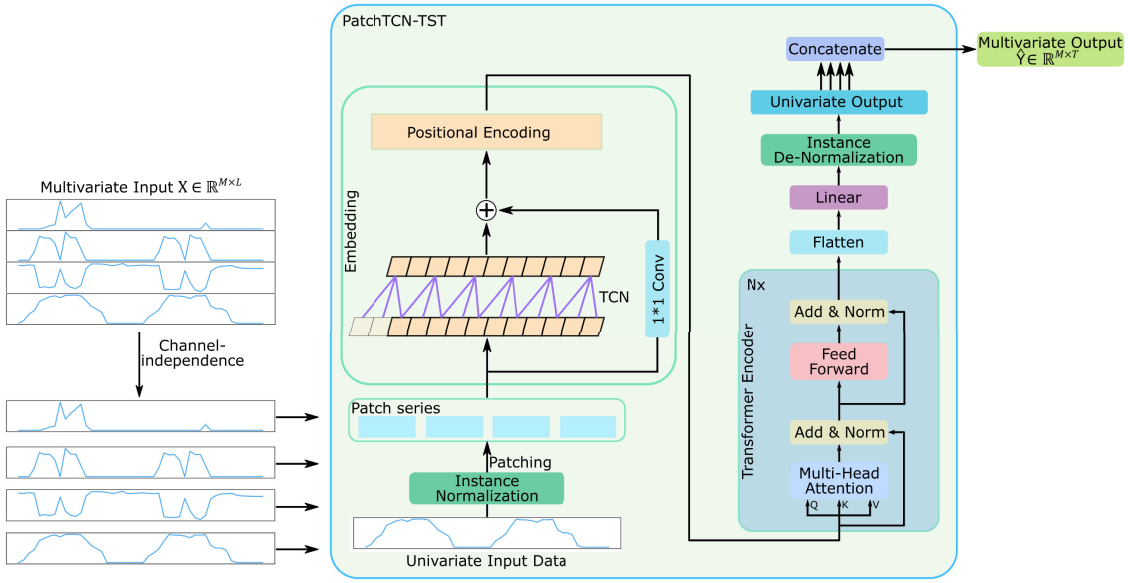


FIGURE 3. The PatchTCN-TST framework.

multivariate time-series forecasting task, given a fixed length L of M dimensional time-series data $X = \{x_1, x_2, \ldots, x_L\} \in \mathbb{R}^{M\times L}$ representing individual loads and indoor environmental features. The goal is to predict the subsequent values for a time horizon T, denoted as $\hat{Y} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_T\}$. The corresponding actual values are denoted as $Y = \{x_{L+1}, x_{L+2}, \ldots, x_{L+T}\}$ where $\hat{Y}$ and $Y \in \mathbb{R}^{M\times T}$. Multiple loads and indoor environmental forecasting can be defined as Multi-task Learning (MTL). The Single-task Learning (STL) and MTL frameworks are depicted in Fig. 1 and 2, respectively. As illustrated in Fig. 1, STL requires several models for each task, which can be time-consuming and computation resource waste. In contrast, MTL is an efficient method by shared parameters to train a unified model for multiple tasks concurrently.

Focusing on the multi-load energy consumption and indoor environmental forecasting, we propose an end-to-end DL-based PatchTCN-TST model without external feature selection and extraction. The framework of the model is shown in Fig. 3, which consists of instance normalization and De-normalization, patching, embedding, and encoder-only structure. PatchTCN-TST predicts multi-load energy demand and indoor environments based on the channel-independent strategy, which individually forecasts multi-load demand for each univariate series. First, the univariate load data is normalized by mean and standard deviation, then segmented into several sub-series. In the embedding stage, the TCN block extracts temporal features and transforms each sub-series to align with the desired model dimensions. Then, positional information is incorporated into the embedded sequence to facilitate the model in discovering the relative positions. Subsequently, stacked Transformer encoders and a linear layer are employed for predicting the normalized single-load energy demand. Finally, multiple individual univariate data are De-normalized and concatenated to generate the multivariate outputs.

A. INSTANCE NORMALIZATION AND DE-NORMALIZATION

In real-world scenarios, most energy load consumption exhibits non-stationarity characteristics. Many studies intended to transfer the series data to stationarity to improve forecasting performance. However, recognizing and accounting for the inherent non-stationarity is essential for accurate forecasting, as it enables the model to capture temporal dependencies effectively. To mitigate the impact of non-stationarity, our model incorporates normalization and de-normalization modules: transfer the non-stationarity series to stationarity at first and revert the initial non-stationarity at the end to obtain the final outputs, respectively. For multi-load and indoor environmental data $X = \{x_1, x_2, \ldots, x_L | x_i \in \mathbb{R}^{M\times 1}\}$ is transformed into the normalized data $X' = \{x_1', x_2', \ldots, x_L' | x_i' \in \mathbb{R}^{M\times 1}\}$. The normalization operation can be defined as follows:

$$\mu_X = \frac{1}{L}\sum_{i=1}^{L} x_i \tag{1}$$

$$\sigma_X{}^2 = \frac{1}{L}\sum_{i=1}^{L} (x_i - \mu_x)^2 \tag{2}$$

$$X' = (X - \mu_X)/\sqrt{\sigma_X + \epsilon} \tag{3}$$

where $\mu_X, \sigma_X \in \mathbb{R}^{M\times 1}$ represent the mean and standard deviation of each measurement unit.

As shown in Fig. 3, the normalized outputs $\hat{Y}' = \{\hat{y}'_{L+1}, \hat{y}'_{L+2}, \ldots, \hat{y}'_{L+T}\} \in \mathbb{R}^{M\times T}$ are de-normalized to recover the original distribution, which can be formulated as:

$$\hat{Y} = \left(\hat{Y}' \times \sigma_X\right) + \mu_X \tag{4}$$

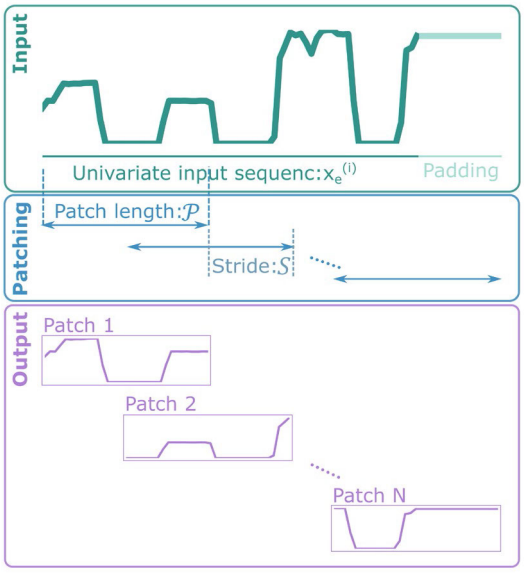


FIGURE 4. Patching operation.

B. PATCHING

Most Transformer-based models commonly employ attention mechanisms to extract point-wise dependencies, which mainly focus on the relationship between points and easily disregard the past series-wise dependencies. Consequently, the scale of point-wise attention has limitations in time-series tasks, as the current data is related to historical data. Considering the constraint of attention mechanisms, we adopt the patching operation to segment the input sequence into several patches regarded as partial features, as shown in Fig. 3. For univariate input sequence $X^i \in \mathbb{R}^{1\times L}$, patch length P and stride S are used to segment the padded univariate data $X^i_{padding} \in \mathbb{R}^{1\times(L+S)}$ where the last boundary value was repeated S times at the end of the input sequence X^i. The generated patches can be denoted as $X^i_{patch} \in \mathbb{R}^{P\times N}$ where $N = \lfloor (L-\mathcal{P})/S \rfloor + 2$ represents the patch number. The patching process is visualized in Fig. 4. Moreover, patching accelerates the computation in attention mechanisms, which shortens the input length L to L/S, approximately.

C. TOKEN EMBEDDING AND POSITIONAL ENCODING

In the token embedding stage, the vanilla Transformer and its variant employ 1D-CNN to transform the input sequence into the embedding dimension d_{model}. However, it should be noted that the output of 1D-CNN at time t is a mapping from the past, current and future data, which is inappropriate for forecasting tasks. To exclude future data, we select the TCN block for token embedding, which consists of two casual convolutional layers, and the detailed structure is shown in Fig. 5. The TCN block operation can be defined as:

$$TCN(X) = ReLU(\mathcal{F}(X) + Conv_{1\times1}(X)) \tag{5}$$

where $\mathcal{F}(\cdot)$ and $Conv_{1\times1}(\cdot)$ represent the casual convolutional and residual connection part. $ReLU(\cdot)$ is the Rectified Linear Unit (ReLU) as an activation function.

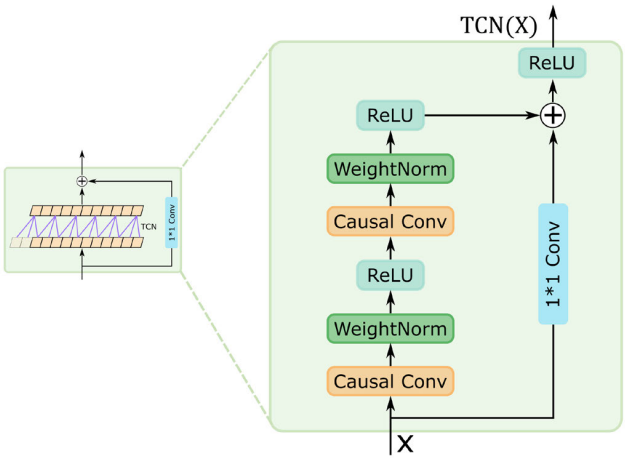


FIGURE 5. The TCN block for embedding.

The TCN block uses casual convolution that the output of time t is only mapping from the elements of time t and past t. In order to speed up the training process and improve generalization, weight normalization is applied to reparameterize the weights within a fixed range after each casual convolutional layer [47]. The casual convolution operation at time t and weight normalization can be defined as follows:

$$F(t) = (x * f)(t) = \sum_{i=0}^{k-1} f(i) \cdot x_{t-i} \tag{6}$$

$$w_{norm} = \frac{w}{\|v\|} g \tag{7}$$

where $*$ denotes the convolutional operation, k is the kernel size in casual convolution, w, w_{norm} denotes the original and normalized weight, $\|v\|$ and g are the Euclidean norm and scalar of the original weight.

In Transformer-based models, attention mechanisms are used to capture the relevant relation to different elements but neglect temporal information. However, the inherent ignorance of temporal information in attention mechanisms can degrade the model performance in sequence tasks. To address this issue, we use sine and cosine functions as positional encoding to add sequential information in time-series inputs, which can be formulated as follows:

$$PE(t,i) = \begin{cases} sin(t/10000^{i/d_{model}}), & if\ i\ is\ even \\ cos(t/10000^{i/d_{model}}), & if\ i\ is\ odd \end{cases} \tag{8}$$

where t is the time t position and $i \in \{0, 1, \ldots, d_{model} - 1\}$ represents i^{th} dimension.

The above equations can be summarized as:

$$X_{d_{model}} = TCN(X) + PE(l_X, d_{model}) \tag{9}$$

where l_X is the length of input X, and $TCN(\cdot)$, $PE(\cdot)$ represent token embedding and positional encoding, respectively.

D. TRANSFORMER ENCODER

As depicted in Fig. 3, our model consists of $N_{encoder}$ stacked encoder blocks. Each block comprises two sub-layers with

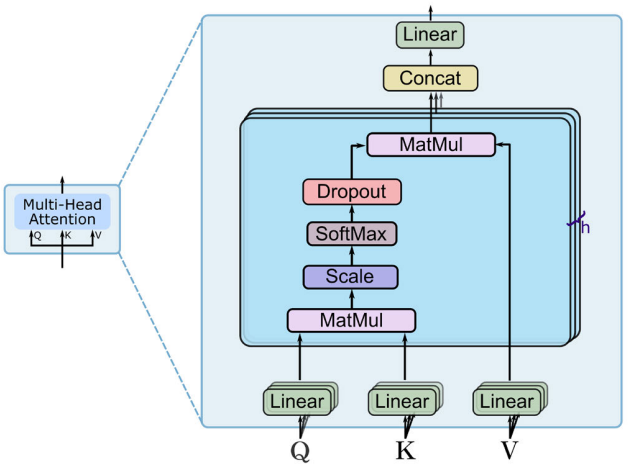


FIGURE 6. Multi-head self-attention mechanism.

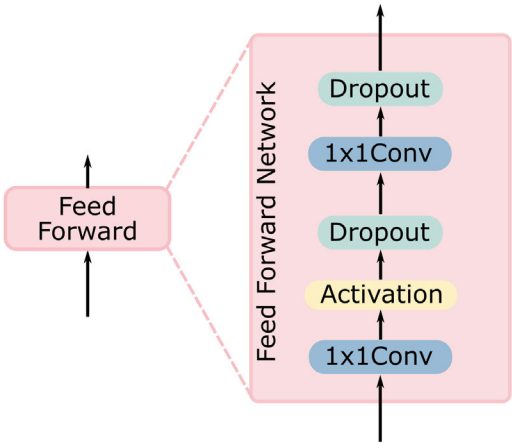


FIGURE 7. Feed-Forward Network.

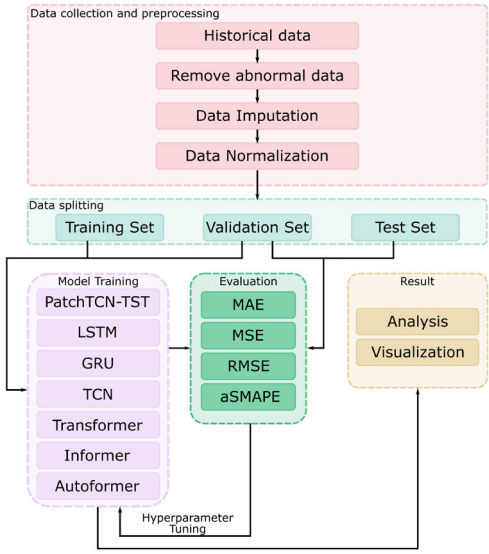


FIGURE 8. The flowchart of our experiments.

residual connection followed by layer normalization: multi-head attention mechanism and feed-forward layer. Hence, the output of i^{th} sub-layer in n^{th} encoder can be expressed as:

$$X_{i+1}^{n} = LayerNorm(X_i^n + SubLayer_i^n(X_i^n)) \tag{10}$$

where X_i^n, X_{i+1}^n denote the input and output of i^{th} sub-layer in n^{th} encoder, $i \in \{1, 2\}$ represents the attention mechanism and feed-forward layer, respectively.

In the first sub-layer, we select the multi-head self-attention mechanism to capture the dependencies between different points, and the structure is shown in Fig. 6. The embedded input $X_{embedding} \in \mathbb{R}^{d_{model} \times N}$ is transformed into queries $\mathcal{Q}^h \in \mathbb{R}^{N \times d_q}$, keys $\mathcal{K}^h \in \mathbb{R}^{N \times d_k}$, and values $\mathcal{V}^h \in \mathbb{R}^{N \times d_v}$ for each head $h \in \{1, 2, \ldots, H\}$ where $d_q = d_k$. For h^{th} head, the weights on values are obtained from the dot-product between $\mathcal{Q}^h$ and $\mathcal{K}^h$. While the large value of d_k increases the value of the dot-product, which causes small gradients after the softmax function. To fix this impact, we rescaled the outputs of attention scores by divide $\sqrt{d_k}$. The outputs of h^{th} scaled dot-product attention can be formulated as:

$$O^h = Attention(\mathcal{Q}^h, \mathcal{K}^h, \mathcal{V}^h) = softmax(\frac{\mathcal{Q}^h \mathcal{K}^{h^T}}{\sqrt{d_k}})\mathcal{V}^h \tag{11}$$

The outputs of each attention head are concatenated and projected by W^O to obtain the final outputs of multi-head attention, which can be calculated as follows:

$$\begin{aligned} O &= MultiHead\ (\mathcal{Q}, \mathcal{K}, \mathcal{V}) \\ &= Concat(O^1, O^2, \ldots, O^h)W^O \end{aligned} \tag{12}$$

where $W^O \in \mathbb{R}^{H*d_v \times N}$ is the trainable projection parameter matrix.

The second sub-layer is the feed-forward network, and the internal structure of the feed-forward network is shown in Fig. 7. It contains two 1×1 convolutional feed-forward layers that process each position separately with different convolutional filters. The equation of the feed-forward network can be expressed as:

$$FeedForward(x) = \sigma(xW_1 + b_1)W_2 + b_2 \tag{13}$$

where W_1, b_1 and W_2, b_2 represent the parameters of the first and second convolutional layer, $\sigma(\cdot)$ is the activation function.

After the stacked encoders, the outputs are flattened and pass through the linear layer to obtain the normalized single-load demand forecasting results $\hat{y}^{i'} \in \mathbb{R}^{1 \times T}$. Afterward, each normalized single-load demand forecasting data is concatenated and De-normalized into the final multi-load forecasting results $\hat{Y} \in \mathbb{R}^{M \times T}$. The pseudocode of the PatchTCN-TST forecasting model is summarized in Algorithm 1.

IV. CASE STUDY

In this section, we compare the other data-driven benchmark models, including LSTM, GRU, TCN, Transformer, Informer, and Autoformer to verify the performance of the

Algorithm 1 Pseudocode for the PatchTCN-TST model

Input: Mini-batch B of historical M features of multiple loads and indoor environmental data for previous L times, $\mathrm{X} \in \mathbb{R}^{\mathrm{B}\times\mathrm{M}\times\mathrm{L}}$;
Output: Mini-batch B of predicted future T time steps data of corresponding M features, $\hat{Y} \in \mathbb{R}^{B\times M\times T}$;

1 $1-$ D Instance Normalization
2 Calculate the mean μ_x and standard deviation σ_x. $\mu_x, \sigma_x \in \mathbb{R}^{B\times M\times 1}$
3 Instance Normalization transform $X' = (x_i - \mu_x)/\sqrt{\sigma_x + \epsilon}$
4
return normalized mini-batch data $X' \in \mathbb{R}^{B\times M\times L}$
5 Patching
6 Pad boundary value on the right side with the size S, $X'_{padding} \in \mathbb{R}^{B\times M\times(L+S)}$
7 Patching $X'_{padding}$ using patch length P and stride S to obtain the N patches sequence
8
return $X'_{patch} \in \mathbb{R}^{B\times M\times P\times N}$
9 Channel-Independent **return** $X'_{ci} \in \mathbb{R}^{(B*M)\times P\times N}$
10 Embedding and positional encoding $X'_{input} \leftarrow TCN\left(X'_{ci}\right) + PE(P, d_{model})$ **return** $X'_{input} \in \mathbb{R}^{(B*M)\times d_{model}\times N}$
11 Transformer encoders
12 **Hyperparameters:** encoder number E, multi-head number H, dimension of query d_q, key d_k and value d_v where $d_q =$
13 $d_k = d_v = d_{model}/H$, feed-forward dimension d_{ff}
14 **Parameters:**
15 For $i \in [E]$:
16 For $h \in [H]$: /* h^{th} head attention parameters for i^{th} encoder */
17 $W^h_{i,q} \in \mathbb{R}^{d_q\times d_{model}}, b^h_{i,q} \in \mathbb{R}^{d_q}, W^h_{i,k} \in \mathbb{R}^{d_k\times d_{model}}, b^h_{i,k} \in \mathbb{R}^{d_k}, W^h_{i,v} \in \mathbb{R}^{d_v\times d_{model}}, b^h_{i,v} \in \mathbb{R}^{d_v}$
18 $W_{i,o} \in \mathbb{R}^{(H*d_v)\times d_{model}}, b_{i,o} \in \mathbb{R}^{(H*d_v)}$, parameters of concatenated multi-head for i^{th} encoder
19 $\gamma^1_i, \beta^1_i, \gamma^2_i, \beta^2_i \in \mathbb{R}^{d_{model}}$, layer normalization parameters of two parts
20 $W_{i,1} \in \mathbb{R}^{d_{ff}\times d_{model}}, b_{i,1} \in \mathbb{R}^{d_{ff}}, W_{i,2} \in \mathbb{R}^{d_{model}\times d_{ff}}, b_{i,2} \in \mathbb{R}^{d_{model}}$, parameters of feed-forward
21 **for** $i = 1, \ldots, E$ **do** /* Encoder number */
22 **for** $h = 1, 2, \ldots, H$ **do** /* Multi $-$ Head Attention */
23 $\mathcal{Q}^h_i \leftarrow W^h_{i,q}X'_{input} + b^h_{i,q}$, queries $\mathcal{Q}^h_i \in \mathbb{R}^{(B*M)\times d_q\times N}$; $\mathcal{K}^h_i \leftarrow W^h_{i,k}X'_{input} + b^h_{i,k}$, keys $\mathcal{K}^h_i \in \mathbb{R}^{(B*M)\times d_k\times N}$
24 $\mathcal{V}^h_i \leftarrow W^h_vX'_{input} + b^h_{i,v}$, values $\mathcal{V}^h_i \in \mathbb{R}^{(B*M)\times d_v\times N}$; $O^h_i \leftarrow Attention(\mathcal{Q}^h_i, \mathcal{K}^h_i, \mathcal{V}^h_i)$, $O^h_i \in \mathbb{R}^{(B*M)\times d_v\times N}$
25 **end**
26 Concate multi-head information $O_i \leftarrow [O^1_i, O^2_i, \ldots, O^H_i]$, $O_i \in \mathbb{R}^{(B*M)\times(H*d_v)\times N}$
27 $X'_{i,atten} \leftarrow W^T_{i,o}O_i + b_{i,o}$, $X'_{i,atten} \in \mathbb{R}^{(B*M)\times d_{model}\times N}$
28 **Add & LayerNorm** $X'_{i,LN\_1} \leftarrow \mathrm{LayerNorm}(X'_{i,atten} + X'_{i-1}|\gamma^1_i, \beta^1_i)$
29 **Feed-forward** $X'_{i,ff} \leftarrow W_{i,2}(\sigma\left(X'_{i,LN\_1}W_{i,1} + b_{i,1}\right)) + b_{i,2}$
30 **Add & LayerNorm** $X'_i \leftarrow \mathrm{LayerNorm}(X'_{i,ff} + X'_{i,LN\_1}|\gamma^2_i, \beta^2_i)$
31 **end**
32 **return** the last encoder output $X'_E \in \mathbb{R}^{(B*M)\times d_{model}\times N}$
33 Obtain normalized forecasting results
34 Reshape and flatten $X'_E \in \mathbb{R}^{B\times M\times(d_{model}*N)}$
35 Linear layer. Parameters: $W_l \in \mathbb{R}^{(d_{model}*N)\times T}$, $b_l \in \mathbb{R}^T$; $\hat{Y}' \leftarrow X'_EW_l + b_l$
36 **return** the normalized result, $\hat{Y}' \in \mathbb{R}^{B\times M\times T}$
37 Obtain the final result by De $-$ normalization
38 De-normalize the $\hat{Y}'$ according to the corresponding mean μ_x and standard deviation σ_x from Step 1
39 **return** forecasting results $\hat{Y} \in \mathbb{R}^{B\times M\times T}$

proposed TCN-PatchTST. To demonstrate the excellence of the model proposed in this study, an experimental workflow was established as shown in Fig. 8. This workflow mainly includes preprocessing, data splitting, and model training and evaluation.

A. DATA DESCRIPTION AND PREPROCESSING

The experimental data (CU-BEMS) were obtained from a large-scale seven-story office building in Bangkok, Thailand [48]. It collected electricity consumption and indoor environmental data at one-minute intervals from 33 zones between July 1, 2018 to December 31, 2019. Each zone includes the electricity consumption data for individual lighting, plug loads, and air conditioning (AC) units, along with indoor environmental measurements of ambient light (lux), relative humidity (%), and temperature ($^\circ C$) by multi-sensors. Fig. 9 (a) and (b) present the floor plans on Floor 1-2 and Floor 3-7, respectively. The red dots on floor plans denote the installation of multi-sensors, except for Floor 1, without environmental sensors. The overall measurements of the office

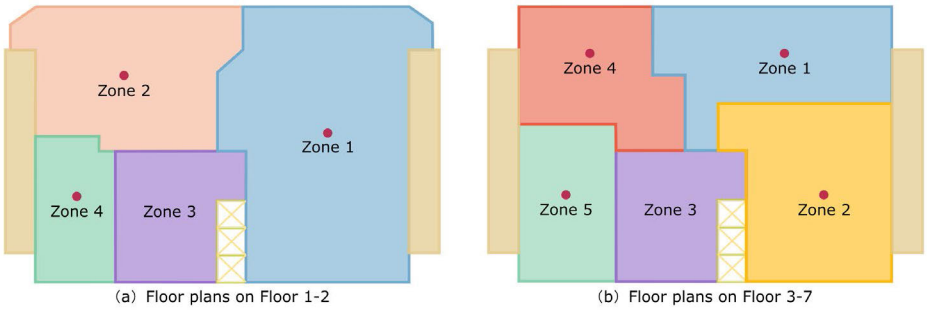


FIGURE 9. The floor plans of smart office building.

TABLE 1. The number of measurements on different floors.

	Number of Power Loads			Indoor Sensors	Total
	AC	Light	Plug load		
Floor 1 (Zone1-4)	[0,4,0,0]	[1,1,1,1]	[0,1,1,1]	[0,0,0,0]	11
Floor 2 (Zone1-4)	[1,14,0,1]	[1,1,1,1]	[1,1,1,1]	[3,3,3,3]	36
Floor 3 (Zone1-5)	[4,1,0,1,1]	[1,1,1,1,1]	[1,1,1,1,1]	[3,3,0,3,3]	29
Floor 4 (Zone1-5)	[4,1,0,1,1]	[1,1,1,1,1]	[1,1,1,1,1]	[3,3,0,3,3]	29
Floor 5 (Zone1-5)	[4,1,0,1,1]	[1,1,1,1,1]	[1,1,1,1,1]	[3,3,0,3,3]	29
Floor 6 (Zone1-5)	[4,1,0,1,1]	[1,1,1,1,1]	[1,1,1,1,1]	[3,3,0,3,3]	29
Floor 7 (Zone1-5)	[4,1,0,1,1]	[1,1,1,1,1]	[1,1,1,1,1]	[3,3,0,3,3]	29

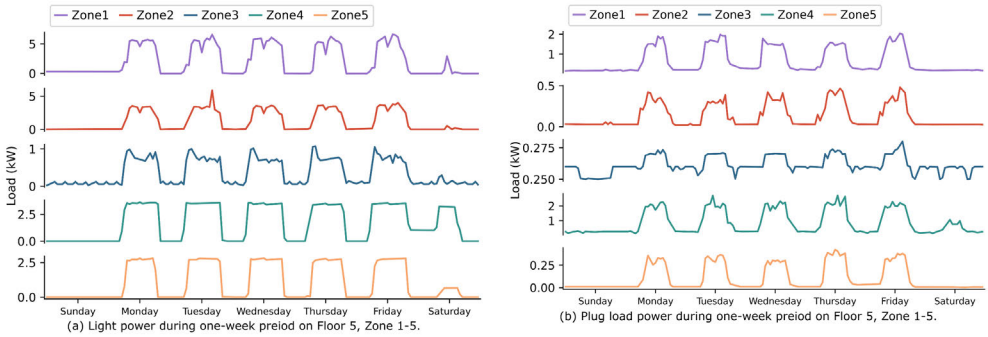


FIGURE 10. The electricity consumption of lighting and plug load in one week.

building contain 55 AC units, 33 lights, 32 plug loads, and 72 sensors, and the corresponding distribution are summarized in Table 1.

Data preprocessing plays an essential role in achieving accurate forecasting owing to the presence of outliers and missing values data during measurement, transmission, and storage. The extreme outliers are identified and removed using box plot analysis, followed by replacement through linear imputation. Missing values are also filled by the linear imputation method. However, Floor 6 is excluded from our research due to the significant amount of missing data. Subsequently, the one-minute interval data are upsampled to an hourly interval. Fig. 10 (a) and (b) display the weekly electricity consumption patterns for lighting and plug load on Floor 5, Zone 1-5. For each zone, the electricity consumption trends for lighting and plug loads are similar on weekdays but with varying magnitudes, while the electricity usage patterns are slightly different on weekends. After that, the first 70% of the data is used for the training set, and the remaining 10% and 20% are used for the validation and test sets, respectively. Additionally, to eliminate the magnitudes of measurements, z-score normalization is applied to rescale the original data into a fixed range, which can be expressed as:

$$x' = \frac{x - \mu}{\sigma} \tag{14}$$

where x represents the original measurement data and x' is the normalized data, μ and σ are the mean and standard deviation values of each measurement.

Moreover, the sliding window method is utilized to obtain the corresponding historical and future data to satisfy the form of input and output in the forecasting model, as shown

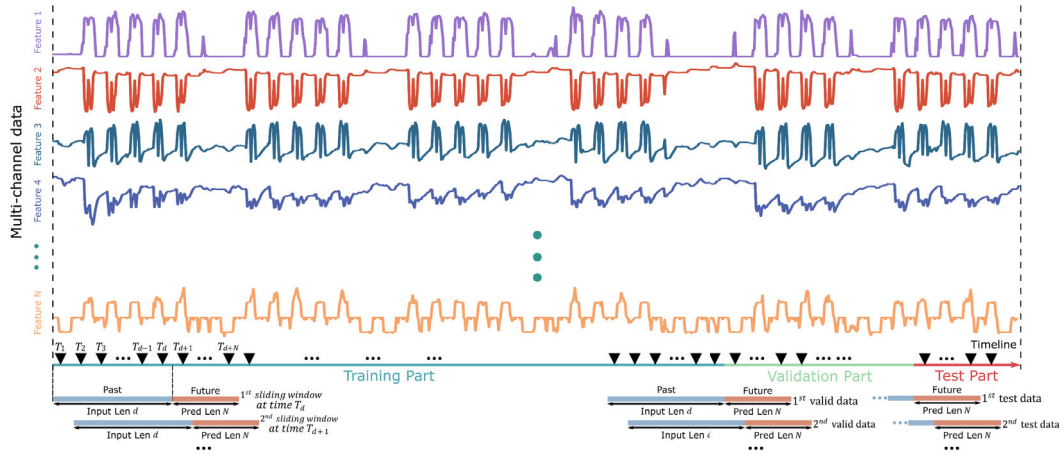


FIGURE 11. Illustration of sliding window method.

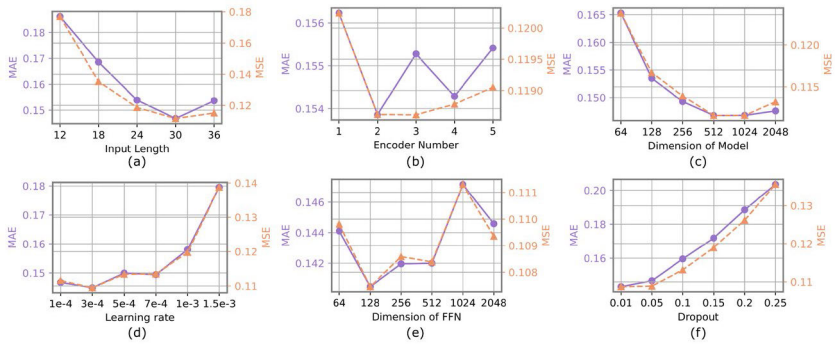


FIGURE 12. The results of hyperparameter selections. (a) the input length, (b) the encoder number, (c) the dimension of model, (d) the initial learning rate, (e) the dimension of feed-forward network, and (f) dropout rate.

in Fig. 11. At time T_d, sliding window method takes the past d hours of multi-load and environmental data as the input and takes the future N hours data as the actual data. The window slides with the stride s until collect the all data.

B. EXPERIMENTAL SETUP AND HYPERPARAMETER TUNING

In this study, we select the Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and adjusted Symmetric Mean Absolute Percentage Error (aSMAPE) metrics to evaluate the forecasting performance, which can be expressed as follows:

$$MAE = \frac{1}{N \times L} \sum_{i=1}^{N} \sum_{j=1}^{L} \left| y_j^i - \hat{y}_j^i \right| \tag{15}$$

$$MSE = \frac{1}{N \times L} \sum_{i=1}^{N} \sum_{j=1}^{L} \left(y_j^i - \hat{y}_j^i \right)^2 \tag{16}$$

$$RMSE = \sqrt{\frac{1}{N \times L} \sum_{i=1}^{N} \sum_{j=1}^{L} \left(y_j^i - \hat{y}_j^i \right)^2} \tag{17}$$

$$aSMAPE = \frac{1}{N \times L} \sum_{i=1}^{N} \sum_{j=1}^{L} \frac{\left| y_j^i - \hat{y}_j^i \right|}{((\left| y_j^i \right| + \left| \hat{y}_j^i \right|)/2) + \varepsilon} \times 100\% \tag{18}$$

where $y_j^i, \hat{y}_j^i \in \mathbb{R}^{M \times 1}$ represent the predicted and actual values at time j, N denotes the number of forecasting data, L is the multi-step forecasting length, and the positive coefficient $\varepsilon = 1$ is added to eliminate the limitation when both forecasted and actual values are close to zero.

In the model training phase, we employ the Adam optimizer with adaptive learning rates to update the model parameters based on MSE loss. In order to avoid overfitting and accelerate the training process, we apply the early stopping method with the patience of 3 epochs and set the batch size to 32. Furthermore, to achieve the best forecasting performance, we conduct experiments to investigate the impact of several hyperparameters and identify the optimal value of hyperparameters. These hyperparameters include input length L_{input}, encoder number $N_{encoder}$, model dimension d_{model}, initial learning rate lr, dimension of feed-forward network d_{ff}, and dropout rate r. The experimental results for various hyperparameters and configurations are presented in Fig. 12. It is evident that both MAE and MSE exhibit similar trends as the hyperparameter values vary, except for the number of encoders. As the incremental length of the input, the reduction of MAE and MSE indicates improved forecasting performance and obtained the best results at an input length of 30. Additionally, the forecasting accuracy improves

TABLE 2. The hyperparameters of comparison models.

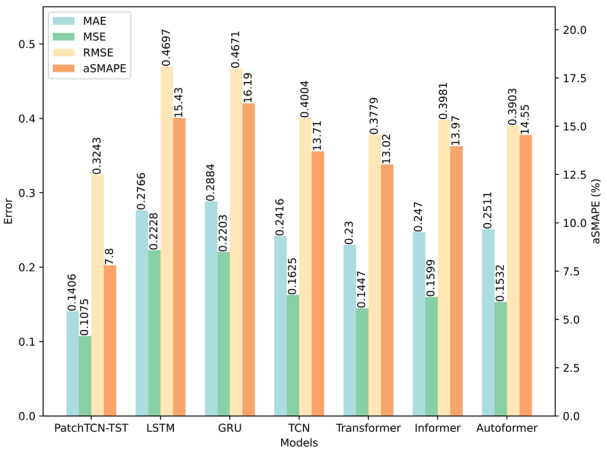
Models	Hyperparameters
LSTM	LSTM layer=1, hidden size=100, batch size=32, learning rate=0.01, dropout rate=0.1, optimizer= Adam, loss function=MSE, epoch=20
GRU	GRU layer=1, hidden size=100, batch size=32, learning rate=0.01, dropout rate=0.12, Optimizer= Adam, loss function=MSE, epoch=20
TCN	TCN block=2, TCN channel size=[64,64], kernel size=1 × 2, batch size=32, learning rate=0.01, dropout rate=0.1, optimizer= Adam, loss function=MSE, epoch=15
Transformer	encoder number=2, decoder number=1, model dimension=512, label length=12, batch size=32, learning rate=0.0001, dropout rate=0.1, Optimizer= Adam, loss function=MSE, epoch=20
Informer	encoder number=2, decoder number=1, model dimension=512, label length=12, prob factor=3,batch size=32, learning rate=0.0001, dropout rate=0.15, Optimizer= Adam, loss function=MSE, epoch=20
Autoformer	encoder number=2, decoder number=1, model dimension=512, label length=12, factor=3, batch size=32, learning rate=0.0001, dropout rate=0.12, Optimizer= Adam, loss function=MSE, epoch=20

with the larger model dimensions, while increased model complexity results in computational resource wastage and time-consuming. Moreover, in Fig. 12 (d) and (f), forecasting results deteriorate with higher learning rates and dropout values. Consequently, a combination of optimal hyperparameters, L_{input}= 30, $N_{encoder}$ = 2, d_{model}= 512, lr= 0.0001, d_{ff} = 64, r= 0.05, is utilized in our proposed model.

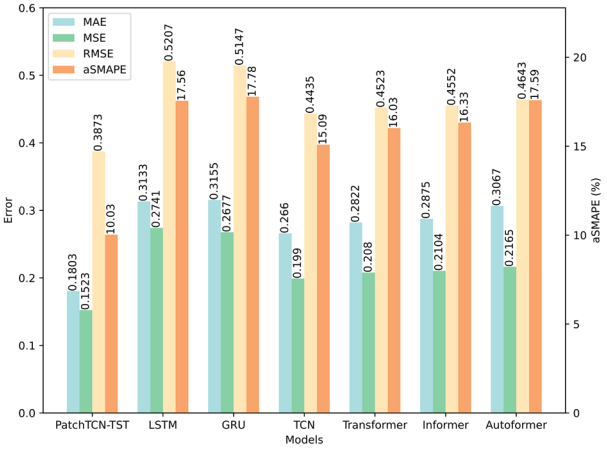
V. COMPARATIVE ANALYSIS

To verify the forecasting performance of PatchTCN-TST, six deep learning models are conducted for comparison, namely LSTM, GRU, TCN, Transformer, Informer and Autoformer. The hyperparameters of comparison models are listed in Table 2. Concretely, LSTM and GRU are recurrent models, and TCN is a variant architecture of the convolutional model that allows parallel computation. Transformer, Informer, and Autoformer belong to the encoder-decoder structure with different attention mechanisms. All of the mentioned models are trained and tested on an Intel Xeon CPU, 64 GB RAM, and NVIDIA A100 40GB GPU using Python 3.8.18 with Pytorch 2.0.1 framework.

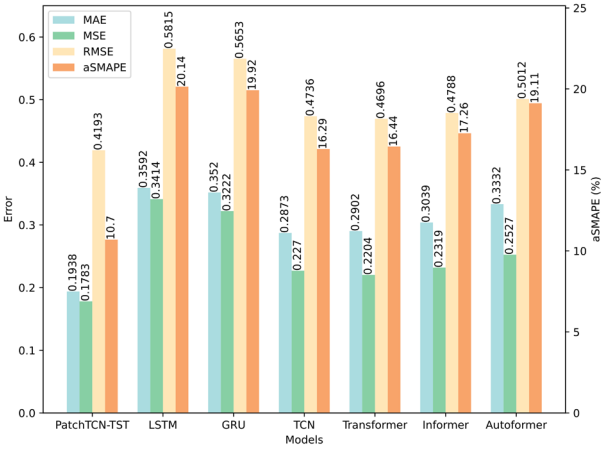
Fig. 13 (a)-(c) present a comprehensive evaluation result of various models' forecasting capabilities across different prediction horizons, which contains one-step ahead, two-step ahead, and three-step ahead predictions, respectively. As evidenced by Fig. 13, the minimal metric error values of PatchTCN-TST indicate that our proposed model surpasses the other benchmark models across the three scenarios with all three metrics. This can be attributed to the usage of channel-independent strategy and patching operation. For two-step ahead predictions, the forecasting performance of the TCN model is better than other compared models. The



(a) one-step ahead forecasting



(b) two-step ahead forecasting



(c) three-step ahead forecasting

FIGURE 13. Evaluation results of forecasting in different models.

recurrent models, namely LSTM and GRU, obtain the highest error rates across three forecasting horizons. Meanwhile, the encoder-decoder models of Transformer, Informer, and Autoformer demonstrate similar errors in each scenario, which might be inappropriate for MTL. Quantitatively, compared to the best benchmark model, PatchTCN-TST yields 34%,

TABLE 3. Performance evaluation of one-step-ahead multi-load and indoor environmental forecast in different models.

Models	Metrics	Floor 2 (11 features)	Floor 2 (36 features)	Floor 3 (29 features)	Floor 4 (29 features)	Floor 5 (29 features)	Floor 7 (29 features)
PatchTCN-TST	MAE	**0.1866**	**0.1193**	**0.1247**	**0.1157**	**0.1274**	**0.1698**
	MSE	**0.1454**	**0.0684**	**0.133**	**0.0744**	**0.0876**	**0.136**
	RMSE	**0.3813**	**0.2615**	**0.3648**	**0.2728**	**0.2961**	**0.369**
	aSMAPE (%)	**10.5853**	**6.7697**	**6.8137**	**6.4153**	**7.04**	**9.1812**
LSTM	MAE	0.2584	0.3042	0.2532	0.2392	0.2613	0.3434
	MSE	0.1857	0.2341	0.215	0.175	0.211	0.3161
	RMSE	0.4309	0.4838	0.4637	0.4183	0.4594	0.5622
	aSMAPE (%)	15.0483	17.1041	14.33	13.2685	14.3993	18.4114
GRU	MAE	0.3239	0.3037	0.2514	0.239	0.2695	0.3426
	MSE	0.2122	0.2155	0.2124	0.1593	0.2103	0.3123
	RMSE	0.4606	0.4642	0.4609	0.3992	0.4586	0.5588
	aSMAPE (%)	18.8371	17.1876	14.3393	13.3691	14.9636	18.4448
TCN	MAE	0.2442	0.2446	0.2502	0.1989	0.2163	0.2954
	MSE	0.1571	0.155	0.1899	0.1065	0.1384	0.2282
	RMSE	0.3964	0.3938	0.4358	0.3264	0.3721	0.4777
	aSMAPE (%)	14.2918	13.7891	14.6275	11.3769	12.0592	16.1043
Transformer	MAE	0.2325	0.2389	0.2111	0.1914	0.2196	0.2865
	MSE	0.1507	0.1291	0.1485	0.0976	0.1334	0.2087
	RMSE	0.3882	0.3593	0.3854	0.3124	0.3652	0.4568
	aSMAPE (%)	13.5363	13.6725	12.2867	10.8769	12.2332	15.5222
Informer	MAE	0.2338	0.2606	0.2342	0.2189	0.2358	0.2987
	MSE	0.1511	0.1547	0.1622	0.1147	0.1555	0.2214
	RMSE	0.3888	0.3933	0.4028	0.3386	0.3943	0.4705
	aSMAPE (%)	13.5785	14.8848	13.6539	12.5092	13.1198	16.0675
Autoformer	MAE	0.277	0.2559	0.2192	0.2178	0.263	0.2737
	MSE	0.1877	0.1423	0.1484	0.115	0.1569	0.169
	RMSE	0.4332	0.3772	0.3853	0.3391	0.396	0.4111
	aSMAPE (%)	16.1244	14.9852	12.755	12.7627	15.2881	15.4058

TABLE 4. Performance evaluation of two-step-ahead multi-load and indoor environmental forecast in different models.

Models	Metrics	Floor 2 (11 features)	Floor 2 (36 features)	Floor 3 (29 features)	Floor 4 (29 features)	Floor 5 (29 features)	Floor 7 (29 features)
PatchTCN-TST	MAE	**0.2265**	**0.1456**	**0.1522**	**0.1414**	**0.2099**	**0.2063**
	MSE	**0.1931**	**0.099**	**0.1734**	**0.1076**	**0.1593**	**0.1815**
	RMSE	**0.4394**	**0.3146**	**0.4164**	**0.328**	**0.3992**	**0.426**
	aSMAPE (%)	**12.8229**	**8.1935**	**8.3403**	**7.7926**	**11.8899**	**11.1227**
LSTM	MAE	0.3225	0.3584	0.2748	0.2683	0.2826	0.373
	MSE	0.2568	0.3256	0.2465	0.2065	0.2406	0.3688
	RMSE	0.5067	0.5688	0.4965	0.4544	0.4905	0.6073
	aSMAPE (%)	18.9429	20.0851	15.598	15.0033	15.6331	20.072
GRU	MAE	0.3303	0.3507	0.2758	0.2733	0.2795	0.3836
	MSE	0.2531	0.2992	0.2444	0.2055	0.227	0.3771
	RMSE	0.5031	0.547	0.4944	0.4533	0.4765	0.6141
	aSMAPE (%)	19.4653	19.8402	15.6891	15.2805	15.6146	20.8008
TCN	MAE	0.2901	0.28255	0.2553	0.2184	0.2333	0.3163
	MSE	0.2148	0.2099	0.2143	0.1341	0.1569	0.2642
	RMSE	0.4634	0.4582	0.463	0.3663	0.3961	0.514
	aSMAPE (%)	17.0441	16.0203	14.6938	12.3893	13.2899	17.0982
Transformer	MAE	0.2861	0.2712	0.2449	0.221	0.3376	0.3325
	MSE	0.2044	0.1745	0.1825	0.132	0.2775	0.277
	RMSE	0.4521	0.4177	0.4272	0.3634	0.5268	0.5263
	aSMAPE (%)	16.7758	15.6293	14.2954	12.465	18.9916	17.9999
Informer	MAE	0.2806	0.287	0.2574	0.2325	0.3326	0.3347
	MSE	0.2009	0.1858	0.1904	0.1339	0.2775	0.2736
	RMSE	0.4482	0.431	0.4363	0.3659	0.5268	0.5231
	aSMAPE (%)	16.4262	16.4984	15.0014	13.2268	18.7606	18.0898
Autoformer	MAE	0.3366	0.3002	0.2957	0.2724	0.2829	0.3525
	MSE	0.2553	0.1884	0.2283	0.1735	0.1874	0.2663
	RMSE	0.5052	0.4341	0.4778	0.4165	0.4359	0.516
	aSMAPE (%)	19.5663	17.5177	17.0316	15.7058	16.1435	19.6014

23%, 12%, and 36.4% averaged improvement on MAE, MSE, RMSE, and aSMAPE, respectively.

Furthermore, the detailed evaluation results of each floor with all forecasting horizons are listed in Table 3-5, and the

TABLE 5. Performance evaluation of three-step-ahead multi-load and indoor environmental forecast in different models.

Models	Metrics	Floor 2 (11 features)	Floor 2 (36 features)	Floor 3 (29 features)	Floor 4 (29 features)	Floor 5 (29 features)	Floor 7 (29 features)
PatchTCN-TST	MAE	**0.2578**	**0.1643**	**0.1729**	**0.1612**	**0.1715**	**0.2349**
	MSE	**0.2321**	**0.1228**	**0.1999**	**0.1363**	**0.1575**	**0.221**
	RMSE	**0.4818**	**0.3504**	**0.4472**	**0.3692**	**0.3969**	**0.4701**
	aSMAPE (%)	**14.6055**	**9.2353**	**9.4904**	**8.8438**	**9.354**	**12.6546**
LSTM	MAE	0.3692	0.4035	0.3088	0.2969	0.3521	0.4244
	MSE	0.3122	0.3946	0.3089	0.2501	0.3226	0.4599
	RMSE	0.5587	0.6282	0.5558	0.5001	0.5679	0.6782
	aSMAPE (%)	21.7988	22.5122	17.3549	16.6274	19.6886	22.836
GRU	MAE	0.3739	0.3848	0.3099	0.2993	0.3273	0.4169
	MSE	0.3075	0.3522	0.2911	0.2461	0.299	0.4371
	RMSE	0.5546	0.5934	0.5395	0.4961	0.5468	0.6611
	aSMAPE (%)	22.1992	21.7464	17.8405	16.8884	18.1917	22.6585
TCN	MAE	0.3205	0.313	0.264	0.2364	0.2435	0.3462
	MSE	0.2502	0.2483	0.2264	0.1543	0.1796	0.3032
	RMSE	0.5002	0.4983	0.4758	0.3928	0.4238	0.5506
	aSMAPE (%)	18.8389	17.7277	15.1356	13.4165	13.7962	18.8073
Transformer	MAE	0.3137	0.2942	0.2645	0.2449	0.2706	0.3534
	MSE	0.2383	0.2014	0.2092	0.1605	0.2071	0.3059
	RMSE	0.4882	0.4488	0.4573	0.4007	0.455	0.5531
	aSMAPE (%)	18.3959	16.9376	15.3707	13.7594	15.0607	19.1368
Informer	MAE	0.3154	0.3131	0.2833	0.2522	0.285	0.3744
	MSE	0.2399	0.2193	0.2241	0.1606	0.2172	0.33
	RMSE	0.4899	0.4683	0.4734	0.4008	0.4661	0.5745
	aSMAPE (%)	18.4909	18.0348	16.4979	14.2757	15.9761	20.3142
Autoformer	MAE	0.3766	0.351	0.2901	0.3169	0.287	0.3775
	MSE	0.3107	0.2477	0.2285	0.2311	0.2008	0.2976
	RMSE	0.5574	0.4977	0.478	0.4807	0.4481	0.5455
	aSMAPE (%)	21.7413	20.5705	16.7367	18.3648	16.3668	20.9123

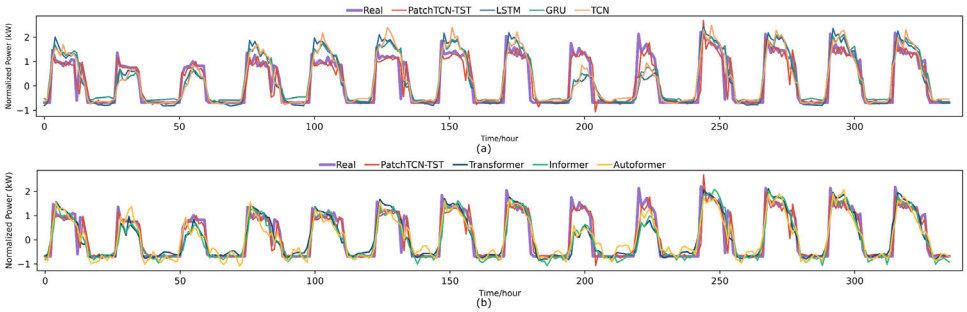


FIGURE 14. Comparative results of one-step-ahead AC1 power forecasting in two weeks on Floor 2, Zone 1.

best results are highlighted in bold. Among three forecasting scenarios, our model consistently outperforms others across different floors. In Table 3, the PatchTCN-TST model demonstrates superior performance on floor 2 with the most feature numbers compared to other floors, which indicates that the forecasting capability of our model is unaffected by the number of features. In contrast, LSTM and GRU obtain similar errors across the floors but generate the worst results, especially on floor 2, which may have potential difficulties in handling multi-channel data. The TCN and Transformer-based models are improved because of the more sophisticated structure and residual connections. Generally, as the prediction horizons increase, the forecasting accuracy diminishes accordingly. From the results of two-step ahead to three-step ahead forecasting in Tables 4 and 5, the metric errors of our model and Transformer-based models increase slightly. Moreover, the three-step ahead forecasting performance on floor 5 of PatchTCN-TST, Transformer and Informer are better than two-step ahead forecasting.

To present the forecasting result, Fig. 14 and 15 show the one-step-ahead power forecasting on AC1 and indoor temperature in two weeks under the different forecasting models. Among different color curves, the bold purple and red curves represent the actual data and forecasting results of our proposed model, respectively. From the two-week AC1 power utilization in Fig. 14 (a), the electricity consumption pattern is based on a daily period with minor variations in electricity consumption magnitude. Among all forecasting results, our proposed PatchTCN-TST model is more accurate than all comparison models. Regarding recurrent models and

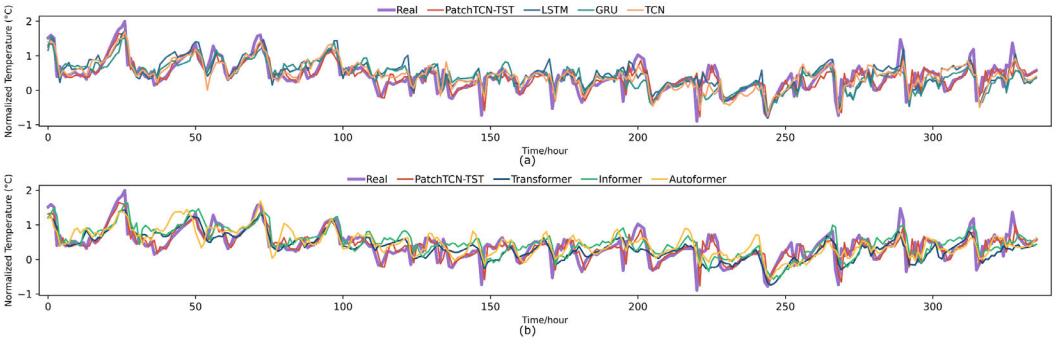


FIGURE 15. Comparative results of one-step-ahead temperature forecasting in two weeks on Floor 2, Zone 1.

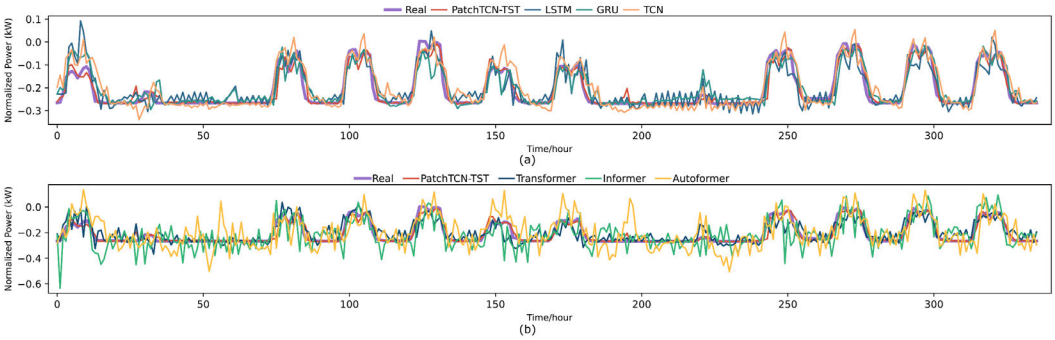


FIGURE 16. Comparative results of two-step-ahead light power forecasting in two weeks on Floor 4, Zone 1.

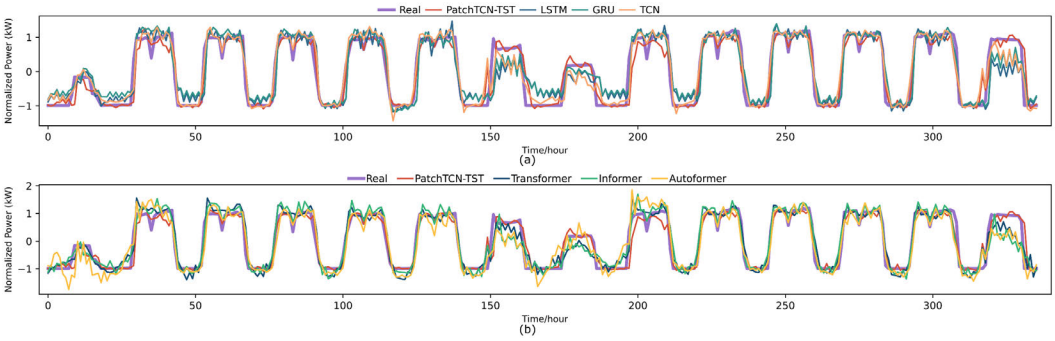


FIGURE 17. Comparative results of two-step-ahead AC1 power forecasting in two weeks on Floor 4, Zone 4.

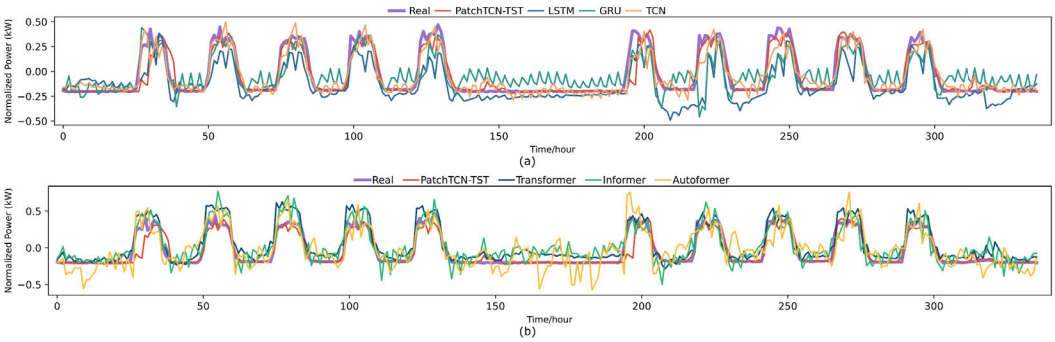


FIGURE 18. Comparative results of three-step-ahead light power forecasting in two weeks on Floor 5, Zone 1.

TCN, they follow the daily trend but cannot precisely predict the power magnitude, which overestimates and underestimates the power demand, especially on the 5^{th}, 6^{th}, 7^{th}, 9^{th}, and 10^{th} days in Fig. 14 (a). In terms of Transformer-based models, it can be observed that the overestimated predictions disappear in Fig. 14 (b), but underestimates still exist on

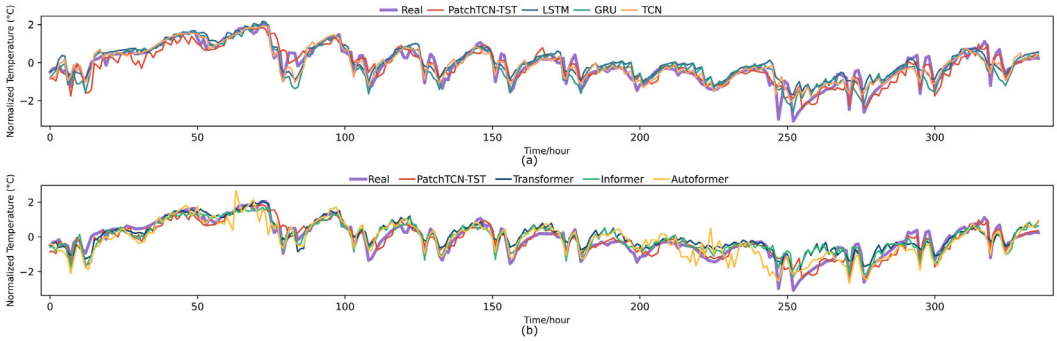


FIGURE 19. Comparative results of three-step-ahead temperature forecasting in two weeks on Floor 5, Zone 4.

the 9th and 10th days. Moreover, the undesired fluctuations occur when the AC1 is powered off, while the Autoformer exhibits the most severe. The corresponding indoor temperature forecasting results on Floor 2, zone 1 are shown in Fig. 15. The temperature degree varies within a certain range and decreases as the air conditioner powers on. Obviously, the predictions of our model could still closely follow the actual temperature trend and obtain the 0.067 MSE score during the plotted two weeks. TCN model also exhibits the admired temperature forecasting results and reaches a 0.0889 MSE score, which is superior to other comparison models.

Fig. 16-19 depict the randomly chosen two-week forecasting results under two-step-ahead and three-step-ahead forecast horizons. Overall, six benchmark models demonstrate drastic deviations from the actual data, while the PatchTCN-TST model could still follow the real variation. For two-step-ahead forecasting, the recurrent models and TCN show sharp fluctuations during the power-off period in Fig. 16 and 17, which does not occur in one-step-ahead forecasting. In Fig. 16 (b), the forecasting results of Informer and Autoformer indicate that the models cannot capture the weekly energy usage pattern. In the predictions of these two weeks, the MSE values of the informer and autoformer models are more than ten times higher than the PatchTCN-TST. From Fig. 18 and 19, the forecasting drawbacks of benchmark models are more evident than two-step-ahead forecasting, especially during the power-off period.

VI. CONCLUSION

This paper proposes an encoder-only PatchTCN-TST model based on the channel-independent strategy for floor-level multi-load and indoor environmental forecasting of an office building in Bangkok, Thailand. The experimental results of hourly forecasting, ranging from 1-h to 3-h ahead, demonstrate that the proposed PatchTCN-TST outperforms six baseline models with smaller evaluation metrics in terms of MAE, MSE, RMSE and aSMAPE. Based on the forecasting performance and evaluation metrics among different models, the superior performance of PatchTCN-TST can be summarized as follows:

1) The channel-independent strategy enables the model to predict multivariate data separately but with shared parameters for multivariate forecasting.
2) Patching operation is applied to segment the input sequence into several sub-series, which enhances the extraction of local dependencies and reduces computational complexity in the attention mechanism.
3) Additionally, the TCN block is used to extract the temporal features rather than the original CNN in embedding.

The poor performance of the baseline models proves that the channel-independent strategy is the main contribution to our methodology. Most models, especially the previously proposed Transformer-based models, primarily focus on multi-step ahead forecasting while neglecting the impact of the mixed channel information, which significantly hinders the forecast performance in multivariate forecasting.

As our method achieves favorable results in multivariate forecasting, one of the most pivotal aspects is the utilization of the channel-independent strategy. On the other hand, the channel-independent strategy disregards inter-channel relationships, such as the strong correlation between AC power and temperature. In future work, we plan to enhance our models by considering high-correlation features rather than relying solely on a channel-independent approach.

REFERENCES

[1] R. Hannah, R. Max, and R. Pablo, "Energy," Our World Data, Tech. Rep., 2022.

[2] A. Mujtaba, P. K. Jena, F. V. Bekun, and P. K. Sahu, "Symmetric and asymmetric impact of economic growth, capital formation, renewable and non-renewable energy consumption on environment in OECD countries," *Renew. Sustain. Energy Rev.*, vol. 160, May 2022, Art. no. 112300, doi: 10.1016/j.rser.2022.112300.

[3] C. Yu, M. Moslehpour, T. K. Tran, L. M. Trung, J. P. Ou, and N. H. Tien, "Impact of non-renewable energy and natural resources on economic recovery: Empirical evidence from selected developing economies," *Resour. Policy*, vol. 80, Jan. 2023, Art. no. 103221, doi: 10.1016/j.resourpol.2022.103221.

[4] A. G. Olabi and M. A. Abdelkareem, "Renewable energy and climate change," *Renew. Sustain. Energy Rev.*, vol. 158, Apr. 2022, Art. no. 112111, doi: 10.1016/j.rser.2022.112111.

[5] S. Fankhauser, S. M. Smith, M. Allen, K. Axelsson, T. Hale, C. Hepburn, J. M. Kendall, R. Khosla, J. Lezaun, E. Mitchell-Larson, M. Obersteiner, L. Rajamani, R. Rickaby, N. Seddon, and T. Wetzer, "The meaning of net zero and how to get it right," *Nature Climate Change*, vol. 12, no. 1, pp. 15–21, Jan. 2022, doi: 10.1038/s41558-021-01245-w.

[6] N. Mararakanye and B. Bekker, "Renewable energy integration impacts within the context of generator type, penetration level and grid characteristics," *Renew. Sustain. Energy Rev.*, vol. 108, pp. 441–451, Jul. 2019, doi: 10.1016/j.rser.2019.03.045.

[7] K. M. Tan, T. S. Babu, V. K. Ramachandaramurthy, P. Kasinathan, S. G. Solanki, and S. K. Raveendran, "Empowering smart grid: A comprehensive review of energy storage technology and application with renewable energy integration," *J. Energy Storage*, vol. 39, Jul. 2021, Art. no. 102591, doi: 10.1016/j.est.2021.102591.

[8] M. Farrokhifar, H. Bahmani, B. Faridpak, A. Safari, D. Pozo, and M. Aiello, "Model predictive control for demand side management in buildings: A survey," *Sustain. Cities Soc.*, vol. 75, Dec. 2021, Art. no. 103381, doi: 10.1016/j.scs.2021.103381.

[9] S. Aslam, S. Aslam, H. Herodotou, S. M. Mohsin, and K. Aurangzeb, "Towards energy efficiency and power trading exploiting renewable energy in cloud data centers," in *Proc. Int. Conf. Adv. Emerg. Comput. Technol. (AECT)*, Feb. 2020, pp. 1–6, doi: 10.1109/AECT47998.2020.9194169.

[10] International Energy Agency. (2023). *Buildings*. [Online]. Available: https://www.iea.org/energy-system/buildings

[11] D. Mariano-Hernández, L. Hernández-Callejo, A. Zorita-Lamadrid, O. Duque-Pérez, and F. S. García, "A review of strategies for building energy management system: Model predictive control, demand side management, optimization, and fault detect & diagnosis," *J. Building Eng.*, vol. 33, Jan. 2021, Art. no. 101692, doi: 10.1016/j.jobe.2020.101692.

[12] M. Zhou, J. Yu, F. Sun, and M. Wang, "Forecasting of short term electric power consumption for different types buildings using improved transfer learning: A case study of primary school in China," *J. Building Eng.*, vol. 78, Nov. 2023, Art. no. 107618, doi: 10.1016/j.jobe.2023.107618.

[13] K. Aurangzeb, "Short term power load forecasting using machine learning models for energy management in a smart community," in *Proc. Int. Conf. Comput. Inf. Sci. (ICCIS)*, Apr. 2019, pp. 1–6, doi: 10.1109/ICCISci.2019.8716475.

[14] L. Han, H.-J. Ye, and D.-C. Zhan, "The capacity and robustness trade-off: Revisiting the channel independent strategy for multivariate time series forecasting," 2023, *arXiv:2304.05206*.

[15] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" 2022, *arXiv:2205.13504*.

[16] Y. Sun, F. Haghighat, and B. C. M. Fung, "A review of the-state-of-the-art in data-driven approaches for building energy prediction," *Energy Buildings*, vol. 221, Aug. 2020, Art. no. 110022, doi: 10.1016/j.enbuild.2020.110022.

[17] A. Foucquier, S. Robert, F. Suard, L. Stéphan, and A. Jay, "State of the art in building modelling and energy performances prediction: A review," *Renew. Sustain. Energy Rev.*, vol. 23, pp. 272–288, Jul. 2013, doi: 10.1016/j.rser.2013.03.004.

[18] J. Al Dakheel, C. Del Pero, N. Aste, and F. Leonforte, "Smart buildings features and key performance indicators: A review," *Sustain. Cities Soc.*, vol. 61, Oct. 2020, Art. no. 102328, doi: 10.1016/j.scs.2020.102328.

[19] B. Dong, V. Prakash, F. Feng, and Z. O'Neill, "A review of smart building sensing system for better indoor environment control," *Energy Buildings*, vol. 199, pp. 29–46, Sep. 2019, doi: 10.1016/j.enbuild.2019.06.025.

[20] M. Khalil, A. S. McGough, Z. Pourmirza, M. Pazhoohesh, and S. Walker, "Machine learning, deep learning and statistical analysis for forecasting building energy consumption—A systematic review," *Eng. Appl. Artif. Intell.*, vol. 115, Oct. 2022, Art. no. 105287, doi: 10.1016/j.engappai.2022.105287.

[21] C. Lu, S. Li, and Z. Lu, "Building energy prediction using artificial neural networks: A literature survey," *Energy Buildings*, vol. 262, May 2022, Art. no. 111718, doi: 10.1016/j.enbuild.2021.111718.

[22] D. H. Vu, K. M. Muttaqi, A. P. Agalgaonkar, and A. Bouzerdoum, "Short-term electricity demand forecasting using autoregressive based time varying model incorporating representative data adjustment," *Appl. Energy*, vol. 205, pp. 790–801, Nov. 2017, doi: 10.1016/j.apenergy.2017.08.135.

[23] P. Sen, M. Roy, and P. Pal, "Application of ARIMA for forecasting energy consumption and GHG emission: A case study of an Indian pig iron manufacturing organization," *Energy*, vol. 116, pp. 1031–1038, Dec. 2016, doi: 10.1016/j.energy.2016.10.068.

[24] T. Fang and R. Lahdelma, "Evaluation of a multiple linear regression model and SARIMA model in forecasting heat demand for district heating system," *Appl. Energy*, vol. 179, pp. 544–552, Oct. 2016, doi: 10.1016/j.apenergy.2016.06.133.

[25] J. L. Speiser, M. E. Miller, J. Tooze, and E. Ip, "A comparison of random forest variable selection methods for classification prediction modeling," *Exp. Syst. Appl.*, vol. 134, pp. 93–101, Nov. 2019, doi: 10.1016/j.eswa.2019.05.028.

[26] A. Bagnasco, F. Fresi, M. Saviozzi, F. Silvestro, and A. Vinci, "Electrical consumption forecasting in hospital facilities: An application case," *Energy Buildings*, vol. 103, pp. 261–270, Sep. 2015, doi: 10.1016/j.enbuild.2015.05.056.

[27] F. Zhang, C. Deb, S. E. Lee, J. Yang, and K. W. Shah, "Time series forecasting for building energy consumption using weighted support vector regression with differential evolution optimization technique," *Energy Buildings*, vol. 126, pp. 94–103, Aug. 2016, doi: 10.1016/j.enbuild.2016.05.028.

[28] Z. Wang, Y. Wang, R. Zeng, R. S. Srinivasan, and S. Ahrentzen, "Random forest based hourly building energy prediction," *Energy Buildings*, vol. 171, pp. 11–25, Jul. 2018, doi: 10.1016/j.enbuild.2018.04.008.

[29] A. Lahouar and J. Ben Hadj Slama, "Day-ahead load forecast using random forest and expert input selection," *Energy Convers. Manage.*, vol. 103, pp. 1040–1051, Oct. 2015, doi: 10.1016/j.enconman.2015.07.041.

[30] M. W. Ahmad, M. Mourshed, and Y. Rezgui, "Trees vs neurons: Comparison between random forest and ANN for high-resolution prediction of building energy consumption," *Energy Buildings*, vol. 147, pp. 77–89, Jul. 2017, doi: 10.1016/j.enbuild.2017.04.038.

[31] J. Yang, C. Ning, C. Deb, F. Zhang, D. Cheong, S. E. Lee, C. Sekhar, and K. W. Tham, "K-shape clustering algorithm for building energy usage patterns analysis and forecasting model accuracy improvement," *Energy Buildings*, vol. 146, pp. 27–37, Jul. 2017, doi: 10.1016/j.enbuild.2017.03.071.

[32] K. Aurangzeb and M. Alhussein, "Deep learning framework for short term power load forecasting, a case study of individual household energy customer," in *Proc. Int. Conf. Adv. Emerg. Comput. Technol. (AECT)*, Feb. 2020, pp. 1–5, doi: 10.1109/AECT47998.2020.9194153.

[33] K. Aurangzeb, M. Alhussein, K. Javaid, and S. I. Haider, "A pyramid-CNN based deep learning model for power load forecasting of similar-profile energy customers based on clustering," *IEEE Access*, vol. 9, pp. 14992–15003, 2021, doi: 10.1109/ACCESS.2021.3053069.

[34] M. Tan, C. Liao, J. Chen, Y. Cao, R. Wang, and Y. Su, "A multi-task learning method for multi-energy load forecasting based on synthesis correlation analysis and load participation factor," *Appl. Energy*, vol. 343, Aug. 2023, Art. no. 121177, doi: 10.1016/j.apenergy.2023.121177.

[35] T.-Y. Kim and S.-B. Cho, "Predicting residential energy consumption using CNN-LSTM neural networks," *Energy*, vol. 182, pp. 72–81, Sep. 2019, doi: 10.1016/j.energy.2019.05.230.

[36] M. Sajjad, Z. A. Khan, A. Ullah, T. Hussain, W. Ullah, M. Y. Lee, and S. W. Baik, "A novel CNN-GRU-based hybrid approach for short-term residential load forecasting," *IEEE Access*, vol. 8, pp. 143759–143768, 2020, doi: 10.1109/access.2020.3009537.

[37] V. H. B. Lemos, J. D. S. Almeida, A. C. Paiva, G. B. Junior, A. C. Silva, S. M. B. Neto, A. C. M. Lima, C. L. S. Cipriano, E. C. Fernandes, and M. I. A. Silva, "Temporal convolutional network applied for forecasting individual monthly electric energy consumption," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2020, pp. 2002–2007, doi: 10.1109/SMC42975.2020.9282960.

[38] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*.

[39] S. Li, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.

[40] N. Kitaev, Ł. Kaiser, and A. Levskaya, "Reformer: The efficient transformer," 2020, *arXiv:2001.04451*.

[41] H. Zhou, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proc. 35th AAAI Conf. Artif. Intell. (AAAI)*, vol. 12B, 2021, pp. 11106–11115, doi: 10.1609/aaai.v35i12.17325.

[42] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2021, pp. 22419–22430.
[43] Y. Nie, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," in *Proc. Int. Conf. Learn. Represent.*, 2022, pp. 1–13.
[44] Z. Zhao, C. Xia, L. Chi, X. Chang, W. Li, T. Yang, and A. Y. Zomaya, "Short-term load forecasting based on the transformer model," *Information*, vol. 12, no. 12, p. 516, Dec. 2021, doi: 10.3390/info12120516.
[45] C. Wang, Y. Wang, Z. Ding, T. Zheng, J. Hu, and K. Zhang, "A transformer-based method of multienergy load forecasting in integrated energy system," *IEEE Trans. Smart Grid*, vol. 13, no. 4, pp. 2703–2714, Jul. 2022, doi: 10.1109/TSG.2022.3166600.
[46] Y. Jiang, T. Gao, Y. Dai, R. Si, J. Hao, J. Zhang, and D. W. Gao, "Very short-term residential load forecasting based on deep-autoformer," *Appl. Energy*, vol. 328, Dec. 2022, Art. no. 120120, doi: 10.1016/j.apenergy.2022.120120.
[47] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 901–909.
[48] M. Pipattanasomporn, G. Chitalia, J. Songsiri, C. Aswakul, W. Pora, S. Suwankawin, K. Audomvongseree, and N. Hoonchareon, "CU-BEMS, smart building electricity consumption and indoor environmental sensor datasets," *Sci. Data*, vol. 7, no. 1, pp. 1–14, 2020, doi: 10.1038/s41597-020-00582-3.

SENFENG CEN was born in Zhejiang, China, in September 1996. He received the bachelor's degree in electrical engineering and automation from the College of Science and Technology, China Three Gorges University, Hubei, China, in 2019. He is currently pursuing the Ph.D. degree with the Department of Computer Engineering, Chonnam National University, South Korea, under the supervision of Prof. Chang Gyoon Lim. His main research interests include machine learning, deep learning, time-series analysis, power quality, energy saving, and demand response.

CHANG GYOON LIM (Member, IEEE) received the Ph.D. degree from the Department of Computer Engineering, Wayne State University, USA, in 1997. Since September 1997, he has been a Professor of computer engineering with Chonnam National University, Yeosu, South Korea. He was the Director of the Home Robot Center, Gwangju Techno Park. He also leads various research projects in his interest areas. His research interests include machine learning, energy optimization, the IoT, big data, virtual power plants, digital twins, and blockchain. He is a member of the Korean Society for Internet Information. He acts as a committee member of several public and private regional institutions. He is also the Chairman of the Board of Coding Insight and WIT.