An Improved Short-Term Power Load Forecasting Mechanism Based on a Deep Learning Algorithm

[1] Hao Dong, [2,*] Chun-Lai Zhou, [1] Zhi-Da Li and [1] Yan-Ju Guo

Abstract

Power load forecasting has been known as one of the key issues in the field of power grids since published in the literature. With the advent of the era of an intelligent power grid, a high level of accuracy and stability need to be reached in load forecasting. Deep Belief Nets [DBN) in deep learning algorithms is proposed as a new solution to regression problems due to its powerful non-linear modeling capacity. With the above-stated features, DBN is applied to solve the short-term load forecasting problems. First of all, training and modeling processes of DBN are detailed herein, and a simple set is then constructed through an analysis on the experimental data provided by the East - Sfovakia Power Distribution Company. Subsequently, the optimal parameters are experimentally determined to build a network with the minimized error, and finally the trained network is applied to load forecast using the DeepLearnToolbox - master in Matlab. For validation purposes, a comparison is made between the forecasted results using DBN and a BP neural network, and DBN is found to outperform the counterpart in terms of forecast accuracy. In short, the deep learning-based algorithm is presented as an improved version of short-term load forecasting mechanisms, and is believed to be a potential solution to relevant issues.

Keywords: Power systems, load forecasting, DBN, deep learning algorithm

1. Introduction

Power system load forecasting stands as the pillar of power grid scheduling, operation, control and other works [1]. Since the conceptualized, it has been one of key issues in power grid studies [4]. The short-term load forecasting of a power system is mainly used to forecast the electric load over the coming hours, a day or even a couple of days. The issue of short-term load forecasting has been addressed since 1960s, and a tremendous progress has been made ever since.

*Corresponding Author: Chun-Lai Zhou
(E-mail: clzhou@cuc.edu.cn).
[1] School of Electrical Engineering, Hebei University of Technology, Tianjin 300130, China
[2] School of Information Engineering, Communication University of China, Beijing 100024, China

Up to now, short-term load forecasting can be mainly categorized into classic, traditional, and intelligent forecasting methods. The intelligent ones become the most common approach due to its accuracy and stability [5], and the mainstream covers neural network-based [11] and support vector regression (SVR)-based forecast techniques [18].

Nowadays, a high performance mechanism must be built for an accurate power load forecast, as the scale of grids increases and an increasing number of smart grids are deployed. The aforementioned neural network and SVR-based algorithms belong to the shallow structure algorithms that have difficulty to express a complicated nonlinear function which effectively gives a limited number of samples. A deep learning algorithm, as presented in [21], which has been developed rapidly over recent years, is adopted herein as an improved way to forecast the power load. Since firstly conceptualized by Hinton et al. in 2006, deep learning algorithms have brought new hope to the disciplines of machine learning and artificial intelligence, and with the improvement on the original proposal, breakthrough has been seen in the fields of speech recognition, computer vision, etc. At the same time, deep learning algorithms are believed to be a potential solution to regression problems since complex functions can be approximated with a nonlinear deep-learning network architecture, and due to the high performance in learning essential characteristics from a small number of samples.
Developed as a deep learning model for unsupervised learning, a deep belief net (DBN) firstly employs an unsupervised greedy algorithm to pre-train data step by step and then use a supervision algorithm to fine-tune the pre-trained data from top to bottom. As can found later, adoption of DBN to deal with power system load forecasting accounts for the outperformance of this work.

2. Deep Belief Nets (DBN)

DBN is essentially a kind of generation models composed by multilayer neurons, and the neurons thereof can be divided into dominant neurons and hidden neurons. The former, which constitute visible layers, are used to receive the input data, and the latter, as the constituents of hidden layers, are responsible for extracting the characteristics of the received input data. The hidden layers at the bottom are designed to extract the characteristic at a low

level. With a great number of layers, features can be expressed smoothly, which makes it easier to model the extracted features. The extracted characteristics of each layer are then treated as the input to the next layer.

2.1 Restricted Boltzmann Machines (RBM)

DBN is composed of restricted Boltzmann machines (RBMs). AS illustrated in Fig.1, an RBM involves a visible layer, and a hidden layer, respectively symbolized as $\mathbf{v}$ and $\mathbf{h}$, and the weight between two layers are represented as W. Neurons in an RBM are characterized as follows. There is no connection between neurons in the same layer, while the neurons in one layer are all connected in the other.

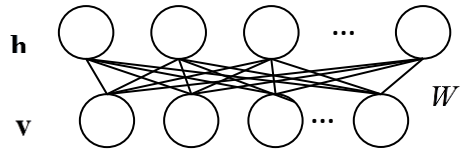


Figure 1: The structure of an RBM

RBM is an energy-based model. Given a set of states ($\mathbf{v}$, $\mathbf{h}$), an energy function is defined as

$$E(\mathbf{v},\mathbf{h}) = -\sum_{i=1} a_i v_i - \sum_{j=1} b_j h_j - \sum_{i=1}\sum_{j=1} v_i h_j w_{ij} \tag{1}$$

where v_i represents the i th visible layer unit; h_j represents the j th hidden layer unit; a_i represents the bias of the i th neurons in the visible layer; b_j represents the bias of the j th neurons in the hidden layer, and w_{ij} the weight between a_i and b_j. The joint probability distribution of states ($\mathbf{v}$, $\mathbf{h}$) can be written as

$$P(\mathbf{v},\mathbf{h}) = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{Z} \tag{2}$$

where Z denotes the normalization factor, also known as the partition function, and expressed as

$$Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \tag{3}$$

In practical applications, $P(\mathbf{v})$ and $P(\mathbf{h})$, the probability distributions of $\mathbf{v}$, $\mathbf{h}$, namely the marginal distribution of $P(\mathbf{v},\mathbf{h})$, must be known, respectively expressed:

$$P(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{v},\mathbf{h}) = \frac{1}{Z}\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \tag{4}$$

$$P(\mathbf{h}) = \sum_{\mathbf{v}} P(\mathbf{v},\mathbf{h}) = \frac{1}{Z}\sum_{\mathbf{v}} e^{-E(\mathbf{v},\mathbf{h})} \tag{5}$$

Once the states of all the neurons in one layer of RBM are known, the activation probability of each neuron in the other layer can be found accordingly. Given the state of the hidden layer, the activation probability of neuron i in the visible layer can be derived as

$$P(v_i = 1|\mathbf{h}) = sigmoid(a_i + \sum_j h_j w_{ij}) \tag{6}$$

The probability of the entire visible layer can be defined as

$$P(\mathbf{v}|\mathbf{h}) = \prod_i P(v_i|\mathbf{h}) \tag{7}$$

Similarly, once the state of the visible layer is known, the activation probability of neuron j in the hidden layer is formulated as

$$P(h_j = 1|\mathbf{v}) = sigmoid(b_j + \sum_i v_i w_{ij}) \tag{8}$$

As in Eq. (7), the probability of the entire hidden layer is expressed as

$$P(\mathbf{h}|\mathbf{v}) = \prod_j P(h_j|\mathbf{v}) \tag{9}$$

Supposed that S is a set composed of training samples, written as

$$S = \{\mathbf{v}^1, \mathbf{v}^2, \cdots, \mathbf{v}^n\} \tag{10}$$

where n is the number of samples, $\mathbf{v}^l$ ($l=1,\cdots,n$) represents a sample, and all of the samples are independently and identically distributed. Subsequently, an RBM training aims to locate the optimal parameters to fit the samples. Letting θ be the parameter matrix, and the training goal is then to maximize the likelihood function

$$L(\theta)=\prod_{l=1}^{n}P(\mathbf{v}^l) \tag{11}$$

For a convenient mathematical treatment, Eq. (11) is redefined in a logarithmic form for the reason that $L(\theta)$ and $\ln L(\theta)$ reach respectively maximum values at the same value of θ. Hence, the training objective is now to maximize the log-likelihood function

$$\ln L(\theta)=\sum_{l=1}^{n}\ln P(\mathbf{v}^l) \tag{12}$$

The likelihood function is then maximized using stochastic gradient ascent for a single training sample $\mathbf{v}^l$, leading to

$$\frac{\partial \ln P(\mathbf{v}^l)}{\partial\theta}=-\sum_{\mathbf{h}}P(\mathbf{h}|\mathbf{v}^l)\frac{\partial E(\mathbf{v}^l,\mathbf{h})}{\partial\theta}+\sum_{\mathbf{v},\mathbf{h}}P(\mathbf{v},\mathbf{h})\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta} \tag{13}$$

The first term on the right side of Eq. (13) refers to the energy function $\frac{\partial E(\mathbf{v}^l,\mathbf{h})}{\partial\theta}$'s expectation under the conditional distribution $P(\mathbf{h}|\mathbf{v}^l)$. $P(\mathbf{h}|\mathbf{v}^l)$ represents the probability distribution of the hidden layer with the data contained in the visible layer as the training sample $\mathbf{v}^l$, and can be easily calculated.

The second term refers to the energy function $\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta}$'s expectation under the conditional distribution $P(\mathbf{v},\mathbf{h})$. $P(\mathbf{v},\mathbf{h})$ represents the joint probability distribution of the visible and hidden layers, which, as opposed to $P(\mathbf{h}|\mathbf{v}^l)$, cannot be easily evaluated in the presence of the normalization factor Z. A way to resolve this problem is to approximate $P(\mathbf{v},\mathbf{h})$ using a sampling algorithm. So far, the Contrastive Divergence (CD) algorithm 22 remains the most popular approach due to its excellent performance.

A k-step CD algorithm (CD-k algorithm) is stated as follows:

1) Symbolize the initial state of the visible layer as $\mathbf{v}^0$;
2) Perform Gibbs sampling k times. In the t th ($t=1,2,\cdots,k$) sampling, get $\mathbf{h}^{t-1}$ using $P(\mathbf{h}|\mathbf{v}^{t-1})$, and then, get $\mathbf{v}^t$;
3) Approximate $\sum_{\mathbf{v},\mathbf{h}}P(\mathbf{v},\mathbf{h})\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta}$ in Eq. (13) as $\sum_{\mathbf{h}}P(\mathbf{h}|\mathbf{v}^k)\frac{\partial E(\mathbf{v}^k,\mathbf{h})}{\partial\theta}$ using the sampling results in step 2.

Through the CD-k algorithm (13) can be approximated, and the value of the parameter matrix θ can be found as well. In most cases, acceptable results can be seen by one time evaluation of the CD algorithm.

2.2 The Modeling and Training of DBN

As explicitly stated previously, a DBN is made up of RBMs, which consists of a visible layer, multiple hidden layers, and an output layer. The visible layer of the first RBM serves as the visible layer and doubles as the input layer of the DBN. The visible layer of the second RBM acts as the hidden layer of the first RBM and doubles as the first hidden layer of the DBN. The hidden layer of the second RBM functions serves as the second hidden layer of the DBN, etc. In this manner, a DBN can be constituted. Suppose that there are a total of m layers, including the hidden layers $h_1,h_2,\cdots,h_{m-1}$ and the output layer h_m; vector $\boldsymbol{x}$ represents the input, and the $m-1$ underlying network layers are made up of RBMs. The first stage of a training is to obtain the weight of the generative model through pre-training using an unsupervised greedy algorithm. In this stage, the first RBM, involving layers h_0,h_1, is trained as described in Section 1.1 to achieve energy balance, and the output thereof is then taken as the input to the second RBM. This process is repeated until the last RBM training is finished. After the pre-training, the target output corresponding to the input $\boldsymbol{x}$ is set to $\boldsymbol{y}$, and the output is set to $\boldsymbol{y}'$. Construct a loss function with the supervisory signal $\boldsymbol{y}$, and train the network in a supervised way using the gradient descent method and fine-tune the parameters 23. The training process is illustrated in Figure 2.

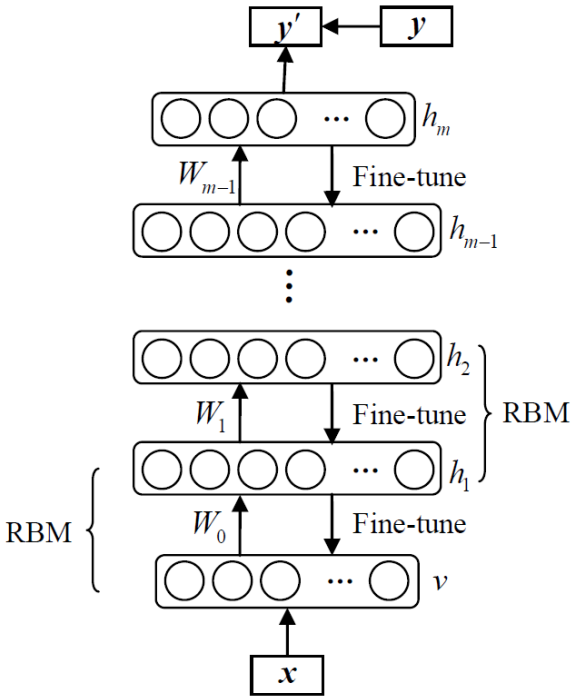


Figure 2: Illustration of a DBN training

3. DBN-Based Modeling and Result Analysis

Design parameters, e.g. the number of the hidden layers, the number of the neurons in a hidden layer, etc, must be specified for the construction of a DBN, according to available load data. In the following section, the above-stated deep learning algorithm will be applied to the load forecasting of a power grid.

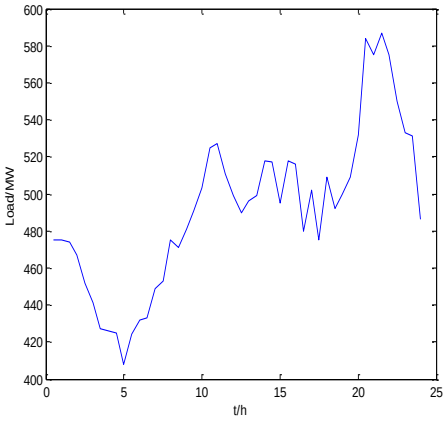
3.1 Data Selection for Training and Testing Purposes

Experimental data is provided by the East - Sfovakia Power Distribution Company, for the load forecasting contest held by the European Network of Excellence on Intelligent Technologies for Smart Adaptive Systems (EUNITE) in 2001. The load data were recorded at a sampling period of 30 minutes over the time span between 1997, 1998 and January 1999.

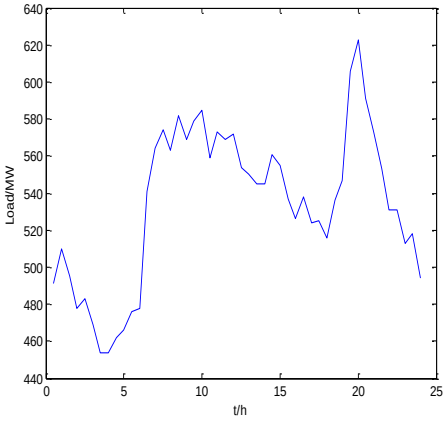
Fig. 3 illustrates a comparison among the one day load curves on randomly selected dates, i.e. 1997.5.10, 1998.9.11 and 1999.1.12. It is found that curves share the same tendency, although not in a good agreement. The curve trend declines first, and then rises gradually to a certain level during the working time. It then oscillates within a certain range, and peaks in the evening after a brief drop. The load change during the day is certain to have a rule to follow, and there must exist a correlation between the load values at neighboring time instants.

Next, an analysis on the load variation over a one-week observation period, i.e. 336 pieces of data, is made as follows. For illustration purposes, Fig. 4 gives three load curves over the time frames 1998.2.9-2.15, 6.22-6.28, and 10.12-10.18. There is an obvious gap between the load on weekdays and weekend, and even an amount of load fluctuation within a workweek too. The observation periods are roughly 4 months away each other, and it is hence evident that the load curve is highly correlated to the dates. In other words, the day must be made for a forecast.

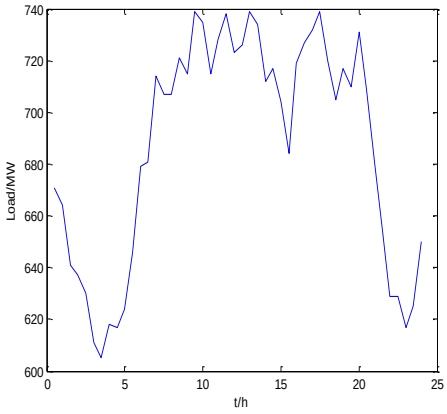
categorized in advanced in terms of the load level, and the time span between the dates of a selected sample and a predicted outcome must be short for the sake of accuracy.



(a)



(b)



(c)

Figure3: Respective load curves measured on: (a) 1997.5.10; (b) 1998.9.11; (c) 1999.1.1

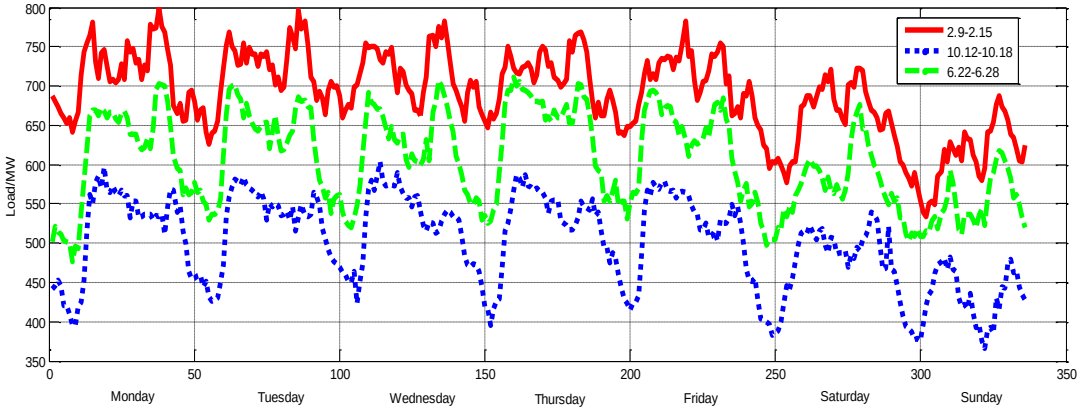


Figure 4: Comparison on load curves over three one-week observation periods in 1998

The load data over the 14 week span: 1998.4.20-1998.7.26 are chosen as a training set, while those over the 6 week time span: 1998.7.27-1998.9.6 are as a test set. When load forecast at a time instant is conducted, the load values at the 5 consecutive time instants right before the above-stated time instant, but in previous days of the same type in terms of power load, are taken as the remaining 5. In simple terms, week days and weekend must be categorized in advance as a way to minimize the load forecast error.

3.2 Results and Discussion

The presented load forecasting algorithm is realized using the DeepLearnToolbox - master in Matlab and all data in the sample set, constructed in 2.1, are normalized to [0,1] by

$$\hat{x} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{14}$$

where x represents the load value at current time instant, $x_{\max}$ and $x_{\min}$ represent the maximum and the minimum load over a whole day, respectively.

Through training and testing, the design parameters of a DBN are chosen as follows. There is one hidden layer containing 8 neurons and 5 numepoch.

For illustration purposes, the effects of the number of neurons in the hidden layer and numepoch on the forecasting accuracy are demonstrated by the example of 1998.8.31 in Figs. 5-6, respectively. In Fig. 5, the red stepwise curve represents the actual load value, while the blue, green, and yellow ones represent the predicted values with 6, 8, and 10 neurons, respectively. In Fig. 6, the piecewise linear curve in red represents the actual load value as in Fig. 5, while the green, yellow, and blue bars represent the predicted values with 3, 5, and 7 numepochs, respectively. Thus, it is evident that the numbers of neurons and numepoch demonstrate a strong effect on the forecast accuracy.

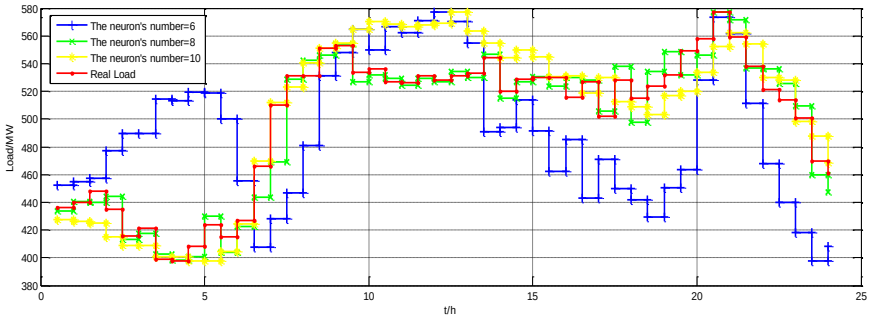


Figure 5: A family of load forecast curves with the number of neurons in the hidden layer as a parameter

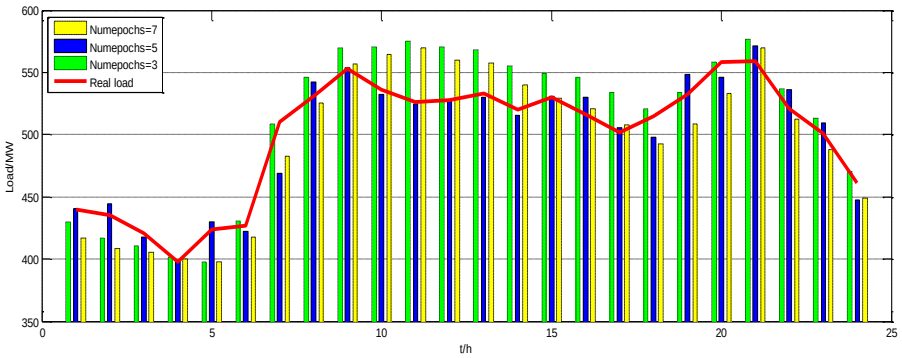


Figure 6: Comparison on forecasted load with different numpochs

For validation purposes, Fig. 7 illustrates a comparison among the forecasted results, using DBN and BP algorithms, and the actual load curve on 1998.8.31, and the comparison, including forecast errors, is presented in a tabular form as

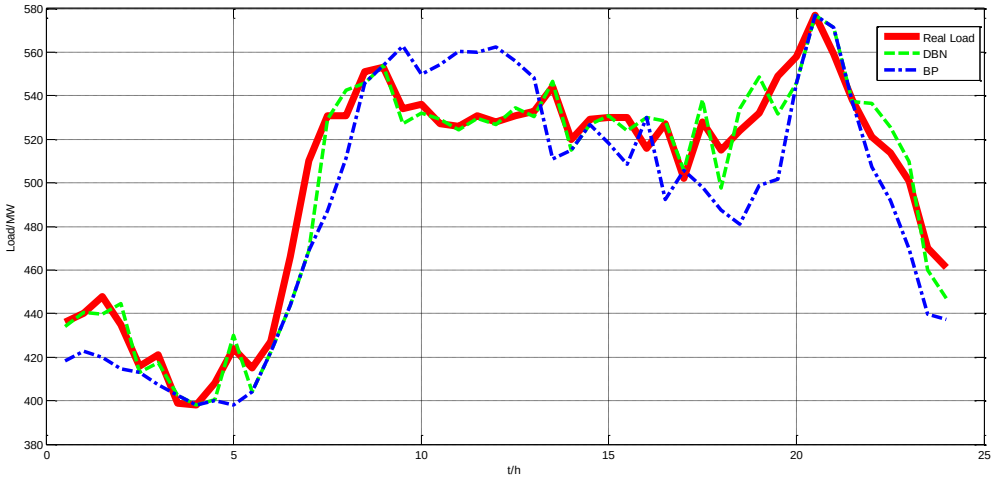


Figure 7: Comparison among a real load curve and the forecasted results using DBN and a BP neural network

Table 1: the Mean Absolute Percentage Error (MAPE) can be calculated according to the data in Table 1, defined as

$$MAPE(X_i) = \frac{1}{l}\sum_{i=1}^{l}\left|\frac{a_i - p_i}{a_i}\right| \qquad (15)$$

where l is the number of the elements contained in the testing sample set, a_i and p_i represent the actual and predicted load values of the i th sample, respectively.

The outperformance of DBN is demonstrated by an MAPE of 1.68% as compared with 3.43% by a BP counterpart, meaning that DBN is validated to provide an improved short-term load forecasting accuracy.

Table 1: Tabular representation of Figure 7

Time	Real	DBN		BP		Time	Real	DBN		BP	
	Load	Forecast	Error/%	Forecast	Error/%		Load	Forecast	Error/%	Forecast	Error/%
1:00	440	440	0	422	4.09	13:00	533	530	0.56	548	-2.81
2:00	435	444	-2.07	415	4.60	14:00	520	515	0.96	515	0.97
3:00	421	417	0.95	407	3.33	15:00	530	531	-0.19	518	2.26
4:00	398	398	0	398	0	16:00	516	530	-2.71	530	-2.71
5:00	424	430	-1.42	398	6.13	17:00	502	506	-0.80	505	-0.60
6:00	427	423	0.94	422	1.17	18:00	515	498	3.30	489	5.05
7:00	510	469	8.03	467	8.43	19:00	532	549	-3.19	499	6.20
8:00	531	542	-1.69	511	3.77	20:00	558	546	2.15	546	2.15
9:00	553	554	-0.18	554	-0.18	21:00	559	571	-2.14	571	-0.21
10:00	536	532	0.75	550	-2.61	22:00	521	536	-2.88	507	2.69
11:00	526	524	0.38	560	-4.56	23:00	501	510	-1.80	470	6.19
12:00	528	527	0.19	562	-6.46	24:00	461	447	3.04	437	5.21

4. Conclusions

This paper presents a deep learning-based algorithm for the short-term load forecast of a power system, and it is as expected to outperform a BP counterpart.

As the scale of micro grids increases, photovoltaic, wind power generation systems and other renewable energy sources are configured in a way that makes it difficult to predict the power load, meaning that the development of an accurate load forecasting mechanism is seen as required, and it is schedule as a successive project in the very near future to forecast the load of a hybrid photovoltaic-wind power generation system.

Acknowledgment

This study is supported by the Hebei Province Science and Technology Support program (No.15212114) and the Tianjin Science and Technology Commissioner Program (No.16JCTPJC50700).

References

[1]. Wu, L.; Shahidehpour, M. A hybrid model for integrated day-ahead electricity price and load forecasting in smart grid. IET Gener. Transmiss. Distrib. 8, 1937-1950, 2014.

[2]. Arora, S.; Taylor, J.W. Short-term forecasting of anomalous load using rule-based triple seasonal methods. IEEE Trans. Power Syst. 28, 3235-3242, 2013.

[3]. Sperandio, M.; Bernardon, D.P.; Bordin, G.; et al. Probabilistic demand forecasting to minimize overtaking the transmission contract. Electrical Power Syst. Res. 112, 27-36, 2014.

[4]. Xu, Y.; Hu, Q.; Li, F. Probabilistic model of payment cost minimization considering wind power and its uncertainty. IEEE Trans. Sustain.4, 716-724, Energy 2013.

[5]. De Jonghe, C.; Hobbs, B.F.; Belmans, R. Optimal generation mix with short-term demand response and wind penetration. IEEE Trans. Power Syst. 27, 830-839, 2012.

[6]. Fan, S.; Methaprayoon, K.; Lee, W.J.; et al. Multiregion load forecasting for system with large geographical area. IEEE Trans. Ind. Appl. 45, 1452-1459, 2009.

[7]. Suganthi, L.; Samuel, A.A. Energy models for demand forecasting-A review. 16, 1223-1240 , Renewable and Sustainable Energy Reviews 2012.

[8]. Matijas, M.; Suykens, J.A.K.; Krajcar, S. Load forecasting using a multivariate meta-learning system. 40, 4427-4437, Expert Systems With Applications 2013.
[9]. De Jonghe, C.; Hobbs, B.F.; Belmans, R. Optimal generation mix with short-term demand response and wind penetration. IEEE Trans. Power Syst. 27, 830–839, 2012.
[10]. Vilar, J.M.; Cao, R.; Aneiros, G. Forecasting next-day electricity demand and price using nonparametric functional methods. Int. J. Electr. Power Energy Syst., 39, 48–55, 2012.
[11]. Khwaja, A.S.; Naeem, M.; Anpalagan, A.; et al. Improved short-term load forecasting using bagged neural networks. 125, 109-11 5 , Electric Power Systems Research 2015.
[12]. Nose-Filho, K.; Lotufo, A.D.P.; Minussi, C.R.; et al. Short-Term multinodal load forecasting using a modified general regression neural network. IEEE Trans. Power Deliv. 26, 2862-2869, 2011.
[13]. Mandal, P.; Senjyu, T.; Urasaki, N.; et al. A neural network based several-hour-ahead electric load forecasting using similar days approach. Int. J. Electr. Power Energy Syst. 28, 367-373, 2006.
[14]. Hao, Q.; Srinivasan, D.; Khosravi, A. Short-term load and wind power forecasting using neural network-based prediction intervals. IEEE Trans. Neural Netw. Learn. Syst. 25, 303-315, 2014.
[15]. Chen, Y.; Luh,P. B.; Guan,C.; et al. Short-term load forecasting: Similar day-based wavelet neural networks. IEEE Trans. Power Syst. 25, 322-330, 2010.
[16]. Nose-Filho, K.; Lotufo, A.D.P.; Minussi, C.R. Short-term multinodal load forecasting using a modified general regression neural network. IEEE Trans. 26, 2862-2869, Power De. 2011.
[17]. Ding, N.; Benoit, C.; Foggia, G.; et al. Neural network-based model design for short-term load forecast in distribution systems. IEEE Trans. Power Syst. 31, 72-81, 2016.
[18]. Ceperic,E.; Ceperic,V.; Baric, A. A strategy for short-term load forecasting by support vector regression machines. IEEE Trans. Power Syst. 28, 4356-4364, 2013.
[19]. Wang, J.J.; Li, L.; Niu, D.X.; et al. An annual load forecasting model based on support vector regression with differential evolution algorithm. Applied Energy, 94, 65-70, 2012.
[20]. Elattar, E.E.; Goulermas, J.; Wu, Q.H. Electric load forecasting based on locally weighted support vector regression. IEEE Trans. Syst., Man, Cybern. C, Appl., Rev. 40, 438-447, 2010.
[21]. Arel, I.; Rose, D.C.; Karnowski,T.P. Deep machine learning—A new frontier in artificial intelligence research [research frontier]. IEEE Comput. Intell. Mag. 5, 13-18, 2010.
[22]. Fischer, A.; Igel, C. Bounding the Bias of Contrastive Divergence Learning. 23, pp. 664-673, Neural Computation 2011.
[23]. Montufar, G.; Ay, N. Refinements of universal approximation results for deep belief networks and restricted Boltzmann machines. Neural Comput. 23, 1306–1319, 2011.

Hao Dong, male, be born in 1992, a master of Electrical Engineering in Hebei University of Technology, one of the researchers of the Hebei Province Science and Technology Support Program (No. 15212114). His research interests is Power System and Automation.

Chun-Lai Zhou, male, be born in 1958, a professor of Control Science and Engineering in Communication University of China, the corresponding author of this paper. His research interests is pattern recognition and intelligent system. E-mail: clzhou@cuc.edu.cn.

Zhi-Da Li, male, be born in 1972, an engineer in Province-Ministry Joint Key Laboratory of Electromagnetic Field and Electrical Apparatus Reliability in Hebei University of Technology. His research interests is control theory and control engineering.

Yan-Ju Guo, female, be born in 1980, a lecturer of Computer Science in Hebei University of Technology, one of the key researchers of the Tianjin Science and Technology Commissioner Program (No.16JCTPJC50700). Her research interests is intelligent information processing.