IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 5, 2019, accepted January 16, 2019, date of publication January 29, 2019, date of current version February 14, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2895604

Deep Learning for Daily Peak Load Forecasting–A Novel Gated Recurrent Neural Network Combining Dynamic Time Warping

ZEYUAN YU, ZHEWEN NIU, WENHU TANG, (Senior Member, IEEE), AND QINGHUA WU, (Fellow, IEEE)
School of Electric Power Engineering, South China University of Technology, Guangzhou 510640, China

Corresponding author: Wenhu Tang (wenhutang@scut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 51477054, and in part by the Fundamental Research Funds for the Central Universities under Grant X2DLD2181850.

ABSTRACT Daily peak load forecasting is an essential tool for decision making in power system operation and planning. However, the daily peak load is a nonlinear, nonstationary, and volatile time series, which makes it difficult to be forecasted accurately. This paper, for the first time, proposes a bespoke gated recurrent neural network combining dynamic time warping (DTW) for accurate daily peak load forecasting. The shape-based DTW distance is used to match the most similar load curve, which can capture trends in load changes. By analyzing the relationship between the load curve and the cycle of human social activities, the some-hot encoding scheme is first applied on the discrete variables to expand the features to further characterize their impact on load curves. Then, a three-layer gated recurrent neural network is developed to forecast daily peak load. The proposed algorithm is implemented on the Theano deep learning platform and tested on the loaded dataset of the European Network on Intelligent Technologies. The simulation results show that the proposed algorithm achieves satisfactory results compared with other algorithms using the same dataset in this paper.

INDEX TERMS Daily peak load forecasting, dynamic time warping, one-hot encoding, gated recurrent unit.

I. INTRODUCTION

Load forecasting is the first phase in power system planning and controlling. Accurate load forecasting is important to utility companies for ensuring reliability and stability of power grids to meet load demands. Load forecasting is divided into very short-term load forecasting, short-term load forecasting, mid-term load forecasting and long-term load forecasting [1]. The daily peak load forecasting is one kind of mid-term load forecasting. It is an important basis for estimating the standby capacity of a power system, daily load rates, and setting the peak and valley electricity prices. Also, it has a significant impact on the operation and production cost of utilities. Therefore, load forecasting algorithms have been widely studied in the last decades. The majority of forecasting models are based on the similarity principle and various optimization algorithms, which are divided into two kinds. One is the classic forecasting algorithms and the other is the intelligent forecasting algorithms.

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

For the classic forecasting algorithms, the main advantages are their fast calculation speed and robustness. The regression analysis and exponential smoothing are simple and easy to implement [2]. However, due to the lack of load curve characteristic analysis, their forecasting accuracy is inadequate. The time series method can reflect the continuous change of load, but this method requires high smoothness of original sequences and fails to account for changes in load factors [3]. The frequency-domain component method and the wavelet analysis method can study the load from the frequency domain but fail to consider the impact of other factors on the load, such as social factors, meteorological factors [4], [5]. The selection of similar day is an effective and practical technique to forecast load. The difficulty of this method lies in how to establish an accurate similar day selection criterion [6]–[8]. There are many ways to measure the similarity between two time series. The common methods are the Euclidean (Euc) distance, the Manhattan (Manh) distance, the cosine angle (Cos) and the correlation (Cor) [9]. They can only describe the degree of similarity for two sequences as a whole, and much local information

is obscured. In order to solve this problem, the shape-based dynamic time warping (DTW) distance, proposed by Itakura in 1975, is adopted in this research to measure the similarity between two curves [10]. The DTW distance can describe the similarity between two sequences on different time scales, which has been widely used in speech recognition [11], [12].

For the intelligent forecasting algorithms, their advantages lie in screening and processing main factors affecting load curves. Artificial Neural Networks (ANN) and Support Vector Machine (SVM) have been widely used for load forecasting [7], [13]. Based on the ANN and SVM methods, Principal Component Analysis Artificial Neural Networks (PCA-ANN) [14], Least Squares Support Vector Machine (LS-SVM) [15] and Chaos-SVM [16] were proposed to improve the accuracy of load forecasting. However, these traditional neural network methods do not consider the time series characteristics of load curves, which are prone to fall into local optimum and overfitting. Hence, the accuracy of load forecasting encountered bottlenecks using traditional neural network methods.

In 2006, Professor Hinton proposed a deep belief network (DBN) [17], which marked the arrival of the deep learning era. On the basis of DBN, Stacked Auto-Encoders (SAE) [18], Convolutional Neural Network (CNN) [19] and Recurrent Neural Networks (RNN) [20] were developed. Compared with traditional neural networks, RNN introduces directional loops that can handle the contextual correlation between the inputs. Some researchers found that using RNN cannot ensure an excellent forecasting effect [21]–[24]. The data pooling technology was then proposed to overcome the problem of overfitting in deep learning and achieved good results in household load forecasting [21]. The long short-term memory (LSTM) is an improvement of RNN, which can tackle the problem of gradient vanishing and gradient explosion [22]. Different long short-term memory-based deep learning forecasting frameworks were proposed to forecast residential load trends [23], [24]. Compared with LSTM, the gated recurrent unit (GRU) has fewer parameters and is easier to converge [25]. Deep learning has achieved many breakthroughs in tackling other sophisticated problems and becomes one of the most promising techniques in the data science community, e.g Alpha Zero [26], face recognition [27], speech recognition [28] and image reconstruction [29], etc.

In order to overcome the shortcomings of the traditional forecasting algorithms, this paper proposes a novel forecasting algorithm based on gated recurrent neural network and dynamic time warping. Firstly, the autocorrelation coefficient is applied to determine the length of the daily peak load curve segment. Then considering the characteristics of load curves, a DTW method is employed to match the most similar load curve. Different from traditional measuring distance methods as mentioned, the DTW distance can capture not only the trend of load curve changes but also local information of load curves. Moreover, for the first time the some-hot encoding scheme is applied on the calendar information to expand forecasting features, which can further characterize their impact on load forecasting. Finally, the gated recurrent neural network is used to forecast the daily peak load, because it can handle the temporal dynamic behavior of a time series and needs fewer parameters.

The rest of the paper is organized as follows: Section II briefly introduces the dynamic time warping distance and the gated recurrent neural network employed in this study. Section III proposes the DTW-GRU algorithm and describes the one-hot encoding scheme to reflect the factors affecting the load. Section IV explains data sources, data analysis, hardware and software platforms, and experiment setup. In Section V, results are demonstrated through comparisons with other common distance methods (Euc, Manh, Cos, Cor), other encoding schemes (all-hot and natural) and other 10 algorithms proposed by previous researchers. Conclusions are drawn in Section VI.

II. INTRODUCTION OF THE DTW-GRU ALGORITHM

A. DYNAMIC TIME WARPING

Dynamic time warping (DTW) is a method for calculating the optimal mapping between two time series, which employs dynamic programming to represent the similarity between two series [10].

Suppose there are two one-dimensional time series, i.e. $x(i),\ i = 1, 2, \cdots, m$ and $y(j),\ j = 1, 2, \cdots, n$. In order to calculate the DTW distance between the two sequences, a distance matrix with dimensional of $m \times n$ should be calculated firstly. Its (i, j) element is denoted as $d(i, j) = (x(i) - y(j))^2$. $d(i, j)$ is called as the local distance, which is the distance between two time points in two time series. When the Euclidean distance is used to calculate the distance between two one-dimensional time series, each pair of corresponding distances at the same time stamp is summed up, and the distance between every two points is the local distance. For the DTW distance, the local distance is no longer the distance between the same two time stamps, and it can be the distance between any two time stamps.

Define the warping path W to represent an alignment or mapping of the sequences x and y.

$$W = (w_k\,(i, j)), \quad k = 1, 2, \cdots, p \tag{1}$$

where $w_k\,(i, j)$ represents that the step k of the element i in the sequence x is mapped with the element j in the sequence y. p represents the length of the warping path W and satisfies $p \in [\max(m, n), m + n - 1]$. For the DTW distance, the same time stamps of two sequences do not necessarily correspond to each other, so a shape-based correspondence needs to be found through the dynamic programming and as a result, the points in the two sequences reflecting the approximate states correspond to each other. For example, a local high point should correspond to a local high point and a local low point should correspond to a local low point, also a point in an upward trend cannot correspond to a point in a downward trend. As shown in Fig. 1, this shape-based correspondence is also called the warping path, that is, a mapping between two sequences.

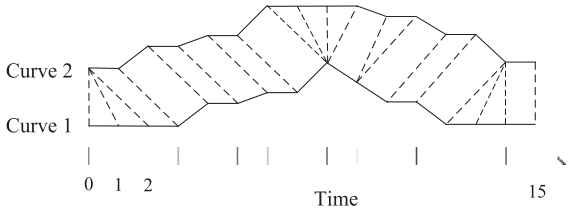

FIGURE 1. The DTW distance of two time series.

The warping path W must satisfy the following three constraints:

1. Boundary conditions

The warping path W must start at $w_1(1, 1)$ and end at $w_p(m, n)$. That is, the selected warping path must start at the bottom left and finish at the top right.

2. Continuity

Adjacent elements $w_{k-1}(a, b)$ and $w_k(a', b')$ of the warping path W must satisfy $a' - a \leq 1$ and $b' - b \leq 1$. That is to say, the point at a certain time can only match the point at the same time and the adjacent moment, and cannot cross the match.

3. Monotonicity

Adjacent elements $w_{k-1}(a, b)$ and $w_k(a', b')$ of the warping path W must satisfy $a' - a \geq 0$ and $b' - b \geq 0$. This makes the mapping in the path monotonically over time, ensuring that there is no crossover of mapping lines between the two sequences.

Obviously there are many warping paths to meet the above three constraints. However, the dynamic time warping needs to find the optimal warping path among them and its goal is the shortest distance. Use DTW(x, y) to represent the shortest distance between time series x and y, i.e. the distance corresponding to the optimal warping path in all the possible warping paths W. The shortest DTW(x, y) distance and the optimal warping path solution is a dynamic programming problem that satisfies the above three constraints:

$$\begin{cases} DTW(x, y) = \min\{r(m, n)\} \\ r(i, j) = d(i, j) + \min\{r(i-1, j-1), r(i-1, j), r(i, j-1)\} \\ r(1, 1) = d(1, 1) \\ r(i, 0) = r(0, j) = 0 \end{cases} \tag{2}$$

where $r(i, j)$ represents the cumulative distance of the local distances from $(1, 1)$ to (i, j) and $r(1, 1)$ represents the initial distance, which equals $d(1, 1)$ in the distance matrix. According to the constraints of continuity and monotonicity, the point (i, j) must start from $(i-1, j-1)$, $(i-1, j)$, or $(i, j-1)$. $\min\{r(i-1, j-1), r(i-1, j), r(i, j-1)\}$ means selecting a point with the smallest accumulated distance among the three points as the starting point. Fig. 1 illustrates the DTW mapping and warping paths for the sequences x and y. Because $r(i, 0)$ and $r(0, j)$ do not exist in practice, their values are defined as 0 for easy calculation. However, calculating the shortest DTW distance is an $O(N^2)$ time and space complexity problem. In other words, if the lengths of the two time series grow linearly, then the time and space needed to calculate the shortest DTW distance grow quadratically, that limits the DTW usefulness to small time series [30].

B. GATED RECURRENT NEURAL NETWORKS

In a traditional neural network model, the input data is fed from an input layer, calculated through one hidden layer or more, and finally output from an output layer. All layers are fully connected, but each node in each layer is not connected. Therefore, a traditional neural network can only characterize the relationship between an input and an output. However, many problems possess time series features. For example, you need to forecast words in a sentence. The last word in the sentence relies on the previous word and the words before, because there is a relationship between all the words in a sentence. Compared with the traditional feedforward neural network (FNN), the recurrent neural network (RNN) increases its storage structure to handle the time-to-time relationship between the input data. Its concrete manifestation is that the current output value in RNN depends on the input value and some previous output values stored. That is to say, the input of the hidden layer includes the input value of the current moment and the output value of the previous moment. A typical RNN is shown in Fig. 2.

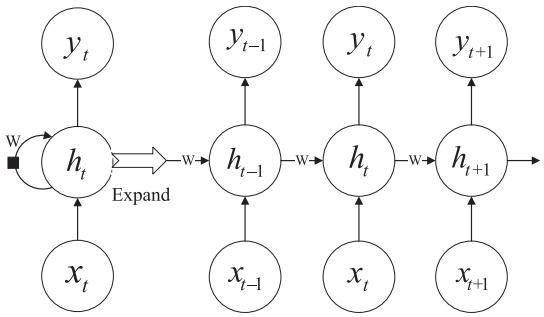

FIGURE 2. A typical recurrent neural network model.

A typical RNN model includes input units, hidden units and output units. The input data set is marked as $\{x_0, x_1, \cdots, x_t, \cdots\}$. The output data set is marked as $\{y_0, y_1, \cdots, y_t, \cdots\}$. The output data set in the hidden cells is marked as $\{h_0, h_1, \cdots, h_t, \cdots\}$. The units in the hidden cells are the main computing units of the RNN. The black square in Fig. 2 denotes the time delay. Fig. 2 shows that there is a flow of information that flows unidirectionally from an input cell to a hidden cell, meanwhile there is another flow of information that flows unidirectionally from a hidden cell to an output cell. In particular, there is a flow of information that flows unidirectionally from a previous hidden cell to a next hidden cell, which means the result of a previous hidden cell is part of a next hidden cell input.

Because of this specific structure, RNN has the advantage in dealing with the problems of time series. However, when the step size between two inputs is too large, gradient

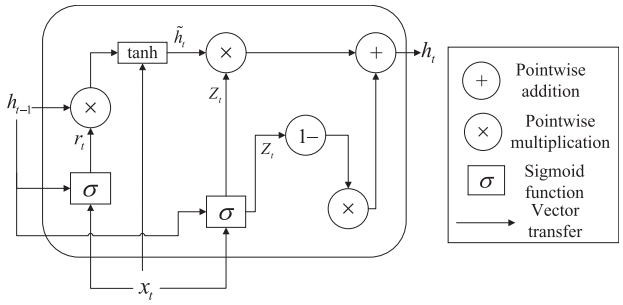

FIGURE 3. The inner structure of a gated recurrent unit.

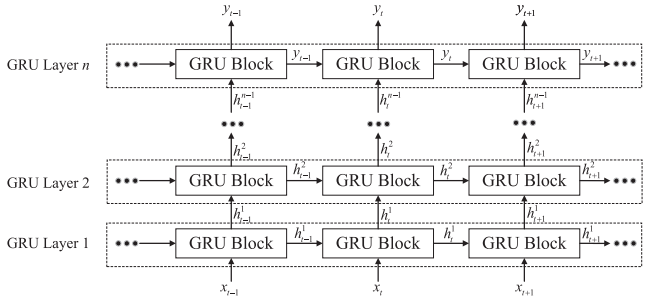

FIGURE 4. The architecture of a multi-layer gated recurrent neural network.

disappearance or explosion occurs, making the RNN difficult to be trained properly [22]. As a variant of RNN, the long short-term memory (LSTM) neural network can effectively solve this problem. Due to the complexity of the LSTM network structure, its network training time may be long. The gated recurrent neural network improves the structure of LSTM neural network by optimizing the number of gates. Its structure is shown in Fig. 3.

In Fig. 3, x_t and h_t represent the input and the output of a gated recurrent unit (GRU) at the current time t respectively, and h_{t-1} is the state at the previous moment of the current time t. Unlike other neural units, r_t and z_t are key structures in a GRU [25], which are called reset gates and update gates respectively. They are both a simple neural network in order to make the output fixed between 0 to 1. The activation function of the neural network uses the sigmoid function. $\tilde{h}_t$ is the value of the output candidate processed by the reset gate. The detailed calculation process is listed in equations (3)-(6).

$$r_t = \sigma(W_{rh}h_{t-1} + W_{rx}x_t) \tag{3}$$
$$z_t = \sigma(W_{zh}h_{t-1} + W_{zx}x_t) \tag{4}$$
$$\tilde{h}_t = \tanh(W_{hh}(r_t \circ h_{t-1}) + W_{hx}x_t) \tag{5}$$
$$h_t = (1 - z_t) \circ \tilde{h}_t + z_t \circ h_{t-1} \tag{6}$$

where W_{rh} represents the connection weight between r_t and h_{t-1}, W_{rx} the connection weight between r_t and x_t, W_{zh} the connection weight between z_t and h_{t-1}, W_{zx} the connection weight between z_t and x_t, W_{hh} the connection weight between h_t and h_{t-1} and W_{hx} the connection weight between h_t and x_t. The operator $\circ$ denotes the multiplication of array elements in turn and σ represents the sigmoid function.

As can be seen from the above equation, the reset gate r_t and the update gate z_t are obtained by using the sigmoid function to calculate the linear combination of the output h_{t-1} and the input x_t. However, their role is different. The reset gate r_t determines how much h_{t-1} information is retained. The closer the r_t value is to 1, the more h_{t-1} information is retained in $\tilde{h}_t$. The update gate z_t determines how much h_{t-1} information is discarded. The smaller z_t represents the more h_{t-1} information is discarded. In particular, when $r_t = 1$ and $z_t = 0$, the gated recurrent neural network degenerates into a traditional RNN.

Due to the sequential nature of the output of a GRU layer, an arbitrary number of GRU layers can be stacked to form a multi-layer gated recurrent neural network. The architecture of the gated recurrent neural network is shown in Fig. 4. The GRU block represents a gate recurrent unit and the sign of "$\rightarrow$" represents the flow of data. For the multi-layer gated recurrent neural network, the input data set is marked as $\{x_0, x_1, \cdots, x_t, \cdots\}$ and the output data set is marked as $\{y_0, y_1, \cdots, y_t, \cdots\}$. For the GRU layer k, the input data set is marked as $\{h_0^{k-1}, h_1^{k-1}, \cdots, h_t^{k-1}, \cdots\}$ and the output data set is marked as $\{h_0^k, h_1^k, \cdots, h_t^k, \cdots\}$, i.e. the output of the previous layer is the input of the next layer. In particular, the input data set of GRU layer 1 is $\{x_0, x_1, \cdots, x_t, \cdots\}$ and the output data set of GRU layer n is $\{y_0, y_1, \cdots, y_t, \cdots\}$.

III. IMPLEMENTATION STEPS OF LOAD FORECASTING

In this section, a new DTW-GRU algorithm is developed for daily peak load forecasting. The details of this methodology are illustrated in Fig. 5.

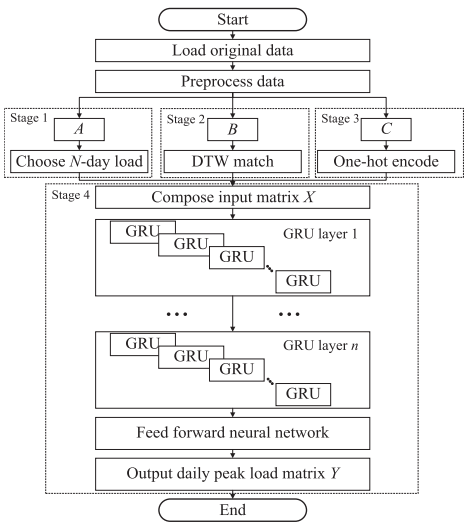

FIGURE 5. The flowchart of the proposed DTW-GRU algorithm.

In general, the proposed algorithm consists of four stages: 1) Choose N-day daily peak load; 2) Match load curve using DTW distance; 3) Encode related influencing factors; 4) Forecast daily peak load with gated recurrent

neural network. The detailed rationale and design of each stage are further discussed as follows.

A. STAGE 1: CHOOSE N-DAY DAILY PEAK LOAD

In the first stage, the length of the daily peak load curve segment is determined by the autocorrelation coefficient. The autocorrelation coefficient is used to describe the degree of correlation of the data itself in different periods, that is, to measure the impact of historical data on the present. For the time series $\{x_t\}$, the correlation coefficient between x_t and x_{t-l} is called the autocorrelation coefficient with an interval l of x_t. The autocorrelation coefficient is denoted as ρ_l and the calculation formula is shown as follows:

$$\rho_l = \frac{Cov(x_t, x_{t-1})}{\sqrt{Var(x_t)Var(x_{t-l})}}, \quad l = 1, 2, \cdots \tag{7}$$

where $Cov(x_t, x_{t-l})$ is the autocovariance, $Var(x_t)$ and $Var(x_{t-l})$ are the variances [31].

The larger the autocorrelation coefficient, the greater the impact of historical data on the present. For the autocorrelation coefficient series $\{\rho_l\}$, its maximum is selected as ρ_N. Then the optimal time interval N is the subscript value of ρ_N. Based on this, the daily peak load of N days are adopted in the fourth stage.

B. STAGE 2: MATCH LOAD CURVE USING DTW DISTANCE

In the second stage, the most similar daily peak load is obtained. Considering the certain regularity and periodicity of power system load, this research uses the method of similarity matching of daily peak load with the smallest DTW distance. Firstly, the original data set is divided into the historical data set and the forecasting data set. Then each segment of the original data set is matched with the historical data set to obtain the most similar segment. Based on the rule of thumb that if the two segments are similar then the corresponding next segments are similar [8], the most similar daily peak load is obtained.

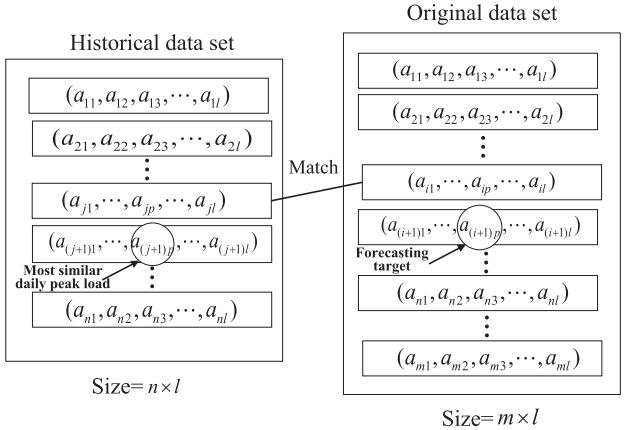

FIGURE 6. The most similar load curve matching.

The following is a detailed description of obtaining the most similar daily peak load. As shown in Fig. 6, there are n segments in the historical data set and m segments in the original data set ($0 < n < m$). Each segment is a $1 \times l$ vector, where l is the length of each segment and its value is obtained in stage 1. For any segment element in the original data set (except the first segment), such as the p^{th} element in the $(i+1)^{th}$ segment is defined as the forecasting target ($i, p > 0$). The DTW matching method is applied to find the most similar segment of the i^{th} segment in the historical data set. Assumed the smallest DTW distance of the i^{th} segment is the j^{th} segment in the historical data set ($i \neq j, j > 0$). Then the p^{th} element of the $(j+1)^{th}$ segment in the historical data set is the most similar daily peak load, which is used in the fourth stage.

C. STAGE 3: ENCODE RELATED INFLUENCING FACTORS

In the third stage, related influencing factors are encoded. As known, there are many factors that affect the daily peak load, which mainly includes two aspects. One type is meteorological factors, such as temperature and precipitation. The other type is social factors, such as the calendar information.

For continuous variables, such as meteorological factors, this research normalizes them and then the natural coding method is used to encode them. The normalized formula is shown as follows,

$$x_c = \frac{x_o - x_{\min}}{x_{\max} - x_{\min}} \tag{8}$$

where x_c is the result of standardization, x_o is the original input data, x_{max} and x_{min} are the maximum and minimum value of the original data respectively.

For discrete variables, this research applies the one-hot encoding scheme [32]. One-hot encoding uses N-bit status registers to encode N states. Each state has its own independent register bit, and at any time, only one of them is valid. That is, for any state, only one bit is one, and the others are zero. The encoding results are shown in Fig. 7.

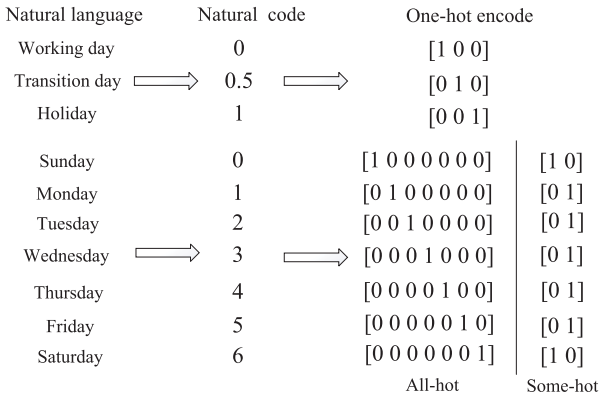

FIGURE 7. Different encoding schemes for calendar information.

As can be deduced from Fig. 7, this encoding scheme defines two attributes for each day. The first attribute is whether it is a holiday. Three conditions are defined for each day. The first is the holiday. The second is the days before and

after holidays, and it is named as the transition day. The third is the working day. The working day's natural encoding result is 0, and its one-hot encoding result is [1 0 0]. The transition day's natural encoding result is 0.5, and its one-hot encoding result is [0 1 0]. The holiday's natural encoding result is 1, and its one-hot encoding result is [0 0 1]. Another attribute is adopted to mark which day of the week it belongs. For Sunday to Saturday, their natural code is 0-6. For their one-hot encoding, this research firstly defines two encoding schemes, which are named as the some-hot encoding scheme and the all-hot encoding scheme. For the all-hot encoding scheme, a 7-bit encoding scheme is used to distinguish seven days in a week. For the some-hot encoding scheme, only a 2-bit encoding scheme is used to distinguish between weekdays and weekends, which can save computing resources and distinguish its essential features.

D. STAGE 4: FORECAST DAILY PEAK LOAD WITH GATED RECURRENT NEURAL NETWORK

The fourth stage employs the gated recurrent neural network to forecast the daily peak load. It contains the following 4 steps: 1) Data preprocessing; 2) Model construction; 3) Model training and testing; 4) Result evaluation.

1) DATA PREPROCESSING

The first step of the load forecasting using the deep learning model is to prepare the data in an appropriate format. In this step, the training set and the test set are built. For each element of the training set and the test set, there are two parts, i.e. matrix X and matrix Y. The matrix X is the input of the gated recurrent neural network and the matrix Y is the output of the gated recurrent neural network. The matrix X is composed of three parts [A B C], where A, B, C are sub-matrixes obtained from the above three stages. The submatrix A is the output of the first stage, which is the daily peak load of N days before the target forecasting day. The submatrix B is the output of the second stage, which is the most similar daily peak load obtained by the DTW matching method. The submatrix C is the output of the third stage, which is the encoding results of the related influencing factors, including the meteorological data and the calendar data. The matrix Y is the daily peak load that needs to be forecasted.

2) MODEL CONSTRUCTION

The second step of the load forecasting is to choose an appropriate deep learning model. Because of the unique reset gate and update gate structure of GRU, it can make the length of the input data changeable. For deep learning models, previous research indicates that the performance of the networks is relatively insensitive to any combination of the layer number and layer size [33]. According to the consistent finding in [34], multiple layers work better than a single layer and the number of hidden nodes should be sufficiently large. In this research, the model structure contains a 3-layer gated recurrent neural network and a 1-layer feed forward neural network.

3) MODEL TRAINING AND TESTING

After the model is constructed, the pre-processed training set is used to train the model. The test set is then employed to test the model. The relevant parameters of the model are detailed in Section IV.

4) RESULT EVALUATION

The results from step 3 are evaluated. Evaluation indicators and results are presented in Section V.

IV. IMPLEMENTATION OF DTW-GRU ALGORITHM

This section introduces the implementation of the proposed algorithm, including data sources, data analysis, hardware and software platforms, and experiment setup.

A. DATA SOURCES

In 2001, the European Network on Intelligent Technologies (EUNITE) organized a load forecasting competition. The organizer of the competition provided the following important information: 1) Electricity load data every 30 mins from January 1997 to January 1999; 2) Daily average temperatures from January 1997 to January 1999; 3) Dates of holidays from 1997 to 1999. The task of the competition was to forecast the daily peak load in January 1999. The task of competitors was to supply the prediction of daily peak load for January 1999. The daily peak load from 1997 to 1998 is shown in Fig. 8 and the daily average temperature from 1997 to 1998 is shown in Fig. 9.

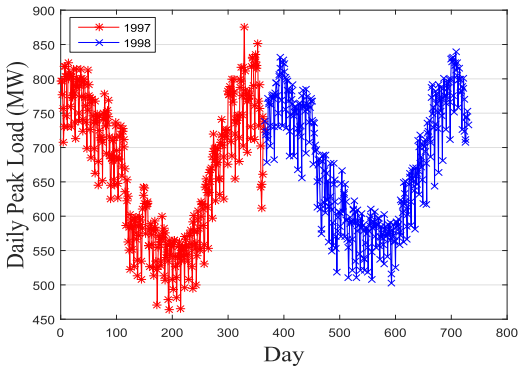

FIGURE 8. Daily peak load from 1997 to 1998.

In this study, the original data containing the daily peak load of 761 days are used. The original data are divided into 109 weeks. Since there are 763 days in the 109 weeks, which are more than 761 days, 0 is used to fill the gap. The first 104 weeks are divided as the historical data set and the last 5 weeks as the forecasting data set.

B. DATA ANALYSIS

1) PRELIMINARY ANALYSIS

Power load is a stochastic process. For a stochastic process, the autocorrelation coefficient is used to determine its appropriate model order. In this study, the autocorrelation

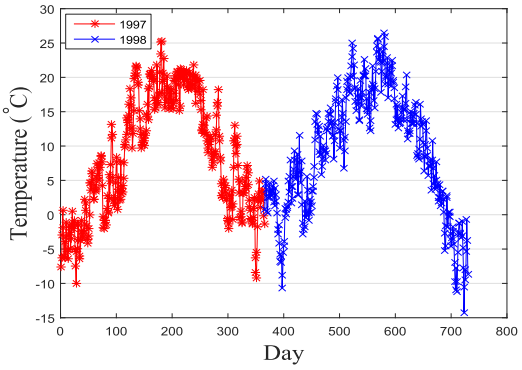

FIGURE 9. Daily average temperature from 1997 to 1998.

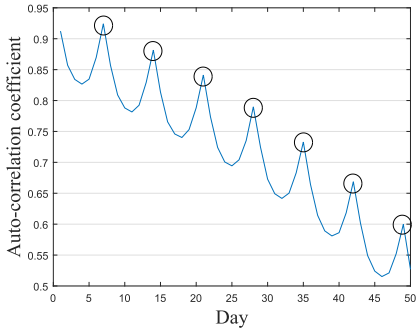

FIGURE 10. Autocorrelation coefficient of daily peak load.

coefficient is calculated by different number of lag days. The result is shown in Fig. 10. As can be deduced from Fig. 10, the maximum of the autocorrelation coefficient is a period of 7 days, which is consistent with the cycle of human social activities for 7 days a week. Hence, a multiple of 7 days is suitable for a load segment.

2) DETAILED ANALYSIS

Calendar information: As can be observed from Fig. 8, the load at weekends (Saturday and Sunday) is usually lower than that of weekdays (Monday to Friday). Furthermore, combining the holiday situation in the area [13], the holidays also have an influence on the load. The load in holidays is usually lower than that of non-holidays. Therefore, it is necessary to use the calendar information, including the weekends and holidays to forecast the daily peak load.

Temperature information: In the dataset, the only climate information provided is the average daily temperature. Comparing Fig. 8 with Fig. 9, the higher temperature corresponds to the lower load. A negative correlation between the load and the daily average temperature is observed, which is calculated as -0.868. Hence, the daily average temperature is an important attribute for developing a forecasting model.

C. HARDWARE AND SOFTWARE PLATFORMS

The program is implemented on a high-performance Sugon workstation equipped with the Ubuntu 14.04 operating system and 2 NVIDIA GTX 1080 units. This deep learning process uses the Keras library [35] with the Theano backend [36] to build. Based on the Theano platform, the Compute Unified Device Architecture (CUDA) [37] is applied to implement GPU acceleration for training gated recurrent neural networks. The programming language used in this research is Python, which is the most popular programming language in data science.

D. EXPERIMENT SETUP

This part presents the details of the algorithm configuration for setting up experiments. In order to obtain the optimal performance of the DTW-GRU algorithm, multiple settings of the DTW-GRU algorithm have been attempted. However, not all results are reported in the result section, and the comparison is made with the results of the optimal settings of the DTW-GRU algorithm and the algorithms reported by other researchers. The parameters of input vector and output vector are introduced in Section III. The training configuration parameters of training batch size, training method, learning rate, training stop strategy and loss function are detailed discussed in [38]. In summary, all the experiment settings and parameters of the DTW-GRU algorithm are presented as follows:

- Input vector $\in \{10, 13, 17, 18, 20, 25\}$
- Output vector $\in \{1\}$
- Hidden neuron number $\in \{200, 300, 500, 800\}$
- Training batch size $\in \{10, 20, 30, 50\}$
- Training method $\in \{AdamOptimizer\}$
- Learning rate $\in \{0.1, 0.05, 0.02, 0.01\}$
- Training stop strategy $\in \{earlystopping\}$
- Loss function $\in \{RMSE\}$

V. RESULT AND DISCUSSION

In this section, the daily peak load forecasting results compared between the DTW-GRU algorithm and ten reported algorithms, as well as the matching results compared between the DTW distance and four popular distance types, are presented using the dataset of EUNITE.

A. COMPARE MATCHING RESULTS

This section gives the matching results using the DTW distance, the Euclidean distance, the Manhattan distance, the Cosine distance and the Correlation coefficient to measure the similarity of load curves. 108 weeks of the original data set (except the last week) are matched with the 103 weeks of the historical data set (except the last week). The matching results of the 104^{th} -108^{th} weeks in the original data set are shown in Fig. 11. Furthermore, the most similar daily peak load is used in the input of the test data set. As seen from Fig. 11, the DTW matching results are somewhat different due to the characteristics of the different distances. It does not focus on the one-to-one correspondence of each point, but pays more attention to the large change tendency, which is effective in solving such a nonstationary problem.

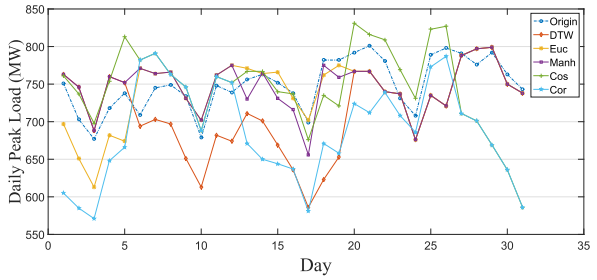

FIGURE 11. Matching results for different distances.

B. COMPARE FORECASTING RESULTS

This section gives a comparison of load forecasting results with other reported algorithms in two aspects, i.e. the Kupiec test and the typical error indicators.

Firstly, the Kupiec test is applied to verify the model performance. The basic idea of the Kupiec test is to calculate the statistical value corresponding to the failure rate during a test period to determine whether the model is valid. The forecasting error is defined as follows:

$$forecasting\ error = \left|\frac{\hat{y}_t - y_t}{y_t}\right| \times 100\% \tag{9}$$

where $\hat{y}_t$ is the forecasting value and y_t is the actual value.

The failure days are the days in which the forecasting error exceeds a given error. Assume that the period is a total of P days, where Q days fail. Then the failure rate f is calculated by $f = Q/P$. Also, assume the time is independent and the number of failure days Q follows the binomial distribution $B(P, f)$. For the significance level α, the statistic likelihood ratio (LR) follows the chi-square distribution with 1 degree of freedom [39]:

$$LR = -2\ln\left[(1-\alpha)^{P-Q}\alpha^{Q}\right] + 2\ln\left\{\left[(1-f)^{P-Q}f^{Q}\right]\right\} \tag{10}$$

Under the $\alpha = 5\%$ level of significance, if the LR value is greater than 5.02 [40], the model is rejected.

Secondly, to assess the performance of the proposed method in conducting load forecasting, three widely used indicators are employed, including the mean absolute percent error (MAPE), the maximal error (ME) and the root mean squared error (RMSE), as defined in equations (11)-(13).

$$MAPE = \frac{1}{N}\sum_{t=1}^{N}\left|\frac{\hat{y}_t - y_t}{y_t}\right| \times 100\% \tag{11}$$

$$ME = \max\left|\hat{y}_t - y_t\right| \quad where\ t = 1, 2, \cdots \mathrm{N} \tag{12}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(\hat{y}_t - y_t)^2} \tag{13}$$

where $\hat{y}_t$ is the forecasting value, y_t is the actual value and N refers to the forecasting set size.

Specifically, the results of daily peak load forecasting are compared in three aspects: different distances, different encoding schemes and different algorithm structures.

TABLE 1. Kupiec test results between the DTW distance and other different distance types.

Distance Type	1.50%	1.75%	2.00%	2.50%	3.00%
DTW	11.28	1.13(✓)	1.13(✓)	0.13(✓)	0.23(✓)
Euc	14.89	2.89(✓)	2.89(✓)	0.23(✓)	0.23(✓)
Manh	27.63	23.08	8.05	2.89(✓)	0.23(✓)
Cos	32.47	18.83	11.28	2.89(✓)	0.13(✓)
Cor	37.57	14.89	11.28	8.05	2.89(✓)

1) DIFFERENT DISTANCES

In this part, the results of different distance types are compared as follows.

The LR values are shown in Table 1 and the sign of (✓) means this distance type passes the Kupiec test under the $\alpha = 5\%$ level of significance. As the error rate increases, more and more distance types pass the Kupiec test. When the threshold of the forecasting error rate is 1.75%, the *DTW* distance and the *Euc* distance pass the Kupiec test. When the threshold of the forecasting error rate is 2.50%, the *Manh* distance and the *Cos* distance pass the Kupiec test. When the threshold of the forecasting error rate is 3.00%, the *Cor* distance passes the Kupiec test. In summary, the *DTW* distance and the *Euc* distance outperform the *Manh* distance, the *Cos* distance and the *Cor* distance.

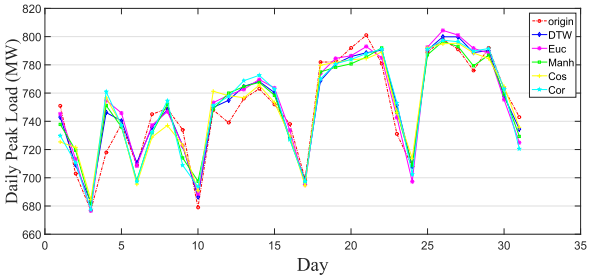

FIGURE 12. Load forecasting results applying different matching distances.

TABLE 2. Comparison between the DTW distance and other different distance types.

Distance Type	*MAPE*/%	*ME*/MW	*RMSE*/MW
DTW	**1.01**	**28.23**	**95.23**
Euc	1.15	36.86	120.74
Manh	1.24	33.19	140.24
Cos	1.31	38.63	162.13
Cor	1.42	42.91	196.73
Improvement from *Euc* to *DTW*	12.17%	23.41%	21.13%
Improvement from *Cor* to *DTW*	28.87%	34.21%	51.59%

It can be seen from Fig. 12 and Table 2, the result of using various distance matching methods is $DTW > Euc > Manh > Cos > Cor$. Compared with the Euc distance, which is the

second-best performance forecasting method, its forecasting accuracy is increased about 12.17%, 23.41% and 21.13% with respect to MAPE, ME and RMSE. Compared with the Cor distance, which is the worst performance forecasting method, its forecasting accuracy is increased about 28.87%, 34.21% and 51.59% with respect to MAPE, ME and RMSE.

Daily peak load is a nonstationary time series. The common distance calculation methods for obtaining the most similar curve focus on the shortest distance between any two points. This type of methods ignore the trend of curve changes and makes them difficult to solve the nonstationary problem of load forecasting. However, the DTW distance can capture the trend of load curve change to obtain the nonstationary information of the load curve to some extent. Also, the unique reset gate and update gate structure of GRU can store and learn the trend of load change obtained by the DTW distance matching.

2) DIFFERENT ENCODING SCHEMES

In this part, the results of different encoding schemes are compared as below.

TABLE 3. Kupiec test results between the DTW distance and other encoding schemes.

Coding Method	1.50%	1.75%	2.00%	2.50%	3.00%
Natural	14.89	5.23	2.89(✓)	0.13(✓)	0.23(✓)
Some-Hot	11.28	1.13(✓)	1.13(✓)	0.13(✓)	0.23(✓)
All-Hot	18.83	11.28	8.05	0.13(✓)	0.23(✓)

The *LR* values are shown in Table 3 and the sign of (✓) means this coding scheme passes the Kupiec test under the $\alpha = 5\%$ level of significance. When the threshold of the forecasting error rate is 1.75%, only the some-hot encoding scheme passes the Kupiec test. When the threshold of the forecasting error rate is 2.00%, the natural encoding scheme passes the Kupiec test. When the threshold of the forecasting error rate is 2.50%, the all-hot encoding scheme passes the Kupiec test. To sum up, the result of using various coding schemes is *some-hot* > *natural* > *all-hot*.

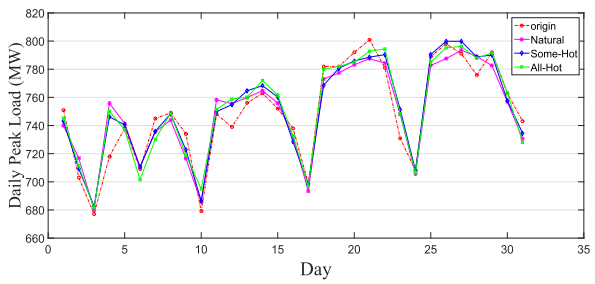

FIGURE 13. Load forecasting results applying different coding schemes.

It can be seen from Fig. 13 and Table 4 that the use of the one-hot encoding schemes, including the some-hot encoding scheme and the all-hot encoding scheme, can effectively improve the accuracy of load forecasting compared with the natural encoding scheme. The some-hot encoding scheme forecasting accuracy is increased about 15.83%, 25.51% and 26.03% with respect to MAPE, ME and RMSE. Compared with the all-hot encoding scheme, the forecasting accuracy of the some-hot encoding scheme is increased about 6.48%, 12.00% and 16.78% with respect to MAPE, ME and RMSE.

TABLE 4. Comparison between the some-hot coding and other encoding schemes.

Coding Method	*MAPE*/%	*ME*/MW	*RMSE*/MW
Natural	1.20	37.90	128.75
Some-Hot	**1.01**	**28.23**	**95.23**
All-Hot	1.08	32.08	114.43
Improvement from Natural to Some-Hot	15.83%	25.51%	26.03%
Improvement from All-Hot to Some-Hot	6.48%	12.00%	16.78%

This is due to the use of one-hot encoding scheme can play a role in expanding the features of discrete variables. The expanded features can characterize the type of load change effectively, which are conducive to solve the volatile problem of the load forecasting. However, for the two kind of one-hot encoding schemes, the forecasting results applied the all-hot encoding scheme is worse than those applied the some-hot encoding scheme. That is because the all-hot encoding scheme expands too many features that the useful features are submerged. Hence, the some-hot encoding scheme achieves the best forecasting results.

TABLE 5. Comparison between the proposed algorithm and other published algorithms.

Network Architecture	*MAPE*/%	Network Architecture	*MAPE*/%
DTW-GRU	**1.01**	GRU	1.49
Winner [13]	1.95	LS-SVM [16]	1.71
SOFNN [42]	1.61	GKPCR [43]	1.59
LW-SVR [44]	1.52	HLC [45]	1.48
GA-CVR [46]	1.43	EMD-SVR [47]	1.25
Chaos-SVM [17]	1.10	ELFFM-AFSGEP [48]	1.09

3) DIFFERENT ALGORITHM STRUCTURES

To futher demonstrate the effectiveness of the proposed DTW-GRU algorithm, the results of the load forecasting algorithms developed by other 10 researchers on the EUNITE test set are compared. Their MAPE results are directly cited from their literatures, as shown in Table 5. In Table 5, the traditional GRU algorithm is verified to be superior to the competition winner, LW-SVR, SOFNN and GKPCR, resulting in a MAPE of 1.49%. Due to the special structure of GRU, the gated recurrent neural network is capable of dealing with nonlinear time series problems. Meanwhile, the obtained results indicate that the DTW-GRU algorithm proposed by

this research outperforms the other 10 algorithms and results in a MAPE of 1.01%. This means the combination of the some-hot encoding scheme and the DTW distance matching method is highly effective for the improvement of traditional gated recurrent neural networks.

VI. CONCLUSION

This research combines the DTW distance with gated recurrent neural networks to forecast the daily peak load for the first time. The DTW distance can capture the trend of load changes effectively and the some-hot encoding scheme can expand the features of discrete variables. Using them as the extra input to the gated recurrent neural network make predictions more accurate. The simulation results on the EUNITE dataset show that the DTW-GRU algorithm can significately improve the forecasting accuracy of the daily peak load, which is superior to the other reported forecasting algorithms in this study.

REFERENCES

[1] D. Ali, M. Yohanna, M. I. Puwu, and B. M. Garkida, "Long-term load forecast modelling using a fuzzy logic approach," *Pacific Sci. Rev. A, Natural Sci. Eng.*, vol. 18, no. 2, pp. 123–127, 2016.

[2] 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.

[3] M. Chaouch, "Clustering-based improvement of nonparametric functional time series forecasting: Application to intra-day household-level load curves," *IEEE Trans. Smart Grid*, vol. 5, no. 1, pp. 411–419, Jan. 2014.

[4] B. Li, J. Zhang, Y. He, and Y. Wang, "Short-term load-forecasting method based on wavelet decomposition with second-order gray neural network model combined with ADF test," *IEEE Access*, vol. 5, pp. 16324–16331, 2017.

[5] X. Zhang, J. Wang, and K. Zhang, "Short-term electric load forecasting based on singular spectrum analysis and support vector machine optimized by Cuckoo search algorithm," *Electr. Power Syst. Res.*, vol. 146, pp. 270–285, May 2017

[6] T. Hong, J. Wilson, and J. Xie, "Long term probabilistic load forecasting and normalization with hourly information," *IEEE Trans. Smart Grid*, vol. 5, no. 1, pp. 456–462, Jan. 2014.

[7] F. L. Quilumba, W.-J. Lee, H. Huang, D. Y. Wang, and R. L. Szabados, "Using smart meter data to improve the accuracy of intraday load forecasting considering customer behavior similarities," *IEEE Trans. Smart Grid*, vol. 6, no. 2, pp. 911–918, Mar. 2015.

[8] Y. Chen *et al.*, "Short-term load forecasting: Similar day-based wavelet neural networks," *IEEE Trans. Power Syst.*, vol. 25, no. 1, pp. 322–330, Feb. 2010.

[9] N. F. Azam and H. L. Viktor, "Spectral clustering: An explorative study of proximity measures," in *Knowledge Discovery, Knowledge Engineering, and Knowledge Management*. Berlin, Germany: Springer-Verlag, 2013, pp. 60–78.

[10] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 23, no. 1, pp. 67–72, Feb. 1975.

[11] J.-C. Junqua, S. Valente, D. Fohr, and J.-F. Mari, "An N-best strategy, dynamic grammars and selectively trained neural networks for real-time recognition of continuously spelled names over the telephone," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Detroit, MI, USA, May 1995, pp. 852–855.

[12] A. H. Mansour, G. Z. A. Salh, and K. A. Mohammed, "Voice recognition using dynamic time warping and mel-frequency cepstral coefficients algorithms," *Int. J. Comput. Appl.*, vol. 116, no. 2, pp. 34–41, 2015.

[13] B.-J. Chen, M.-W. Chang, and C.-J. Lin, "Load forecasting using support vector machines: A study on EUNITE competition 2001," *IEEE Trans. Power Syst.*, vol. 19, no. 4, pp. 1821–1830, Nov. 2004.

[14] F. M. Bianchi, E. De Santis, A. Rizzi, and A. Sadeghian, "Short-term electric load forecasting using echo state networks and PCA decomposition," *IEEE Access*, vol. 3, pp. 1931–1943, 2015.

[15] T.-T. Chen and S.-J. Lee, "A weighted LS-SVM based learning system for time series forecasting," *Inf. Sci.*, vol. 299, pp. 99–116, Apr. 2015.

[16] H. Wu and X. Chang, "Power load forecasting with least squares support vector machines and chaos theory," in *Proc. 6th World Congr. Intell. Control Automat. (WCICA)*, Dalian, China, Jun. 2006, pp. 4369–4373.

[17] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[18] S. Lange and M. Riedmiller, "Deep auto-encoder neural networks in reinforcement learning," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Barcelona, Spain, Jul. 2010, pp. 1–8.

[19] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Paris, France, May/Jun. 2010, pp. 253–256.

[20] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Vancouver, BC, Canada, May 2013, pp. 6645–6649.

[21] H. Shi, M. Xu, and R. Li, "Deep learning for household load forecasting—A novel pooling deep RNN," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 5271–5280, Sep. 2018.

[22] H. Sak, A. Senior, and F. Beaufays. (2014). "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition." [Online]. Available: https://arxiv.org/abs/1402.1128

[23] W. Kong, Z. Y. Dong, D. J. Hill, F. Luo, and Y. Xu, "Short-term residential load forecasting based on resident behaviour learning," *IEEE Trans. Power Syst.*, vol. 33, no. 1, pp. 1087–1088, Jan. 2018.

[24] 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.

[25] K. Cho *et al.* (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation." [Online]. Available: https://arxiv.org/abs/1406.1078

[26] D. Silver *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[27] A. Polyak and L. Wolf, "Channel-level acceleration of deep face representations," *IEEE Access*, vol. 3, pp. 2163–2175, 2015.

[28] K. Noda, Y. Yamaguchi, K. Nakadai, H. G. Okuno, and T. Ogata, "Audio-visual speech recognition using deep learning," *Appl. Intell.*, vol. 42, no. 4, pp. 722–737, 2015.

[29] G. Wang, "A perspective on deep imaging," *IEEE Access*, vol. 4, pp. 8914–8924, 2016.

[30] X. Wang, F. Yu, and W. Pedrycz, "An area-based shape distance measure of time series," *Appl. Soft Comput.*, vol. 48, pp. 650–659, Nov. 2016.

[31] I. Monedero, F. Biscarri, C. León, J. I. Guerrero, J. Biscarri, and R. Millán, "Detection of frauds and other non-technical losses in a power utility using Pearson coefficient, Bayesian networks and decision trees," *Int. J. Elect. Power Energy Syst.*, vol. 34, no. 1, pp. 90–98, 2012.

[32] B. Jin, C. Che, Z. Liu, S. Zhang, X. Yin, and X. Wei, "Predicting the risk of heart failure with EHR sequential data modeling," *IEEE Access*, vol. 6, pp. 9256–9261, 2018.

[33] A.-R. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 1, pp. 14–22, Jan. 2012.

[34] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[35] K. Choi, D. Joo, and J. Kim. (2017). "Kapre: On-GPU audio preprocessing layers for a quick implementation of deep neural network models with Keras." [Online]. Available: https://arxiv.org/abs/1706.05781

[36] R. Al-Rfou *et al.* (2016). "Theano: A python framework for fast computation of mathematical expressions." [Online]. Available: https://arxiv.org/abs/1605.02688

[37] L. Christino and F. Osório, "GPU-services: GPU based real-time processing of 3D point clouds applied to robotic systems and intelligent vehicles," in *Robotics*. Uberlândia, Brazil: Springer, 2016, pp. 152–171.

[38] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?" *J. Mach. Learn. Res.*, vol. 11, pp. 625–660, Feb. 2010.

[39] P. H. Kupiec, "Techniques for verifying the accuracy of risk management models," *J. Derivatives*, vol. 3, no. 2, pp. 73–84, 1995.

[40] R. E. Walpole and R. H. Myers, *Probability and Statistics for Engineers and Scientists*. Buffalo, New York, USA: Pearson Education, 1993, p. 388.

[41] H. Mao, X.-J. Zeng, G. Leng, Y.-J. Zhai, and J. A. Keane, "Short-term and midterm load forecasting using a bilevel optimization model," *IEEE Trans. Power Syst.*, vol. 24, no. 2, pp. 1080–1090, May 2009.

[42] M.-R. Ghandour and J. Li, "Short term electric load prediction by incorporation of kernel into features extraction regression technique," *Smart Grid Renew. Energy*, vol. 8, no. 1, pp. 31–45, 2017.
[43] E. E. Elattar, J. Goulermas, and Q. H. Wu, "Electric load forecasting based on locally weighted support vector regression," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 40, no. 4, pp. 438–447, Jul. 2010.
[44] A. M. De Silva, F. Noorian, R. I. A. Davis, and P. H. W. Leong, "A hybrid feature selection and generation algorithm for electricity load prediction using grammatical evolution," in *Proc. 12th Int. Conf. Mach. Learn. Appl. (ICMLA)*, Miami, FL, USA, Dec. 2013, pp. 211–217.
[45] Y. Li, R. Ma, L. Yang, and P. Chen, "A short term load forecasting model using core vector regression optimized by memetic algorithm," *Int. J. Control Automat.*, vol. 9, no. 6, pp. 365–378, 2016.
[46] B. Bican and Y. Yaslan, "A hybrid method for time series prediction using EMD and SVR," in *Proc. 6th Int. Symp. Commun., Control Signal Process. (ISCCSP)*, Athens, Greece, May 2014, pp. 566–569.
[47] S. Deng, C. Yuan, L. Yang, and L. Zhang, "Distributed electricity load forecasting model mining based on hybrid gene expression programming and cloud computing," *Pattern Recognit. Lett.*, vol. 109, pp. 72–80, Jul. 2018.

ZEYUAN YU received the B.Sc. degree in information and computing science from North China Electric Power University, Baoding, China, in 2015. He is currently pursuing the Ph.D. degree with the South China University of Technology, China. His major research interests include deep learning applications in power systems and resilience modeling of power grids.

ZHEWEN NIU received the B.Sc. degree in electrical engineering from the Taiyuan University of Technology, Taiyuan, China, in 2016. He is currently pursuing the Ph.D. degree in electrical engineering with the South China University of Technology, China. His research interests include big data, machine learning, and deep leaning applications in power systems.

WENHU TANG (M'05–SM'13) received the B.Sc. and M.Sc. degrees in electrical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 1996 and 2000, respectively, and the Ph.D. degree in electrical engineering from The University of Liverpool, Liverpool, U.K., in 2004, where he was a Postdoctoral Research Associate and subsequently a Lecturer, from 2004 to 2013. He is currently a Distinguished Professor and the Dean of the School of Electric Power Engineering, South China University of Technology, Guangzhou, China. He has authored or co-authored more than 100 research papers, including 40 journal papers and one Springer research monograph. His research interests include power systems risk assessment, renewable energy integration in power grids, condition monitoring and fault diagnosis for power apparatus, multiple-criteria evaluation, and intelligent decision support systems. He is a Fellow of the IET.

QINGHUA WU (M'91–SM'97–F'11) received the Ph.D. degree in electrical engineering from Queen's University Belfast (QUB), Belfast, U.K., in 1987, where he was a Research Fellow and subsequently a Senior Research Fellow, from 1987 to 1991. He joined the Department of Mathematical Sciences, Loughborough University, U.K., in 1991, as a Lecturer, where subsequently he was appointed as a Senior Lecturer. In 1995, he joined The University of Liverpool, Liverpool, U.K., to take up his appointment as the Chair of electrical engineering at the Department of Electrical Engineering and Electronics. He is currently with the School of Electric Power Engineering, South China University of Technology, Guangzhou, China, as a Distinguished Professor and as the Director of the Energy Research Institute. He is a Chartered Engineer and also a Fellow of InstMC. He has authored or co-authored more than 440 technical publications, including 240 journal papers, 20 book chapters, and three research monographs published by Springer. His research interests include nonlinear adaptive control, mathematical morphology, evolutionary computation, power quality, and power system control and operation. He is a Fellow of the IET.