IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 10, 2021, accepted April 25, 2021, date of publication April 28, 2021, date of current version May 11, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3076313

Short-Term Load Forecasting Based on Adabelief Optimized Temporal Convolutional Network and Gated Recurrent Unit Hybrid Neural Network

HANHONG SHI[1], LEI WANG[1,2], RAFAŁ SCHERER[3], (Member, IEEE), MARCIN WOŹNIAK[4], PENGCHAO ZHANG[1], AND WEI WEI[2], (Senior Member, IEEE)

[1]Shaanxi Key Laboratory of Industrial Automation, Shaanxi University of Technology, Hanzhong 723001, China
[2]Shaanxi Key Laboratory for Network Computing and Security Technology, Xi'an University of Technology, Xi'an 710048, China
[3]Institute of Computational Intelligence, Czestochowa University of Technology, 42-200 Czestochowa, Poland
[4]Faculty of Applied Mathematics, Silesian University of Technology, 44-100 Gliwice, Poland

Corresponding author: Lei Wang (leiwang@xaut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61773314, in part by the Shaanxi Provincial Natural Science Basic Research Program under Grant 2019JZ-11, in part by the Scientific Research Project of the Education Department of the Shaanxi Provincial Government under Grant 19JC011, in part by the Key Research and Development Program of Shaanxi Province under Grant 2018ZDXM-GY-036, and in part by the Shaanxi Key Laboratory of Intelligent Processing for Big Energy Data under Grant IPBED7.

ABSTRACT To fully mine the relationship between temporal features in load data, improve the accuracy and efficiency of short-term load forecasting and overcome the difficulties caused by load nonlinearity and volatility in accurate load forecasting. In this paper, a hybrid neural network short-term load forecasting model based on temporal convolutional network (TCN) and gated recurrent unit (GRU) is proposed. Firstly, the correlation between meteorological features and load is measured with the distance correlation coefficient, and the fixed-length sliding time window method is used to reconstruct the features. Next, temporal convolutional network is adopted to extract the hidden historical information and time relationship including meteorological features, electricity price, etc., and a better-performing gated recurrent unit is utilized for perdition. Furthermore, the state-of-the-art AdaBelief optimizer and Attention mechanism are utilized to enhance the prediction accuracy and efficiency. The effectiveness and superiority of the proposed model are verified by load and weather data from Spain and PJM power system data. Short-term load forecasting results in different periods and comprehensive comparisons with the performance of different models show that the proposed model can provide accurate load forecasting results rather quickly. The highlights of this paper are that temporal convolutional network and gated recurrent unit are combined for load forecasting for the first time, and the forecasting performance is improved by the novel optimizer AdaBelief and feature selection based on distance correlation coefficient.

INDEX TERMS Short-term load forecasting, distance correlation coefficient, adabelief, temporal convolution network, gated recurrent unit, attention mechanism.

I. INTRODUCTION

Accurate and fast short-term load forecasting has become an essential task throughout the development of the market and smart grid. It can effectively ensure the safe operation of the power grid, reduce the cost of power generation and improve socioeconomic benefits [1]. However, the fluctuation of short-term power load sequence has obvious randomness and nonlinearity, and the influencing factors are diversified and complex (weather, electricity price, holidays, etc.), which bring huge challenges to accurate forecasting [2].

In response to these difficulties, there are numerous studies on short-term load forecasting currently. The invention of various forecasting technologies promotes the progress of short-term load forecasting. These methods are mainly divided into statistical methods, time series prediction methods, and machine learning methods. The statistical method is represented by Kalman filter [3], Shaima *et al.* [4] proposed

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

a blind Kalman filter method for short-term load forecasting. The experimental results showed that the method is superior to the state-of-the-art technology in load profile estimation and peak load forecasting through the improvement of the traditional Kalman filter. Time series prediction method mainly include linear regression method [5], exponential smoothing method [6] and autoregressive integrated moving average (ARIMA) method [7]. Arima is the most commonly used time series analysis method among these. Cao *et al.* [8] used the ARIMA model and similar day method to forecast the load within one day. By grouping meteorological similar days and target days, the load was predicted according to the average demand of the target day. It was proved that the ARIMA model performs better on ordinary days while a similar day method outperforms on special days.

However, all of the above methods have higher requirements for the stationarity of time series. Their data regression capabilities are all weak. Machine learning methods which are widely applied short-term load forecasting methods at present, have good nonlinear data fitting ability and parameter learning ability. Jiang *et al.* [9] proposed a hybrid prediction model based on support vector regression (SVR) and hybrid parameter optimization algorithms. After SVR is optimized by a two-step hybrid optimization algorithm, this model can deal with nonlinear problems better, and it wins upon other models in short-term load forecasting tasks. Shi *et al.* [10] introduced a continuous multi-day peak load forecasting method based on series-parallel ensemble learning. This method adopts XGBoost [11] serial ensemble algorithm and Bagging parallel ensemble algorithm to forecast peak load, and uses particle swarm optimization algorithm to tune parameters. It was demonstrated that the ensemble algorithm makes full use of the advantages of the two models, which maintain small bias and small variance respectively, thus improving the prediction accuracy.

In addition to the above-mentioned traditional methods and tree-based methods for machine learning methods, deep learning algorithms have also been continuously developed and have received extensive attention in recent years. In the existing literature, the work of Kong *et al.* [12] introduced a long short-term memory neural network (LSTM) into the short-term residential load forecasting with high volatility and uncertainty. Meanwhile, the method of aggregating individual forecasts was adopted. The proposed LSTM framework achieved the best prediction performance in the data set. Jiao *et al.* [13] proposed an LSTM-based method to predict the load of non-residential consumers by using multiple correlated sequence information. The daily load curve of non-residential consumers was analyzed by K-means, which was also superior to other load forecasting methods in experimental results. Nevertheless, considering that a large number of relevant features and historical information cannot be ignored in load forecasting. Single RNN algorithms such as LSTM or GRU [14] could consider the historical information of temporal data, but it needs to construct the feature relationship manually [15]. Convolution neural network also has the ability to process time information, and the feature extraction ability of CNN is higher than that of RNN [16]. Combined with CNN, the RNN model can get better prediction results in power load forecasting [17].

Facing the problem, feature extraction techniques are considered to be practicable methods. There are gradually some researches dedicated to combining feature extraction technologies with forecasting models to form a hybrid model for short-term load forecasting. Alhussein *et al.* [18] proposed a deep learning framework based on the combination of convolutional neural networks (CNN) and LSTM. The hybrid CNN-LSTM model utilizes the CNN layer to extract features from input data with LSTM layers for time series learning. It proved that the model is preferable to other deep learning models. Yao *et al.* [19] and Xu *et al.* [20] also used CNN to extract features, combined with GRU and LightGBM [21] to complete short-term load forecasting tasks, and both achieved good accuracy.

Based on the above analysis, temporal convolutional network (TCN) [22] is an improved one-dimensional CNN for time series problems. Experiments testify that TCN is superior to RNNs such as LSTM and GRU in certain application scenarios. Zhang *et al.* [23] introduced a quantile Huber function guided TCN for load probability forecasting to quantify the variability and uncertainty of consumer side load forecasting. And it achieved better performance than other models in residential and industrial users. Wang [24] proposed a hybrid algorithm combining TCN and LightGBM, then conducted experiments on three different types of industrial user load data sets. It was established that the method has the best feature extraction ability and the highest accuracy among all contrast models.

At present, there are fewer researches on TCN or hybrid algorithm of TCN and neural networks in the field of the power system. At the same time, a large number of studies have adopted grid search as parameters adjustment method and applied Adam optimizer to train the neural networks. In Table 1, we list some references to indicate the state-of-the-art development of load forecasting based on hybrid models. Therefore, based on the hybrid model of TCN with GRU, in this work, we firstly analyze the correlation of weather features, electricity price, and date features in load data. Then we use Tree-structured Parzen Estimator (TPE [25]), which is an optimized approach based on Bayesian optimization algorithm [26] to tune parameters. Finally, we use an improved optimizer AdaBelief and Attention mechanism to further improve the accuracy and efficiency of short-term load forecasting. Our proposed method uses the hourly power load series from 2015 to 2018 in Spain and PJM power system load data for load forecasting. The resolution of both datasets is 1 hour.

Lastly, about STLF models, there is a very important point that can not be ignored. Nowadays, more and more power grids perform an optimal operation considering day-ahead STLF [27]. In many studies about day-ahead STLF, direct multistep-ahead forecasting [28] is better than iterative

TABLE 1. Studies for load forecasting based on hybrid models.

Reference	Models	Features	Optimizer	Hyper parameters selection	Resolution	Number of datasets	Criteria	Pros & Cons
[9]	GTA-PSO-SVR	Historical load data	-	Grid Traverse Algorithm & Particle Swarm optimization	1min & 1hour	2	MAPE	Better performance of parameter selection, higher requirements for stationarity of data
[10]	Bagging-XGBoost	Temperature, Historical load, time index	-	Particle Swarm optimization	30min	1	MAPE, RMSE, MAX error	High accuracy, strong generalization, but without considering meteorological factors
[18]	CNN-LSTM	Historical load, time index	Adam	-	30min	1	MAPE	Outperforms other rival techniques in forecasting individual household energy consumption, but parameters selected by previous experience
[19]	CNN-GRU	Meteorological features, Date features, Similar day load	Adam	Grid Search	1day	1	MAX error, MAPE, time, FA-mean	High prediction accuracy and fast training speed, not considered diversified load types and load price
[20]	CNN-QRLightGBM	Temperature, Energy price, Historical load	Adam	Tree-structured Parzen Estimator	1hour	1	MAPE, RMSE	High prediction accuracy, fast parameters selection, but the dataset in evaluation was limited
[24]	TCN-LightGBM	Electrical features, Temperature, Date features	Adam	Grid Search	15min	3	MAE, MAPE, RMSE, time	Better features extraction ability and forecasting performs, relatively long time to select parameters
[50]	EEMD-LSTM-MLR	Historical load, Meteorological features, time index	Adam	-	1hour	3	MAPE, RMSE	High prediction accuracy in local details, even over different types of datasets
[48]	QR-TCN-Attention	Meteorological features, Date features, Historical load, Seasonal features	RMSProp	-	15min	1	MAPE, CP, MWP	High accuracy in load probability density prediction, but parameters selected by previous experience

one-step-ahead forecasting when the data resolution is 1 hour [29], [30]. In our article, all models will be designed for day-ahead load forecasting.

The main contributions of this paper are presented as follows:

1. A relatively novel data preprocessing method: we use the distance correlation coefficient to analyze the non-linear correlation between various meteorological features and short-term load, comprehensively consider the features (weather, electricity price, holidays, working hours, etc.). After feature engineering, the difficulty of prediction is effectively reduced and the prediction efficiency is improved.

2. A new hybrid algorithm: due to the lack of convolution, traditional single RNN model cannot well extract the hidden information [24], which limits the prediction accuracy of the model [15], [16]. Thus, a TCN-GRU algorithm with strong feature extraction capability is proposed to improve the performance of short-term load forecasting while avoiding gradient disappearance problem. TPE algorithm is also applied to select the optimal parameters of model.

3. Improved optimizer and Attention mechanism: we compare the learning performance of different optimizers on the training set and validation set. The state-of-the-art AdaBelief optimizer based on Adam is adopted to greatly improve the accuracy and efficiency of model operation. The Attention layer is also introduced into the neural network structure to avoid the problem of long-distance information weakening and improve the prediction accuracy of the model.

4. A comprehensive evaluation of the predictive performance of the proposed model: by comparing other single models and hybrid models, combined with multiple statistical parameter evaluation indicators (MAPE, RMSE, MAE, Training Time), in-depth analysis of the pros and cons of each model. And the effectiveness of our proposed model in short-term load forecasting is demonstrated by using two datasets from different regions. And considering the difference in load fluctuations between working days and holidays, it is verified that this method has strong adaptability.

The remainder of this paper is organized as follows. In Section II, we introduce the basic theory of algorithms adopted in this article. Section III presents the main steps of the proposed hybrid model in detail, including the principles of the model, the prediction framework and the structure of TCN-GRU, and some details of feature engineering. Section IV provides specific process and parameter settings of the experiment, describes the comparison with other models, also analyzes and discusses the reasons for the different evaluation indicators obtained by different models. The conclusion of this paper and the future work are given in Section V.

II. METHODOLOGY

A. CORRELATION ANALYSIS

Weather features are vital factors affecting short-term load [31]. Due to actual geographical differences, which weather features should be selected for short-term load forecasting should be analyzed in detail according to the actual situation, and the factors with high correlation should be selected to build the input data set. The distance correlation coefficient [32] (DCC) is improved based on the Pearson correlation coefficient (PCC), which can measure the non-linear correlation. According to the theory of statistics, the distribution function represents the unique property of the random vector itself, and the conditional distribution function represents the distribution property of a certain random vector under certain conditions. If we want to investigate the correlation between any two random vectors X and Y, we can compare the distribution $F(Y)$ of Y with the conditional distribution function $F(Y|X)$ of Y under the condition of X. The higher the similarity between the two, the less influence of X on Y, the weaker the correlation between X and Y.

For the convenience of calculation, the distribution function is replaced by the following characteristic functions:

$$f_{XY}(s,t) = E\exp[i\langle s,X\rangle + i\langle t,Y\rangle] \quad (1)$$

$$f_X(s) = f_{XY}(s,0) = E\exp[i\langle s,X\rangle] \quad (2)$$

$$f_Y(t) = f_{XY}(0,t) = E\exp[i\langle t,Y\rangle] \quad (3)$$

where E denotes Expectation, i represents imaginary unit, s and t represents Real vector, $\langle\rangle$ accounts for dot product operation. If and only if $f_{XY}(s,t) - f_X(s) f_Y(t) = 0$, X and Y are not correlated, otherwise they are correlated. The following distance covariance and variance are defined according to the Euclidean norm:

$$D^2_{cov}(X,Y) = \|f_{XY}(s,t) - f_X(s) f_Y(t)\|^2 \quad (4)$$

$$D^2_{cov}(X) = \|f_{XX}(s,t) - f_X(s) f_X(t)\|^2 \quad (5)$$

$$D^2_{cov}(Y) = \|f_{YY}(s,t) - f_Y(s) f_Y(t)\|^2 \quad (6)$$

The expression of the distance correlation coefficient is constructed by formula (4) - (6) as

$$D_{cor} = \begin{cases} \dfrac{D^2_{cov}(X,Y)}{\sqrt{D^2_{cov}(X) D^2_{cov}(Y)}}, & D^2_{cov}(X) D^2_{cov}(Y) > 0 \\ 0, & D^2_{cov}(X) D^2_{cov}(Y) = 0 \end{cases} \quad (7)$$

The various meteorological features and load data are brought into above formula to calculate the correlation distance coefficient and Pearson correlation coefficient. The distance correlation coefficient has only non-negative values. The larger the value, the stronger the correlation between this feature and short-term load, while the Pearson correlation coefficient is between -1 and 1, the closer the absolute value to 1, then the stronger the linear correlation between features and short-term load.

B. TPE ALGORITHM

The selection of hyperparameters in machine learning will directly affect the performance of the model. In the past, grid search and random search were used for hyperparameters optimization, but the execution time of this type of method will increase proportionally with the expansion of the hyperparameter scale, resulting in insufficient efficiency. Other tuning algorithms, such as particle swarm optimization [33], are relatively time-consuming and not conducive to practical application. TPE algorithm constructs a probability model of the objective function in order to intelligently evaluate each group of hyperparameters, reduce the searching time of hyperparameters, and find the best hyperparameters for the machine learning model more effectively.

TPE algorithm obtains the posterior probability distribution $p(y|x)$ by parameterizing the probability distribution $p(x|y)$ and the prior probability distribution $p(y)$. $p(x|y)$ is defined as follows [34]:

$$p(x|y) = \begin{cases} \ell(x), & y < y^* \\ g(x), & y \geq y^* \end{cases} \quad (8)$$

where $\ell(x)$ is the density formed by model observation, y denotes the value of loss function generated by the model, y^* is the set threshold, and $g(x)$ represents the density formed by residual observation.

The expected improvement (EI) in TPE is optimized [35]:

$$EI_{y^*}(x) = \int_{-\infty}^{y^*} (y^* - y) \frac{p(x|y)p(y)}{p(x)} dy \quad (9)$$

Let $\gamma = p(y < y^*)$, $\int_R p(x|y)p(y)dy = \gamma \ell(x) + (1-\gamma)g(x)$, we can get:

$$EI_{y^*}(x) \propto (\gamma + \frac{g(x)}{\ell(x)}(1-\gamma))^{-1} \quad (10)$$

Equation (10) represents that in order to maximize improvement, we hope the $\ell(x)$ probability to be high and the $g(x)$ probability to be low at point x. In each epoch, the algorithm is trying to improve the prediction model better by returning the candidate algorithm x^* with the largest EI.

C. AdaBelief OPTIMIZER

The so-called AdaBelief [36] refers to adjusting the training stride according to the Belief in the gradient direction. Compared with the Adam optimizer, the improved steps are decomposed as Step 1-Step 4:

Step 1: Intialize θ_0, and set $M_0 \leftarrow 0$, $s_0 \leftarrow 0$, $t \leftarrow 0$;

Step 2: While θ is not converged, set $t \leftarrow t+1$, $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$, $M_t \leftarrow \beta_1 M_{t-1} + (1-\beta_1) g_t$, $s_t \leftarrow \beta_2 s_{t-1} + (1-\beta_2)(g_t - M_t)^2$;

Step 3: Bias correction to M_t and s_t;

Step 4: Update $\theta_t \leftarrow \prod_{\mathcal{F}, \sqrt{s_t}} \left(\theta_{t-1} - \alpha \frac{M_t}{\sqrt{s_t + \varepsilon}}\right)$ using M_t and s_t corrected in Step 3.

Where g_t represents the t-th step, M_t represents the exponential moving average (EMA) of g_t, and α is learning rate. Consequently, AdaBelief replaces v_t in Adam with s_t. v_t and s_t are EMA of ${g_t}^2$ and $(g_t - M_t)^2$ respectively. In the proposed method for load forecasting, the optimizer will adopt this improvement to consider curvature of the loss function, instead of taking a large (small) step where the gradient is large (small). In other words, this method considers not only modulus size of the past gradient of parameters, but also the consistency of the gradient direction in the past.

III. SHORT-TERM LOAD FORECASTING BASED ON TCN-GRU

Short-term load forecasting can denote as:

$$\hat{Y}_i = f\left(X_i, \hat{W}, \hat{b}\right) \quad (11)$$

where X_i is the input data after correlation analysis and preprocessing; $\hat{W}$ and $\hat{b}$ are the optimal estimates of weight parameter and bias parameter. Based on massive input data, we continuously iterate in the direction of the fastest gradient descent with the goal of the smallest loss function value. Finally, the non-linear implicit relationship f between the short-term load and the input data is learned, and then obtain the optimal estimation value $\hat{Y}$ of short-term load forecasting. In this paper, the hidden information of data features is extracted by temporal convolution network and input it into the prediction model of gated recurrent unit to obtain the short-term load forecasting value.

A. TEMPORAL CONVOLUTIONAL NETWORK

Temporal Convolutional Network (TCN) is a neural network algorithm proposed by Bai *et al.* [22] in 2018 to analyze

time series data. Causal convolution, expansion convolution and residual connection are introduced in TCN, which solve the problem of feature extraction of long-term time series information. The structure and principle are described as follows:

1) CAUSAL CONVOLUTION

Causal convolution plays two key roles in TCN: making the network produce an output with the same length as the input, and avoiding the leakage from future to past. Figure 1 shows the principle of causal convolution structure. In causal convolution, convolution operations are performed strictly in chronological order, that is, the convolution operation at time t only occurs on the data before $t-1$ and $t-1$ in the previous layer. It is worth noting that causal convolution is easily restricted by the receptive field, so that it can only accept short history information for prediction.

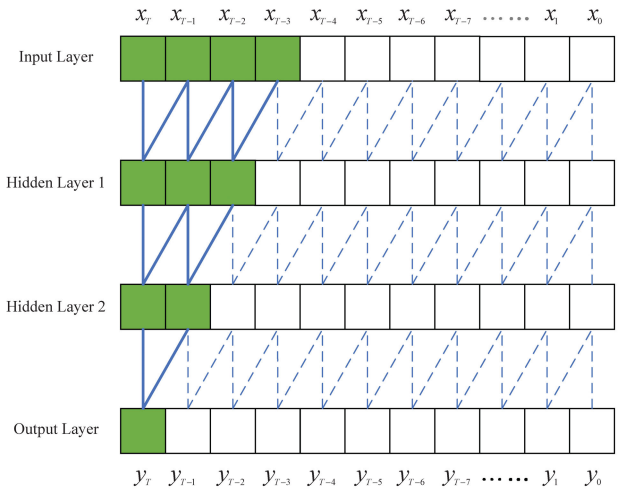

FIGURE 1. The principle of causal convolution structure.

2) DILATED CONVOLUTION

Distinct from traditional convolution, to solve the restricted receptive field problem, the TCN convolution structure allows interval sampling of convolution input, which is called dilated convolution [37]. For one-dimensional time series input $X = (x_0, x_1, \ldots, x_T)$ and filter $f : \{0, 1, 2, \ldots, n-1\} \rightarrow R$, the dilated convolution operation $F(\cdot)$ is defined as follows:

$$F(T) = (X *_d f)(T) = \sum_{i=0}^{n-1} f(i) \cdot x_{T-d \cdot i} \tag{12}$$

where d denotes the dilated factor, n represents the filter size and $T - d \cdot i$ accounts for the direction of the past.

By increasing dilation factor d and filter size n, the TCN is able to extend the receptive field. It allows the top layer output to receive a wider range of input information. The principle of dilated causal convolution structure is presented in Figure 2. The filter size $n = 2$, the top layer $d = 1$, means that every point is sampled during input; the middle

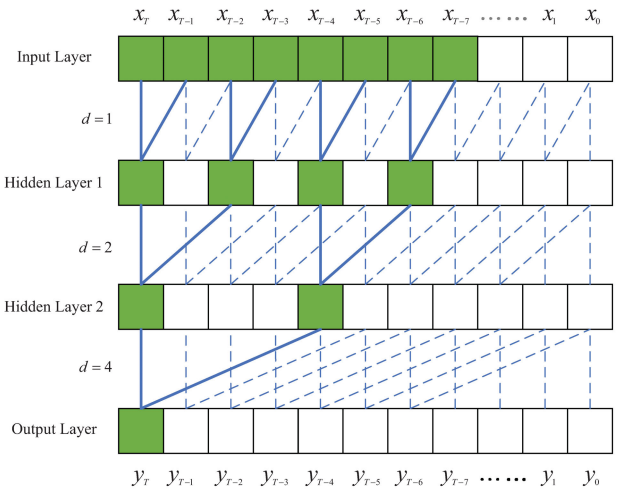

FIGURE 2. The principle of dilated causal convolution structure.

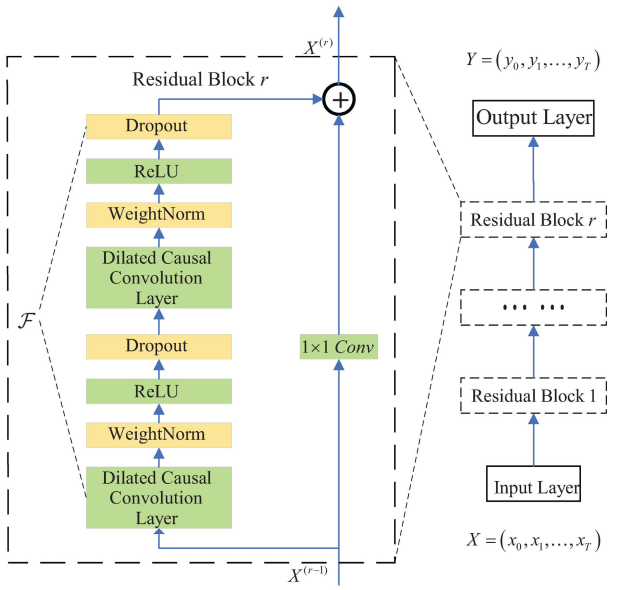

FIGURE 3. The details of residual block in a deep TCN.

layer $d = 2$ denotes that one of every two points is sampled as the input. It can be seen from Figure 2 that the deeper the layer, the larger the dilated coefficient. The size of the effective window in the TCN convolution structure increases exponentially with the number of layers. Therefore, TCN can obtain a larger receptive field with less layers.

3) RESIDUAL CONNECTION

TCN can also expand the receptive field by adding hidden layers. In order to avoid the disappearance of the gradient caused by the hidden layer being too deep, TCN applies the residual block to the deep network to solve this problem [38]. More specifically, each layer of TCN contains multiple filters for feature extraction. We replace the convolutional layer with a general residual block. The details of the residual block and a deep TCN structure are shown in Figure 3. It can be seen that a deep TCN consists of several residual blocks.

One branch of the residual block performs a transformation operation $\mathcal{F}$ on the input $X^{(r-1)}$, adding a branch for a $1 \times 1 Conv$ transformation to ensure that the element wise addition accepts tensors of the same shape [39]. The output $X^{(r)}$ of the r-th residual block can denote as:

$$X^{(r)} = Activation\left(\mathcal{F}\left(X^{(r-1)}\right) + X^{(r)}\right) \tag{13}$$

where activation function utilizes Rectified Linear Unit (ReLU) [40]. $\mathcal{F}$ is a series of transformations, including dilated causal convolution layer, weight normalization, activation layer and dropout layer. In details, for normalization, we apply weight normalization [41] to the convolution filter. In addition, a Dropout [42] is added after the activation layer of each convolution for regularization.

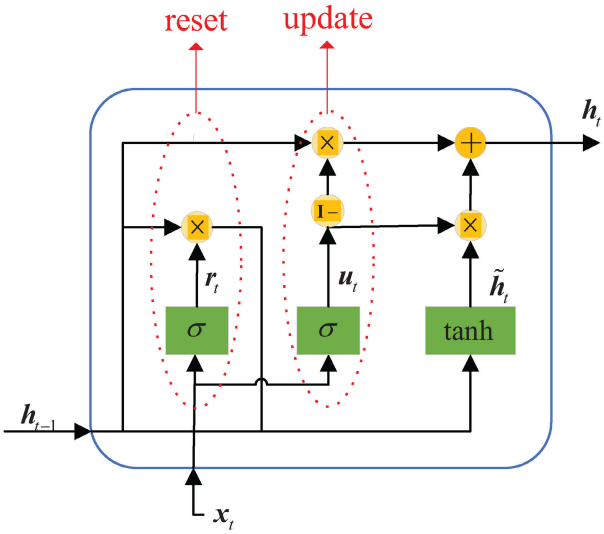

FIGURE 4. The structure of GRU network.

B. GATED RECURRENT UNIT & ATTENTION MECHANISM

1) GRU MODEL

GRU network is an improved model of LSTM network. It optimizes the three gate structures of LSTM, integrates forget gate and input gate into a single update gate [43], and mixes neuron state and hidden state. It can effectively alleviate "gradient disappearance" of RNN and reduce the number of parameters of the LSTM network unit. Thus, GRU shortens the training time of the model. The structure of the GRU network is shown in Figure 4. The mathematical formulas are described in formulas (14)-(18).

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{14}$$
$$u_t = \sigma(W_u \cdot [h_{t-1}, x_t]) \tag{15}$$
$$\tilde{h}_t = \phi(W_{\tilde{h}} \cdot [r_t \times h_{t-1}, x_t]) \tag{16}$$
$$h_t = (I - u_t) \times h_{t-1} + u_t \times \tilde{h}_t \tag{17}$$
$$y_t = \sigma(W_o \cdot h_t) \tag{18}$$

where x_t, h_{t-1}, h_t, r_t, u_t, $\tilde{h}_t$ and y_t are respectively the input vector, the state memory variable at previous moment, the state memory variable at current moment, the state of reset gate, the state of update gate, the state of the current candidate set and the output vector at current moment; W_r, W_u, $W_{\tilde{h}}$, W_o are weight matrices for the corresponding inputs of the network activation functions; I stands for Identity matrix; [] denotes vector connection; $\cdot$ represents matrix dot product; $\times$ is matrix cross product; σ represents sigmoid activation function; ϕ denotes tanh activation function. The mathematical description of σ and ϕ are as follows:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{19}$$
$$\phi(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{20}$$

The core module of GRU network is update gate and reset gate. The splicing matrix of the input variable x_t and the state memory variable h_{t-1} at previous moment is input into the update gate after sigmoid nonlinear transformation, which determines the degree of state variable brought into current state at previous moment. The reset gate controls the amount of information written to the candidate set at previous time. The information of previous time is stored by $I - u_t$ times h_{t-1}, and the information of current time is recorded by u_t times $\tilde{h}_t$, and adds the two as current time output.

2) ATTENTION MECHANISM

Attention mechanism [44] is the simulation of biological attention through the algorithm, which can dynamically adjust the attention degree of a deep neural network to different features according to the needs of different prediction tasks. It mainly changes the attention to information, thereby increasing important and useful information, suppressing and ignoring useless information. The attention mechanism is used to effectively highlight the key features that affect the power load in the prediction results of the GRU layer, and improve the prediction performance of the model. These features are processed through formula (21) and are used as a standard to measure the importance of different features. Then through equation (22), they are quantified as attention weight value between 0 and 1, finally through formula (23), summing all the products of attention weights and features, we get the final output features. Compared with the original features, features processed by the above steps will be more direct and effective. The above steps are carried out in a dynamic loop, so as to realize the highlight of important features, and then make more effective use of different features in different prediction situations. The formula is expressed as follows:

$$z_{ki} = u \tanh(W h_k + U h_i + b) \tag{21}$$
$$\alpha_{ki} = \frac{\exp(z_{ki})}{\sum_{j=1}^{n} \exp(z_{kj})} \tag{22}$$
$$H = \sum_{i=1}^{n} \alpha_{ki} h_i \tag{23}$$

where z_{ki} denotes the measure standard of feature importance; u, W, U stand for the weight parameter matrices; b accounts for the bias; h_k stands for the hidden layer state corresponding

TABLE 2. Correlation coefficient comparison.

Category	Temperature	Humidity	Pressure	Wind_speed	Wind_direction	Rain	Cloud
DCC	**0.681**	**0.315**	0.110	**0.437**	0.107	0.145	0.139
PCC	0.499	-0.195	-0.081	0.202	0.050	0.068	0.051

to the last input; h_i stands for the hidden layer state corresponding to the i-th element of the input sequence; tanh is activation function; α_{ki} represents the attention weight of the hidden state of historical input to the current input; H is final output feature.

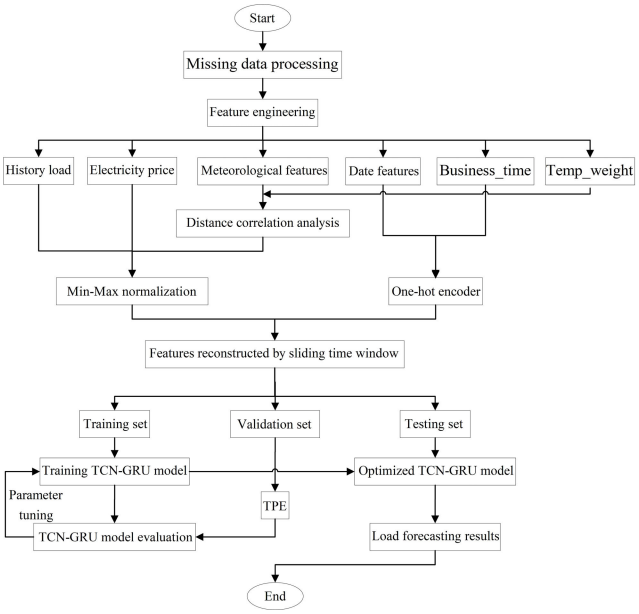

FIGURE 5. The short-term load forecasing framework based on TCN-GRU model.

C. THE LOAD FORECASTING FRAMEWORK BASED ON TCN-GRU MODEL

Due to the combination of CNN's extraction capabilities and RNN's time-domain modeling capabilities, TCN can extract the temporal correlation of features [45]. Besides, considering the load data scale of the whole area, GRU not only increases the depth of the model but also processes big data more accurately and quickly than LSTM. Thus, the model can forecast load data with multiple features according to the extracted feature information. The short-term load forecasting framework based on the TCN-GRU model is illustrated in Figure 5. The framework is established by four main steps: missing data processing, distance correlation analysis, feature engineering, and load forecasting based on the TCN-GRU model. The first three steps together are the data preprocessing. Each step in the framework is described in detail as follows:

1) MISSING DATA PROCESSING

The raw data usually have a large amount of missing data because of interrupted signal transmissions or acquisition equipment failures. There are not many missing values in the dataset used in this paper. To avoid the adverse impact of missing data on load forecasting, missing data are filled with data from the same moment of the previous day.

2) DISTANCE CORRELATION ANALYSIS

In this paper, we analyze the correlation between the data and compare the Pearson coefficient and distance correlation coefficient between the meteorological features and the predicted load. Among them, because some features and the predicted quantity are not simply linear, the distance correlation coefficient that can measure the non-linear relationship is better than the traditional Pearson correlation coefficient in the correlation analysis. Table 2 analyzes the correlation coefficient comparison of meteorological features. Each weather feature is the value obtained after the weighted average. The original weather data is from the hourly data of five major cities in Spain (Madrid, Barcelona, Valencia, Seville, Bilbao) [46].

From Table 2, although the DCC score of the same feature is significantly better than PCC, the overall correlation coefficients are still low for the following reasons. It is difficult and tedious to collect and process the national weather data. We select data of major cities to reflect the overall national weather changes because large cities with large populations account for a larger share of the national power generation. These five major cities make up one-third of Spain's total population. The correlation coefficient of weather features after weighted average is relatively low ($<$0.8), but it can still reflect the correlation with the total power generation to a certain extent. We retain the three weather features of temperature, humidity, and wind speed (DCC values are relatively high), removing other weather features (DCC values are about 0.1) that have little impact on the results [47], [48].

TABLE 3. Forecasting results of proposed model with different features.

Features	X_{MAPE} (%)	X_{RMSE} (MW)	X_{MAE} (MW)	Calculation time per Epoch (s)
All weather features	1.22	497.71	349.40	163
Weather features selected via DCC	1.24	493.61	346.09	94

Further, Table 3 analyzes the forecasting results of the proposed model with or without irrelevant weather features. It is proved that the weather variables selected via the Distance correlation coefficient are able to bring better forecasting performance and faster prediction speed.

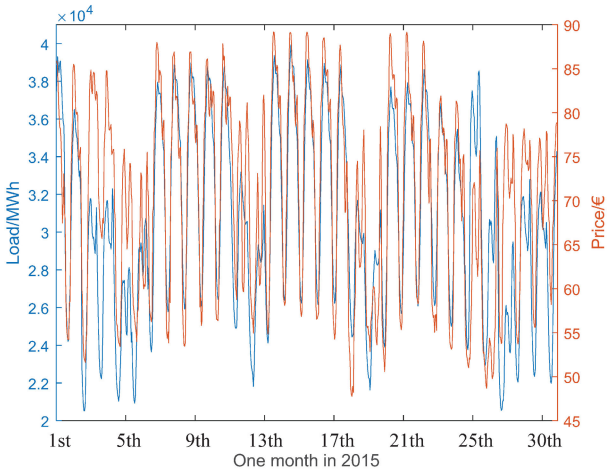

FIGURE 6. Load and electricity price for one month in 2015.

3) FEATURE ENGINEERING

Our work mainly conducts feature engineering on five aspects of data: historical load data, electricity price data, meteorological features, date features, and other features. Electricity price is also considered to be a vital relevant factor affecting load changes. Load and electricity price for one month in 2015 is described in Figure 6. It can be seen that the total load is positively correlated with the electricity price. Meanwhile, the distance correlation coefficient also proves the correlation between electricity price and load, and the correlation coefficient score is 0.577. In this paper, this correlation coefficient score is higher, so we choose the price feature as an important feature of load forecasting.

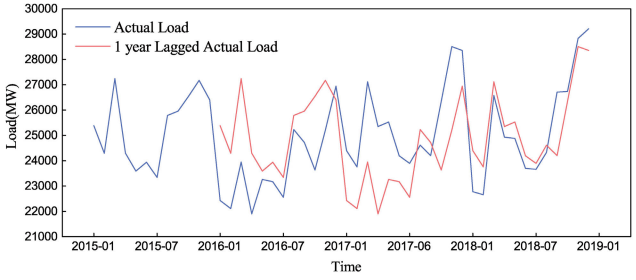

FIGURE 7. Actual electricity load (monthly frequency) and 1-year lagged load.

This research mainly considers the date features such as week, month and holiday. Figure 7 presents the actual electricity load (Monthly frequency) and 1-year lagged load. From this point, we can find that the load has seasonal pattern on a monthly scale. At the same time, we not only consider whether a given time belongs to a weekend, but also specifically distinguish between Saturday and Sunday. It is appropriate to take the date features as the relevant factor for total load forecasting.

Finally, we generate some important features. After seeing that there is a high correlation among the temperatures of the different cities, we will also try creating a weighted temperature features by taking into account each city's population. The specific formulas are as follows:

$$w_x = \frac{p_x}{P} \tag{24}$$

$$Temp = \sum w_x \cdot Temp_x \tag{25}$$

where w_x is the temperature weight of each city, p_x stands for the population of each city, P stands for total population of 5 cities, $Temp_x$ is the temperature in each city and $Temp$ is the temperature features input into the model.

We have also generated a very useful feature based on the peculiarities of Spain, that is, whether a given period is in business time. Not all companies have weekdays between 9 a.m. and 5 p.m., because there is a siesta. The most common business hours are from Monday to Saturday, 9:30 a.m. to 1:30 p.m., and then from 4:30 p.m. to 8:00 p.m.

In this paper, we adopt the one-hot encoding process for category features. Due to the different feature dimensions of the collected discrete data, to improve the robustness of the model, the data is normalized, and the formula is as follows:

$$X_n = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{26}$$

where X_n is the normalized data, X is the original data, $X_{\max}$ and $X_{\min}$ are the maximum and minimum values of each feature in the sample.

TABLE 4. Classification and processing methods of input features.

Type of feature	Feature	Processing method
History Load	Load	Linear interpolation is used, reconstructed by sliding time window and normalized by the Min-Max method
Electricity Price	Price	Reconstructed by sliding time window and normalized by the Min-Max method
Meteorological features	Temperature Humidity Wind_speed	Weighted average is used, reconstructed by sliding time window and normalized by the Min-Max method
Date features	Week	Feature mapping: 0 represent Monday to Friday, 1 represent Saturday, 2 represent Sunday
	Month	Feature mapping: 0-11 represent January to December
	Holiday	Feature mapping: 0 for a non-holiday and 1 for a holiday
Other features	Temp_weight	Temperature feature is weighted according to population proportion
	Business_time	Feature mapping: 2 for given hour being in "Business time", 1 for "Siesta", 0 for other given hours

The classification and processing methods of input features are described in Table 4. Owing to a large number of feature types, we also use 24 previous time steps [49] to extract or generate features in order to avoid redundancy during data operation and reduce model performance. Since the past features may have a significant impact on the current load

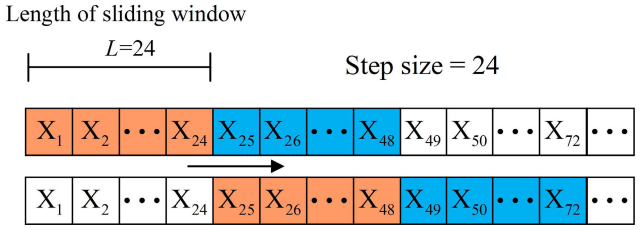

FIGURE 8. The principle of sliding time window.

forecasting, the network can understand the past deeply and extract time-varying features. The principle of the sliding time window is shown in Figure 8 below. The step size is set to 24, which means that each time the sliding window in Figure 8 moves 24 steps to the right to reconstruct features. In other words, we perform day-ahead short-term load forecasting (direct multistep-ahead forecasting) with 24 steps of historical data [50], [51].

4) LOAD FORECASTING BASED ON TCN-GRU MODEL

With the input of multi-features, the difficulty of load forecasting has been improved a lot. Fortunately, the TCN network can effectively extract the input feature X. The data reconstructed by sliding time window is divided into a training set, a validation set and a testing set. The training set is utilized to train the TCN-GRU model, and the validation set is used for parameters tuning by the TPE algorithm. After several iterations, the testing set is input into the optimized TCN-GRU model to predict the short-term load and evaluate the model performance.

Otherwise, to further improve the performance of the model, this article also uses two techniques to make innovations: AdaBelief optimizer and Attention mechanism. We have added different optimizer comparison experiments, compared the performance of Adam, SGD (stochastic gradient descent) [52], and AdaBelief in terms of training loss and prediction accuracy. The Attention mechanism is also added to the neural network structure to help us better predict the fluctuation of the load at details [48]. The structure of our proposed TCN-GRU model is shown in Figure 9.

5) PERFORMANCE EVALUATION

In this paper, mean absolute percentage error (MAPE), root mean squared error (RMSE) and mean absolute error (MAE) are selected as evaluation metrics of load forecasting model. The formulas are as follows:

$$X_{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{y_i} \times 100\% \tag{27}$$

$$X_{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{28}$$

$$X_{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{29}$$

where n represents the number of testing samples. y_i and $\hat{y}_i$ respectively denote the actual load value and model predicted load at time T.

IV. CASE STUDY

A. EXPERIMENTAL SETTINGS

In this paper, the electricity and energy data of Spain from January 1, 2015 to December 31, 2018 (1h sampling interval) is used as the research object. The data is retrieved from ENTSOE, which is a public portal of Transmission Service Operator (TSO) data. The electricity price data is from TSO Red Electric Espana. Weather data is open-source on Kaggle [46], obtained from the open meteorological API of the five largest cities in Spain. The datasets are split into a training set, a validation set and a testing set according to the proportion of 8:1:1, i.e., the training set accounts for 28052 hours, the validation set and test set each contain 3506 hours of data.

In this paper, firstly Experiment 1 is set up to prove the improvement effect of the proposed model, and the effects of different optimizers and Attention mechanism on the learning performance and prediction error of TCN-GRU are compared.

Then, in order to verify the effectiveness and superiority of the proposed model, we compared the prediction results of TCN-GRU on the dataset with other comparative models in Experiment 2. It is noted that the proposed model is compared and analyzed with various single prediction models (SVR, XGBoost, LSTM, GRU, TCN) and hybrid models (CNN-LSTM, CNN-GRU, TCN-LSTM). In Experiment 3, the proposed model is tested on PJM power system load data to verify the forecasting effectiveness in other scenarios. All models adopt the TPE algorithm for parameter tuning, and the prior distribution of hyperparameters is set according to prior experience to ensure that the compared models have relatively consistent complexity. All experimental models run in the Python 3.7 programming environment. The deep learning architecture is based on the Tensorflow framework, the version is tensorflow-gpu 2.2.0, the hardware is a laptop with Intel Core i5-4210, the GPU is NVIDIA GeForce GTX 840M, and 16GB of memory.

B. EXPERIMENT 1: THE INFLUENCE OF AdaBelief OPTIMIZER AND ATTENTION MECHANISM ON LOAD FORECASTING

In this experiment, we use the same learning rate (lr = 1e-2) and the same training epoch (n = 60) to test the changes of loss function of TCN-GRU in the states of AdaBelief, Adam, and SGD. It is noted that mean squared error (MSE) is used as the loss function to compare the performance of the Adabelief optimizer with other optimizers. The formula is:

$$Loss = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{30}$$

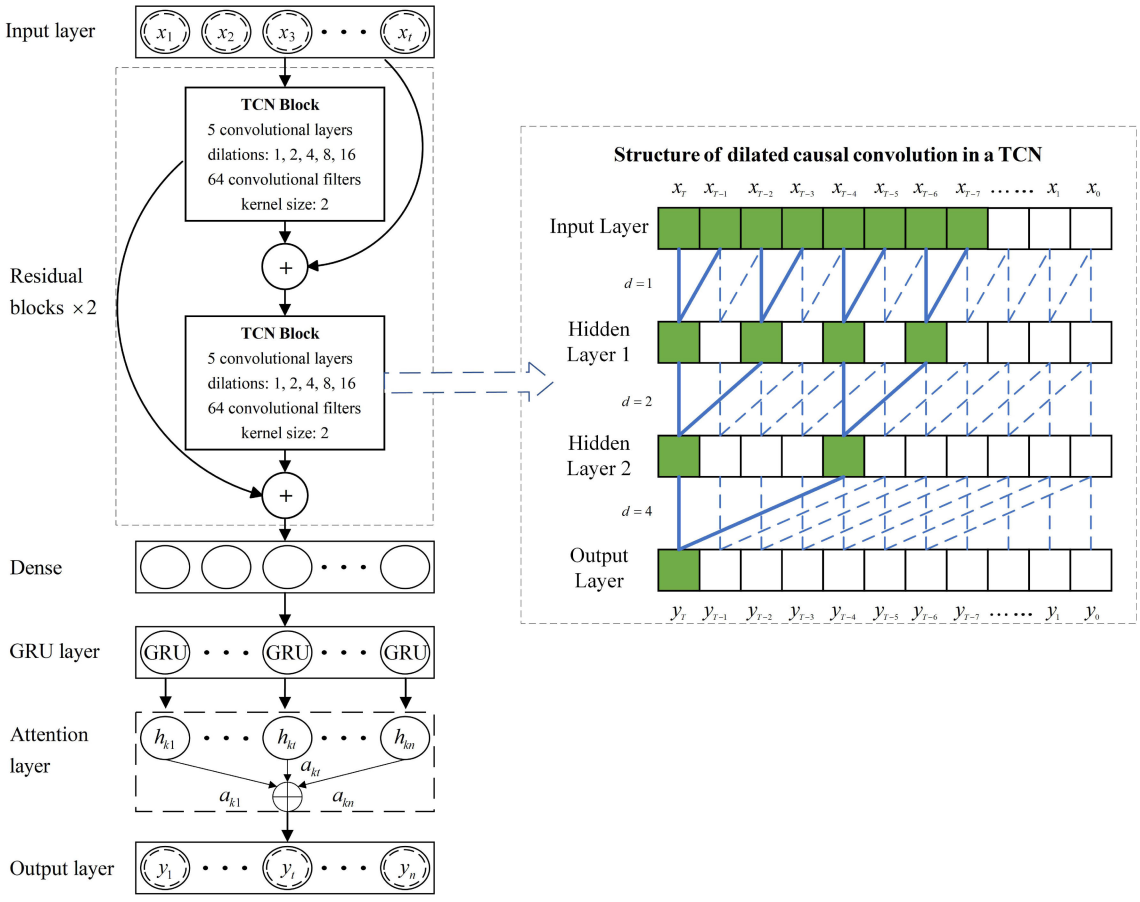

FIGURE 9. The proposed TCN-GRU model structure.

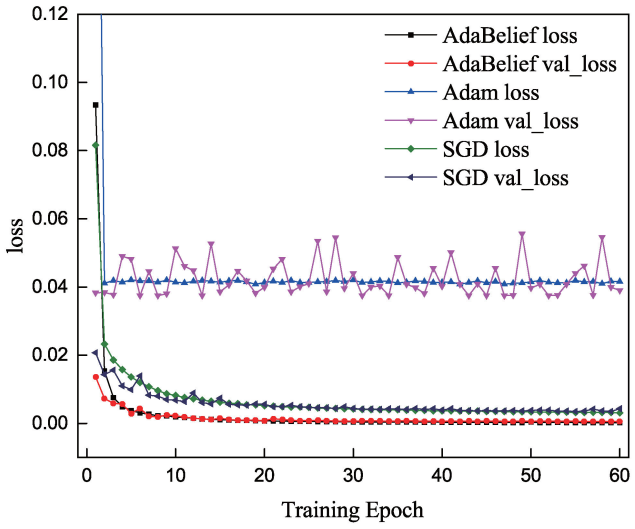

FIGURE 10. MSE loss trend of TCN-GRU on training set with different optimizers.

where n represents the number of samples. y_i stands for the actual value. $\hat{y}_i$ stands for the predicted value of the model.

Figure 10 shows the MSE loss trend of TCN-GRU on training set with different optimizers. It is obvious that compared with Adam and SGD, AdaBeilief has the lowest loss on the training set and validation set, and the learning speed is also the fastest.

TABLE 5. Load forecasting evaluation on the testing set with different optimizers.

	X_{MAPE} (%)	X_{RMSE} (MW)	X_{MAE} (MW)	Calculation time per Epoch (s)
AdaBelief	1.24	493.61	346.09	94
Adam	7.02	2211.86	1892.94	125
SGD	3.88	1386.23	1104.60	101

The comparison of prediction error of different optimizers on testing set is presented in Figure 11. It is obvious that AdaBelief has the largest number of samples of small errors. In addition, combined with the results of various metrics in Table 5, Adam has a large prediction error at this learning rate (RMSE>2000MW), and the training effect is obviously inferior to AdaBelief and SGD. On the one hand, from the trend of the loss curve, it can also be seen that Adam optimizer has encountered a barrier in this case. The val_loss fluctuates around 0.04, and the learning rate needs to be reduced to ameliorate. On the other hand, the calculation speed of each epoch of AdaBelief is also significantly faster than the other

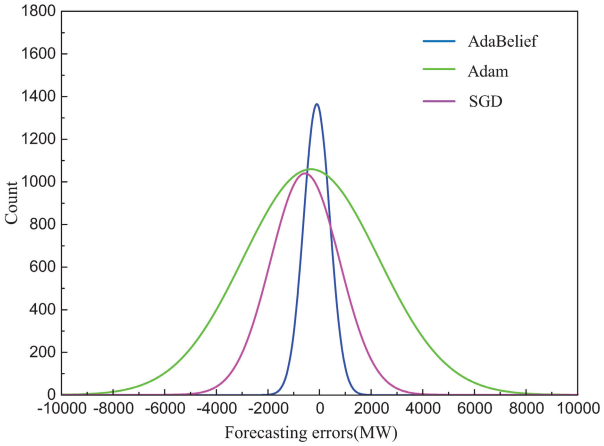

FIGURE 11. Comparison of prediction error of different optimizers.

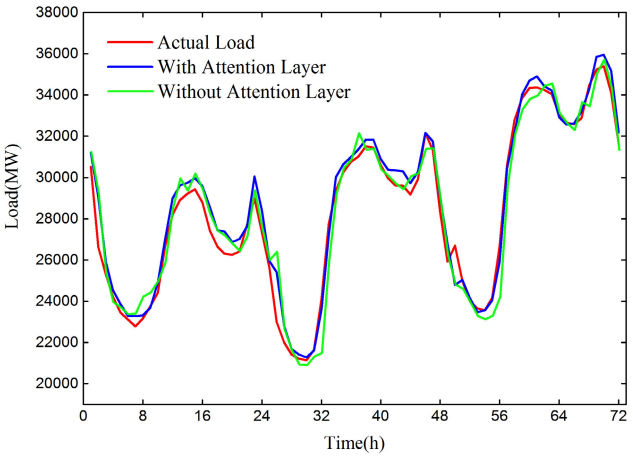

FIGURE 12. Short-term load forecasting with or without attention mechanism.

two optimizers (computation time is reduced by 24.8% and 6.9% respectively compared with Adam and SGD). In general, in this load forecasting task, the AdaBelief optimizer has only 1.24% on MAPE, which brings a very outstanding performance improvement.

TABLE 6. Short-term load forecasting evaluation with or without attention mechanism.

	X_{MAPE} (%)	X_{RMSE} (MW)	X_{MAE} (MW)	Calculation time per Epoch (s)
With Attention	1.90	700.17	524.73	96
Without Attention	2.26	887.51	608.52	90

After determining AdaBelief as the optimizer of the proposed method, this article also sets up an experiment to judge whether the Attention mechanism can improve the performance of the model. While ensuring the same other parameters and experimental environment (especially, the same optimizer), we compared the performance of TCN-GRU with or without the Attention layer. The selected three-day (8.8-8.11 in 2018) short-term load prediction experimental results are shown in Table 6, and the prediction curve results are shown in Figure 12. It is clear that the model with Attention layer is superior to the model without Attention layer in various statistical metrics. The progress in MAPE, RMSE and MAE are 16.0%, 21.1%, 13.8% respectively. But after adding the attention mechanism, the calculation time of each Epoch is slightly increased. We are able to draw a conclusion that the TCN-GRU load prediction model with Attention layer has improvement in prediction accuracy compared to the model without the Attention layer, but the time required for each Epoch increases.

C. EXPERIMENT 2: LOAD FORECASTING OF DIFFERENT MODELS ON SPAIN LOAD DATASET

In this experiment, according to the aforementioned split rule, the training set time ranges from January 1, 2015 to March 14, 2018, and the validation set is from March 15, 2018 to August 7, 2018. The rest (August 8, 2018 to December 31, 2018) is the testing set. All models are trained with the training set and the validation set is utilized for parameter tuning.

The parameters in this article are tuned by the TPE algorithm provided by Hyperopt [53]. The prior distribution of hyperparameters are as follows: learning rate is set as logarithmic uniform distribution between 0 and 1, the batch size is set to [8, 16, 32, 64], and the distribution set by dropout [54] is [0.2, 0.3, 0.4, 0.5], the number of epochs is set to an integer uniform distribution from 50 to 200, and the number of nodes in hidden layer of neural network is distributed as [32, 64, 128, 256]. The remaining parameters are set according to past experience. It is worth noting that the deep learning models all utilize the AdaBelief optimizer and Attention mechanism.

Finally, the parameters of each model optimized by TPE are summarized as follows.

(1) TCN: The algorithm is built using the Keras library in Tensorflow framework. The number of filters is 64, the kernel size in every convolution layer is 2, and the dilation factor is set to [1, 2, 4, 8, 16]. The stack number of the residual block is 2, activation function using ''ReLU''.

(2) GRU: The number of hidden layers is 2 and the number of nodes in hidden layer is set to 256/128, activation function using ''tanh''.

(3) LSTM: The number of hidden layers is 3 and the number of nodes in hidden layer is set to 128/128/64, activation function using ''tanh''.

(4) CNN: The number of convolution layers is 1, the number of filters in convolution layer is 64, the kernel size in every convolution layer is 2, the number of fully connected layers is 2 and the number of neurons in fully connected layer is set 100/1, activation function using ''ReLU''.

(5) XGBoost: The learning rate is 0.01, the max depth of trees is 8, iteration is set to 60, colsample equals 0.8, alpha is 0.1, lamda is 0.2, gamma is 0.1 and the minimum child weight is 3.

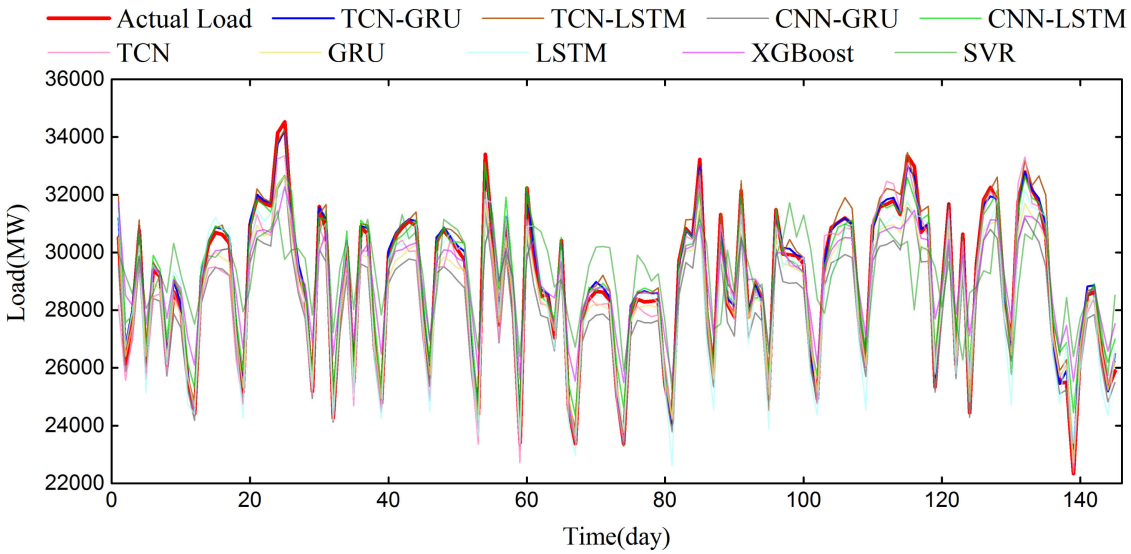

FIGURE 13. Load forecasting profiles of all models on the testing set.

(6) SVR: The kernel function is ''rbf'', the penalty factor is set to 5 and other parameters are default settings.

The prediction results of all models on the testing set are shown in Figure 13. In order to better display, we process the forecast results every 24 hours into the average hourly load of the day. Definitely, there is volatility in the load on different days.

TABLE 7. Load forecasting evaluation on the testing set.

Time	From 08-Aug-2018 to 31-Dec-2018		
Models	X_{MAPE} (%)	X_{RMSE} (MW)	X_{MAE} (MW)
SVR	7.51	2685.67	2093.22
XGBoost	5.86	1991.44	1608.04
LSTM	4.24	1705.70	1202.37
GRU	2.44	1009.52	741.02
TCN	4.10	1468.02	1156.46
CNN-LSTM	3.43	1295.60	974.25
CNN-GRU	3.13	1160.62	964.76
TCN-LSTM	2.21	839.45	628.34
Proposed model	**1.24**	**493.61**	**346.09**

Load forecasting evaluation of all models on the testing set is presented in Table 7. Distinctly, the TCN-GRU hybrid neural network model proposed in this paper has achieved the best results in MAPE, RMSE, and MAE. Compared with the comparison models, the proposed method has a significant improvement in performance, and the forecasting load error is smaller. In the individual models, GRU has the best performance, while SVR model performs relatively poorly (RMSE>2000MW). When comparing in the hybrid models, it can be seen that the extraction ability of TCN is superior to that of CNN. As far as MAPE is concerned, TCN is 35.6% lower than CNN in LSTM, and TCN is 60.4% lower than CNN in GRU. Among all comparison models, the method proposed in this paper combines the best performance GRU in a single prediction model and the TCN with better extraction capability. The MAPE is reduced to 1.24%, our proposed model obtains the best prediction performance as a result.

TABLE 8. Calculation time of each epoch for all neural network models.

Models	Calculation time per Epoch (s)
LSTM	45
GRU	35
TCN	30
CNN-LSTM	55
CNN-GRU	49
TCN-LSTM	106
Proposed model	94

Further, we also researched the calculation time of all neural network models on the testing set. The calculation time for each epoch of different models is shown in Table 8. The results present that the TCN model has the fastest load prediction speed in the individual models, and the GRU prediction speed is better than LSTM [14]. In the hybrid models, whether CNN or TCN is adopted as the extraction method, the calculation time of GRU and LSTM is undoubtedly increased. Among them, the model combined with TCN is more time-consuming, but TCN-GRU still has a shorter calculation time than TCN-LSTM. Although the calculation time of the proposed model is relatively higher than single models, with the development of cloud computing and the

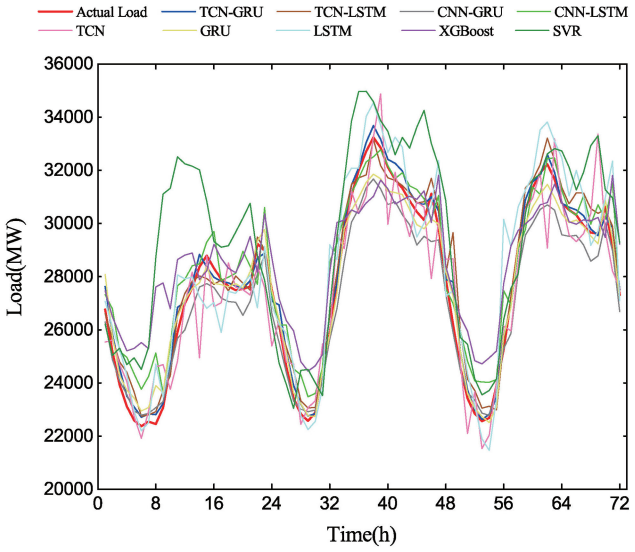

FIGURE 14. Short-term load forecasting of all models on three-day working day.

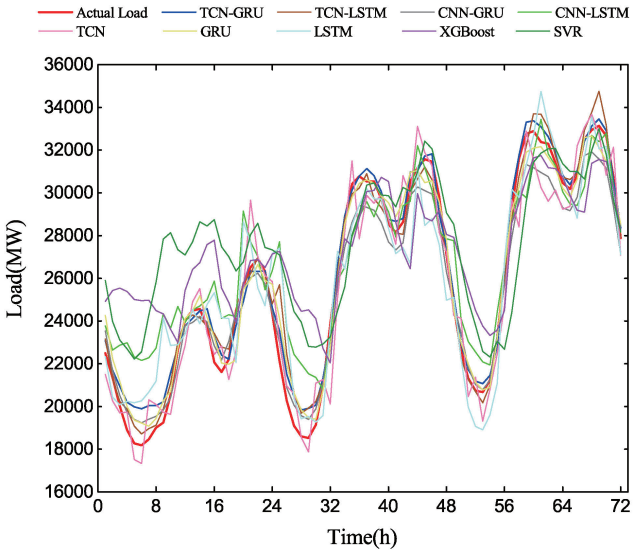

FIGURE 15. Short-term load forecasting of all models on three-day holiday.

advancement of GPU computing power in future, it is still satisfactory in practical applications.

In order to evaluate the performance of the model more comprehensively, we used different time periods (three days in August, 8.15-8.17 and three days during the Christmas holiday, 12.25-12.27) for short-term load forecasting and further analysis. The forecast results are shown in Figure 14 and Figure 15. The results show that all models can roughly fit the rising and falling trend of load whether it is during normal working days or during the peak period of holiday power consumption. In terms of the peak, valley, stable or fluctuation range of the actual load in the testing set, the model proposed in this article has the smallest bias. Compared with several comparative models, it can best fit and capture the changing trend of the actual load.

Considering that there is a large difference in electricity consumption between weekdays and holidays, we compared and analyzed short-term load forecasting metrics for the three days of normal working days (mid-August) and Christmas holidays. Table 9 shows two representative short-term load forecasting results during working days and holidays. Obviously, in normal working days, the proposed model has achieved the best prediction results in all statistical metrics, and the MAPE is only 1.24%. In holiday short-term load forecasting, TCN-LSTM is slightly better than TCN-GRU on MAPE, and TCN-GRU is slightly better than TCN-LSTM on MAE. When comparing RMSE, the proposed model forecasting error is distinctly lower than other models. Generally speaking, the proposed model has apparent advantages and strong robustness in load forecasting performance.

D. EXPERIMENT 3: LOAD FORECASTING OF DIFFERENT MODELS ON PJM POWER SYSTEM

In this experiment, to evaluate our proposed model performance over different scenarios, a set of hourly load data are added from PJM, which is a famous benchmark. According to the data source information, the market region is EAST and the time is from 2017-7-3 00:00 to 2018-7-3 00:00. The specific data is shown in Figure 16.

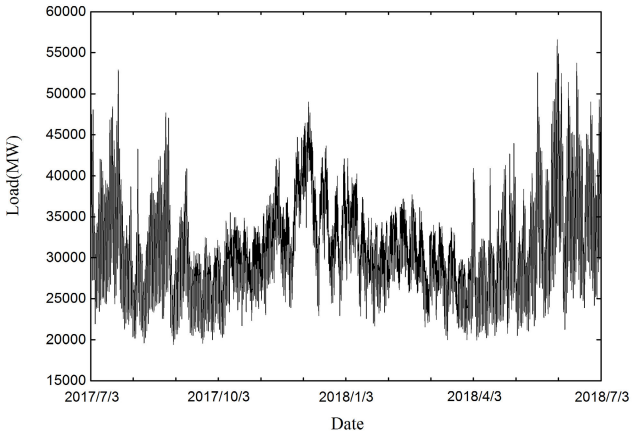

FIGURE 16. PJM hourly metering load data.

Similarly, the dataset is split into a training set, a validation set and a testing set according to the proportion of 8:1:1. The parameters of each model are set as follows.

(1) TCN: The algorithm is built using the Keras library in Tensorflow framework. The number of filters is 64, the kernel size in every convolution layer is 2, and the dilation factor is set to [1, 2, 4, 8]. The stack number of the residual block is 2, activation function using "ReLU".

(2) GRU: The number of hidden layers is 1 and the number of nodes in hidden layer is set to 128, activation function using "tanh".

(3) LSTM: The number of hidden layers is 1 and the number of nodes in hidden layer is set to 128, activation function using "tanh".

TABLE 9. Short-term load forecasting evaluation for each model in two representative period.

Time	Three working days in August 2018			Three holidays in Xmas 2018		
Models	$X_{MAPE}(\%)$	X_{RMSE}(MW)	X_{MAE}(MW)	$X_{MAPE}(\%)$	X_{RMSE}(MW)	X_{MAE}(MW)
SVR	7.05	2422.72	1890.08	11.24	3190.94	2532.74
XGBoost	5.38	1668.34	1392.71	10.78	3063.04	2399.27
LSTM	3.67	1328.57	1007.66	5.85	1785.54	1413.30
GRU	1.67	573.12	457.60	2.59	795.49	619.34
TCN	3.31	1235.94	916.75	4.10	1293.40	1038.60
CNN-LSTM	3.29	1075.97	856.77	7.12	2081.83	1614.73
CNN-GRU	2.38	816.86	689.30	3.13	944.35	826.32
TCN-LSTM	1.97	701.93	525.81	**2.36**	797.99	569.91
Proposed model	**1.24**	**424.40**	**336.32**	2.39	**684.12**	**537.53**

(4) CNN: The number of convolution layers is 1, the number of filters in convolution layer is 16, the kernel size in every convolution layer is 2, the number of fully connected layers is 2 and the number of neurons in fully connected layer is set 32/1, activation function using "ReLU".

(5) XGBoost: The learning rate is 0.05, the max depth of trees is 6, iteration is set to 60, colsample equals 0.8, alpha is 0.1, lamda is 0.2, gamma is 0.1 and the minimum child weight is 3.

(6) SVR: The kernel function is "rbf", the penalty factor is set to 10 and other parameters are default settings.

TABLE 10. Calculation time of each Epoch for all neural network models.

Models	Calculation time per Epoch (s)
LSTM	37
GRU	25
TCN	22
CNN-LSTM	45
CNN-GRU	40
TCN-LSTM	74
Proposed model	58

The prediction results based on different models for PJM load data on 2018-6-21 are shown in Figure 17. Similar to experiment 2, the computational speed and statistical metrics of each model are computed and shown in Table 10 and Table 11. From these results, although PJM load dataset does not contain meteorological features and electricity price data, the proposed model still achieves the best forecasting performance. Based on TCN-GRU model, the MAPE is only 1.16% with less features. In terms of MAPE, TCN-GRU is 59.7% lower than CNN-GRU. It can be concluded that our proposed is competent for different STLF situations. Among all the comparison models, our proposed model is mediocre in prediction speed but has obvious advantages in forecasting accuracy.

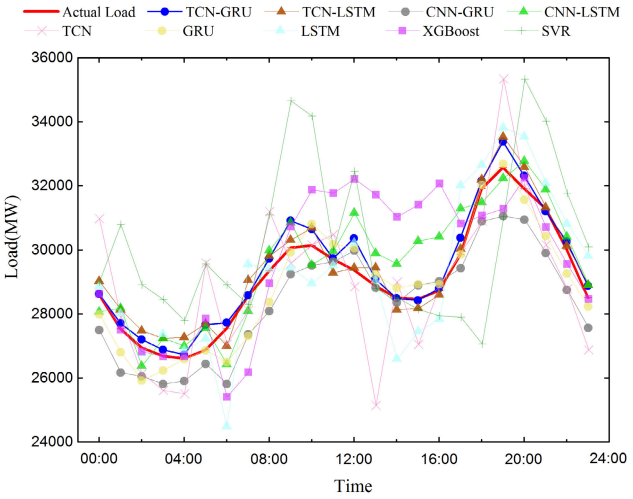

FIGURE 17. The prediction results based on different models for PJM load data on 2018-6-21.

TABLE 11. Load forecasting evaluation on the testing set.

Models	X_{MAPE} (%)	X_{RMSE} (MW)	X_{MAE} (MW)
SVR	6.46	2359.14	1903.21
XGBoost	4.24	1653.73	1234.88
LSTM	3.11	1150.75	909.17
GRU	1.72	610.51	495.48
TCN	3.84	1463.03	1111.53
CNN-LSTM	2.75	918.08	795.87
CNN-GRU	2.88	956.08	839.55
TCN-LSTM	1.50	488.97	431.10
Proposed model	**1.16**	**429.55**	**318.08**

E. DISCUSSION AND ANALYSIS

In this paper, the load data considering electricity price, weather and date are utilized to simulate the experiment. Experimental results from all aspects present that the

proposed model is superior to the comparison models in terms of prediction accuracy and efficiency. Furthermore, the following information can be analyzed from the results:

(1) Experiment 1 proved that the proposed model was further improved with the blessing of two technologies. Compared with the traditional Adam in load forecasting tasks, the AdaBelief optimizer has a faster convergence rate, better stability, obvious accuracy advantages and stronger normalization. Theoretically, AdaBelief mainly modifies the adaptive learning rate tuning item in Adam and considers curvature information. Although the change is little, it is very prominent in the prediction effect of the time series. Besides, the accuracy is slightly improved with the added Attention. In principle, Attention solves the problem that RNN cannot be operated in parallel. Meanwhile, Attention can grasp the key points of information and avoid the weakening of long-distance information to some extent. Although adding attention mechanism inevitably increases the calculation time slightly, we mainly focus on prediction accuracy. In terms of load forecasting, the optimization of neural networks is still worthy of study.

(2) In the individual models, GRU fits the actual load best, and the convergence rate is faster than LSTM. In theory, GRU merges the input gate and forget gate in LSTM into one update gate, eliminating redundant gate mechanism. Moreover, the GRU has fewer parameters, and when the data set is not particularly large, it can improve the prediction efficiency and alleviate the problem of gradient disappearance.

(3) In the hybrid models, the performance of the proposed model is better than CNN-LSTM, CNN-GRU and TCN-LSTM. And the prediction error of TCN-LSTM is lower than that of CNN-LSTM. These demonstrate that TCN is superior to CNN in feature extraction. In principle, TCN has a wider receptive field due to dilated convolution and can capture historical data and time relationships in a long-term range. TCN also introduces residual block to avoid gradient disappearance. Due to the limited receptive field, the CNN model has a poor ability to capture long-distance features, and even the prediction accuracy of the CNN-GRU hybrid model is worse than that of the single GRU model. This paper proposes a model that combines the advantages of TCN and GRU to achieve the best prediction effect. However, in terms of calculation time, due to features extraction, forecast requires more time than single models, but the prediction accuracy is improved. In the practical application of load forecasting, forecast accuracy should be primarily considered.

(4) In this article, we also discuss the short-term load forecasting during working days and holidays. No matter what the load fluctuation is, the proposed model achieves the best prediction results. It can be summarized that the model maintains a high precision prediction effect from the overall period to the local time, and the model has a certain capability to adapt to the fluctuation of the load. Moreover, the proposed model is tested on PJM power system to verify the forecasting effectiveness in other STLF scenarios. Finally, Table 3 also shows the importance of data preprocessing in load forecasting.

V. CONCLUSION

This paper proposes a hybrid TCN-GRU model for short-term load forecasting. First, the correlation between meteorological features and short-term load are analyzed, using fixed-length sliding time windows to reconstruct various features as input features of TCN. Next, the historical information and time relationship hidden in features can be extracted well by TCN, and then use the prediction advantages of the GRU model. With the support of AdaBelief optimizer and Attention mechanism, the proposed TCN-GRU model improves the accuracy and efficiency of short-term load forecasting. Finally, the model parameters tuning adopts the TPE algorithm which consumes less time. The experimental results in this paper present that the model has good adaptability to short-term load forecasting in different periods and different types of load data, in other words, the proposed model has strong robustness. It can be concluded that the combination of more new technologies and neural networks is a tendency for the improvement of short-term load forecasting in the future.

In future work, it is necessary to consider more detailed weather data and higher-dimensional data features on the performance of the prediction model and verify the prediction effect of this method in other scenarios. In addition, some novel load data preprocessing methods that can improve the prediction accuracy should be explored.

REFERENCES

[1] B. A. Hoverstad, A. Tidemann, H. Langseth, and P. Ozturk, "Short-term load forecasting with seasonal decomposition using evolution for parameter tuning," *IEEE Trans. Smart Grid*, vol. 6, no. 4, pp. 1904–1913, Jul. 2015, doi: 10.1109/TSG.2015.2395822.

[2] W. Li, Q. Shi, M. Sibtain, D. Li, and D. E. Mbanze, "A hybrid forecasting model for short-term power load based on sample entropy, two-phase decomposition and whale algorithm optimized support vector regression," *IEEE Access*, vol. 8, pp. 166907–166921, 2020, doi: 10.1109/ACCESS.2020.3023143.

[3] F. Zhao, B. Sun, and C. Zhang, "Cooling, heating and electrical load forecasting method for CCHP system based on multivariate phase space reconstruction and Kalman filter," *Proc. CSEE*, vol. 36, no. 2, pp. 399–406, 2016.

[4] S. Sharma, A. Majumdar, V. Elvira, and E. Chouzenoux, "Blind Kalman filtering for short-term load forecasting," *IEEE Trans. Power Syst.*, vol. 35, no. 6, pp. 4916–4919, Nov. 2020, doi: 10.1109/TPWRS.2020.3018623.

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

[6] J. F. Rendon-Sanchez and L. M. de Menezes, "Structural combination of seasonal exponential smoothing forecasts applied to load forecasting," *Eur. J. Oper. Res.*, vol. 275, no. 3, pp. 916–924, Jun. 2019, doi: 10.1016/j.ejor.2018.12.013.

[7] J. C. Lopez, M. J. Rider, and Q. Wu, "Parsimonious short-term load forecasting for optimal operation planning of electrical distribution systems," *IEEE Trans. Power Syst.*, vol. 34, no. 2, pp. 1427–1437, Mar. 2019, doi: 10.1109/TPWRS.2018.2872388.

[8] X. Cao, S. Dong, Z. Wu, and Y. Jing, "A data-driven hybrid optimization model for short-term residential load forecasting," in *Proc. IEEE Int. Conf. Comput. Inf. Technol., Ubiquitous Comput. Commun., Dependable, Autonomic Secure Comput., Pervas. Intell. Comput.*, Oct. 2015, pp. 283–287, doi: 10.1109/CIT/IUCC/DASC/PICOM.2015.41.

[9] H. Jiang, Y. Zhang, E. Muljadi, J. J. Zhang, and D. W. Gao, "A short-term and high-resolution distribution system load forecasting approach using support vector regression with hybrid parameters optimization," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3341–3350, Jul. 2018, doi: 10.1109/TSG.2016.2628061.

[10] J. Shi, L. Ma, and C. Li, "Daily peak load forecasting based on sequential-parallel ensemble learning," *Proc. CSEE*, vol. 40, no. 14, pp. 4463–4472 and 4726, 2020.

[11] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794, doi: 10.1145/2939672.2939785.

[12] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on LSTM recurrent neural network," *IEEE Trans. Smart Grid*, vol. 10, no. 1, pp. 841–851, Jan. 2019, doi: 10.1109/TSG.2017.2753802.

[13] R. Jiao, T. Zhang, Y. Jiang, and H. He, "Short-term non-residential load forecasting based on multiple sequences LSTM recurrent neural network," *IEEE Access*, vol. 6, pp. 59438–59448, 2018, doi: 10.1109/ACCESS.2018.2873712.

[14] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*. [Online]. Available: http://arxiv.org/abs/1412.3555

[15] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," *Comput. Sci.*, Dec. 2014.

[16] J. Lu, Q. Zhang, and Z. Yang, "Short-term load forecasting nethod based on CNN-LSTM hybrid neural network model," *Autom. Electric Power Syst.*, vol. 43, no. 8, 2019, doi: 10.7500/AEPS20181012004.

[17] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 4580–4584, doi: 10.1109/ICASSP.2015.7178838.

[18] M. Alhussein, K. Aurangzeb, and S. I. Haider, "Hybrid CNN-LSTM model for short-term individual household load forecasting," *IEEE Access*, vol. 8, pp. 180544–180557, 2020, doi: 10.1109/ACCESS.2020.3028281.

[19] C. Yao, P. Yang, and Z. Liu, "Load forecasting method based on CNN-GRU hybrid neural network," *Power Syst. Technol.*, vol. 44, no. 9, pp. 3416–3424, 2020.

[20] J. Xu, X. Wang, and J. Yang, "Short-term load density prediction based on CNN-QR lightGBM," *Power Syst. Technol.*, vol. 44, no. 9, pp. 3409–3416, 2020.

[21] G. Ke, Q. Meng, and T. Finley, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 3146–3154.

[22] S. Bai, J. Zico Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*. [Online]. Available: http://arxiv.org/abs/1803.01271

[23] Z. Zhang, H. Chen, Y. Huang, and W.-J. Lee, "Quantile huber function guided TCN for short-term consumer-side probabilistic load forecasting," in *Proc. IEEE/IAS Ind. Commercial Power Syst. Asia*, Weihai, China, Dec. 2020, pp. 322–329, doi: 10.1109/ICPSAsia48933.2020.9208545.

[24] Y. Wang, J. Chen, X. Chen, X. Zeng, Y. Kong, S. Sun, Y. Guo, and Y. Liu, "Short-term load forecasting for industrial customers based on TCN-lightGBM," *IEEE Trans. Power Syst.*, vol. 36, no. 1, pp. 1984–1997, May 2020, doi: 10.1109/TPWRS.2020.3028133.

[25] H. P. Nguyen, J. Liu, and E. Zio, "A long-term prediction approach based on long short-term memory neural networks with automatic parameter optimization by tree-structured parzen estimator and applied to time-series data of NPP steam generators," *Appl. Soft Comput.*, vol. 89, pp. 106–116, 2020, doi: 10.1016/j.asoc.2020.106116.

[26] Y. Xia, C. Liu, Y. Li, and N. Liu, "A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring," *Expert Syst. Appl.*, vol. 78, pp. 225–241, Jul. 2017, doi: 10.1016/j.eswa.2017.02.017.

[27] 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.

[28] M. Cai, M. Pipattanasomporn, and S. Rahman, "Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques," *Appl. Energy*, vol. 236, pp. 1078–1088, Feb. 2019, doi: 10.1016/j.apenergy.2018.12.042.

[29] A. Khoshrou and E. J. Pauwels, "Short-term scenario-based probabilistic load fore-casting: A data driven approach," *Appl. Energy*, vol. 238, p. 1258–1268, Apr. 2019, doi: 10.1016/j.apenergy.2019.01.155.

[30] D. Geysen, O. De Somer, C. Johansson, J. Brage, and D. Vanhoudt, "Operational thermal load forecasting in district heating networks using machine learning and expert advice," *Energy Buildings*, vol. 162, pp. 144–153, Mar. 2018, doi: 10.1016/j.enbuild.2017.12.042.

[31] D. Liu, L. Zeng, C. Li, K. Ma, Y. Chen, and Y. Cao, "A distributed short-term load forecasting method based on local weather information," *IEEE Syst. J.*, vol. 12, no. 1, pp. 208–215, Mar. 2018, doi: 10.1109/JSYST.2016.2594208.

[32] G. J. Székely, M. L. Rizzo, and N. K. Bakirov, "Measuring and testing dependence by correlation of distances," *Ann. Statist.*, vol. 35, no. 6, pp. 2769–2794, Dec. 2007, doi: 10.1214/009053607000000505.

[33] F. Marini and B. Walczak, "Particle swarm optimization (PSO). A tutorial," *Chemometrics Intell. Lab. Syst.*, vol. 149, pp. 153–165, Dec. 2015, doi: 10.1016/j.chemolab.2015.08.020.

[34] J. Bergstra, R. Bardenet, and Y. Bengio, "Algorithms for hyperparameter optimization," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2011, pp. 2546–2554.

[35] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, vol. 1. Cambridge, MA, USA: MIT Press, pp. 326–366, 2016.

[36] J. Zhang, T. Tang, and Y. Ding, "Adabelief optimizer: Adapting stepsizes by the belief in observed gradients," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 33, 2020, pp. 1–9.

[37] Z. Zhang, X. Wang, and C. Jung, "DCSR: Dilated convolutions for single image super-resolution," *IEEE Trans. Image Process.*, vol. 28, no. 4, pp. 1625–1635, Apr. 2019, doi: 10.1109/TIP.2018.2877483.

[38] K. Zhang, M. Sun, T. X. Han, X. Yuan, L. Guo, and T. Liu, "Residual networks of residual networks: Multilevel residual networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 6, pp. 1303–1314, Jun. 2018, doi: 10.1109/TCSVT.2017.2654543.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

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

[41] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," 2016, *arXiv:1602.07868*. [Online]. Available: http://arxiv.org/abs/1602.07868

[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[43] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," 2014, *arXiv:1409.1259*. [Online]. Available: http://arxiv.org/abs/1409.1259

[44] A. Vaswani, N. Shazeer, and N. Parmar, "Attention is all you need," *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 5998–6008.

[45] J. Song and G. Xue, "Hourly heat load prediction model based on temporal convolutional neural network," *IEEE Access*, vol. 8, pp. 16726–16741, Jan. 2020, doi: 10.1109/ACCESS.2020.2968536.

[46] W. F. Set. *Weather For 5 Largest Cities in Spain*. [Online]. Available: https://www.kaggle.com/nicholasjhana/energy-consumption-generation-prices-and-weather?select=weather_features.csv

[47] D. Edelmann, T. F. Móri, and G. J. Székely, "On relationships between the pearson and the distance correlation coefficients," *Statist. Probab. Lett.*, vol. 169, Feb. 2021, Art. no. 108960, doi: 10.1016/j.spl.2020.108960.

[48] H. Pang, J. Gao, and Y. Du, "A short-term load probability density prediction based on quantile regression of time convolution network," *Power Syst. Technol.*, vol. 44, no. 4, pp. 1343–1350, 2020.

[49] T. Fang and R. Lahdelma, "Optimization of combined heat and power production with heat storage based on sliding time window method," *Appl. Energy*, vol. 162, pp. 723–732, Jan. 2016, doi: 10.1016/j.apenergy.2015.10.135.

[50] J. Li, D. Deng, J. Zhao, D. Cai, W. Hu, M. Zhang, and Q. Huang, "A novel hybrid short-term load forecasting method of smart grid using MLR and LSTM neural network," *IEEE Trans. Ind. Informat.*, vol. 17, no. 4, pp. 2443–2452, Apr. 2021, doi: 10.1109/TII.2020.3000184.

[51] A. Rahman, V. Srikumar, and A. D. Smith, "Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks," *Appl. Energy*, vol. 212, pp. 372–385, Feb. 2018, doi: 10.1016/j.apenergy.2017.12.051.

[52] L. Bottou, "Stochastic gradient descent tricks," *Neural Networks: Tricks Tradeing*. Berlin, Germany: Springer, 2012, pp. 421–436.

[53] J. Bergstra, D. Yamins, and D. Cox, "Hyperopt: A Python library for optimizing the hyperparameters of machine learning algorithms," in *Proc. 12th Python Sci. Conf.*, 2013, p. 20, doi: 10.25080/Majora-8b375195-003.
[54] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour, "Dropout improves recurrent neural networks for handwriting recognition," in *Proc. 14th Int. Conf. Frontiers Handwriting Recognit.*, Sep. 2014, pp. 285–290, doi: 10.1109/ICFHR.2014.55.

HANHONG SHI is currently pursuing the master's degree with the School of Electrical Engineering (EE), Shaanxi University of Technology. His research interest includes machine learning and its application in smart grid, especially load forecasting.

LEI WANG received the B.S. degree and M.S. degree in computer science and technology from the Xi'an University of Technology, Xi'an, China, in 1994 and 1997, respectively, and the Ph.D. degree in electronic science and technology from Xidian University, Xi'an, in 2001. He is currently a Professor with the Shaanxi Key Laboratory of Industrial Automation, Shaanxi University of Technology, Hanzhong, Shaanxi, China. His current research interests include evolutionary algorithms, neural networks, and big data.

RAFAŁ SCHERER (Member, IEEE) received the M.S. degree in electrical engineering from the Department of Electrical Engineering, Czestochowa University of Technology, and the Ph.D. degree in computer science (methods of classification using neuro-fuzzy systems) from the Department of Mechanical Engineering and Computer Science, Czestochowa University of Technology. He is currently an Associate Professor with the Institute of Computational Intelligence, Czestochowa University of Technology. He is also a co-coordinator of the Microsoft Dynamics Academic Alliance Program, Czestochowa University of Technology. He was a Principal Investigator of the Polish Ministry of Science and Higher Education Project "Computational Intelligence Methods in Data Mining" and a Researcher of the Polish-Singapore Research Project "Development of Intelligent Techniques for Modeling, Controlling and Optimizing Complex Manufacturing Systems." He has authored a book on multiple classification techniques published in Springer. He has authored more than 80 research articles. His research interests include developing new methods in computational intelligence and data mining, ensembling methods in machine learning, and content-based image indexing. He was a reviewer for major computational intelligence journals. He is also a Co-Editor of the *Journal of Artificial Intelligence and Soft Computing Research*. He co-organizes every year or two years the International Conference on Artificial Intelligence and Soft Computing in Zakopane, which is one of the major events on computational intelligence.

MARCIN WOŹNIAK received the M.Sc. degree in applied mathematics from the Silesian University of Technology, Gliwice, Poland, in 2007, and the Ph.D. and D.Sc. degrees in computational intelligence from the Czestochowa University of Technology, Czestochowa, Poland, in 2012 and 2019, respectively. He is currently an Associate Professor with the Faculty of Applied Mathematics, Silesian University of Technology. He is also a Scientific Supervisor in editions of "The Diamond Grant" and "The Best of the Best" programs for highly talented students from the Polish Ministry of Science and Higher Education. He participated in various scientific projects (as a Lead Investigator, a Scientific Investigator, the Manager, or a Participant) at Polish and Italian universities. He was a visiting researcher with universities in Italy, Sweden, and Germany. He has authored/coauthored more than 100 research articles in international conferences and journals. His current research interest includes neural networks with their applications together with various aspects of applied computational intelligence. He was an Editorial Board Member or an Editor for *Sensors*, IEEE Access, *Frontiers in Human Neuroscience*, *PeerJ CS*, the *International Journal of Distributed Sensor Networks*, *Computational Intelligence and Neuroscience*, and *Journal of Universal Computer Science*. He was the Session Chair at various international conferences and symposiums, including the IEEE Symposium Series on Computational Intelligence and the IEEE Congress on Evolutionary Computation.

PENGCHAO ZHANG received the B.Eng. degree in automation from the Shaanxi University of Technology (SNUT), Hanzhong, China, and the M.Eng. degree in traffic control engineering from Northwestern Polytechnical University (NPU), Xi'an, China, where he is currently pursuing the Doctoral degree. He is also an Associate Professor with SNUT. His current research interests include industrial robot and mobile robotics.

WEI WEI (Senior Member, IEEE) received the M.S. and Ph.D. degrees from Xi'an Jiaotong University, Xi'an, China, in 2005 and 2011, respectively. He is currently an Associate Professor with the School of Computer Science and Engineering, Xi'an University of Technology, Xi'an. He ran many funded research projects as a principal investigator and a technical member. He has published around 100 research articles in international conferences and journals. His current research interests include the area of wireless networks, wireless sensor networks application, image processing, mobile computing, distributed computing, pervasive computing, the Internet of Things, and sensor data clouds. He is a Senior Member of the China Computer Federation. He is also an Editorial Board Member of the *Future Generation Computer System*, IEEE Access, *Ad Hoc & Sensor Wireless Networks*, the *Institute of Electronics, Information and Communication Engineers*, and *KSII Transactions on Internet and Information Systems*. He is also a TPC member of many conferences and a regular Reviewer of the IEEE Transactions on Parallel and Distributed Systems, the IEEE Transactions on Image Processing, the IEEE Transactions on Mobile Computing, the IEEE Transactions on Wireless Communications, the *Journal of Network and Computer Applications*, and many other Elsevier journals.