Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

Short-term Electricity Load Forecasting Using a MapReduce-based Elman Networks with Coarse-grained Parallel Genetic Optimization

Guangqing Bao
College of Electrical and Information Engineering
Lanzhou University of Technology
Lanzhou, Gansu Province, China
gqbao@lut.cn

Fuchao Liu and Xin He
Institute of Electric Power Science
State Grid Gansu Electric Power Company
Lanzhou, Gansu Province, China
Liufc8127@sina.com

Gang Zhang
College of Water Resources and Hydropower
Xi 'an University of Technology
Xi 'an, Shanxi Province, China
zhanggang3463003@163.com

***Abstract* - How to make full use of high-dimensional massive power data to improve the accuracy and efficiency of short-term electrical load forecasting is a challenging problem to be solved. This paper discusses a parallel load forecasting technique with an Elman Neural Network(ENN) based on the MapReduce(MR) programming model. Specially, a Coarse-Grained Parallel Genetic Algorithm(CPGA) is introduced into the process of ENN training for obtaining the promising weights and thresholds. In the framework of MR, a parallel load-forecasting model of MR-CPGA-ENN is established. Case study with different scenario-based electrical load data sets and meteorological information confirm that both the accuracy and efficiency of the developed model are superior to the conventional models compared.**

Index Terms - Electrical load forecasting; Elman neural network; coarse-grained parallel genetic algorithm; MapReduce.

I. INTRODUCTION

Since electricity cannot be stored effectively in large quantities, the amount generated at any given time must cover the workload needs as well as power losses. Accurate tracking of the load by the system generation at all times is a basic requirement in the operation of power systems. The forecasting carried out for a single day to several weeks ahead is usually referred to as short-term electrical load forecasting(STELF). As the scale of development and construction of new renewable energy in China continuous to expand, enhance the capacity of wind and PV power utilization is becoming increasingly prominent. Electricity load forecasts have become a fundamental input to energy companies' decision-making mechanisms. Especially in retail energy markets, where consumers, generators, and traders can interact, supplier obligations are settled even on hourly or sub-hourly basis, which giving a new dimension to the problem of STELF. According to Chinese electrical guide rules, about 2%-5% of the installed capacity must be prepared for the spinning reserve capacity. Thus, with a 1% improvement in the reserve forecasting accuracy, about 80,500 to 201,300 MWh electricity can be saved for one day. From Hobbs' survey, the electric power companies would accrue 76 million USD yearly from improved dispatch and power purchases if the mean absolute percentage error of load forecasting is reduced by 1.5%[1].

The interest in improving the accuracy of STELF is also encouraged in part by methodological advances. Up to present, various approaches have been proposed to conduct STELF, which are divided principally into two categories: classical statistics methods[2] and artificial intelligence techniques. The statistics schemes include linear regression, Kalman filtering method, and state space model etc., which perform well in predicting linear load, but tend not to provide any insight into the cause of mode structural changes. Support vector machine seems to offer excellent generalization properties on STELF. The parameters selection of support vector machine is crucial and influences the forecasting accuracy. Artificial intelligence techniques include expert system, fuzzy logic and artificial neural network(ANN) etc., which achieve competitive advantages for nonlinear load mapping and generalization, however present challenges to criteria making and parameter setting. Sometimes severe issues such as "over fitting" and "curse of dimensionality associated with high-dimensional input" appear in the design of STELF.

With smart grid constructions, the yearly data volume of power industry grows from the current level of GB to TB, or even the PB level. As the power utilities need to analyze big data within seconds or minutes, traditional forecasting tools can hardly ever meet the requirements of the accuracy and efficiency simultaneously. Therefore, based on the MapReduce(MR) software platform, a parallel load forecasting technique with Elman Neural Network (ENN) is put forward. However, ENN cannot always assure the desired real time forecasting properties for long time iteration process and possible premature convergence. Accordingly, the coarse-grained parallel genetic algorithm (CPGA) is introduced into the weight and threshold optimization of ENN. MR is a framework for processing parallelizable problems across large data sets using a great number of computer nodes, collectively referred to as a cluster. Applying massive historical load data to perform load requirement evaluation on demand response, the CPGA-ENN is implemented based on MR to reduce the computer time as well as to improve the accuracy of STELF. The classical example from Europe is carried out. According

978-1-7281-6321-5/19//$31.00 ©2019 IEEE

to the quantitative analysis of error metric, the proposed model predicts the future load data with mean absolute percentage error less than 1.77%. The CPU time is reduced by 36% averagely when comparing with other models cited in this paper. The speedup ratio analysis verifies the parallel characteristics of the proposed algorithm, when the cluster is same, the larger the data size, the higher the speedup ratio is. The remainder of this paper is organized as follows, the framework of MapReduce is introduced in section II, the revised ENN model with CPGA strategy for future short-term load profile is developed in section III, Section IV presents the case study results and section V concludes the paper.

II. MAPREDUCE FRAMEWORK

MR is a kind of programming models that is associated with implementation for processing big data sets with parallel, distributed algorithms on clusters. An MR program is composed of a Map procedure (method) that performs filtering and sorting (such as sorting students by the first name into queues) and a Reduce method that performs summary operations (such as counting the number of students in each queue, yielding name frequencies). The "MapReduce System", also called "infrastructure" or "framework" orchestrates the processing by marshalling the distributed servers, running various tasks in parallel, managing all communications and data transfers between various parts of the system, and providing for redundancy and fault tolerance. A popular open-source implementation is a part of Apache Hadoop.

MR adopts the thought of divide-and-conquer, which will contribute to improve the efficiency of the computing. As shown in Fig.1, a MR model comprises both Map tasks that project a given data set called input records, then convert into another data set called intermediate records, and a Reduce task that combines intermediate records to get a desired final result.

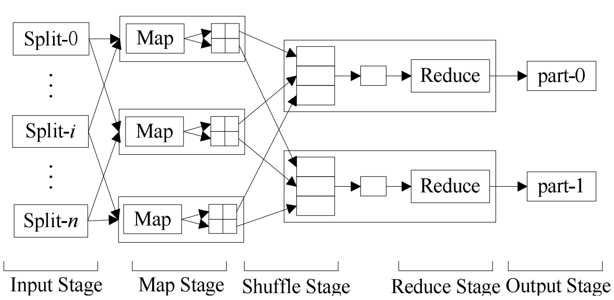

Fig. 1 The flowchart of MapReduce.

Another way to look at MR is a 5-step parallel and distributed computation:

1) Input stage: read the input data from a distributed file system and divide it into data slices, then allocate the slices to different maps;

2) Map stage: users need to write the Map function, which generates a set of intermediate <key/value> by running the function;

3) Shuffle stage: in this stage, transfer the <key/value> from a Map node to the Reduce node, merge the same key, and sort the key;

4) Reduce stage: users also need to create a reduce function, which merge the value according to the same key;

5) Output stage: output the results in the specified location of the distributed file system.

III. CPGA-ENN BASED ON MR

The predictive ability of neural networks is heavily dependent on their own topological structures and learning algorithms. As a class of recurrent neural networks, ENN integrates the initial and past states with the current states of neurons, exhibiting inherent dynamic behavior of nonlinear natures of electric load. In this paper, a revised ENN model with CPGA is developed for STELF, avoiding the slow convergence and randomicity resulted from the initial weight and threshold selection.

A. Coarse-Grained Parallel Genetic Algorithm

Genetic algorithms are generally used to generate high-quality solutions to optimization and search problems by relying on bio-inspired operators such as mutation, crossover and selection. The search mechanism of genetic algorithms(GAs) relies on stochastic process, which sometimes sticking to local optima or take considerable time when the number of data increases. The Coarse-Grained model is a distributed or island-based model, which can be the most adaptable and widely used in parallel genetic algorithms. In this model, the population is divided into a number of sub-populations or demes. Different processors maintain each of these sub-populations, which perform genetic operations concurrently and independently. By introducing migration operator, the model is able to exploit the differences in various sub-populations.

The Coarse-Grained model aims at simulating the behavior of complex systems using their coarse-grained (simplified) representation. It is possible to realize ideal speedup. The highest fitness individual can migrate to next sub-population via the circulation of comparing with the fitness values, selecting a new highest fitness individual, and migrating to the next sub-population again. Execute the above steps in cycle, and traverse all the sub-populations until the best individual is available. In brief, the Coarse-Grained model runs in parallel as a whole, and searches the global optimum with circular migration to traverse all the sub-populations.

B. Elman Neural Network

Fig.2 represents the basic structure of ENN, which consists in four layers: input layer, hidden layer, context layer and output layer. The neighbor layers are connected by adjustable weights and thresholds. As a kind of dynamic recurrent networks, ENN is considered as a special kind of feed-forward neural networks with additional memory neurons and the local feedback. The self-connections of the context nodes in ENN make it sensitive to the historical input data, which is very useful in dynamic system models.

The state space expression of the network is provided as:

$$y_h(k) = h(w_1 u(k-1) + w_2 y_c(k)) + b_1 \, . \tag{1}$$

$$y_c(k) = y_h(k-1)\,. \tag{2}$$

$$y_o(k) = e(w_3 y_h(k) + b_2)\,. \tag{3}$$

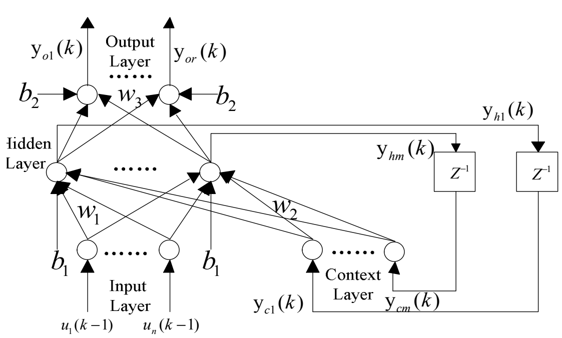

Fig. 2 The structure of ENN.

Where $u(k\text{-}1)$ is the input variables, w_1, w_2 and w_3 are the connection weights, b_1 and b_2 are the thresholds of the network. $y_c(k)$, $y_h(k)$ and $y_o(k)$ are the output of the context layer, the hidden layer and the whole network respectively. The transformation function $e(k)$ of the output layer is described by purelin expression, while the active function $h(k)$ of the hidden layer is defined by sigmoid function $h_{sig}(k)$

$$h_{sig}(k) = 1/(1+e^{-k})\,. \tag{4}$$

A wide variety of sigmoid functions has been used as the activation function of artificial neuron. During the training processes, sigmoid function exhibits a progression from small beginnings that accelerates and approaches a climax over time, making the convergence rate of a network slow and easy to falling into local extremum. To compensate, the constant a, gain parameter b, independent variable factor c and adjustable parameter d are introduced into the classical sigmoid function, one can obtain an improved active function as

$$h_{active}(k) = a + \frac{b}{1+e^{-c(k+d)}}\,. \tag{5}$$

Where, a and d affects the vertical and horizontal position of sigmoid function respectively. The learning speed of the network is related to the derivative of formula (6)

$$h'_{active}(k) = -\frac{c(h_{active}(k) - \frac{2a+b}{2})}{b} + \frac{cb}{4}\,. \tag{6}$$

It can be easily observed from (6), when $h_{active}(k)$ is close to $(2a+b)/2$, the fastest convergence rate is available, which is decided by b and c. In order to achieve a balance between convergence rate and prediction accuracy, the four parameters of a, b, c and d are chosen as a=-1, b=2, c=0.5 and d=2. Considering the integrity of the paper, the process of parameters grouping optimization of the four parameters is no longer discussed in detail.

C. Implementation of MR-CPGA-ENN

1) ENN with CPGA Optimization

In electrical engineering, a load profile is a graph of the variation in the electrical load versus time, which varies according to customer type (typically includes residential, commercial and industrial), temperature and holiday seasons. For a non-linear time series, the performance of ENN model is closely related to its network structure and parameters. A coarse-grained parallel genetic algorithm(CPGA) is used to search the optimal weights and thresholds of ENN in this paper. There are superiorities of the use of CPGA compared to alternative optimization algorithms. CPGA converges faster than classical GAs since the smaller size of the demes and parallel implementations while keeping code simple and organized. The migration operation prevents every group of demes from prematurely converging. With the basic process described in Fig.3, a single-threaded optimization is conducted, which includes the following steps:

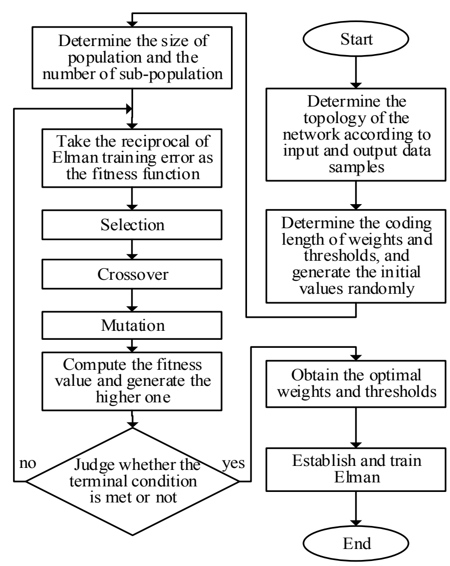

Fig.3 The weights and thresholds optimization flow chart of ENN.

step1: according to the input and output data samples, assume that S_1 denotes the number of neurons in the input layer, S_2 denotes that in the hidden layer, S_3 denotes that in the output layer, then the topology is denoted as S_1-S_2-S_3;

step2: based on the topology, the encoding length S can be deduced by

$$S = S_1 \times S_2 + S_2 \times S_2 + S_2 \times S_3 + S_2 + S_3\,. \tag{7}$$

Where $S_1 \times S_2$ denotes the number of connecting weights from the input layer to the hidden layer, $S_2 \times S_2$ denotes that from the context layer to the hidden layer, and $S_2 \times S_3$ denotes that from the hidden layer to the output layer;

step3: in the light of the encoding length, determine the size of the population and the number of sub-populations;

step4: take the reciprocal of the ENN training error as the fitness function

$$f_{fitness} = \frac{1}{(y_i + \hat{y}_i)^2}\,. \tag{8}$$

Where $f_{fitness}$ denotes the fitness value of the individual, y_i denotes the observed value of the *i-th* datum, $\hat{y}_i$ denotes the predicted value of the *i-th* datum. From the expression, the higher the fitness value, the better the result;

step5: do the operations of election based on the individual selection probability, which can be computed by

$$p_i = \frac{f_{fitness}}{\sum_{i=1}^{M} f_{fitness}}. \tag{9}$$

Where p_i denotes the selection probability of the *i-th* individual, M denotes the number of individuals;

step6: do the operations of crossover to generate new excellent individuals G'_{kj} and G'_{li}, using the following equations

$$\begin{cases} G'_{kj} = G_{kj}(1-v) + G_{kj} \cdot v \\ G'_{lj} = G_{lj}(1-v) + G_{lj} \cdot v \end{cases}. \tag{10}$$

which means that the *j-th* gene of the chromosome G_k crosses with the *j-th* gene of the chromosome G_l, v is a random number in the range of [0,1];

step7: do the operations of mutation to produce new individuals g'_{ij} with a high fitness value, then we have

$$g'_{kj} = \begin{cases} g_{ij}(g_{ij} - g_{\max}) \times f_{mutation}(k); & when \;\; m > 0.5 \\ g_{ij}(g_{\min} - g_{ij}) \times f_{mutation}(k); & when \;\; m \le 0.5 \end{cases}. \tag{11}$$

$$f_{mutation}(k) = r(1 - \frac{k}{k_{\max}})^2. \tag{12}$$

Where g_{max} and g_{min} correspond to the upper and lower limit of the mutation gene g_{ij} respectively, $f_{mutation}(k)$ is the coefficient of mutation, r is also a random number with range[0,1], k is the current iterations, k_{max} is the maximum number of iterations, m is the proportion of mutation. Normally m should be set between 0 and 1, where m>0.5 means the mutation proportion is over the half of the original genes, and m≤0.5 indicates that smaller or equal to the half of the original genes mutation are happened.

step8: compute the fitness value and reserve the better one; then judge whether the number of iterations meets the default set value, if yes, obtain the best weights and thresholds, establish and train the ENN, then complete the forecasting task; if not, go back to the genetic operations.

2) CPGA-ENN with MR Platform

It is unusual for an individual optimization algorithm to perform well in all cases. Especially in the bulk electrical data application, the optimal result will be raised when CPGA combines with MR, because the shuffle operation of the platform can be fully exploited. The searching time is probably decreased when a good trade-off between the computation and the communication costs is chosen. The procedure includes the following steps:

step1: generate N sub-populations, which contain from sub-pop 1 to sub-pop N;

step2: assign the N sub-populations to the N maps;

step3: conduct the genetic operation and compute the individual fitness;

step4: do the operation of the ring migration, select the highest fitness individual migrate to adjacent sub-population circularly, and compete with other individuals in the new environment to ensure excellent global solutions;

step5: each sub-population also has a reducer, and each reducer needs judge whether the generated individual is the best; if yes, output the results; if not, go to step3.

In tuning performance of MR, the complexity of mapping, shuffle, sorting, and reducing has to be taken into account. The mapping function is used to do the genetic operations to generate better individuals, while the reducing function is used to emit the best individual by merging the intermediate (Writable Key, Text individuals).

IV. Verification and Discussion

In general, besides the accuracy, the universality and stability of a forecasting model in different data sets should be evaluated. Accordingly, the case study with different electrical load characteristics are considered as the overall assessment of the proposed model.

A. Data Sources

As an example, the two-year half-hour load data set is provided by European Network on Intelligent Technologies [3]. Fig.4 describes the average daily load demand from 1997 to 1998. It is evident that the numbers are not intermixed accidentally, but are subject to regular periodicity. First, the demand has some time features: high demand for electricity in winter while low demand in summer. This feature implies the electricity usage is closely related to the weather conditions in different seasons. Besides, if the data set is further examined, there is another load feature that could be observed: a load periodicity happens in every week. Load demand in Saturday-to-Monday is usually lower than that of weekdays. In addition, electricity demand on Saturday is somewhat higher than that on Sunday. Further investigation of load data, such as statistical relations between the temperature and load change can be found, since the data set contains more details.

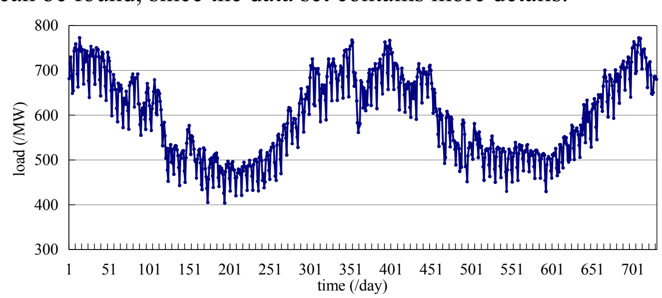

Fig. 4 The historical average daily load.

Although many factors have effects on the daily load demand, according to the data available, this paper would, without loss of generality for the forecasting method evaluation, focus on the temperature values and date type. As can be seen from Fig.5, the temperature and average daily load are approximately linear, and the correlation degree is approximately -0.873. Generally, load on Sunday or Saturday is lower than that on weekday (from Monday to Friday).

The similarity coefficient of date factors is calculated by the following formula:

$$COE = 1 - \left| r(x_i) - r(x_0) \right| . \tag{13}$$

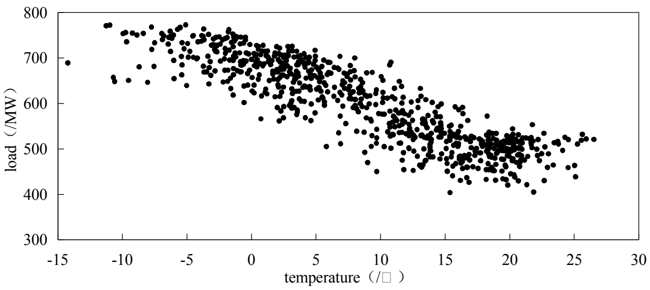

Fig. 5 The relationship between ambient temperature and average daily load.

Where *COE* denotes the correlation coefficient, x_i denotes the *i-th* date type, x_0 denotes the type of the predict day, $r(x_i)$ and $r(x_0)$ are the mapping value of x_i and x_0.

B. Forecasting Model

The learning and training process for the MR-CPGA-ENN model is shown in Fig.6.

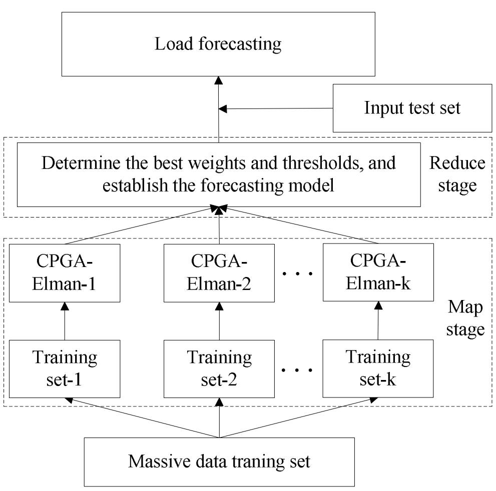

Fig. 6 The process of MR-CPGA-ENN implementation.

Firstly, receive different items of the training set.Secondly, execute mapping tasks, each mapping receives one training item and then computes all update-values for the weights in the network using this training item. Thirdly, execute several reducer tasks, each reducer gathers update-values for one weight, calculates an average value of these values, and outputs the average value as the update value for the weight. Fourthly, determine the best weights and thresholds in the network, and establish the forecasting model. Lastly, input the test set, and conduct the load forecasting.

C. Result Analysis

In order to improve the precision of load forecasting, the historical load data have to be cleaned in the data pre-processing procedure. Except for avoiding saturation phenomenon, the dimensions of different input samples need to be unified in advance. The normalization procedure for the input layer data is performed by formula (14), while the inverse normalization for the output layer data is accomplished by formula (15).

$$B = \frac{(B_{\max} - B_{\min}) \times (A_{\max} - A_{\min})}{A_{\max} - A_{\min}} + B_{\min} . \tag{14}$$

$$A = \frac{(A_{\max} - A_{\min}) \times (B_{\max} - B_{\min})}{B_{\max} - B_{\min}} + A_{\min} . \tag{15}$$

Where B_{min} and B_{max} are the minimum and maximum normalized values, respectively, A_{min} and A_{max} are the boundary values of sample data.

The simulation applied to all data sets is performed in the MATLAB environment. In this paper, ENN and SVR(support vector regression prediction algorithm) are employed as the benchmark models to evaluate the proposed model. To guarantee the fair experiments between MR-CPGA-ENN and the benchmark models, the number of neurons of the benchmark models is set according to the most accurate one after series of trials. The Hadoop platform, which contains ten nodes, is built to conduct this experiment. Generally, there are no clear rules to help to set the number of nodes of a Hadoop platform. In this study, the number of nodes is selected by trial and efficiency. Forecasting effectiveness can be measured by the absolute percentage error (APE), the mean absolute percentage error (MAPE), and sometimes the skewness and kurtosis of the distribution need consideration. When the task is the prediction of the maximum daily values of electrical load for January 2009, namely predicting maximum daily load of the next 31 days, the load of predicted value to the actual value are depicted in Fig.7. The statistic MAPE and MAE of different schemes are the main concern of our comparisons, which are widely selected to validate models as follows:

$$MEA = \frac{1}{N} \sum_{i=1}^{N} \left| y_i - \hat{y}_i \right| . \tag{16}$$

$$MEPA = \frac{1}{N} \left(\sum_{i=1}^{N} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100\% \right. . \tag{17}$$

Where N is the total number of data used for the performance evaluation and comparison.

With different schemes, a series of experiments are conducted. In Table □, the statistic MAPE and MAE values are for the accuracy test, meanwhile the response times are also recorded. Among all the models compared, the proposed MR-CPGA-ENN achieves the most satisfying MAPE, but only by a very small margin when compared with SVR.

Everything has two sides. Of course, the parallel genetic algorithm CPGA is benefited to optimize the parameters and weights in ENN. However, it should be noted the proposed MR-CPGA-ENN requires longer CPU time than ENN does.

$$\mathrm{var}(\hat{y}) = E(\hat{y} - \frac{1}{N} \sum_{i-1}^{N} \left| y_i - \hat{y}_i \right|)^2 . \tag{18}$$

$$MEPA = \frac{1}{N} \sum_{i-1}^{N} \hat{y}_i . \tag{19}$$

The performance variance(18) states the forecasting stability of the model, where (19)represents the expectation of the predicted value over all the predicted data. A smaller $Var(\hat{y})$ implies a more stable forecasting performance. The average variance of MR-CPGA-ENN, ENN and SVR for 31 days duration is 35.822, 36.101 and 30.407, respectively, which means that forecasting fluctuation in MR-CPGA-ENN needs to be improved in future research. Based on MapReduce weighted online sequential extreme learning machine MR-OSELM-WA[4] and FNN (Functional Neural Network)[5] are used to fulfill the comparative analysis. Although combining cloud computing technology, the prediction accuracy of FNN is obviously lower than that of MR-CPGA-ENN. As illustrated in TableII, when considering all the factors, the superiority of MR-CPGA-ENN is verified.

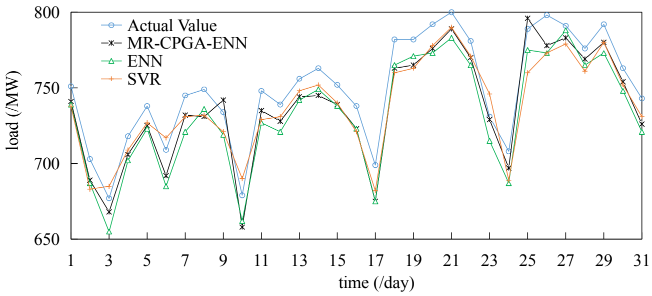

Fig. 7 The month load comparison of the predicted value to actual value.

TABLE □
THE FORECASTING RESULTS COMPARISON

Model	MAPE	MAE/MW	CPU Time/s
MR-CPGA-ENN	1.77%	13.19	479
ENN	2.66%	16.81	416
SVR	1.86%	14.26	1032
FNN	3.43%	19.13	683
MR-OSELM-WA	1.95%	15.85	574

In addition, in order to exploit the parallel performance of MR-CPGA-ENN, the speedup ratio analysis is studied by running the load forecasting experiment with 1, 5, 10, 20 distributed clusters for different data sizes of 820MB, 1640MB and 3280MB. The speedup ratio $S_{speedup}$ is to assess the parallel characteristic of the proposed algorithm.

$$S_{speedup} = \frac{t}{T}. \tag{20}$$

Where t is the running time of a single PC, T is the total cluster running time.

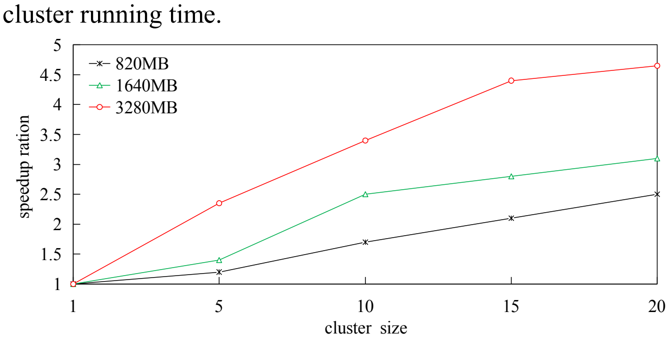

Fig. 8 The speedup ratio analysis.

As can be seen from Fig.8, the speedup ratio of MR-CPGA-ENN shows a nearly linear growth with the increase of cluster size. When cluster is same, the larger data size, the higher speedup ratio. However, for a determined data volume, when the cluster size increased to a certain value, the speedup ratio enters the saturated growth period. In this sense, the optimal cluster selection lies in the knee point for a particular scope.

V. VERIFICATION AND DISCUSSION

Electricity Load forecasting involves the accurate prediction of magnitudes and geographical locations over the different periods of the planning horizon. The basic quantity of interest is typically the hourly system load. In this paper, a parallel load forecasting technique with an Elman Neural Network based on the MapReduce programming model has been discussed and analyzed, which utilizing two separate historical hourly load data as illustrative examples to evaluate the effectiveness of the developed model. The key is to address both accuracy and efficiency of the forecasting model, which worked out from the background art of electric power big data. In addition, a coarse-grained parallel genetic algorithm is used to optimize the weights and thresholds of Elman Neural Network. Furthermore, MapReduce software platform accomplishes the parallel computing. In fact, the power big data are still in their early R&D stage. It is believed what have been done in this paper has a significant impact on electric power dispatch and market orientation.

ACKNOWLEDGMENT

This paper is supported by National Natural Science Foundation of China (No. 51967012), Scientific Research and Innovation Team Project of Gansu Education Department (No. 2018C-09) and State Grid Gansu Power Company Science and Technology Project (No. 522722170018).

REFERENCES

[1] Benjamin F Hobbsa, et al. Artificial neural networks for short-term energy forecasting: Accuracy and Economic Value,Neuro-computing, vol. 23, no.1-3, pp.71-84, Dec.1998.
[2] ChiragDeb, et al. A review on time series forecasting techniques for building energy consumption,Renewable and Sustainable Energy Reviews, vol.74, pp.902-92, July. 2017
[3] http://neuron-ai.tuke.sk/competition.
[4] BaoYi Wang, et al. A Distributed load forecasting algorithm based on cloud computing and extreme learning machine,Power System Technology, vol. 38, no. 2, pp. 526-531, Feb. 2014. (in Chinese)
[5] DeWen Wang and ZhiWei Sun. Big data analysis and parallel load forecasting of electric power user side,Proceedings of the Chinese Society for Electrical Engineering, vol. 35, no. 3, pp. 527-537, Feb. 2015.(in Chinese)