PhaCIA-TCNs: Short-Term Load Forecasting Using Temporal Convolutional Networks With Parallel Hybrid Activated Convolution and Input Attention

Zhenghua Xu, Zhoutao Yu, Hexiang Zhang, Junyang Chen, Junhua Gu, Thomas Lukasiewicz, Victor C. M. Leung, *(Life Fellow, IEEE)*

***Abstract*—Temporal convolution networks (TCNs) are recently proposed to be used in the short-term load forecasting (STLF) tasks in modern smart grids, however, TCNs have two shortcomings, i.e., redundant convolutional operation and equal input importance problems. Therefore, we propose a novel TCN-based backbone model, called PhaCIA-TCNs, to achieve a more accurate short-term load forecasting, where parallel hybrid activated convolution (PhaC) and input attention (IA) are proposed to resolve the above problems, respectively. Specifically, IA is proposed to highlight important input elements while depressing irrelevant ones, which thus rises the model's forecasting accuracies but also brings additional time-cost; then PhaC is further proposed to remedy the efficiency problem and to further enhance the forecasting accuracies by shortening the convolutional learning path to overcome the redundant convolutional operation problem. Extensive experimental results show that (i) PhaCIA-TCNs significantly outperform all state-of-the-art RNN-based and TCNs-based baselines in forecasting-error-based evaluation metrics on all datasets; (ii) ablation studies show that PhaC and IA are both effective and essential for PhaCIA-TCN to achieve the superior forecasting accuracies in STLF tasks, and by integrating IA and PhaC with TCN, the proposed PhaCIA-TCN not only greatly outperforms TCN in forecasting accuracies but also keeps similar (sometimes even better) learning efficiency.**

***Index Terms*—Short-Term Load Forecasting, Temporal Convolution Networks, Input Attention, Hybrid Convolution**

I. Introduction

SMART grid can use information and communication technology to adjust the production, transmission, and distribution of electricity to save energy and enhance the reliability of the power system [3]. Power load forecasting is an essential module in smart grids, which uses historical load information to predict the future load and to help power system planning [13]. Short-term load forecasting (STLF) is an important load forecasting task, which aims to predict the power load of the grid in the near future, e.g., load over the following hours, days, or weeks, to guarantee that the generated energy always matches the power load demand precisely [7].

In recent years, deep learning techniques have been widely applied in short-term load forecasting tasks [13]. Recurrent neural networks (RNNs), which utilize a cyclic structure to learn sequential features based on historical information, are the most widely adopted deep learning methods for load forecasting [19]. However, vanilla RNNs have the inherent shortcomings of gradient explosion and vanishing, which greatly restricts their practical performance. Consequently, long short-term memory (LSTM) networks [9] are further proposed to overcome this problem by using gate mechanisms to capture long-term dependencies, which has also been applied in STLF [12]. Nevertheless, the structure of LSTMs is relatively complex with a large number of parameters, which thus makes the model difficult to learn.

Therefore, gated recurrent unit (GRU) networks [2] have further been proposed to simplify LSTMs by combining the input and forget gates into a single update, which have been proved to achieve better prediction accuracies than LSTMs in the tasks of STLF [4]. Furthermore, bidirectional models [17], e.g., bidirectional RNNs (Bi-RNNs), bidirectional LSTMs (Bi-LSTMs), and bidirectional GRUs (Bi-GRUs), that consider not only historical but also future information, have also been applied to STLF and achieved state-of-the-art performances [20].

Despite achieving some great successes, the performances of RNNs and their advanced variants in STLF tasks are still unsatisfactory. This is because their inherent cyclic structure makes them have to learn non-linear relationships between elements of sequences in a lengthy and unbalanced way [6]. Specifically, although RNN-based models are capable of utilizing long-term historical information for sequential prediction, the amount of non-linear processing for different elements in a given sequence is different, e.g., the first element is transformed many times (over-processed) while the last element is processed only once (under-processed). So, this unbalanced non-linearity problem inevitably limits the learning capability of RNN-based models for historical information, and thus

This work was supported by the National Natural Science Foundation of China under the grants 62276089, 61906063 and 62102265, by the Natural Science Foundation of Hebei Province, China, under the grant F2021202064, and by the Natural Science Foundation of Guangdong Province, China, under Grant 2022A1515011474. (Corresponding authors: Zhenghua Xu, Hexiang Zhang, Junhua Gu; e-mails: zhenghua.xu@hebut.edu.cn, 202222901005@stu.hebut.edu.cn, jhgu@hebut.edu.cn.)

Zhenghua Xu and Hexiang Zhang are with the State Key Laboratory of Reliability and Intelligence of Electrical Equipment and the School of Health Sciences and Biomedical Engineering, Hebei University of Technology, Tianjin, China.

Zhoutao Yu is with the State Key Laboratory of Reliability and Intelligence of Electrical Equipment, Hebei University of Technology, Tianjin, China.

Junhua Gu is with the State Key Laboratory of Reliability and Intelligence of Electrical Equipment and the School of Artificial Intelligence, Hebei University of Technology, Tianjin, China.

Junyang Chen and Victor C. M. Leung are with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China

Thomas Lukasiewicz is with the Institute of Logic and Computation, TU Wien, 1040 Vienna, Austria, and the Department of Computer Science, University of Oxford, Oxford, United Kingdom.

weakens their forecasting performance.

Therefore, a convolution-based sequential learning model, temporal convolution networks (TCNs), has recently been proposed to remedy this problem, where a succession of dilated casual convolution layers are used to learn non-linear temporal relationships between elements in a shorter and balanced way [1]. Specifically, stacked casual convolution operations make all elements be processed by the same number of non-linearities, and the dilated convolution operations are used to extend the receptive field and ensure that the model can learn long-term sequential relationships by a shorter path [5]. Despite having been applied in many sequential prediction tasks [8], there exist only a limited number of research works using TCNs for STLF [26], where the existing TCNs are directly used without any improvement. However, TCNs are not flawless: (i) **redundant convolutional operation problem**: each layer of TCNs consists of two consecutive convolutional operations, which improve the model's learning capability but also inevitably increase the path of backpropagation and the time-cost of each learning epoch; and (ii) **equal input importance problem:** the same number of non-linearities make TCNs treat all input elements equally, however, some elements actually contain more important information than others, e.g., elements at peaks or valleys of a load sequence.

Consequently, in this work, we propose a new TCN-based backbone model, called PhaCIA-TCN, to replace the existing TCNs models and achieve a more accurate STLF. Two innovative improvements, namely, parallel hybrid activated convolution (PhaC) and input attention (IA), respectively, are proposed to resolve the above shortcomings of the existing TCNs-based models. Specifically, inspired by the attention mechanism [25], the first contribution of PhaCIA-TCN is to develop a novel input attention (IA) module to assign importance weights to elements of the input load sequences. Specifically, the weights are computed in IA using one linear and two non-linear fully-connected layers to estimate self-dependencies between the elements of the input sequences, which thus highlights the elements that are important for STLF tasks and suppresses the inessential ones. Our experiments demonstrate that IA can not only be applied to TCN-based models to help them achieve a superior STLF performance, but it can also be easily incorporated with all existing RNN-based deep models to greatly enhance their forecasting accuracies. This thus proves the effectiveness, applicability, and portability of IA in STLF.

However, although the input attention module can greatly rise the forecasting accuracies, due to additional attention operations, it will inevitably bring additional computational costs and thus decrease the learning efficiency of the forecasting model. Consequently, parallel hybrid activated convolution (PhaC) modules are further proposed to replace the original residual blocks (which contain two consecutive convolution operations) in the existing TCNs to remedy this efficiency problem and further increase the model's forecasting accuracies. Specifically, PhaC consists of solely a dilated causal convolution operation, which greatly reduces the model's convolution depth and learning time-cost; then, two different parallel-arranged non-linear activation operations are used to avoid the decline of learning ability caused by reducing convolution operations. Our experiments show that using PhaC not only decreases the TCN model's convergence time but also further increases the model's forecasting accuracy in STLF tasks. Consequently, by integrating the proposed IA and PhaC modules with TCN, the proposed PhaCIA-TCN will be guaranteed to greatly outperform TCN in forecasting accuracies (because both modules boost the forecasting accuracies), and keep similar (sometimes even better) learning efficiency as TCN (because IA introduces additional time-costs but PhaC remedies it, the two modules complement each other, making the learning efficiency remains relatively stable).

The contributions of this paper are summarized as follows:

- We identify existing shortcomings of RNN-based and TCN-based STLF models, and propose a novel model, called PhaCIA-TCN, to remedy these problems and achieve a more accurate power load forecasting on three public datasets. To our knowledge, PhaCIA-TCN is the first work that optimizes the structure of TCNs so as to obtain a better TCN-based backbone model for sequential learning and forecasting tasks (including but not limited to STLF).
- A highly applicable and portable attention mechanism, input attention (IA), is first proposed to assign importance weights to elements of input sequences, which thus resolves the equal input importance problem and helps STLF models better utilize elements with important information in the learning process. Then, to remedy the additional computational costs of IA, a parallel hybrid activated convolution (PhaC) module is further proposed to shorten the convolutional learning path, which can decrease the time-cost of TCN-based models and further enhance the forecasting accuracies by overcoming the redundant convolutional operation problem.
- We conducted extensive experiments on three public power load datasets, and the results show that (i) PhaCIA-TCN significantly outperforms the state-of-the-art RNN-based and TCNs-based STLF baselines in various STLF tasks in terms of both evaluation metrics, (ii) ablation studies show that the proposed two improvement modules, PhaC and IA, are both effective and essential for PhaCIA-TCN to achieve the superior forecasting accuracies in STLF tasks, and by integrating IA and PhaC with TCN, the proposed PhaCIA-TCN not only greatly outperforms TCN in forecasting accuracies but also keeps similar (sometimes even better) learning efficiency.

The rest of this paper is organized as follows. Section II reviews the related works, and PhaCIA-TCN is introduced in Section III. Section IV presents the experimental studies and evaluates the results. Section V contains some conclusions and potential future works.

II. Related Work

RNN-Based Models for STLF. RNN-based models are the most widely used deep-learning-based methods for STLF. Shi et al. propose a novel pooling-based deep RNN for STLF to learn the high volatility and uncertainty of load profiles to

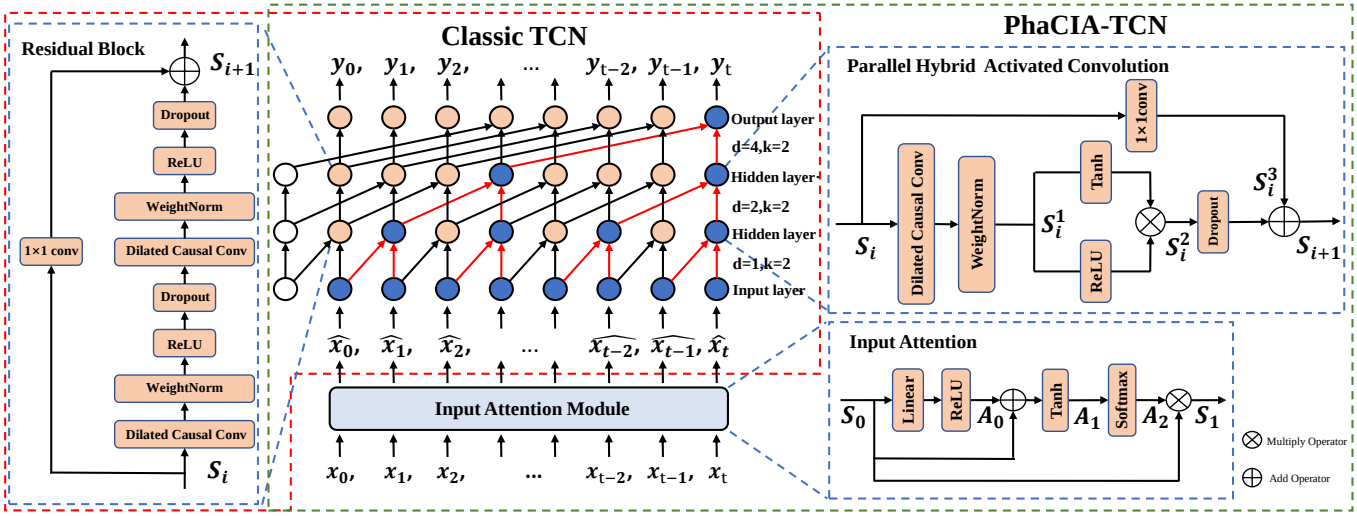


Fig. 1: Overall Structure of the existing TCNs and the proposed PhaCIA-TCNs. Orange circles represent the nodes of TCN and black arrows represent the information processing modules (i.e., residual block for classic TCN and parallel hybrid activated convolution (PhaC) block for the proposed PhaCIA-TCN) between adjacent layer nodes; in addition, blue circles and red arrows are the same things as orange circles and blue arrows, and their different colors are used to highlight a processing procedure example of dilated convolution between different layers with dilated sizes $d = 1, 2, 4$.

predict short-term household loads in Ireland [19]. Kong et al. use LSTM for short-term residential load forecasting in Australia to tackle the high volatility and uncertainty issue in load profiles and report very reliable and robust results [12]. A feature-selection-based empirical-model-decomposition GRU is proposed in [4] to achieve a good STLF accuracy in three American datasets and a Chinese power load dataset [4]. Liu et al. propose a combined model based on Bi-directional for STLF, and the experiments on a Chinese power load dataset exhibit accurate forecasting results [15]. However, the existing RNN-based models all suffer from the unbalanced non-linearity problem, so in this work, we use TCNs as the backbone model for the prediction of sequential data.

TCN-Based Models for STLF. Although TCNs have been widely applied in many sequential prediction tasks, there exists only a limited number of works that use TCNs for STLF tasks. A MTCN is proposed in [26] to learn both nonlinear feature and time series characteristics of load data, where a multi-temporal-spatial-scale method is used for data preprocessing to reduce the data noise and enhance the time series characteristics. Wang et al. integrate TCNs with a light gradient boosting machine (lightGBM) to achieve STLF for industrial customers, where TCNs are used to extract hidden features and long-term temporal relationships, and lightGBM is used for result forecasting in power load sequences from China, Australia and Ireland [22]. However, all the existing TCN-based STLF models use vanilla TCNs directly without any structure optimization; as identified above, vanilla TCNs also encounter two shortcomings: redundant convolutional operations and equal input importance. So, parallel hybrid activated convolution (PhaC) and input attention (IA) modules, respectively, are proposed in PhaCIA-TCN to resolve these shortcomings. To our knowledge, PhaCIA-TCN is the first TCN-based STLF work to optimize the structure of TCNs.

Attention Mechanism in STLF Works. The attention mechanisms [24] have also been applied in the existing RNN-based [21] and TCN-based [18] STLF works. Wang et al. introduce additional attention and rolling update modules into Bi-LSTM, where the attention module is used to estimate the weights of possessed load sequence coming from the rolling update module to highlight the effective features and achieve better results on two Australian STLF datasets [21]. Shi et al. propose a hybrid neural network that combines vanilla TCNs with GRUs for STLF, where an attention layer is added after a GRU layer to highlight the key features generated by GRU [18]. Differently from these works, our input attention (IA) is added before TCNs, aiming to assign different importance weights to elements of the input load sequences; due to its high applicability and portability, the proposed IA module can also be incorporated into the above works to further improve their forecasting performances.

CNN or TCN based models for other forecasting tasks. CNN and TCN are also widely used in many other sequential data based forecasting tasks that are closely related to STLF. For example, CNN and TCN are widely used in the task of traffic forecasting to model the temporal information within traffic data. Specifically, a Graph WaveNet model is proposed in [23] to predict the future traffic condition, where TCN and GCN are used to learn the temporal and spatial features, respectively. Then, a GAMCN is further proposed in [16] where a variant of GCN named LPGCN and a multi-path CNN (which has a similar structure as TCN) are proposed to better model the spatial and temporal factors of traffic data. Besides traffic forecasting, CNN or TCN is also used in many other sequential forecasting tasks, e.g., medical time series classification [14], dynamic prediction [11], and weather forecasting [8], to model the sequential features. Similar to our work, these CNN or TCN based forecasting methods also use convolutional networks to model the long-term dependency of sequential data. However, the CNN or TCN models used in these works also encounter redundant convolutional operation and equal input importance problems. Since PhaCIA-TCN focus on resolving these two problems to

Algorithm 1 PhaCIA-TCNs

Require: Input data $x_0, x_1, x_2, ..., x_t$, max epoch t_{max}, batch size S.

1: **for** $t = 0$ to t_{max} **do**
2: Get the $S_0, S_1, ...S_i$ according to the batch size and input sequence data
3: Obtaining a intermediate attention vector A_0 using Eq. 1
4: Obtaining a *completed attention vector* A_1 using Eq. 2
5: Generating the *weighted load sequence* S_1 using Eq. 3. Then, getting intermediate data sequence $\widehat{x_0}, \widehat{x_1}, \widehat{x_2}, ..., \widehat{x_t}$
6: Obtaining an *intermediate sequence* S_i^1 using Eq. 4
7: Getting an *activated sequence* S_i^2 using Eq. 5
8: Obtaining a *residual sequence* S_i^3 using Eq. 6
9: Using an element-wise summation Eq 7 to obtain the output sequence of PhaC S_{i+1}. Get output sequence $y_0, y_1, y_2, ..., y_t$ according to the batch size
10: **end for**

optimize the structure of TCNs and obtain a better TCN-based backbone model for sequential data forecasting, PhaCIA-TCN can also be integrated with these works to further enhance their performances in the corresponding forecasting tasks.

III. Methodology

In order to resolve the existing shortcomings of using vanilla TCNs in STLF tasks, in this work, a novel advanced TCN model, called PhaCIA-TCN, is proposed. As shown in Fig. 1, PhaCIA-TCNs adopt the classic TCN structure as the model's backbone to learn the features and temporal relationships from the input power load sequences. However, differently from vanilla TCN, parallel hybrid activated convolution (PhaC) modules are proposed in PhaCIA-TCNs to replace the original residual blocks, where we first remove a "redundant" convolution operation to reduce the model's convolution depth and enhance the learning efficiency, then we further use two different parallel non-linear activation operations to ensure sufficient non-linearities for the model to maintain a good learning capability. Moreover, an additional input attention (IA) module is integrated into PhaCIA-TCNs to estimate the self-dependencies between elements of the input sequences to highlight the important elements by assigning higher weights. The pseudo-codes of the proposed PhaCIA-TCNs model are shown in algorithm 1.

A. Classic TCN Structure

As shown in Fig. 1, the classic TCN structure effectively learns non-linearity features by a stack of dilated casual convolutional layers. Specifically, causal convolutions are the central structure of TCN, which are designed to make the model capable of processing sequential data and utilizing historical information using convolutional operations. Furthermore, zero-padding is utilized to ensure the sufficient processing of boundary elements. Due to the limited receptive field, for long sequential data, it needs many layers of causal convolution to make full use of all historical information. Dilated convolutions are thus introduced into TCNs to remedy this problem by enlarging the receptive field of each convolution operation, and make TCNs capable of capturing long-range dependencies using a reasonable number of convolution layers. Finally, skip connections are used in adjacent layers to improve the model's feature learning and convergence capability. See [1] for more details about TCNs.

B. Input Attention Module

In PhaCIA-TCN, we first integrate an input attention module (IA) into the classic TCN backbone to highlight the important information of the input sequences, e.g., elements at peaks or valleys of a power load sequence. IA mainly consists of a fully connected linear layer (denoted Linear in Fig. 1), two fully connected non-linear layers using ReLu and Tanh as activation functions (denoted ReLU and Tanh), respectively, and two residual connections. Generally, the linear and non-linear fully connected layers are utilized to estimate the linear and non-linear dependencies between elements of input sequences, respectively, while the two residual connections are used to enhance the feature learning and convergence ability of the IA module. Finally, the estimated element-wise dependencies are normalized by a Softmax operation to obtain a probability distribution vector, which is then multiplied by the input load sequence. With the help of IA, the important elements in the input load sequence are assigned higher weights.

The formal definitions and detailed operations of IA are as follows. As shown in Fig. 1, the input load sequence is denoted as S_0, where $S_0 \in \mathbb{R}^T$, and T is the length of S_0. In the IA module, S_0 is first sent to a fully connected linear layer, and then further processed by a fully connected non-linear layer (with ReLU as activation function) to obtain an *intermediate attention* vector A_0. Formally,

$$A_0 = ReLU(Linear(S_0)). \tag{1}$$

Then, A_0 is added with S_0 by an element-wise sum operation, and the summation result is further processed by the other fully connected non-linear layer (with Tanh as activation function) to obtain a *completed attention vector* A_1. A_1 is then normalized by a Softmax operation to obtain the final *weighted importance vector* A_2. Formally,

$$\begin{aligned} A_1 &= Tanh(A_0 + S_0) \\ A_2 &= Softmax(A_1). \end{aligned} \tag{2}$$

Finally, the weighted importance vector A_2 is applied to the original input load sequence S_0, using an element-wise multiplication to generate the *weighted load sequence* S_1.

$$S_1 = A_2 \otimes S_0. \tag{3}$$

C. Parallel Hybrid Activated Convolution Module

In order to reduce the redundant convolution operations in vanilla TCNs while maintaining a superior learning capability, we creatively propose a parallel hybrid activated convolution (PhaC) module to replace the original residual block in TCNs. As shown in Fig. 1, the original residual block of vanilla TCNs

consists of two consecutively connected basic convolutional modules, which are identical with the same dilated causal convolutional layer, normalization layer, ReLU operation, and dropout. Although such a "redundant" structure may increase the model's learning capability with the help of additional non-linearity, it inevitably increases the complexity of the model and makes the model much more difficult to converge.

Differently, the proposed parallel hybrid activated convolution (PhaC) module consists of solely a dilated causal convolution operation (so, the model's convolution depth is greatly reduced) and utilizes two different parallel-arranged non-linear activation operations (ReLU and Tanh) to maintain a sufficient non-linearity. Experimental results show that PhaC not only greatly enhances the model's learning efficiency, but also increases the forecasting accuracy; this may be because two different activation functions in PhaC may bring better diversity than two identical activation functions.

Specifically, as shown in Fig. 1, given an input sequence S_i generated from the previous layer i, the PhaC module first processes S_i using a dilated casual convolution (with dilated rate d and kernel size k) and a normalization to obtain an *intermediate sequence* S_i^1. Formally,

$$S_i^1 = WeightNorm(Dilated\_Conv_k^d(S_i)). \tag{4}$$

Then, S_i^1 is sent to the parallel-arranged non-linear activation operations, whose results are multiplied to get an *activated sequence* S_i^2:

$$S_i^2 = ReLU(S_i^1) \times Tanh(S_i^1). \tag{5}$$

A skip connection is also incorporated into the PhaC module, where the input sequence S_i is processed by a 1×1 convolution block to obtain a *residual sequence* S_i^3. Formally,

$$S_i^3 = Conv\_1 \times 1(S_i). \tag{6}$$

Finally, after a dropout operation, the activated sequence S_i^2 is added with the residual sequence S_i^3 using an element-wise summation to obtain the output sequence of PhaC S_{i+1}:

$$S_{i+1} = Sum(Dropout(S_i^2), S_i^3) \tag{7}$$

IV. Experiments

A. Description of Dataset

To evaluate the performance of our proposed PhaCIA-TCN model for STLF, we have conducted extensive experiments to compare the STLF performances of PhaCIA-TCN with those of the state-of-the-art RNN-based and TCN-based deep models on three public datasets, namely, PJM[1], GEFcom (GEF) [10], and Entsoe (ENT)[2]. Specifically, PJM provides load data within the Northeast and Midwest regions of United States; GEF is released in the Global Energy Forecasting Competition 2014 (GEFCom2014) and includes global load data; the load data of ENT are collected from 35 European countries. Each dataset contains hourly-sampled power load sequential data for an entire calendar year (2019 for PJM, 2004 for GEF, and 2015 for ENT), where the data points are divided into three subsets, i.e., training set (containing data points generated from January to September), validation set (containing data points generated in October), and testing set (containing data points generated in November and December). Specifically, the number of data points in the training sets of PJM, GEF and ENT datasets are 6575, 6587, and 6552, respectively; those in the validation sets of PJM, GEF and ENT datasets are 720, 732, and 744, respectively; and those in the testing sets are 1437 for all three datasets.

[1]link: www.pjm.com
[2]link: www.entsoe.eu.

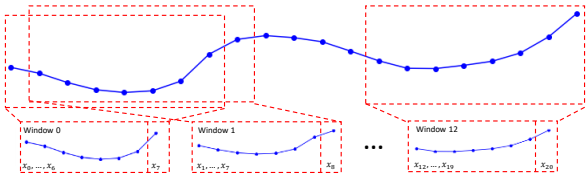


Fig. 2: Generating subsequences using sliding windows.

B. Preprocessing

Normalization. Data normalization is applied to ensure the values of the input data for each dataset are within the scale of $[0, 1]$. In this work, we use max-minimum standardization for data normalization, i.e., assuming the value of a given sample in a dataset is x, the new value of this sample after data normalization $\hat{x}$ can be obtained by the following equation.

$$\hat{x} = \frac{x - \min D}{\max D - \min D}, \tag{8}$$

where $\min D$ and $\max D$ are the minimum and maximum load values, respectively, in the given dataset.

Preprocessing for Daily Total/Peak Load Forecasting. In STLF, we are often concerned about the following three aspects of load forecasting tasks: (i) hourly load in the next few hours, (ii) daily total load in the next few days, and (iii) daily peak load in the next few days. So, we will evaluate the forecasting performances of PhaCIA-TCN and all state-of-the-art baselines on these three STLF tasks. However, since each data point in the datasets represents the hourly load value, to forecast the daily total load, we need to do further data pre-processing to sum up the 24-hour load values belonging to the same day to obtain the total load value of the given day; similarly, for daily peak load forecasting, the maximum one among the 24-hour load values is selected as the peak load value of the given day. Consequently, the total number of data points for daily total/peak load forecasting is 365 for each dataset; and the numbers of data points in the training, validation and testing sets for daily total/peak load forecasting are respectively 273, 31 and 61 for each dataset.

Subsequence Generation. Finally, similarly to [18], [22], we apply sliding windows to divide the sequential load data into subsequences. Specifically, assuming that we want to use the load values of the past m hours (or days) as the inputs to predict the load values of the following n hours (or days), given the length of the original sequence as L, the sliding window method (with $m + n$ as window size and 1 as step size) is applied to obtain $L - m - n + 1$ subsequences with

TABLE I: Accuracy results in three different load forecasting tasks of PhaCIA-TCNs and the state-of-the-art baselines.

	Module	PJM				GEF				ENT			
		RMSE (WM)		MAPE		RMSE (WM)		MAPE		RMSE (WM)		MAPE	
		w/o IA	w/ IA	w/o IA	w/ IA	w/o IA	w/ IA	w/o IA	w/ IA	w/o IA	w/ IA	w/o IA	w/ IA
daily total load (7-1)	RNN	21172.2	20691.1	4.80	4.67	78540.3	78308.9	11.84	11.84	18271.7	16862.9	6.70	6.28
	LSTM	21103.1	20379.4	4.83	4.64	76605.6	65758.7	11.86	9.28	19145.3	17633.3	7.59	6.56
	GRU	21001.3	20893.2	4.77	4.63	72666.8	71731.0	10.20	10.10	19103.5	18282.8	7.63	6.77
	Bi-RNN	20882.1	20622.3	4.72	4.66	72517.8	71952.8	10.65	10.55	18216.3	16970.9	6.66	6.11
	Bi-LSTM	21266.8	20437.8	4.78	4.66	69373.9	69356.4	9.72	9.72	19089.3	16272.9	7.66	5.85
	Bi-GRU	21376.9	20794.7	4.89	4.75	65761.2	65247.9	9.24	9.24	19012.3	15767.7	7.58	5.58
	TCN	20773.5	20291.4	4.69	4.54	65567.6	64540.5	9.08	9.01	15560.7	14704.6	5.59	5.25
	TCN-GRU [18]	20168.2	20074.6	4.58	4.46	64475.2	64087.9	9.23	9.01	14444.3	14403.1	5.44	5.23
	TCN-LightGBM [22]	22863.9	21074.3	5.11	4.87	65276.9	64372.6	10.40	9.06	14620.9	14584.1	5.88	5.45
	MTCN [26]	20559.7	19438.4	4.62	4.50	65154.0	64306.3	11.62	9.44	14628.1	14507.0	5.37	5.13
	PhaCIA-TCN	**18680.5**	–	**4.11**	–	**63985.0**	–	**8.90**	–	**14321.6**	–	**5.07**	–
daily peak load (7-1)	RNN	1124.9	1099.8	5.70	5.56	4581.5	4566.4	14.04	13.98	860.6	813.5	6.43	6.14
	LSTM	1076.4	1038.3	5.58	5.38	4627.6	4566.4	14.15	13.98	941.9	848.1	7.88	6.64
	GRU	1057.0	1049.3	5.58	5.54	4556.7	4542.5	13.88	13.84	939.9	897.3	7.81	7.06
	Bi-RNN	1145.7	1134.4	5.61	5.57	3790.8	3725.9	11.65	11.49	859.3	817.5	6.40	6.09
	Bi-LSTM	1140.7	1072.1	5.96	5.54	4271.3	4267.6	12.81	12.79	937.2	766.3	7.80	5.74
	Bi-GRU	1074.3	1067.2	5.60	5.58	4298.8	4290.4	12.89	12.87	927.3	782.3	7.66	5.86
	TCN	1048.8	1030.5	5.45	5.32	3755.0	3705.1	11.29	11.27	778.2	692.2	5.63	5.13
	TCN-GRU [18]	1020.7	1013.2	5.35	5.24	3719.6	3701.4	11.82	11.43	697.3	684.7	5.13	5.04
	TCN-LightGBM [22]	1039.6	1024.3	5.43	5.31	3715.9	3698.1	12.06	11.76	741.6	690.1	6.12	5.43
	MTCN [26]	1033.1	1017.9	5.42	5.27	3695.2	3687.0	11.62	11.25	704.5	688.4	5.27	5.09
	PhaCIA-TCN	**1003.2**	–	**5.17**	–	**3683.0**	–	**11.15**	–	**679.3**	–	**5.01**	–
hourly load (24-4)	RNN	506.4	489.1	2.49	2.41	2625.0	2606.1	9.41	9.36	391.5	371.0	3.42	3.17
	LSTM	840.7	828.1	4.35	4.26	1506.9	1493.4	5.61	5.58	350.9	337.3	3.01	2.91
	GRU	532.4	524.9	2.69	2.62	1507.8	1425.2	5.87	5.29	313.7	309.1	2.68	2.66
	Bi-RNN	531.2	484.0	2.69	2.38	2367.5	2223.8	8.66	8.25	350.8	324.1	3.08	2.90
	Bi-LSTM	647.1	546.9	3.31	2.82	1706.8	1699.9	6.57	6.56	394.5	369.8	3.36	3.15
	Bi-GRU	583.5	550.8	3.03	2.87	1480.8	1464.5	5.44	5.38	377.0	364.1	3.24	3.15
	TCN	474.9	468.5	2.40	2.35	1420.5	1394.5	5.38	5.22	291.0	279.4	2.45	2.33
	TCN-GRU [18]	428.6	427.9	2.26	2.20	1404.7	1347.7	5.06	4.98	288.8	277.3	2.54	2.44
	TCN-LightGBM [22]	468.0	447.6	2.35	2.27	1732.1	1407.2	6.72	5.44	283.0	274.6	2.46	2.37
	MTCN [26]	446.7	431.3	2.37	2.29	1345.2	1324.0	5.32	5.07	284.0	275.1	2.40	2.34
	PhaCIA-TCN	**423.7**	–	**2.16**	–	**1312.5**	–	**4.96**	–	**267.1**	–	**2.31**	–

length $m+n$, where the first m data points are used as inputs, and the last n data points are ground truth labels of forecasting. Fig. 2 shows an example, where $L = 20$, $m = 7$, and $n = 1$.

C. Baselines and Implementation Details

In the experiments, six RNN-based state-of-the-art deep models, namely, RNNs, LSTMs, GRUs, Bi-RNNs, Bi-LSTMs, and Bi-GRUs are first used as the baselines [27]. Furthermore, since the goal of this work is to propose a more effective and efficient TCN-based backbone model to replace vanilla TCNs in STLF, the vanilla TCNs and three state-of-the-art TCN-based SLTF models, TCN-GRU [18], TCN-LightGBM [22], and MTCN [26] are further used as the baselines in this work.

PhaCIA-TCNs and all baselines are implemented in PyTorch and run on an NVIDIA GeForce GTX 2080 GPU. The number of neurons in each hidden layer is 384 for all models, the kernel size of vanilla TCNs and PhaCIA-TCNs is 5, and the dilated rate is 2^i in vanilla TCNs and PhaCIA-TCNs. TCN-GRU [18], TCN-LightGBM [22], and MTCN [26] are all implemented following the settings in their corresponding original works. All other models are trained using the Adam optimizer, where the hyperparameters of Adam are $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. The initial learning rate is 0.00001, and learning rate decay is adopted (multiplied by 0.95 every 50 epochs) to decrease the learning rate automatically as the number of iterations increases. The batch size is 32 in daily total/peak load forecasting and 64 in hourly load forecasting.

D. Evaluation Metrics

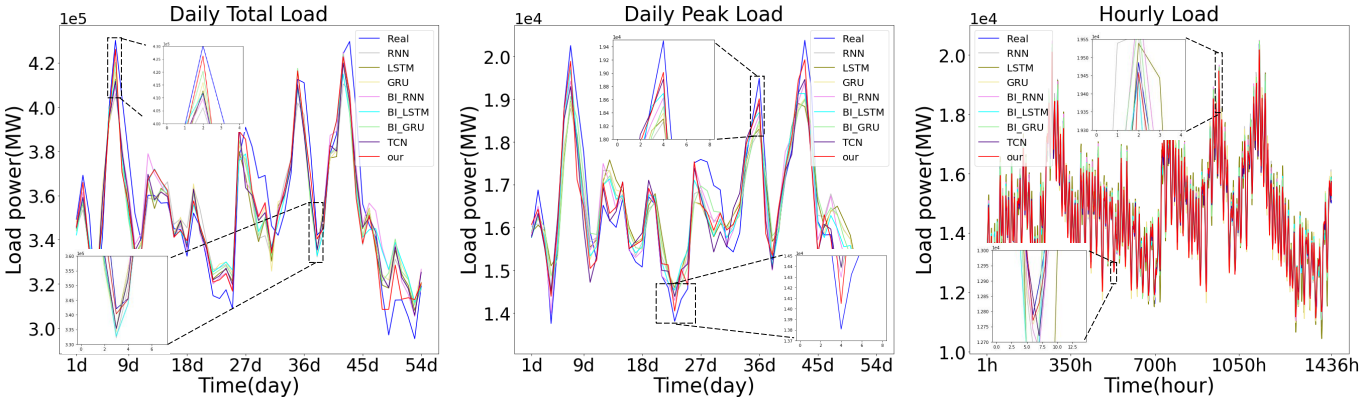
The load forecasting accuracy is the most important metric to evaluate the quality of STLF models. In this paper, two metrics, root mean square error (RMSE) and mean absolute percentage error (MAPE), are used to evaluate the load forecasting accuracy of PhaCIA-TCNs and all baselines. The lower the values of RMSE and MAPE, the better this accuracy:

$$\begin{aligned} RMSE &= \sqrt{\frac{1}{N}\sum_{i=1}^{n}(y_i - \hat{y_i})^2}, \\ MAEP &= \frac{100\%}{N}\sum_{i=1}^{n}|\frac{\hat{y_i} - y_i}{y_i}|, \end{aligned} \tag{9}$$

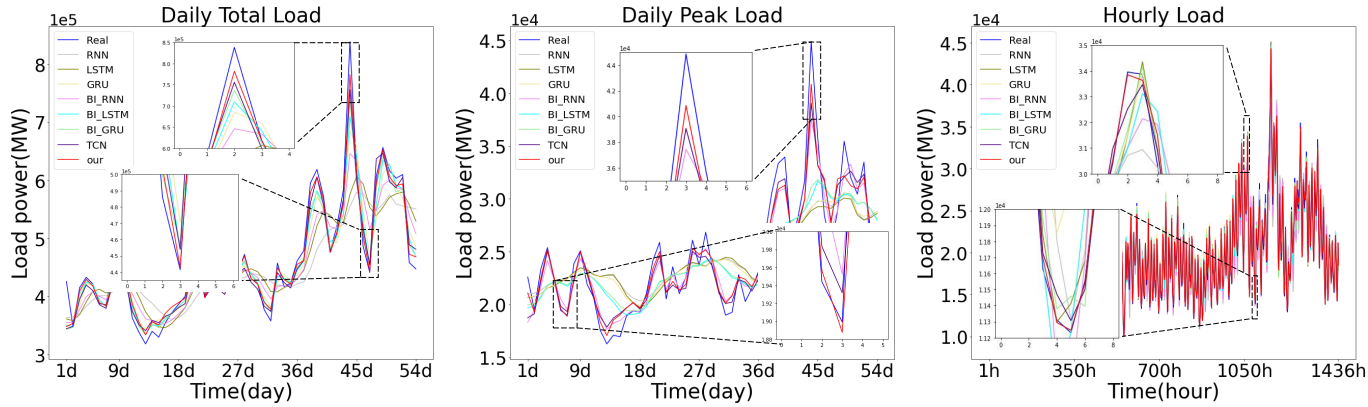
where N is the total number of predicted load values, y_i is the true load value, and $\hat{y_i}$ is the predicted load value.

E. Main Results

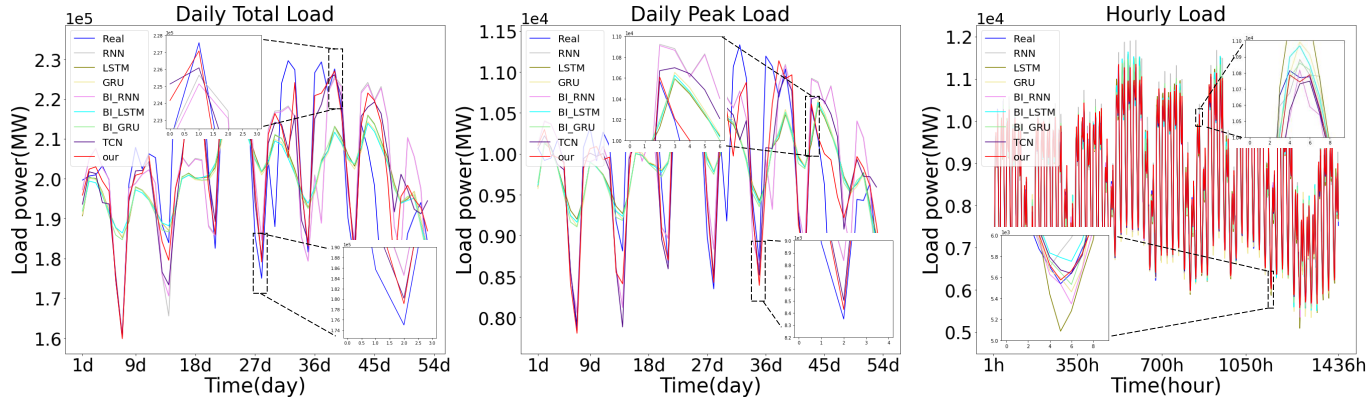
The short-term load forecasting results of PhaCIA-TCNs and ten state-of-the-art baselines in terms of RMSE and MAPE on three datasets are shown in Table I. As for daily total/peak load forecasting, we evaluate the task of using the total/peak load information of the past seven days to predict that of the



(a) Results on the PJM dataset



(b) Results on the GEF dataset

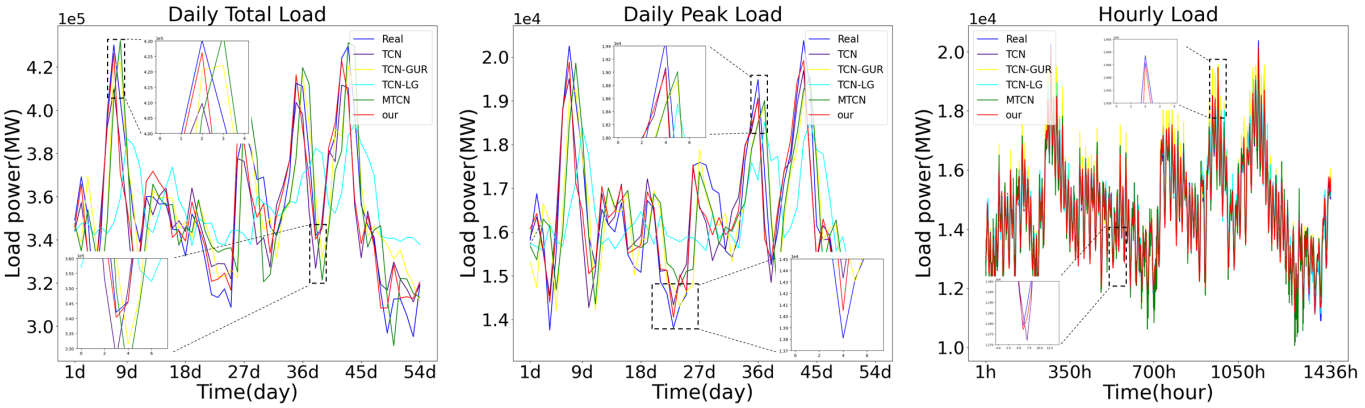


(c) Results on the ENT dataset

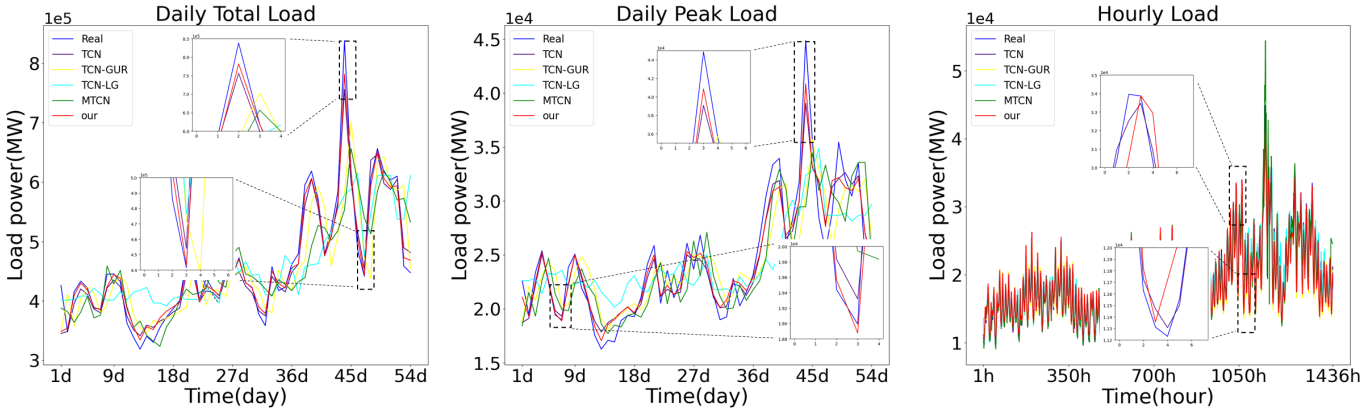
Fig. 3: Visualized forecasting results of PhaCIA-TCN, TCN and RNN-based baselines on three datasets.

next day (denoted 7-1); as for hourly load forecasting, the hourly load information of the past 24 hours is used to predict those of the following 4 hours (denoted 24-4).

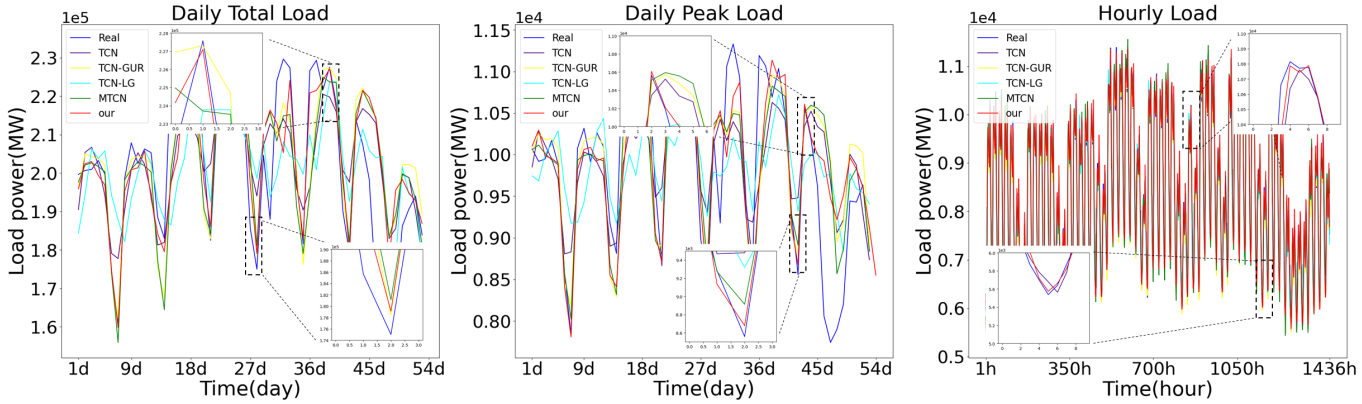
In Table I, we find that PhaCIA-TCNs always outperform all the state-of-the-art STLF baselines in both evaluation metrics for all forecasting tasks, which proves that the proposed PhaCIA-TCNs are a better STLF backbone model than all RNN-based and TCNs-based deep models. Furthermore, the following additional observations are also found in Table I: (i) We note that the TCNs-based results are generally better than those of all six RNN-based models in all STLF tasks; this demonstrates that there exists the unbalanced non-linearity problem in RNN-based models, and TCNs can overcome this problem by using stacked casual convolution operations to process all elements by the same number of non-linearities and thus achieve a better STLF performance. (ii) PhaCIA-TCNs constantly outperform not only the classic vanilla TCN but also all three state-of-the-art TCN-based STLF baselines, TCN-GUR [18], TCN-LightGBM [22] and MTCN [26], under all settings; this is because the existing TCN-based works have the redundant convolutional operation and equal input importance problems, which are resolved by parallel hybrid



(a) Results on the PJM dataset



(b) Results on the GEF dataset



(c) Results on the ENT dataset

Fig. 4: Visualized forecasting results of PhaCIA-TCN and TCN-based baselines on three datasets, where TCN-LG stands for TCN-LightGBM.

activated convolution (PhaC) and input attention (IA) modules in PhaCIA-TCNs, respectively. Finally, we integrate IA modules to all seven SOTA baselines, and the results in Table I show that the STLF accuracies of all baselines are significantly enhanced after adding IA modules, which thus proves that IA is with great effectiveness, applicability, and portability in STLF tasks.

To illustrate the superior STLF accuracies of PhaCIA-TCNs, Fig. 3 exhibits the visualized forecasting results of PhaCIA-TCNs, vanilla TCN and all RCNN-based baselines on the test sets of all three datasets, while Fig. 4 visualizes the forecasting results of PhaCIA-TCNs, vanilla TCN and all TCN-based baselines on the test sets of all three datasets. It can be seen from the result curves that the forecasting errors mainly come from the peak and valley positions of the results curve, i.e., the prediction of peak and valley values is generally more difficult. By enlarging some peak and valley positions, we further witness that the predicted values of PhaCIA-TCN are generally much closer to the true values than those of the state-of-the-art baselines at both peak and valley positions. This

TABLE II: Forecasting accuracies in ablation studies, where the improvement rate (denoted imp) is based on RMSE.

	Task (in-out)	TCN		TCN-IA			TCN-PhaC			PhaCIA-TCN		
		RMSE	MAPE	RMSE	MAPE	Imp(%)	RMSE	MAPE	Imp(%)	RMSE	MAPE	Imp(%)
PJM	daily total load (7-1)	20773.5	4.69	20291.4	4.54	2.32	20275.2	4.55	2.39	**18680.5**	**4.11**	**10.07**
	daily peak load (7-1)	1048.8	5.45	1030.5	5.32	1.74	1023.9	5.36	2.37	**1003.2**	**5.17**	**4.34**
	hourly load (24-4)	474.9	2.40	468.4	2.35	1.37	430.8	2.20	9.29	**423.7**	**2.16**	**10.78**
GEF	daily total load (7-1)	65567.6	9.08	64540.5	9.01	1.56	64869.4	9.02	1.06	**63985.0**	**8.90**	**2.41**
	daily peak load (7-1)	3755.0	11.29	3705.1	11.27	1.33	3694.6	11.21	1.61	**3683.0**	**11.15**	**1.92**
	hourly load (24-4)	1420.5	5.38	1394.5	5.22	2.98	1364.3	5.09	3.42	**1312.5**	**4.96**	**7.60**
ENT	daily total load (7-1)	15560.7	5.59	14704.6	5.25	5.50	14733.1	5.18	5.32	**14321.6**	**5.07**	**7.96**
	daily peak load (7-1)	778.2	5.63	692.2	5.13	11.04	722.2	5.61	7.19	**679.3**	**5.01**	**12.71**
	hourly load (24-4)	291.0	2.45	279.4	2.33	3.99	272.6	2.35	6.32	**267.1**	**2.31**	**8.21**

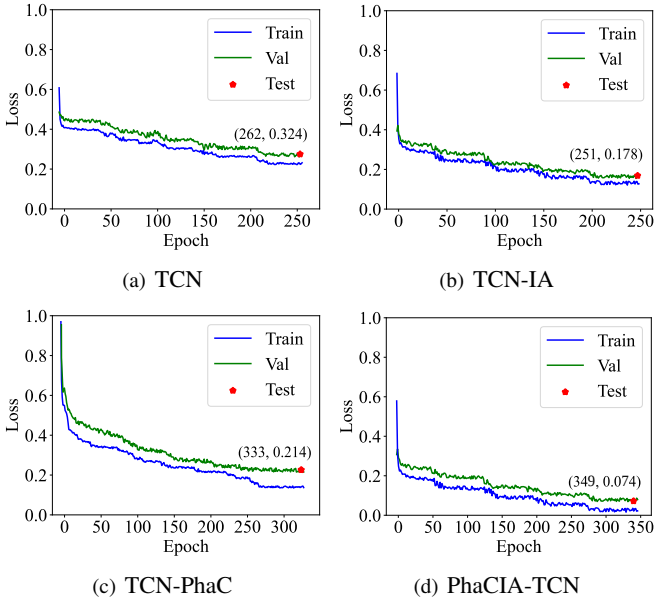
thus argues that the superior STLF accuracies of the proposed PhaCIA-TCN are very likely due to the following reason, by remedying the shortcomings of the SOTA baselines, PhaCIA-TCNs can achieve a better performance in forecasting the peak and valley load values.

F. Ablation Studies

To investigate the effects of the two proposed advanced modules, input attention (IA) and parallel hybrid activated convolution (PhaC), we conducted ablation studies, where the two modules are combined with vanilla TCNs, resulting in two intermediate models, TCNs with IA (TCN-IA) and TCNs with PhaC (TCN-PhaC). Table II shows the models' forecasting accuracies for all three forecasting tasks on the three datasets. Then, the training, validation and testing losses of these methods on the PJM dataset are plotted in Fig. 5 to show the advantages of integrating IA and PhaC modules with vanilla TCN (note that the resulting figures on GEF and ENT datasets are similar, so they are omitted to keep it simple). Furthermore, since the training efficiency is also important for STLF models, we also compare the training time-costs of vanilla TCNs, two intermediate models, and PhaCIA-TCNs in Table III (only the hourly load forecasting results are shown, those for daily load forecasting are similar).

Effectiveness of IA. We first compare the forecasting accuracies and training time-costs of vanilla TCNs with those of TCN-IA. The results show that with the help of IA, TCN-IA always outperforms vanilla TCNs on all STLF tasks in terms of both metrics; this thus demonstrates the effectiveness of IA in introducing importance weights to highlight key elements within the input sequences and make sufficient use of important information. However, the additional attention operations inevitably increase the average training time of each epoch (denoted Avg.T in Table III) and thus make TCN-IA cost more time in training (denoted Total T) than vanilla TCNs.

Effectiveness of PhaC. We then compare the performances of vanilla TCNs and TCN-PhaC in Tables II and III. The results show that using PhaC can not only increase the model's forecasting accuracy in STLF tasks but also decreases the model's convergence time-costs. This greatly supports our argument that using the PhaC module to replace the original residual block in vanilla TCNs can resolve the problem of redundant convolutional operations (thus enhancing the efficiency), while maintaining a good sequential learning capability using two different parallel-arranged activation functions (thus enhances the accuracy).



(a) TCN (b) TCN-IA (c) TCN-PhaC (d) PhaCIA-TCN

Fig. 5: Training, validation and testing losses on PJM dataset.

Effectiveness of using both IA and PhaC. Finally, we find that by using both the IA and the PhaC module, PhaCIA-TCNs outperform TCN-IA and TCN-PhaC in terms of both RMSE and MAPE in Table II. This is because the IA and PhaC modules are designed to resolve different problems in the existing TCNs-based works, and can complement each other to improve the deep model's STLF accuracies. In addition, by comparing the time-costs of all models in Table III, we observe that the training time-costs of PhaCIA-TCNs are between those of TCN-IA and TCN-PhaC, and are similar to (sometimes even lower than) those of the existing TCNs models. So all the above observations demonstrate that PhaCIA-TCNs are an effective and efficient TCN-based backbone model for STLF, and both IA and PhaC modules are effective and essential for PhaCIA-TCNs to achieve superior performances.

Visualizing training, validation and testing losses. To visualize the advantages of the proposed IA and PhaC modules in STLF tasks, we further plot the training, validation and testing losses of TCN, TCN-IA, TCN-PhaC, and PhaCIA-TCN in Fig. 5 (note that the testing set is only used once on the well-trained model, so it only results in one loss point). By

TABLE III: Training time-costs in ablation studies.

Dataset	Task (24-4)	TCN	TCN -IA	TCN -PhaC	PhaCIA -TCN
PJM	Avg.T(s)	1.81	2.15	1.26	1.40
	# of Epoch	262	251	333	349
	Total T(s)	476.59	538.79	421.48	487.41
GEF	Avg.T(s))	1.95	2.19	1.34	1.53
	# of Epoch	118	152	109	111
	Total T(s)	230.28	332.99	146.38	169.82
ENT	Avg.T(s)	1.74	2.05	1.24	1.39
	# of Epoch	379	354	445	466
	Total T(s)	659.37	725.64	553.89	646.87

comparing the results of TCN-IA and TCN, we find that the proposed IA module can greatly reduce all losses (e.g., the testing loss is reduced from 0.324 to 0.178), which thus helps the model learn more accurate features. Then, by comparing TCN-PhaC (Fig. 5(c)) with TCN (Fig. 5(a)), we observe that the proposed PhaC module can also enhance the model's learning capability to make it converge to lower losses. Finally, by using both IA and PhaC modules, the proposed PhaCIA-TCN can reach very low training, validation, and testing losses (e.g., the testing loss of PhaCIA-TCN is only 0.074). Consequently, these observations demonstrate that both IA and PhaC modules effectively reduce the training, validation and testing losses, making PhaCIA-TCN work much better than the vanilla TCN in STLF tasks.

Training efficiency. We further investigate the training efficiency of PhaCIA-TCN and the intermediate models in ablation studies as follows. As shown in Table III, we can first observe that, due to the additional attention operations, the average time-cost of each epoch (denoted Avg.T) in TCN-IA is higher than that of vanilla TCN, which thus increases the model's total training time-cost (denoted Total T), as the number of training epochs of TCN-IA is similar to that of TCN. Then, we also find that, with the help of PhaC modules, the Avg.T of TCN-PhaC is greatly reduced to about two-thirds of TCN; therefore, although the number of training epochs of TCN-PhaC sometimes increases slightly (because, as shown in Fig. 5 and discussed in the above paragraph, the PhaC module needs slightly more epochs to achieve lower losses, learn more features, and further enhance the forecasting accuracies), the total training time-cost of TCN-PhaC is always lower than that of TCN. Finally, by integrating both IA and PhaC modules with TCN, the Avg.T of the proposed PhaCIA-TCN is slightly higher than that of TCN-PhaC but much lower than those of TCN and TCN-IA, and the number of epochs of PhaCIA-TCN is similar to that of TCN-PhaC and slightly higher than those of TCN and TCN-IA in some cases; consequently, the total time-cost of PhaCIA-TCN is slightly worse than that of TCN-PhaC, better than that of TCN-IA, and similar (sometimes even better) to that of TCN. In summary, the above observations sufficiently support our arguments that, by integrating IA and PhaC with TCN, the proposed PhaCIA-TCN not only greatly outperforms TCN in forecasting accuracies but also keeps similar (sometimes even better) learning efficiency.

TABLE IV: Forecasting accuracies of PhaCIA-TCNs for different lengths of input and output sequences on three datasets.

	in-out	PJM		GEF		ENT	
		RMSE	MAPE	RMSE	MAPE	RMSE	MAPE
total load	7-1	18680.5	4.11	63985.0	8.90	14321.6	5.07
	7-2	25239.9	5.44	75097.8	11.22	15124.6	5.32
	14-1	18419.3	4.05	69065.0	9.44	15525.3	5.76
	14-2	23534.2	5.17	78910.4	11.28	16791.9	6.22
peak load	7-1	1003.2	5.17	3683.0	11.15	679.3	5.01
	7-2	1276.9	6.55	4368.4	12.99	751.7	5.43
	14-1	967.6	5.09	3791.9	11.27	750.2	5.38
	14-2	1181.5	6.17	4618.3	13.12	783.3	5.64
hourly load	24-4	423.7	2.16	1312.5	4.96	267.1	2.31
	24-6	523.8	2.66	1672.1	6.30	301.2	2.56
	48-4	443.7	2.24	1430.6	5.58	278.8	2.42
	48-6	541.8	2.70	1770.2	6.47	306.7	2.59

G. Effect of Varying Lengths of Input and Output Sequences

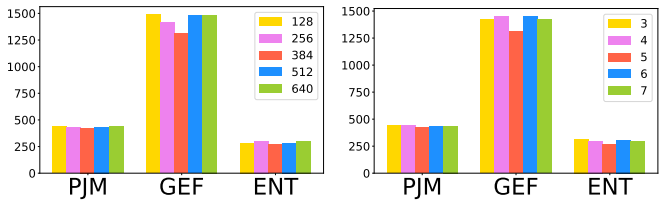
In real-world STLF applications, the forecasting demands are various, so additional experiments have been conducted to investigate the strategies for selecting the lengths of the input and output sequences in practical usage.

First, the results in Table IV show that with the increase of the lengths of output sequences, the forecasting accuracies of PhaCIA-TCNs always become worse. This is because more information needs to be predicted when the output sequence becomes longer, i.e., the prediction task becomes more difficult, so when the learning ability (i.e., other settings) of the model remains unchanged, the forecasting accuracy of the model will inevitably deteriorate. Therefore, on the premise of satisfying the practical needs, we should set the length of the predicted sequences as short as possible. This is also why we impose preprocessing and learn the model on daily-sample data for daily load forecasting, instead of directly predicting the load values of the next 24 hours based on hourly-sampled data.

Second, we note that the effect of changing the length of the input sequence is uncertain: the forecasting accuracies increase in some cases (e.g., daily total/peak load on PJM) but decrease in others (e.g., daily total/peak load on GEF and ENT) with the rise of the lengths of input sequences. This may be due to the differences in data characteristics of different STLF tasks: Input sequences of the same length may be too short to contain the whole latent sequential features in some tasks, but may be too long to introduce unnecessary noise in others; so, increasing the input length will improve forecasting accuracies in the former cases, but makes the forecasting worse in the latter. Therefore, pre-experiments are needed to properly tune the length of input sequences in PhaCIA-TCNs before usage.

H. Effect of Varying Hyperparameters

The number of neurons in each hidden layer (denoted N) and the size of kernels (denoted K) are two important hyperparameters for TCN-based models. So, experiments are conducted to investigate the effects of varying these two hyperparameters on the model's training quality in terms of RMSE on the validation sets for the hourly load forecasting



(a) varying # of neurons (N) (b) varying size of kernel (K)

Fig. 6: RMSE of PhaCIA-TCNs on validation sets of three datasets.

tasks on three datasets. To enhance the efficiency of computing using GPU, the value of N is searched from 128 to 640 with a step of 128, and the value of K is searched within $[3, 4, 5, 6, 7]$. The results in Fig. 6 show that the RMSE of PhaCIA-TCNs fluctuates with the increase of the values of K (resp., N) on all three validation sets, but the best forecasting accuracies are achieved coincidentally under the same setting, i.e., when $K = 384$ (resp., $N = 5$), on all three datasets. Therefore, we set the number of neurons to 384 and the kernel size to 5.

V. Conclusion and Future Work

In this paper, we first identified existing shortcomings of RNN-based (i.e., unbalanced non-linearity problem) and TCN-based (i.e., redundant convolutional operation and equal input importance problems) STLF models, and then proposed a novel TCN-based backbone model, PhaCIA-TCN, to remedy these problems and a achieve better short-term load forecasting performance. Extensive experiments were conducted, and their results proved that (i) the proposed PhaCIA-TCNs achieved much better STLF accuracies than the state-of-the-art baselines, (ii) the advanced modules, parallel hybrid activated convolution (PhaC) and input attention (IA), were both effective and essential for PhaCIA-TCNs to achieve the superior STLF performances, and (iii) even integrated with PhaC and IA modules, the training time-cost of PhaCIA-TCNs is similar (and sometimes even lower) to that of the existing TCNs-based solutions.

In the future, we intend to apply PhaCIA-TCNs to more STLF tasks in practice to verify their applicability and scalability. It may also be interesting to extend PhaCIA-TCNs to consider additional information (e.g., weather [8]) and multi-modal learning [28] for more accurate STLF. In addition, a 'smaller peak value" phenomenon (i.e., the prediction results of all methods around the peaks are usually smaller than the real load values) is noticed in Figs 3 and 4; this may because the RNN and CNN (including TCN) based sequential modeling methods aim to learn the universal characteristics of the sequential data, so it may treat the peak and valley values as outliers and give "conservative" predictions; considering the importance of peak and valley values in STLF tasks, it is an interesting future work to further discover a solution to remedy this issue.

References

[1] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[2] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[3] Xi Fang, Satyajayant Misra, Guoliang Xue, and Dejun Yang. Smart grid—The new and improved power grid: A survey. *IEEE Communications Surveys & Tutorials*, 14(4):944–980, 2011.

[4] Xin Gao, Xiaobing Li, Bing Zhao, Weijia Ji, Xiao Jing, and Yang He. Short-term electricity load forecasting model based on emd-gru with feature selection. *Energies*, 12(6):1140, 2019.

[5] Jonas Gehring, Michael Auli, David Grangier, and Yann N Dauphin. A convolutional encoder model for neural machine translation. *arXiv preprint arXiv:1611.02344*, 2016.

[6] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the International Conference on Machine Learning*, pages 1243–1252, 2017.

[7] George Gross and Francisco D Galiana. Short-term load forecasting. *Proceedings of the IEEE*, 75(12):1558–1573, 1987.

[8] Pradeep Hewage, Ardhendu Behera, Marcello Trovati, Ella Pereira, Morteza Ghahremani, Francesco Palmieri, and Yonghuai Liu. Temporal convolutional neural (tcn) network for an effective weather forecasting using time-series data from the local weather station. *Soft Computing*, 24(21):16453–16482, 2020.

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[10] Tao Hong and Shu Fan. Probabilistic electric load forecasting: A tutorial review. *International Journal of Forecasting*, 32(3):914–938, 2016.

[11] Daniel Jarrett, Jinsung Yoon, and Mihaela van der Schaar. Dynamic prediction in clinical survival analysis using temporal convolutional networks. *IEEE Journal of Biomedical and Health Informatics*, 24(2):424–436, 2019.

[12] Weicong Kong, Zhao Yang Dong, Youwei Jia, David J Hill, Yan Xu, and Yuan Zhang. Short-term residential load forecasting based on LSTM recurrent neural network. *IEEE Transactions on Smart Grid*, 10(1):841–851, 2017.

[13] Jian Li, Daiyu Deng, Junbo Zhao, Dongsheng Cai, Weihao Hu, Man Zhang, and Qi Huang. A novel hybrid short-term load forecasting method of smart grid using mlr and lstm neural network. *IEEE Transactions on Industrial Informatics*, 17(4):2443–2452, 2020.

[14] Lei Lin, Beilei Xu, Wencheng Wu, Trevor W Richardson, and Edgar A Bernal. Medical time series classification with hierarchical attention-based temporal convolutional networks: A case study of myotonic dystrophy diagnosis. In *Proceedings of the IEEE / CVF Computer Vision and Pattern Recognition Conference*, pages 83–86, 2019.

[15] Yuecan Liu, Kun Zhang, Shuai Zhen, Yongming Guan, and Yuliang Shi. WRL: A combined model for short-term load forecasting. In *Proceedings of the Asia Pacific Web and Web-Age Information Management Joint International Conference on Web and Big Data*, pages 35–42, 2019.

[16] Jianzhong Qi, Zhuowei Zhao, Egemen Tanin, Tingru Cui, Neema Nassir, and Majid Sarvi. A graph and attentive multi-path convolutional network for traffic prediction. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[17] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[18] Hanhong Shi, Lei Wang, Rafał Scherer, Marcin Woźniak, Pengchao Zhang, and Wei Wei. Short-term load forecasting based on adabelief optimized temporal convolutional network and gated recurrent unit hybrid neural network. *IEEE Access*, 9:66965–66981, 2021.

[19] Heng Shi, Minghao Xu, and Ran Li. Deep learning for household load forecasting-A novel pooling deep RNN. *IEEE Transactions on Smart Grid*, 9(5):5271–5280, 2017.

[20] Xianlun Tang, Yuyan Dai, Ting Wang, and Yingjie Chen. Short-term power load forecasting based on multi-layer bidirectional recurrent neural network. *IET Generation, Transmission & Distribution*, 13(17):3847–3854, 2019.

[21] Shouxiang Wang, Xuan Wang, Shaomin Wang, and Dan Wang. Bi-directional long short-term memory method based on attention mechanism and rolling update for short-term load forecasting. *International Journal of Electrical Power & Energy Systems*, 109:470–479, 2019.

[22] Yuanyuan Wang, Jun Chen, Xiaoqiao Chen, Xiangjun Zeng, Yang Kong, Shanfeng Sun, Yongsheng Guo, and Ying Liu. Short-term load forecasting for industrial customers based on tcn-lightgbm. *IEEE Transactions on Power Systems*, 36(3):1984–1997, 2020.

[23] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121*, 2019.

[24] Zhenghua Xu, Shijie Liu, Di Yuan, Lei Wang, Junyang Chen, Thomas Lukasiewicz, Zhigang Fu, and Rui Zhang. ω-net: Dual supervised medical image segmentation with multi-dimensional self-attention and diversely-connected multi-scale convolution. *Neurocomputing*, 500:177–190, 2022.
[25] Zhenghua Xu, Xudong Zhang, Hexiang Zhang, Yunxin Liu, Yuefu Zhan, and Thomas Lukasiewicz. Efpn: Effective medical image detection using feature pyramid fusion enhancement. *Computers in Biology and Medicine*, page 107149, 2023.
[26] Linfei Yin and Jiaxing Xie. Multi-temporal-spatial-scale temporal convolution network for short-term load forecasting of power systems. *Applied Energy*, 283:116328, 2021.
[27] Mingfei Zhang, Zhoutao Yu, and Zhenghua Xu. Short-term load forecasting using recurrent neural networks with input attention mechanism and hidden connection mechanism. *IEEE Access*, 8:186514–186529, 2020.
[28] Shuo Zhang, Jiaojiao Zhang, Biao Tian, Thomas Lukasiewicz, and Zhenghua Xu. Multi-modal contrastive mutual learning and pseudo-label re-learning for semi-supervised medical image segmentation. *Medical Image Analysis*, 83:102656, 2023.

Zhenghua Xu received a M.Phil. in Computer Science from The University of Melbourne, Australia, in 2012, and a D.Phil in Computer Science from University of Oxford, United Kingdom, in 2018. From 2017 to 2018, he worked as a research associate at the Department of Computer Science, University of Oxford. He is now a professor at the Hebei University of Technology, China, and a awardee of "100 Talents Plan" of Hebei Province. He has published dozens of papers in top AI or database conferences and journals, e.g., NeurIPS, AAAI, IJCAI, ICDE, Medical Image Analysis, etc. His current research focuses on deep learning, medical artificial intelligence, and computer vision.

Zhoutao Yu received the B.E. degree from Anhui University of Technology, in 2019. He is currently pursuing the M.S. degree in Electrical Engineering from Hebei University of Technology, Tianjin, China. His research interests are deep learning, electricity market, and smart grid.

Hexiang Zhang is currently a master's student at Hebei University of Technology, China. He received B.Eng. degree in Automation from Yanshan University, China, in 2022. His research interests lie in medical image processing using deep learning methods.

Junyang Chen received the Ph.D. degree in computer and information science from the University of Macau, Macau, China, in 2020. He is currently an Assistant Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His research interests include graph neural networks, text mining, deep learning, and recommender systems.

Junhua Gu received the B.S degree in mathematics from Shanghai Jiaotong University, Shanghai, China, in 1988, the M.S. degree in computer science and the Ph.D. degree in electrical engineering from the Hebei University of Technology, Tianjin, China, in 1993 and 1997, respectively. He is currently a Professor at the Hebei University of Technology, China. He has authored more than 70 papers. His current research interests include big data, intelligent control, and intelligent transportation systems. Prof. Gu was awarded the Hebei New Century "333 Talent Project" Second Level Suitable Person.

Thomas Lukasiewicz is a Professor of Computer Science at the Department of Computer Science, University of Oxford, UK, heading the Intelligent Systems Lab within the Artificial Intelligence and Machine Learning Theme. He currently holds an AXA Chair grant on "Explainable Artificial Intelligence in Healthcare" and a Turing Fellowship at the Alan Turing Institute, London, UK, which is the UK's National Institute for Data Science and Artificial Intelligence. He received the IJCAI-01 Distinguished Paper Award, the AIJ Prominent Paper Award 2013, the RuleML 2015 Best Paper Award, and the ACM PODS Alberto O. Mendelzon Test-of-Time Award 2019. He is a Fellow of the European Association for Artificial Intelligence (EurAI) since 2020. His research interests are especially in artificial intelligence and machine learning.

Victor C. M. Leung (Life Fellow, IEEE) is currently a Distinguished Professor of computer science and software engineering with Shenzhen University, Shenzhen, China. He is also an Emeritus Professor of electrical and computer engineering and the Director of the Laboratory for Wireless Networks and Mobile Systems, University of British Columbia (UBC), Vancouver, BC, Canada. He is a Fellow of Royal Society of Canada, Canadian Academy of Engineering, and Engineering Institute of Canada. He is named in the current Clarivate Analytics list of "Highly Cited Researcher". His research interests include wireless networks and mobile systems.