2017 14th International Symposium on Pervasive Systems, Algorithms and Networks & 2017 11th International Conference on Frontier of Computer Science and Technology & 2017 Third International Symposium of Creative Computing

# Everything is Image: CNN-based Short-term Electrical Load Forecasting for Smart Grid

Liangzhi Li, Kaoru Ota, Mianxiong Dong
Department of Information and Electronic Engineering
Muroran Institute of Technology, Japan
E-mail:{16096502, ota, mxdong}@mmm.muroran-it.ac.jp

***Abstract*—Electrical load forecasting is of great significance to guarantee the system stability under large disturbances, and optimize the distribution of energy resources in the smart grid. Traditional prediction models, which are mainly based on time series analyzing, have been unable to fully meet the actual needs of the power system, due to their non-negligible prediction errors. To improve the forecasting precision, we successfully transform the numerical prediction problem into an image processing task, and, based on that, utilize the state-of-the-art deep learning methods, which have been widely used in computer image area, to perform the electrical load forecasting. A novel deep learning based short-term forecasting (DLSF) method is proposed in the paper. Our method can perform accurate clustering on the input data using a deep Convolutional Neural Network (CNN) model. And ultimately, another neural network with three hidden-layers is used to predict the electric load, considering various external influencing factors, e.g. temperature, humidity, wind speed, etc. Experimental results demonstrate that the proposed DLSF method performs well in both accuracy and efficiency.**

## I. Introduction

In a power system, the electricity generation, transportation and distribution must be performed simultaneously. Therefore, the power generation should always be consistent with the actual electrical load, which not only benefits the power system but reduces the energy waste. Therefore, the major problem is how to obtain a precise prediction regarding the future power load. As a power system for the future [1]–[4], the smart grid gives out a practicable solution, i.e. record real-time electrical data and utilize them to forecast future load value. However, it is very difficult to conduct an accurate load forecasting, due to the numerous influence factors. To address this problem, we focus on the short-term load forecasting task and implement an effective forecasting method in the paper.

Short-term load forecasting is of great significance for the dynamic equilibrium of the power system. And there are mainly two categories of the short-term load forecasting methods, i.e., the traditional methods and intelligent methods [5]. The traditional methods include regression analysis methods, timing analysis methods and exponential smoothing methods. Regression analysis methods [6] seek the relationship between independent and dependent variables, according to the variation patterns of historical data and the influence factors of load changes. Based on that, they can determine the regression equation and estimate the power load at some point in the future. Regression analysis methods have a fast calculation speed due to its simple mechanism. However, these methods cannot describe the effects of various impact factors, which could make a significant impact on the forecasting process, and also can be easily interfered by the outlier data, resulting in low prediction accuracy. Timing analysis methods [7] regard the load data as a periodically-varying time sequence, and analyze the difference between actual load and estimated load as a smooth stochastic process. Since these methods only need a small number of historical data, they are simple, convenient and easy to calculate. Nonetheless, it is very difficult for these methods to completely eliminate the influences caused by environmental factors, and as a result, the prediction errors increase with the time distance. The main principle of exponential smoothing methods is a weighted average, which reflects the impact of the historical data on the future electrical load. Smoothing effect is used to eliminate random fluctuation in the forecasting process. However, the influence of external factors, such as economy and weather, is not considered in these methods.

The intelligent forecasting methods have been emerged along with the progress and development of artificial intelligence technology. In recent years, the load forecasting methods using the intelligent methods such as expert system, wavelet transform [8], neural network, fuzzy set theory [9], Artificial Neural Networks (ANN) and support vector machine (SVM) [10] have attracted much more attentions than the traditional methods. The expert system is a computer system based on knowledge and rules, but has no autonomous learning ability. The wavelet transform is a time-frequency domain analysis method. By transforming load data to time domain signals, this method can obtain several signals, derived from the load sequences, in different frequency band. However, it cannot analyze the influence factors. The fuzzy set theory [11] can be used to describe the fuzzy factors related with load forecasting, e.g. weather and date, etc., which is able to handle the uncertainty of load change. SVM is built on the statistic learning theory, and utilizes an algorithm of optimization theory. SVM can improve the convergence of genetic algorithms and gets optimal solution with high speed. However, with a large sample size and high dimensional feature data, SVM does not work well. Although the above intelligent methods have made certain progress in load forecasting area, manpower is still required to set the rules and features, which have direct impact on the final forecast result.

With the huge success of the deep learning methods, es-

2375-527X/17 $31.00 © 2017 IEEE
DOI 10.1109/ISPAN-FCST-ISCC.2017.78

pecially the Convolutional Neural Network (CNN) for image processing area, we attempt to adopt CNN models in the load forecasting task in order to improve the prediction precision. And we successfully find a solution to transform the numerical prediction problem into an image processing task. Based on that, we utilize several state-of-the-art deep learning methods, which have been widely used in computer image area, to perform the electrical load forecasting. A novel deep learning based short-term forecasting (DLSF) method is proposed in the paper. Our method can perform accurate clustering on the input data using a deep CNN model. And ultimately, another neural network with three hidden-layers is used to predict the electric load, considering various external influencing factors, e.g. temperature, humidity, wind speed, etc. Experimental results demonstrate that the proposed DLSF method performs well in both accuracy and efficiency.

The rest of the paper is organized as follows. Section II gives a brief introduction regarding the deep learning networks. Section III introduces the core algorithm of the proposed load forecasting method, and gives out its implementations in detail. Experimental evaluation is shown in Section IV. Finally, conclusions are drawn in Section V.

## II. Deep Learning Networks

One huge challenge in the electrical load forecasting problem is that, the short-time electrical load can be influenced by numerous factors, e.g. environmental factors, economic factors and geographical factors. As we know, a simple three-layered neural network with only one hidden layer can approximate any continuous nonlinear functions with any precision. Therefore, a deep neural network having multiple hidden layers has a strong ability in modeling the load forecasting problem under complicated conditions. Deep learning [12] is one of the deep neural networks. It belongs to unsupervised learning methods, which are able to determine which features are essential and which features can be omitted without manpower. As a result, it has a satisfactory robustness and an excellent generalization ability. Layer-by-layer data mapping and feature extraction are adopted in deep learning methods to improve the precision of forecasting and classification, which brings huge success in various research fields, e.g. pattern recognition, classification, clustering, dimensionality reduction, forecasting, recommendation, and information retrieval [13].

Traditional neural networks learn new patterns by calculating and tuning the weights. The algorithms for adjusting weights in shallow networks is usually very simple and limited in learning ability. Hinton and David Rumelhart proposed the back-propagation algorithm, adding a hidden layer in neural networks, which addressed the representation problem of perceptron and achieved much better performance. However, with the increase of network layers, deep networks frequently encounter overfitting problems, which result in poor generalization ability and bad performance, even compared with some shallow networks. Moreover, the vanishing gradient is also a severe problem in deep networks [14]. When sigmoid is used as the activation function, the back-propagation gradient decays to one quarter of original value with every propagation between two layers. Therefore, the weights near input end receive little back-propagated training signals. Neural networks with multiple hidden-layers are easily trapped in local optima, instead of global optima. And with the increase of network layers, non-convex objective function become progressively complex and local optima increase exponentially, which makes it extremely difficult to train deep networks [14]. In contrast, SVM, based on structural risk minimization inductive principle, seeks the optimal solution with limited samples. Therefore, SVM has an excellent generalization ability. SVM optimization problems can be transformed into a quadratic optimization problem to obtain global optima. For the situation that data samples are linearly inseparable, SVM adopts kernel function to map the low-dimensional input to high-dimensional space, in which linear discriminant is conducted. These exceptional properties give SVM huge advantages in both recognition and regression than traditional networks. In essence, SVM is still a neural network with one hidden-layer, and the optimizing process is a multivariate quadratic optimizing problem, which can be turned into a quadratic programming problem. Theoretically, if a neural network with multiple hidden-layers is well trained, it could have a much better performance than SVM [15]. However, there is no really good way to train a neural network with multiple hidden-layers, which, in essence, is an optimizing problem of high degree polynomials.

To address the training problem of deep networks, Hinton proposed the strategy to pre-train the network layer-by-layer in advance and conduct fine-tuning to the existing weights [12]. But this method cannot fundamentally solve the training problem of deep networks. The main reason is, when the sigmoid activation function is close to saturation region, the function changes too slowly and its derivative approaches zero, which results in the vanishing gradient. In 2010, Glorot Bengio used ReLU function to train a neural network, and found it can not only reach a lower error rate, but have a better sparsity [16]. This finding meets the rules of biological brain that there are only one to four percent of total neutrons activated at a time [17]. To solve the overfitting problem, Hinton proposed a novel training strategy calling dropout in 2012 [18]. Its main principle is, in each iteration, each neutron is forbided to perform the forward and backward calculation with a certain probability $p$. So the network structure changes in every iteration of the training process, which brings a better robustness. Deep networks can, to a great extent, avoid overfitting with dropout. The incidence of some unexpected situations that the final output changes significantly with a small fluctuation of the input data, which is common for the traditional training method, decreases greatly. In addition, ReLU also zeroes the outputs of several neurons, which results in sparse networks, and, as a result, decreases the interdependence between parameters and reduces the incidence of overfitting.

With the further research in the pre-process of the raw data and several other novel approaches, such as ReLU and

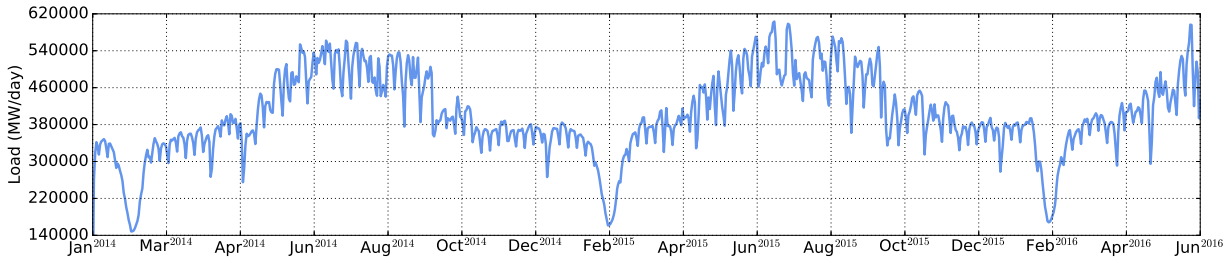



Fig. 1. The electrical load of a large city in China (From January 2014 to June 2016)

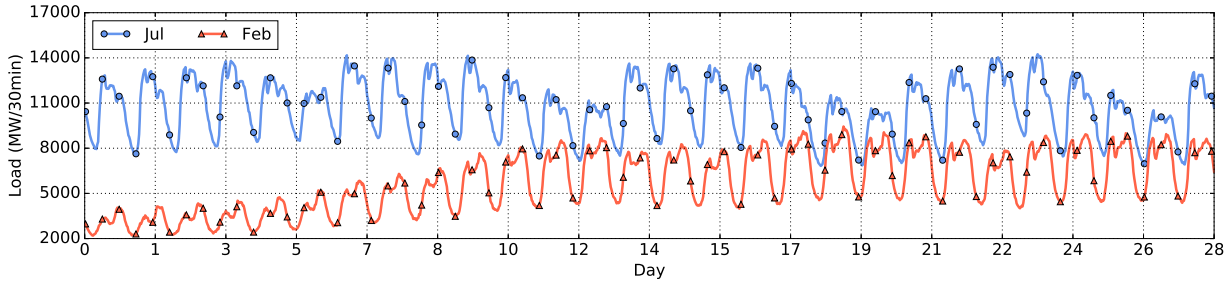



Fig. 2. The comparison of electrical load in February and July (2014)

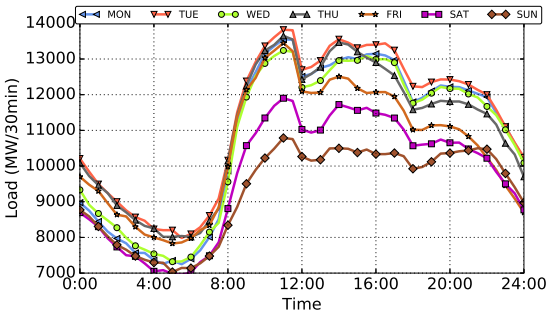



Fig. 3. Daily electrical load of different weekdays.

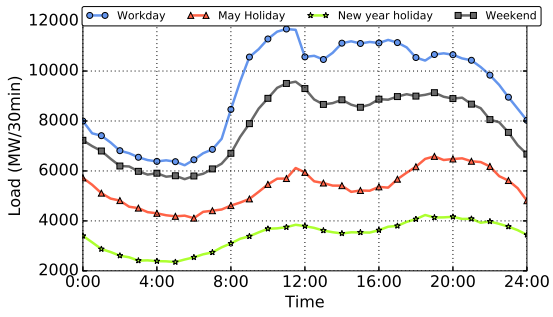



Fig. 4. Daily electrical load of different day types.

Dropout, both the vanishing gradient problem and the local optima problem can be effectively avoided [14]. Furthermore, the Graphic Processing Units (GPU) which have powerful parallel computing ability are widely used in network training, as a result, even a network with huge amount of data can get trained in an acceptable time.

## III. Mechanism

The load of power system is under the influence of various internal and external factors. So in essence, power load is uncontrollable. And the load forecasting is mainly based on the predictive model extracted from historical data. On the one hand, there is a very large uncertainty in estimates of the future power load, due to the stochastic factors, e.g. weather change, unexpected accidents, etc. On the other hand, from several perspectives, the power load changes regularly. Therefore, it is valuable for load forecasting, especially short-term forecasting, to analyze the load variations and interferences to find useful patterns.

The electrical load of a large city in China (From January 2014 to June 2016) is used in the research. The data was recorded every 5 minutes, and there are 257,184 records in total. As shown in Fig. 1, these records reflect the annual periodicity of the power load. The power load peaks between June and October every year, and hits bottom around February, which has significant seasonal characteristics. Fig. 2 presents the comparison of electrical load in February and July. In February, there is the most famous Chinese traditional holiday, the Spring Festival, therefore, the load reaches a low level. In contrary, July is the hottest month in China, and the load reaches a highest level in the whole year. As can be seen from Fig. 2, there is also a obvious daily periodicity in the load value. Fig. 3 shows the daily loads of a single week. And we can see, the power load peaks around 11:00 am, and

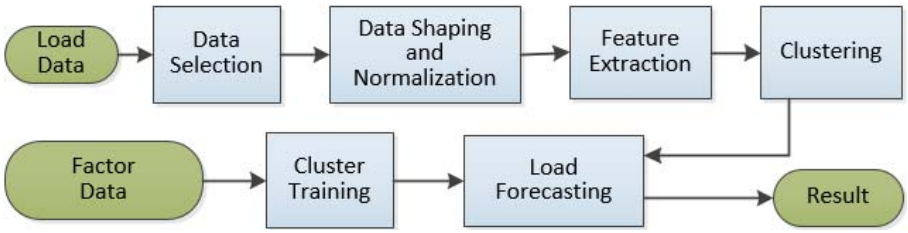



Fig. 5. Prediction procedure of the proposed method.

hits bottom around 4:30 pm.

Besides, through data analysis, we found that the day type, to a certain extent, decides the shape of daily load curve, as shown in Fig. 4. The load curves of workdays have completely different shapes with the ones of non-workdays. Even the daily load curves with the same day type differentiate in y-coordinate (the load value) according to different months, which is mainly caused by different temperature rather than months. For example, in Fig. 1, the load curve from March to June in 2014 and the one of the same period in 2015 show two different trends.

We believe, these patterns can be well recognized by the deep learning methods. Therefore, based on the above analysis, we designed a forecasting model composed of two parts. The first part performs clustering on the daily load curves, and divides the daily load data to several categories according to the corresponding curve shape. Fig. 4 shows the differences between these categories. At the clustering stage, we do not care the y-coordinate of load curves. And, the clustering model only focuses on the specific shapes. The second part is a prediction network trained for each category, using the results of the first part, and its input is the corresponding parameters of the power system and external environment. The output is the prediction value, i.e. the electric load forecasting result.

Fig. 5 presents the procedure of the proposed method. Firstly, our method obtains the daily load data as the input, and transforms them into graphical representation. Then, the proposed method extracts their features using a CNNmodel and, based on that, decides their categories. Several deep network model will be trained for each category. Ultimately, the trained networks can output the final prediction value.

### A. Clustering Model

Before transforming into graphical representation, pre-processing is required for the raw input data, including sampling and normalization. At first, one record is sampled from the load data every five minutes. Then, normalization is performed to the samples, mapping the load data to $[0, 1]$, as shown in Eq (1). $Y$ is the set of daily load data, such as $y_1, y_2, \cdots, y_{49}$. $y_i^*$ is the y-coordinate value after normalization. Since the images generated after normalization cannot reflect the load value in each sampled time point, RGB color is utilized to map the load values.

$$y_i^* = \frac{y_i - \min(Y)}{\max(Y) - \min(Y)} \qquad i = 1, 2, \cdots, 49 \tag{1}$$

Self-Organized Map (SOM) is widely used in clustering of load curves [19]. SOM uses the distance between input vector and weight vector to seek the responding neuron of output layer, which can be expressed as the following equation.

$$W_s = \arg\min_{\omega} \left\| X_i - W_j \right\| \tag{2}$$

where $W_s$ is the nearest weight vector from the input vector $X_i$. This weight vector $W_s$ corresponds to a certain neuron in the input layer of SOM network, i.e. the winning neuron. The vectors corresponding with the neurons in the neighborhood region of winning unit will be adjusted according to Gaussian attenuation, which can be represented by the following equation.

$$\mathrm{Near}_i = W_s + \alpha(s) \cdot h(s, i) \cdot (X_s - \mathrm{Near}_i) \tag{3}$$

where $\mathrm{Near}_i$ is the neurons in the neighborhood region of winning unit, function $\alpha(s)$ is the current learning rate, $h(s, i)$ represents the neighborhood function of the winning unit, which usually is Gaussian attenuation function. SOM method simply links the input to the output, and then adjust the weights. Currently, almost every clustering analysis method of daily load curve uses euclidean distance or manhattan distance, shown in Eq (4) and (5) respectively, as the reference to decide the similarity of two vectors, $X_i$ and $X_j$.

$$d_{ij} = \sqrt{\sum_{k=1}^{n} |x_{ik} - x_{jk}|^2} \tag{4}$$

$$d_{ij} = \sum_{k=1}^{n} |x_{ik} - x_{jk}| \tag{5}$$

However, there are limitations and restrictions with both euclidean distance and manhattan distance for the time series curves. The reason is, time series data is extremely susceptible to the interferences, e.g. noises, shift, deformation, etc. Time series data is time-variant and with characteristics of randomicity. Therefore, it is insufficient to simply consider the geometrical distance between points in the space. In order to calculate the similarity of two load curves, shape comparing based method is adopted in the paper, and a dual branch deep network is proposed to judge whether two curves match each other. The proposed network utilizes a deep neural network to extract the descriptors, obtain feature vectors, and, ultimately, calculate the similarity.

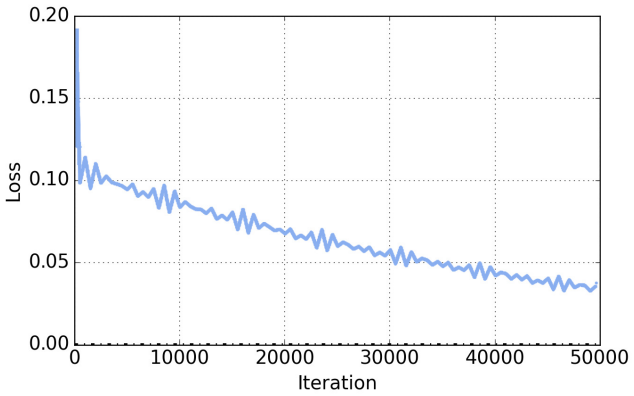



Fig. 6. Convergence curve of the loss function.

The input of the dual branch network must be pairs of training data. The input images, *patch1* and *patch2*, are processed by two branch respectively to obtain their features, which will get vectorized in a fully-connected-layer. The feature extraction process is equivalent to a mapping function $f_\theta()$, which maps the input $I_1$ and $I_2$ to $f_\theta(I_1)$, $f_\theta(I_1)$. Then the loss function, Eq (6) or (7), is adopted to train the network [20].

$$L = \sum_{I_1, I_2} \log(1 + \exp(\delta(I_1, I_2)(\alpha \cdot d(f_\theta(I_1), f_\theta(I_2)) - \beta))) \tag{6}$$

where $I_i$ is the information data of input picture, $f_\theta$ is a convolutional network, $d()$ is the distance function of two vectors, $\alpha$ and $\beta$ are distance adjustment parameters, and $\delta()$ checks whether pictures belong to the same category. If so, $\delta()$ equals 1, otherwise $-1$.

The loss function in comparative analysis is shown below.

$$E = \frac{1}{2N} \sum_{1}^{N} (y) \|a_n - b_n\|^2 + (1 - y) \max(\text{margin} - \|a_n - b_n\|^2, 0) \tag{7}$$

where $a_n$ and $b_n$ are the vectors of input images, $y$ represents whether these two pictures have significant differences. If they are same, $y$ is 1, otherwise $-1$.

As mentioned above, the input must be pairs of images, due to the dual-branch structure. And input data will be marked 1 if they are same type, otherwise $-1$. Moreover, the two branches own same shared weights. The feature extracted from input data $I_1$ and $I_2$ will be entered to the fully-connected-layer, whose function is mapping the features obtained in the convolutional layers to vectors. Therefore, similar input data will result in similar input vector, which represents that a well trained network can map similar input data to neighboring vectors. After training, only one branch is needed to perform the clustering. Compared with training network, the clustering network does not have the loss layer.

In the training process, learning rate $base_{lr}$ is set to 0.01, $momentum$ is set to 0.9, $weight_decay$ is 0.0005, the $gamma$ value of learning rate's polynomial error decay is 0.001, and $power$ is 0.75. The convergence curve of the loss function is shown in Fig. 6.

After the convergence of loss function, the network has the ability of feature extraction and mapping. If a full-connected-layer having two output neurons is added behind test network, the output data could be, though not strictly necessary, mapped to a two-dimensional plain, as shown in Fig. 8. Load data is time-series, and is always continuous curves. Therefore, the transition data usually has the character of multiple classes.

Nine groups of data are obtained after the clustering based on shape similarity, different types of load curves have different features. Fig. 7 includes partial data of three clustered classes. In all nine classes, two of them have very few data, due to some load abnormality caused by random occurrences, which can be regarded as noise data. Therefore, only seven clustered classes are obtained.

In this section, we transform the input numerical data into graphical representation, and adopt a CNN model to extract their features, and ultimately, use SOM to conduct the final clustering. Several neural networks will be trained for each category to perform the subsequent load forecasting task, which is detailed in the following section.

### *B. Prediction Model*

As mentioned above, the daily electrical load fits some certain distributions, which are decided by the features in each day, for example, working day or non-working day. Because of the numerous factors, even the same type of daily load has a difference in actual load value. Therefore, the daily load can be described as the sum of base load, which is only related to time, and the load caused by various influence factors. Usually, multiple records are needed to describe these influence factors, and vector $x^j = (x_1^j, x_2^j, \cdots, x_n^j)$ is adopted to represent a single influence factor. Then, daily electrical load can be expressed as

$$power(t) = b(t) + \sum_{x^j} \text{eff}_{x^j}(t) \tag{8}$$

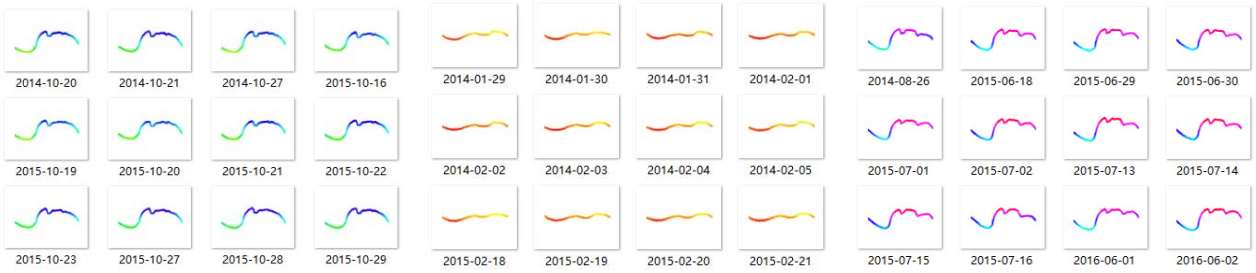
where $b(t)$ is the base load, which is not affected by other influence factors, and $\text{eff}_{x^j}(t)$ represents the load caused by the influence factor $x^j$.

Load forecasting can be described as a process to find a forecasting function $f()$ in the training set $\{(x_1, y_1), (x_2, y_2), \cdots, (x_m, y_m)\}$ to minimize $\sum_i^m (f(x_i) - y_i)^2$. A neural network with multi hidden layers can approximate any nonlinear functions with any precision, and this process can be expressed as the following formula, which looks for a set of weights minimizing the cost function of the network.

$$\text{loos}(\omega) = \sum_j L(y^j(\omega^T x^j)) + \lambda R(\omega) \tag{9}$$

where $\omega$ is the weight matrix, $L$ is the error function, and $R$ is the regularization term.

A deep network is designed to perform load forecasting. This network has two hidden layers, and their neuron numbers




(a) (b) (c)

Fig. 7. Clustering results of the load variation (graphical representations).

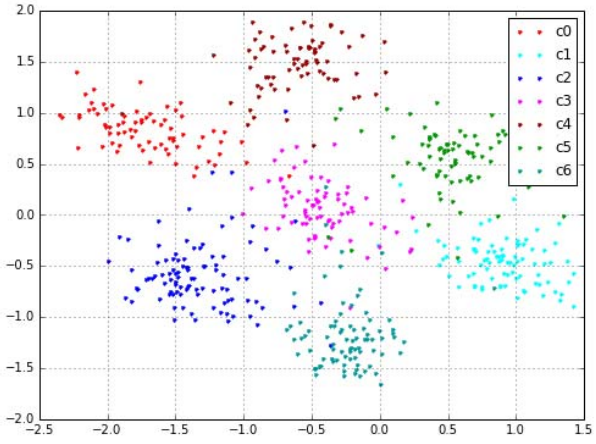



Fig. 8. Clustering results

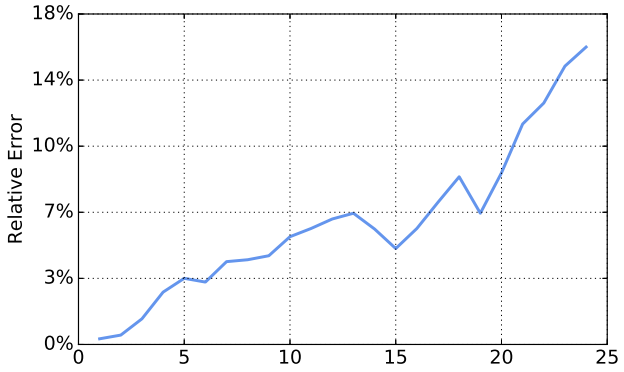



Fig. 9. Relative error of time series analyzing methods.

are 30 and 25, respectively. There is only one output value. It adopts ReLU as the activation function, and with Dropout strategy. The damping factor of weights is set to 0.1, and learning rate set to 0.01. All the input data must be normalized as required.

According to Grey Relational Theory [21], some factors, e.g. day type, mean daily temperature, daily maximum temperature, daily minimum temperature, humidity and wind speed have significant influence on electrical load. Therefore, the proposed model mainly focuses on these important factors. Given the load data and weather data, the input structure includes *time* vector, *day type* vector, *temperature* vector, *weather condition* vector and *electrical load* vector.

Vector *time* consists of current year, month, date, and hour. The following formula, Eq (10), is adopted for the calculation of month normalization. And Eq (11) for time normalization. The last coefficients are used to avoid zero values.

$$-\cos\left(\frac{\pi}{12}\cdot(month+0.01)\right) \tag{10}$$

$$-\cos\left(\frac{\pi}{48}\cdot(time+0.5)\right) \tag{11}$$

Vector *day type* represents the type of current day, including weekday, workday or not and holiday type. Eq (12) is used in the normalization of weekday. Workday element equals 1 or $-1$ according to actual situations. And holiday type consists of short and long vacations.

$$-\cos\left(\frac{\pi}{7}\cdot(week+0.01)\right) \tag{12}$$

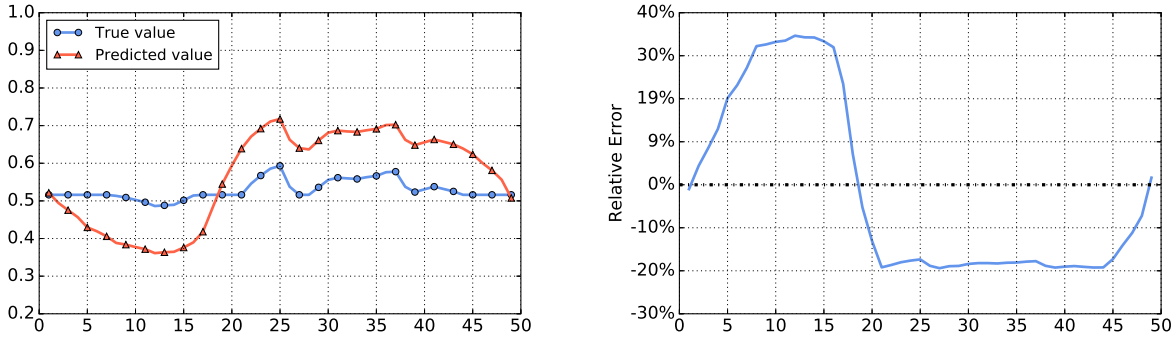
Vector *electrical load* includes the last load record, the minimum load in the last day, and the maximum load in last three days, etc.

Vector *temperature* includes current temperature, estimated highest temperature of current day, estimated lowest temperature of current day, highest temperature of last day, and lowest temperature of last day. Vector *weather condition* includes air pressure, humidity, wind speed and dew point temperature, and is calculated using min-max normalization.

After training process, the proposed neural networks obtain the ability to accurately forecast the short-term load variations, which is demonstrated in the following evaluation section.

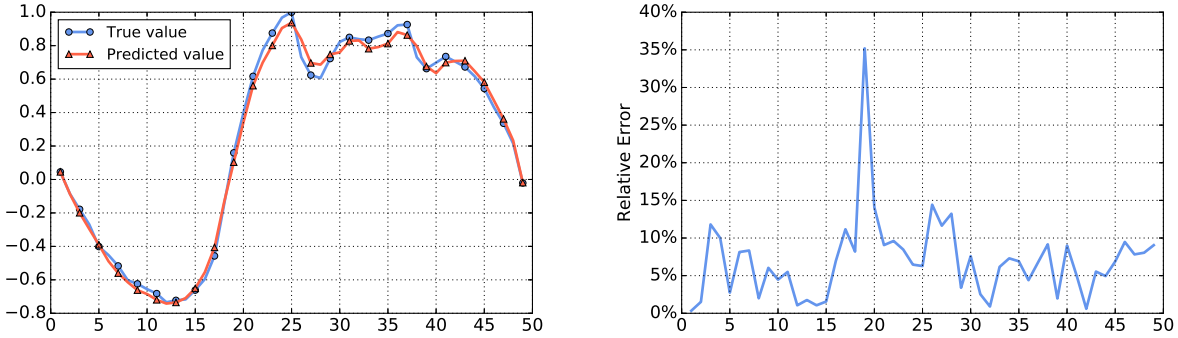
## IV. Performance Evaluation

The load data of a large city in China is used in the following experiments. Both the clustering network and the prediction network are tested, in order to give a comprehensive evaluation of the proposed forecasting method.




(a) Predicted value and true value (b) Relative error

Fig. 10. Prediction results of SVM method




(a) Predicted value and true value (b) Relative error

Fig. 11. Prediction results of DLSF method

### A. Prediction Accuracy

The prediction accuracy is analyzed using relative error, as shown below.

$$RE(t) = \left| \frac{y^*(t) - y(t)}{y(t)} \right| \times 100\% \tag{13}$$

**Time series analyzing methods.** Traditional time series analyzing based methods use vector $load$ to represent the related data, which includes the load value in last period, the load value in last two periods, etc. These methods have small prediction errors in extremely short time load forecasting [7], but there are accumulative errors in the prediction process using such series forecasting methods. Predictions are conducted every five minutes in the future time. The results are presented in Fig. 9. As we can see, time series analyzing methods always perform the prediction along the direction of load change, and the first several points have little errors, but with the increase of prediction time, the errors increase dramatically.

**SVM methods.** The standard regression function in LIBSVM [22] to predict the load data. In this experiment, Radial Basis Function (RBF) is adopted as the kernel function, SVM type is set to epsilon-SVR, the $\gamma$ parameter of kernel function is set to $0.03125$, the $\epsilon$ parameter of loss function is set to $0.01$, and the input data has 20 dimensions. We predict 48 results in different time everyday, as shown in Fig. 10. As we can see, SVM methods perform poorly when the input data has a high dimension or the number of training samples is too big. In addition, the sample selection has a huge impact on the final results, and noise samples could also bring a negative effect to the prediction precision.

**DLSF method.** We also conduct a load forecasting experiment using the proposed DLSF method. 48 predictions are generated for a single day, as shown in Fig. 11. Except for one prediction result which has a relative error larger than $15\%$, most forecasting results match well with the actual load. These evaluation results of the prediction methods demonstrate that, the proposed prediction network can be well applied to the actual smart grid scenarios, and outperform the existing methods in the forecasting accuracy.

### B. Clustering Performance

The clustering effect can be evaluated using Davies-Bouldin Index (DBI), as shown in Eq (14). A smaller DBI represents a better clustering performance.

$$\mathrm{DBI} = \frac{1}{7} \sum_{1}^{7} \max_{i \neq j} \left( \frac{\mathrm{avg}(C_i) + \mathrm{avg}(C_j)}{\mathrm{dcen}(u_i, u_j)} \right) \tag{14}$$

where $C$ represents clustering classes, $u$ is clustering centers. Function $\mathrm{dcen}(u_i, u_j)$ represents the distance between two clustering centers, which is calculated using Minkowski distance, as shown in Eq (15), and $p$ is set to 2.

$$d_{mikv}(x_i, x_j) = \left( \sum_{l=1}^{n} |x_i^l - x_j^l|^p \right)^{\frac{1}{p}} \tag{15}$$

Clustered centers can be calculated by Eq (16).

$$u = \frac{1}{C} \sum_{1 \le i \le |C|} x_i \tag{16}$$

Function $\mathrm{avg}(C_i)$ represents the average distance between samples in clustered class $C_i$, which is calculated using Eq (17).

$$\mathrm{avg}(C) = \frac{2}{|C|(|C|-1)} \sum_{1 \le i \le j} \mathrm{dist}(x_i, x_j) \tag{17}$$

As a result, four classes were obtained using a common SOM method, and its DBI is $16.21\%$. For comparison, the SOM in our DLSF method uses a CNN model to extract the features and, as a result, has a much lower DBI, which is $14.68\%$. Moreover, the SOM in our DLSF method can recognize and remove the abnormal data, and DBI can decrease to $8.57\%$ after the denoise.

## V. Conclusion

A deep learning based short-term forecasting method (DLSF) is proposed in the paper. We transform the load forecasting task into an image processing problem, and design a dual-branch deep convolutional network to extract the features from the input data for the subsequent clustering process. For load prediction, a multi-layer neural network is proposed to forecast future load variations. Compared with other simple prediction models and SVM methods, the proposed DLSF method takes into account all the external influencing factors, e.g. temperature, humidity, wind speed, etc., and outperforms the existing methods in the forecasting accuracy.

Future work includes developing a more effective cost function in order to further improve the training efficiency. We will also extend our work to other areas for a more comprehensive evaluation.

## Acknowledgment

This work is partially supported by JSPS KAKENHI Grant Number JP16K00117, JP15K15976, and KDDI Foundation.

## References

[1] Z. Liu, C. Zhang, M. Dong, B. Gu, Y. Ji, and Y. Tanaka, "Markov-decision-process-assisted consumer scheduling in a networked smart grid," *IEEE Access*, vol. 5, pp. 2448–2458, 2017.

[2] J. Wu, M. Dong, K. Ota, Z. Zhou, and B. Duan, "Towards fault-tolerant fine-grained data access control for smart grid," *Wireless Personal Communications*, vol. 75, no. 3, pp. 1787–1808, 2014.

[3] X. Deng, L. He, C. Zhu, M. Dong, K. Ota, and L. Cai, "Qos-aware and load-balance routing for IEEE 802.11s based neighborhood area network in smart grid," *Wireless Personal Communications*, vol. 89, no. 4, pp. 1065–1088, 2016.

[4] L. Guo, M. Dong, K. Ota, J. Wu, and J. Li, "A name-based secure communication mechanism for smart grid employing wireless networks," in *2016 IEEE Global Communications Conference, GLOBECOM 2016, Washington, DC, USA, December 4-8, 2016*, 2016, pp. 1–6.

[5] M. Q. Raza, Z. Baharudin, B. Islam, M. Zakariya, and M. M. Khir, "Neural network based stlf model to study the seasonal impact of weather and exogenous variables," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 6, no. 20, pp. 3729–3735, 2013.

[6] N. Amral, C. Ozveren, and D. King, "Short term load forecasting using multiple linear regression," in *Universities Power Engineering Conference, 2007. UPEC 2007. 42nd International*. IEEE, 2007, pp. 1192–1198.

[7] Y. Chakhchoukh, P. Panciatici, and L. Mili, "Electric load forecasting based on statistical robust methods," *IEEE Transactions on Power Systems*, vol. 26, no. 3, pp. 982–991, 2011.

[8] Z. Hua and Z. Lizi, "The factor analysis of short-term load forecast based on wavelet transform," in *Power System Technology, 2002. Proceedings. PowerCon 2002. International Conference on*, vol. 2. IEEE, 2002, pp. 1073–1076.

[9] G. Lambert-Torres, W. Marra, W. Lage, C. de Moraes, and C. Costa, "Data mining in load forecasting: an approach using fuzzy techniques," in *2006 IEEE Power Engineering Society General Meeting*. IEEE, 2006, pp. 8–pp.

[10] V. Vapnik, *The nature of statistical learning theory*. Springer Science & Business Media, 2013.

[11] M. Rejc and M. Pantos, "Short-term transmission-loss forecast for the slovenian transmission power system based on a fuzzy-logic decision approach," *IEEE Transactions on Power systems*, vol. 26, no. 3, pp. 1511–1521, 2011.

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

[13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[14] D. Erhan, P.-A. Manzagol, Y. Bengio, S. Bengio, and P. Vincent, "The difficulty of training deep architectures and the effect of unsupervised pre-training." in *AISTATS*, vol. 5, 2009, pp. 153–160.

[15] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Deep big simple neural nets excel on handwritten digit recognition," *CoRR*, vol. abs/1003.0358, 2010. [Online]. Available: http://arxiv.org/abs/1003.0358

[16] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks." in *Aistats*, vol. 15, no. 106, 2011, p. 275.

[17] P. Lennie, "The cost of cortical computation," *Current biology*, vol. 13, no. 6, pp. 493–497, 2003.

[18] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[19] L. Hernández, C. Baladrón, J. M. Aguiar, L. Calavia, B. Carro, A. Sánchez-Esguevillas, P. García, and J. Lloret, "Experimental analysis of the input variables relevance to forecast next days aggregated electric demand using neural networks," *Energies*, vol. 6, no. 6, pp. 2927–2948, 2013.

[20] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 539–546.

[21] F. Li and X. Zhao, "The application of genetic algorithm in power short-term load forecasting," in *2012 International Conference on Image, Vision and Computing (ICIVC 2012)*. IACSIT Press.

[22] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.