Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Integration of Deep Boltzmann Machine and Generalized Radial Basis Function Network for Photovoltaic Generation Output Forecasting

Shota Ogawa*, Hiroyuki Mori*

**Department of Network Design, Meiji University, Nakano-ku, Tokyo 164-8525, Japan (Tel:+81- 3-5343-8292; e-mail: hmori@meiji.ac.jp).*

Abstract: In this paper, an efficient method is proposed to deal with photovoltaic generation output forecasting with Deep Boltzmann Machine. In recent years, the penetration of photovoltaic generation has been widely spread in the world due to clean energy. However, it has brought about uncertainties for generation schedules in a way that power system operators have to consider the significant variations of generation output. As a result, the forecasting model of the generation output with high accuracy is required in the industries. This paper proposes a Deep Neural Network (DNN) model that integrates Deep Boltzmann Machine with Generalized Radial Basis Function Network (GRBFN) of Artificial Neural Network (ANN). The proposed model is tested for real data of photovoltaic generation output.

Keywords: Renewable energy systems, Photovoltaic generation, Forecasts, Time series analysis, Deep Neural networks, Artificial Neural Network, Optimization, Evolutionary computation, Swarm Intelligence

1. INTRODUCTION

In recent years, renewable energy is positively used to suppress the emission of carbon dioxide in thermal power plants that brings about global warming in the world. As renewable energy, photovoltaic generation, wind power generation, biomass generation, geothermal power generation, *etc.* are well–spread in the world. Japan has positively taken a policy to install photovoltaic generation plants due to the limitation of wind power generation sites. According to REN21 GSR2019 Report, Japan had photovoltaic generation amounts of 56GW into power systems as cumulative capacity in 2018, which corresponded to the third largest in the world and 6.5% of the total generation. However, photovoltaic generation output is significantly affected by weather conditions such as solar radiation, cloudiness, temperature, *etc*., which increases a lot of uncertainties occur in generation schedules like Economic Load Dispatching (ELD) and Unit Commitment (UC). As a result, it is necessary to take measures to smooth power system operation and planning. One of them is to come up with more accurate models for photovoltaic generation output forecasting. If the predicted generation output is available precisely, power system operators and planners are allowed to reduce the degree of uncertainties in generation scheduling. Therefore, it is imperative to develop new forecasting tools in power system operation and planning. Looking over the methods for photovoltaic generation output forecasting, there exist two categories. One is statistical methods that are based on classical time series analysis in signal processing. Chowdhury and Rahman developed Autoregressive Integrated Moving Average (ARIMA) model (Box, *et al.,* 2015) to investigate the model errors with several indices and sampling intervals (Chowdhury and Rahman, 1987). ARIMA and its variants have been used in engineering fields for long time, but it is well-known that their models provide more erroneous results for lack of nonlinear approximation functions. The other is Artificial Neural Networks (ANNs) that are mainly based on Multi-layer Perceptron (MLP) or Radial Basis Function Network (Powell, 1987) that makes use of the function of pattern recognition for time-series forecasting. A method with RBFN was presented for one-day ahead 24 hours forecasting (Yona, *et al.,* 2007). An MLP-based method with the improved backpropagation algorithm was developed for one-day ahead 24 hours forecasting (Ding, *et al.*, 2011).

Recently, as a new trend in ANNs, Deep Neural Networks (DNNs) have been widely spread to deal with nonlinear systems in image processing which is different from time series problems in a way that they often include nonlinear dynamic behaviour in a sense of prediction. Thus, it is necessary to consider what types of DNNs are applicable to time-series prediction problems. In DNNs, this paper focuses on techniques called pre-training that play a role to extract features of input variables in front of ANN and reduce the number of them from a standpoint of dimension reduction. Specifically, the following pre-training techniques are given:

a) Auto Encoder (AE) (Hinton & Salakhutdinov, 2006)

b) Restricted Boltzmann Machine (RBM) (Hinton & Salakhutdinov, 2006; Hinton, 2010)

Item a) is constructed by making test data same as learning data in the learning process and using only input and hidden layers, where it is assumed that the number of neurons at hidden layer is less than that at input layers and that brings about dimension reduction of the input variables. Also, Item b) is created by estimating the stochastic model from data distribution through Maximum Likelihood Estimation. Both models are not applicable to photovoltaic generation output forecasting because they are originally developed for image processing with binary variables. To overcome the problem, Gaussian-Gaussian RBM (GGRBM) with continuous variables was developed (Hinton, 2010). Ogawa and Mori applied GGRBM to photovoltaic generation output forecasting (Ogawa & Mori, 2019a, 2019b). However, there is still room for improvement for the above GGRBM method.

In this paper, a new method with Deep Boltzmann Machine (DBM) is proposed for photovoltaic generation output forecasting, where Deep Boltzmann Machine means GGRBM with deep layers. The differences between the previous works (Ogawa & Mori, 2019a, 2019b) and the proposed method may be described as follows:

1) DBM is used to improve the performance of GGRBM in terms of feature extraction of input variables.
2) Generalized Radial Basis Function Network (GRBFN) (Wettscheck & Dietterich, 1992) with better performance than RBFN is employed as the predictor in the proposed DNN. It should be noted that DNNs with pre-training need both pre-training ANN as the extractor and ANN as the predictor.
3) Scatter Search Predator Prey Brain Storm Optimization (SS-PPBSO) of high performance evolutionary computation is developed to estimate better weights in DNN. Scatter Search is a framework that makes the most of evolutionary computation efficiently (Glover, 1998) while PPBSO (Duan, Li, & Shi, 2013; Mori, Ogawa & Chiang, 2018) is the modified BSO that introduces Predator-Prey Strategy into BSO (Shi, 2011) to strengthen both intensification and diversification in search process.
4) The weight decayed method (Hinton, 1987) is applied to the cost function of the learning process to avoid overfitting of the prediction model.

The proposed method is successfully applied to real data of photovoltaic generation output forecasting in Japan.

2. DEEP BOLTZMANN MACHINE

This section describes Deep Boltzmann Machine (DBM) that plays a key role as pre-training in DNN. Hinton and Salakhutdinov extended RBM (Hinton & Salakhutdinov, 2006; Hinton, 2010) into DBM in a way that the number of layers increases from 2 to 3 and more. Basically, RBM consists of two layers that are called visible and hidden layers as shown in Fig. 1, where ν_i is state variable at visible unit i, and h_k is state variables of hidden unit k. There is high possibility that the increased layers bring about effective dimension reduction to improve the feature extraction of input variables. As the first stage of deep RBM with three layers and more, DBM means GGRBM with three layers for simplicity in this paper. In other words, DBM consists of a visible and two hidden layers as shown in Fig. 2, where ν_i is a state variable at visible unit i, and h^1_k and h^2_k are state variables of hidden unit k at hidden layers 1 and 2, respectively. Also, Symbols $\mathbf{W}^{(1)}$ and

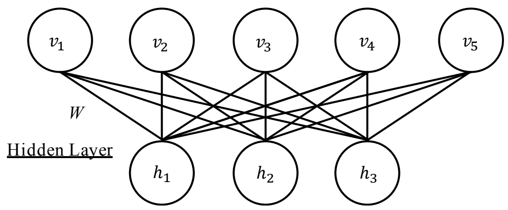

Fig. 1. Structure of DBM.

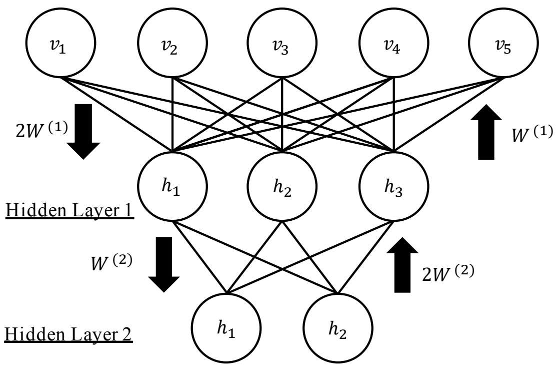

Fig. 2. Learning process in DBM with three layers.

$\mathbf{W}^{(2)}$ denote weight between neurons. Now, let us consider the mathematical formulation. Suppose the logarithmic likelihood function of each unit in DBM. It may be written as

$$log\, L(\boldsymbol{\theta}) = \sum_{\boldsymbol{n=1}}^{\boldsymbol{N}} log\, p(\boldsymbol{V_n}|\boldsymbol{\theta}) \tag{1}$$

where

L(•): likelihood function of •

N : No. of learning data

$\mathbf{V_n}$: n-th data

θ: model parameters

ν : state variables of visible units

p(•): probabilistic distribution of •

p(**ν**;**θ**): probabilistic distribution such as

$$p(\boldsymbol{v};\boldsymbol{\theta}) = \frac{1}{Z(\theta)} \sum_{\boldsymbol{h^1},\boldsymbol{h^2}} exp(-\phi(\boldsymbol{v},\boldsymbol{h^1},\boldsymbol{h^2};\boldsymbol{\theta})) \tag{2}$$

where Z (**θ**): partition function such as

$$Z(\theta) = \sum_{x,h} exp(-\phi(\boldsymbol{v},\boldsymbol{h^1},\boldsymbol{h^2},\boldsymbol{\theta})) \tag{3}$$

where Φ (•): energy function such as

$$\phi(\boldsymbol{v},\boldsymbol{h^1},\boldsymbol{h^2},\boldsymbol{\theta}) = -\sum_i a_i v_i - \sum_j b_j{}^{(1)} h_j{}^{(1)} - \sum_i \sum_j w_{ij}{}^{(1)} v_i h_j{}^{(1)} - \sum_j b_k{}^{(2)} h_k{}^{(2)} - \sum_i \sum_j w_{jk}{}^{(2)} h_j{}^{(1)} h_j{}^{(2)} \tag{4}$$

where

a_i: bias of visible unit i

ν_i: state of visible unit i

b_j: bias of visible unit j

h_j:: state of visible unit j

w_{ij}: weights

The parameters are obtained by maximizing (1) in terms of weights between neurons and parameters of energy function E, evaluating the updating terms and adding them to the initial values. The updating terms consists of the expected values of data and the model. Constructive Divergence of the sampling method is used to evaluate the updating terms since the expected value of the model is not estimated directly (Hinton, 2010). The algorithm of DBM with 3 layers may be summarized as follows:

Step 1: Provide learning data with Vsible Layer

Step 2: Evaluate output at Hidden Layer 1 by the Gaussian random number derived from data at Visible Layer.

Step 3: Use the results of Step 2 to calculate output at Visible Layer.

Step 4: Repeat Steps 2and 3 T times to obtain the expected value of weights between Visible Layer and Hidden Layer 1.

Step 5: Substitute the results of Step 4 and the expected value of data into the updating terms to update the parameters.

Step 6: Stop the learning process of Steps 1-5 to determine the parameters between Visible Layer and Hidden Layer 1 if the termination conditions are satisfied, and move to the next step.

Step 7: Provide learning data with Visible Layer and calculate output at Hidden Layer 1.

Step 8: Evaluate output at Hidden Layer 2 by the Gaussian random number derived from data at Hidden Layer 1.

Step 9: Calculate output at Hidden Layer 1 by the Gaussian random number derived from data at Hidden Layer 2.

Step 10: Repeat Steps 8 and 9 T times to obtain the expected value of weights between Hidden Layers 1 and 2.

Step 11: Substitute the results of Step 10 and the expected value of data into the updating terms to update the parameters.

Step 12: Stop the learning process of Steps 7-11 to determine the parameters between Hidden Layer 1 and 2 if the termination conditions are satisfied.

It should be note that there exist double weights between Visible Layer and Hidden Layer 1, and between Hidden Layers 1 and 2 because Hidden Layer 1 receives input from both Visible Layer and Hidden Layer 2 in Fig. 2. That enables to make the learning process of weights between Visible Layer and Hidden Layer 1, and between Hidden Layers 1 and 2 independent.

3. GRBFN

In this section, GRBFN of ANN is outlined. It is an extension of RBFN that consists of a weighted sum of the Gaussian functions. RBFN is based on the idea of Finite Mixture Model (FMM) (McLauchlan & Peel, 2000) and is different from MLP in the followings:

1) The weights between input and hidden layers are set to unity in RBFN although MLP needs optimize them by learning process.
2) RBFN makes use of Gaussian functions as the nonlinear mapping, but MLP employs the sigmoid function.

The mathematical formulation of RBFN may be written as

$$y = \sum_{i=1}^{n} w_i a_i \tag{5}$$

where
y: output
n: number of Gaussian functions
w_i: weights between hidden and output layers
a_i: output of unit i at hidden layer such as

$$a_i = exp\left(\frac{\|x-u_i\|^2}{\sigma_i^2}\right) \tag{6}$$

where

x: input vector

$\mathbf{u}_i$: centre of Gaussian function of unit i

σ_i^2: variance of Gaussian function of unit i

RBFN has the function of better nonlinear function approximation than MLP in dealing with nonlinear times-series forecasting (Mori & Iwashita, 2005). On the other hand, it has the following challenge:

- How to determine the parameters of the Gaussian functions, *i.e.*, the centres and the variances

To overcome the challenge, GRBFN was developed to evaluate them in the learning process as well as the weights between the hidden and output layers.

GRBFN evaluates the parameters by minimizing the following cost function in terms of w_i, u_i, and σ_i:

$$f = \frac{1}{2}\sum_{j=1}^{J}\left(y_j - t_j\right)^2 \tag{7}$$

where
f: cost function
J: number of learning data
y_j: j-th output
t_j: teaching signal for data y_j

4. PPBSO

This section describes PPBSO (Duan, *et al.,* 2013) that is an extension of BSO (Shi, 2011) of evolutionary computation. It stems from the analogy of brainstorming in the process of idea creation. To improve the performance of BSO in terms of search process, Predator-Prey Strategy (PPS) was applied to BSO. PPS plays an important role to imitate the relationship between predators and preys in ecosystems and strengthen intensification and diversification in solution search process. It consists of Predator and Prey Strategies that are selected randomly. Predator Strategy aims at intensively finding better solutions around good ones like the behaviour that Predator focuses on Prey. On the other hand, Prey Strategy evaluates better solutions by escaping from good solutions like the behaviour that Prey runs away from Predator. The use of PPS improves the performance of BSO by making use of information on the best solution and the centres.

The algorithm of PPBSO may be described as follows:

Step 1: Set the initial conditions.

Step 2: Prepare a set of initial solution candidates, classify them into clusters by k-means and set the best solution at each cluster to be the centre.

Step 3: Exchange the cluster with others in a certain probability, where one of the following rules is selected:

Rule 1: To select one of other centres

Rule 2: To select one solution candidate excluding the centre

Rule 3: To select two centres to create a new solution

Rule 4: To select two solution candidates excluding the centres to create a new solution.

Step 4: Select either Predator Strategy or Prey Strategy for the obtained solution at Step 3 randomly and update the solution as follows:

Moving rule of Predator Strategy:

$$X_{new}^d = X_{selected}^d + \xi N(\mu,\sigma) + w_{predator}(X_{g_{best}}^d - X_{selected}^d) \quad (8)$$

Moving Rule of Prey Strategy:

$$X_{new}^d = X_{selected}^d + \xi N(\mu,\sigma) - P \cdot a \cdot sgn(X_{center}^d - X_{selected}^d) e^{-b|X_{center}^d - X_{selected}^d|} \quad (9)$$

where

X_{new}^d: solution of d dimension after moving

$X_{selected}^d$: selected solution of d dimension

ξ : weight

$N(\mu,\sigma)$: Gaussian random numbers with mean μ and standard deviation σ

$w_{predator}$: weight for Predator

$X_{g_{best}}^d$: best solution in individuals

P: binary number to determine if Prey should run away from Predator

$sgn(\cdot)$: sign function

X_{center}^d: center in cluster

x_{span}: search space region

a: search region

b: search coefficient

Steps 1-4 are repeated to evaluate better solutions until the termination conditions are satisfied.

6. PROPOSED METHOD

6.1 SS-PPBSO

Before proposing SS-PPBSO, Scatter Search (SS) (Glover, 1998 ; Mori & Shimomugi, 2007) is described. It is a framework to improve the performance of evolutionary computation with several strategies. They consist of the following five concepts:

(A) Diversification Generation Method: A group of solution candidates are created.

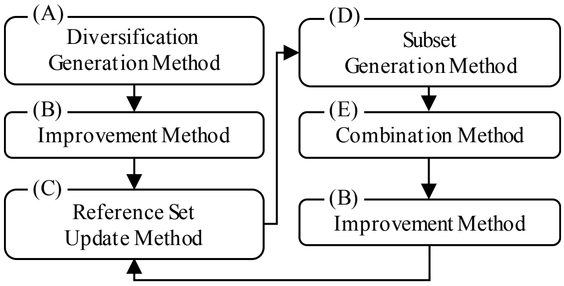

Fig. 3. Algorithm of Scatter Search.

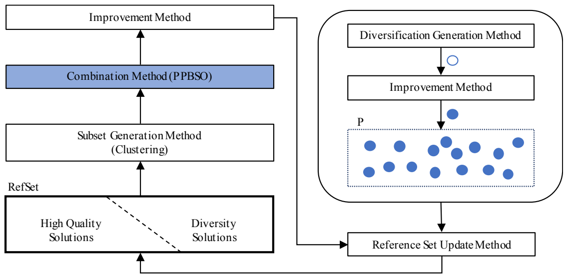

Fig. 4. Algorithm of SS-PPBSO.

(B) Improving Method: The obtained solutions are improved by finding better solutions around them with local search.

(C) Reference Set Update Method: The reference set which consists of better and diverse solutions selected from the solutions with Improving Method is updated.

(D) Subset Generation Method: The solution candidates are classified into several subsets in the reference set.

(E) Combination Method: New solutions are created by combing several solution candidates in a subset with those in other subsets with a certain method such as the center of gravity, linear combinations of solution candidates, etc.

SS is useful for improving the existing evolutionary computation since it includes both intensification and diversification strategies in search process. Fig. 3 shows the algorithm of SS, where the above concepts (A)-(E) are used to find better solutions and SS enhances the solutions of Combination Method with Improving Method and makes use of the improved solutions in Reference Set Update Method as shown in Fig. 3. The process is repeated until the termination conditions are satisfied. Fig. 3 shows an example of SS and there is possibility to construct a new structure of SS.

In this paper, the idea of SS is introduced into PPBSO, which is referred to as SS-PPBSO. It is necessary to consider the framework of SS that is suitable for PPBSO. The main feature is that furthermore it strengthens both intensification and diversification in search process. Fig. 4 shows the algorithm of SS-PPBSO. The algorithm may be described as follows:

Step 1: Generate a set of initial solutions by Diversification Generation Method and improved them by Multiple Start Local Search (MSLS) of Improving Method.

Step 2: Store the obtained solution candidates in Solution Set P and Select the solutions a by the rule of Reference Set Update Method.

Step 3: Decompose them into several subsets by Subset Generation Method.

Step 4: Carry out PPBSO of Combination Method to obtain better solutions.

Step 5: Run Multiple Start Local Search (MSLS) of Improving Method to improve solution candidates.

The above Steps 2-4 are repeated until the termination conditions are satisfied.

6.2 DNN-based Photovoltaic Generation Output Forecasting

This paper proposes a new method for photovoltaic generation output forecasting as shown in Fig. 5, where DNN consits of DBM and GRBFN. The features are given as follows:

1) To use DBM as pre-training of ANN to extract features of input variables
2) To employ GRBFN of ANN as a predictor that is more flexible for nonlinear approximations
3) To use SS-PPBSO for minimizing the ANN cost function
4) To apply Weight Decay Method to minimization of ANN cost function to avoid model overfitting

Regarding 1), 2) and 3), this paper already explained the concept, but item 4) is not explained. The mathematical formulation may be written as

$$f = \frac{1}{2}\sum_{j=1}^{I}(y_j - t_j)^2 + \lambda \sum_i p_i^2 \tag{10}$$

where
f : cost function
I : number of learning data
y_j: j-*th* output
t_j : teaching signal for data y_j
λ: penalty coefficient
pi: parameter i

It should be noted that the second term plays a role to avoid the model overfitting. The forecasting model with smaller parameters bring about high accuracy. The idea is very useful since time-series of photovoltaic generation output has high nonlinearity due to weather conditions.

7. SIMULATION

7.1 Conditions

a) The proposed method was applied to real data of photovoltaic generation output in Japan. The number of learning and test data are given as follows:

No. of learning data: 38851, No. of test data: 5760

The sampling time was 1 [min]. The proposed model used the following input and output variables:

Input variables:

x_1^t: photovoltaic generation output at time t

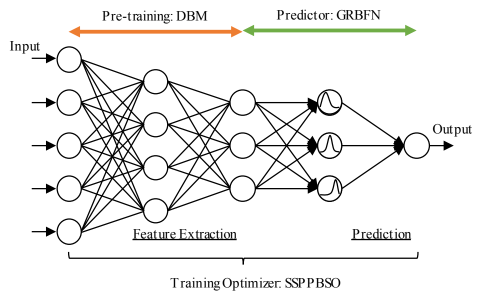

Fig. 5. Proposed method.

x_2^t: temperature of panel surface at time t

x_3^t- x_8^t: variances of photovoltaic generation output between time t and time t-k *(k=5,10,15,20, 25, 30)*

x_9^t- x_{14}^t: variances of panel surface between time t and time t-k *(k=5,10,15,20, 25, 30)*

x_{15}^t- x_{20}^t: first-order difference of photovoltaic generation output between time t and time t-k *(k=5,10,15,20, 25, 30)*

x_{21}^t- x_{26}^t: first-order difference of panel surface between time t and time t-k *(k=5,10,15,20, 25, 30)*

x_{27}^t- x_{32}^t: second-order difference of photovoltaic generation output between time t and time t-k *(k=5,10,15,20, 25, 30)*

x_{33}^t- x_{38}^t: second-order difference of panel surface between time t and time t-k *(k=5,10,15,20, 25, 30)*

x_{39}^t: temperature at time t

x_{40}^t: sunshine in minutes at time t

output variable:

y_1^{t+30}: 30minute ahead photovoltaic generation output at time t

b) The proposed method was compared with the conventional methods. For convineience, the following methods were defined :

Method A: MLP

Method B: AE+MLP

Method C: GGRBM+MLP

Method D: DBM+MLP

Method E: DBM+MLP+WD

Method F: DBM+MLP+PSO+WD

Method G: DBM+MLP+BSO+WD

Method H: DBM+MLP+PPBSO+WD

Method I: DBM+MLP+SS-PPBSO+WD

Method J: DBM+GRBFN+SS-PPBSO+WD(Proposed Method)

c) As the performance indices of forecasting errors, this paper employed the maximum, average and standard deviation (STD) of the errors

Table 1. Forecasting errors in each method

		Error indices		
		Max	Ave.	STD
A	MLP	0.650 (1)	0.143 (1)	0.188 (1)
B	AE+MLP	0.601 (0.92)	0.053 (0.37)	0.062 (0.33)
C	GGRBM+MLP	0.591 (0.91)	0.052 (0.36)	0.060 (0.32)
D	DBM+MLP	0.564 (0.87)	0.047 (0.33)	0.057 (0.31)
E	DBM+MLP +WD	0.557 (0.85)	0.044 (0.31)	0.056 (0.30)
F	DBM+MLP +PSO+WD	0.551 (0.84)	0.042 (0.29)	0.055 (0.29)
G	DBM+MLP +BSO+WD	0.544 (0.83)	0.038 (0.26)	0.053 (0.28)
H	DBM+MLP +PPBSO+WD	0.539 (0.82)	0.038 (0.26)	0.053 (0.28)
I	DBM+MLP +SS-PPBSO+WD	0.535 (0.82)	0.037 (0.26)	0.052 (0.28)
J	DBM+GRBFN +SS-PPBSO+WD (Proposed Method)	0.498 (0.76)	0.034 (0.24)	0.050 (0.26)

Note) Values in parentheses indicate data normalized by that of MLP.

7.2 Results

Table 1 shows the results for each method, where the values in parentheses show the errors normalized by the error of MLP. Method J of the proposed method succeeded in reducing the maximum, average and SD of MLP by 24%, 76%, and 74%, respectively. Also, looking at Method B with Auto Encoder (AE), Method E reduced the maximum, average and SD of Method B by 17%, 35%, and 21%, respectively. Next, compared with Method C of simple RBM, the proposed method reduced the maximum, average and SD of Method C by 16%, 33%, and 19%, respectively. It can be observed that DMB was much better than AE and RBM was a little bit better than AE. Compared with Method D, Method E provided better results due to the use of Weight Decay Method. Also, looking at Methods F-I, Method I gave better results than others, which means that SS-PPBSO is better than PSO, BSO, and PPBSO as evolutionary computaion. Method J outperformed Method I in the maximum, average and SD due to the use of GRBFN. Therefore, Method J of the proposed method was better than others. In particulr, it was a remarkable acivement that Method J reduced the average error of MLP by 76%.

8. CONCLUSIONS

This paper has proposed a new DNN-based method for photovoltaic generation output forecasting. The proposed method is based on the integration of DBM of extractor and GRBFN of predictor. Also, Weight Decay Method was introduced to the cost function of the learning process to avoid the model overfitting for highly nonlinear time-series of photovoltaic generation output. SS-PPBSO of high-performance evolutionary computation was developed to minimize the cost function. The proposed method was applied to real data in Japan. A comparison was made between the proposed and conventional methods in terms of the maximum and average errors as well as the standard deviation.The simulation results indicated that the proposed method provided better results Thus, the proposed method allows power system operators and planners to deal with photovoltaic generation output forecasting easily.

REFERENCES

Box, G.E.P Jenkins. G.M., Reinsel, G.C. and Lijung, G. M. (2015): Time Series Analysis : Forecasting and Control (5th Edition). Wiely, New York, USA.

Chowdhury, B.H. and Rahman S. (1987): Forecasting Sub-hourly Solar Irradiance for Prediction of Photovoltaic Output. Proc. of IEEE Photovoltaic Specialists Conference 1987, pp. 171-176.

Ding, M., Wang, L. and Bi, R. (2011): An ANN-based Approach for Forecasting the Power Output of Photovoltaic System. Procedia Environmental Sciences (Elsevier), Vol. 11, pp.1308-1315.

Duan, H., Li, S. and Shi, Y. (2013): Predator-Prey Brain Storm Optimization for DC Brushless Motor. IEEE Trans. on Magnetics, Vol. 49, No. 10, pp. 5336-5340.

Glover, F (1998): A Template for Scatter Search and Path Relinking. Artificial Evolution LNCS 1363, pp. 13-54.

Hinton, G.E. (1987): Learning Translation Invariant Recognition in a Massively Parallel Network. Proc. of PARLE 1987, pp. 1-13.

Hinton, G.E. and Salakhutdinov, S.S. (2006): Reducing the Dimensionality of Data with Neural Networks. Science, Vol. 313, No. 5786, pp. 504-507.

Hinton, G.E (2010): A Practical Guide to Training Restricted Boltzmann Machines. Technical Report 2010-003, Univ. of Toronto.

McLauchlan, G.J. and Peel, D. (2000): Finite Mixture Models. Wiley, New York, NY, USA.

Mori, H. and Iwashita, D. (2005): ANN-Based Risk Assessment for Short-Term Load Forecasting. Proc. of IEEE ISAP2005, pp. 446-451.

Mori, H. and Shimomugi, K. (2007): Transmission Network Expansion Planning with Scatter Search. Proc. of IEEE SMC2017, 6 pages.

Mori, H., Ogawa, S. and Chiang, H.D. (2018): Optimal Power Network Decomposition for Voltage and Reactive Power Control with Predator-Prey Brain Storm Optimization. Proc. of IFAC the 10th Symposium on CPES 2018, 6 pages.

Ogawa, S. and Mori, H. (2019a): Application of Evolutionary Deep Neural Network to Photovoltaic Generation Forecasting. Proc. of IEEE ISCAS2019, 4 pages.

Ogawa, S. and Mori, H. (2019b): A Gaussian-Gaussian-Restricted-Boltzmann-Machine-Based Deep Neural Network Technique for Photovoltaic System Generation Forecasting. Proc. of IFAC CSGRES2019, 6 pages.

Powell, M.J.D. (1987): Radial Basis Functions for Multivariable Interpolation: A Review in J.C. Mason and M.G. Cox, Eds., Algorithm for Approximation of Functions and Data, pp. 143-167, Oxford Univ. Press, UK.

Shi, Y. (2011): An Optimization Algorithm Based on Brainstorming Process. Int. J. Swarm Intel. Res., Vol. 2, No. 4, pp. 35-62.

Wettscheck, W.D. and Dietterich, T. (1992): Improving the Performance of Radial Basis Function Network by Learning Center Location. Advances in Neural Information Processing Systems 4, pp. 1133-1140.

Yona, A., Senjyu, T., Saber, A.Y., Funabashi, T., Sekine, H., and Kim, C.-H. (2007): Application of Neural Network to One-Day-Ahead 24 hours Generating Power Forecasting for Photovoltaic System. Proc. of IEEE ISAP2017, 6 pages.