


Integrating Gray Data Preprocessor and Deep Belief Network for Day-Ahead PV Power Output Forecast

Gary W. Chang , *Fellow, IEEE*, and Heng-Jiu Lu, *Member, IEEE*

Abstract—Generation output forecasting is a crucial task for planning and sizing of a photovoltaic (PV) power plant. The purpose of this paper is to present an effective model for day-ahead forecasting PV power output of a plant based on deepbelief network (DBN) combined with gray theory-based data preprocessor (GT-DBN), where the DBN attempts to learn high-level abstractions in historical PV output data by utilizing hierarchical architectures. Test results obtained by the proposed model are compared with those obtained by other five forecasting methods including autoregressive integrated moving average model, back propagation neural network, radial basis function neural network, support vector regression, and DBN alone. It shows that the proposed model is superior to other models in forecasting accuracy and is suitable for day-ahead PV power output prediction.

Index Terms—Neural networks, power generation planning, renewable energy, supervised learning, time series analysis.

ACRONYMS AND ABBREVIATIONS

| | |
|--------|---|
| AGO | Accumulated generating operation |
| AI | Artificial intelligence |
| ANN | Artificial neural networks |
| AR | Autoregressive |
| ARIMA | Autoregressive integrated moving average |
| ARMA | Autoregressive moving average |
| BPNN | Back propagation neural network |
| CD | Contrastive divergence |
| DBN | Deep belief network |
| FFN | Feed-forward network |
| GA | Genetic algorithm |
| GT-DBN | Grey theory combined with deep belief network |
| IAGO | Inverse accumulating generation operator |
| MAPE | Mean absolute percentage error |
| ML | Machining learning |
| PV | Photovoltaic |
| RBFNN | Radial basis function neural network |
| RBM | Restricted Boltzmann machine |
| RMSPE | Root mean squared percentage error |
| RNN | Recurrent neural networks |

Manuscript received April 13, 2018; revised August 10, 2018 and November 13, 2018; accepted December 12, 2018. Date of publication December 18, 2018; date of current version December 18, 2019. This work was supported in part by the Ministry of Science and Technology of Taiwan under Grants MOST 106-2221-E-194-041-MY2, and MOST 107-3113-E-194-001. Paper no. TSTE-00296-2018. (*Corresponding author: Gary W. Chang.*)

The authors are with the Department of Electrical Engineering, National Chung Cheng University, Chia-Yi 62102, Taiwan (e-mail: garywkchang@mail.com; lhengjiu@gmail.com).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TSTE.2018.2888548

SVM Support vector machine
SVR Support vector regression

I. INTRODUCTION

RENEWABLE energy such as the photovoltaic (PV) power generation is regarded as a potential resource of electricity in many areas worldwide. Though the power output of the PV plant helps reduce carbon dioxide and mitigates global warming, high penetration of PV generation also carries several challenges in grid operations and planning, mainly due to its intermittent generation nature [1]. It is known that short-term forecasts are useful in power generation planning and for electricity trading in power markets where PV power and energy storage can be traded or hedged. With the increasing penetration of PV generation, accurate forecasting techniques of PV power output play a critical role in improving energy market efficiency and reducing the amount of reserves while maintaining the system security.

In practice, there are different input data models for forecasting methods according to the time horizons required for the study. For instance, statistical models such as autoregressive (AR), autoregressive moving average (ARMA), and artificial neural networks (ANN) are for short-time forecasts. Artificial intelligence (AI) and machining learning (ML) based approaches are frequently adopted for PV output power forecasting, where commonly seen techniques include back propagation neural network (BPNN) [2], recurrent neural networks (RNN) [3], radial basis function neural network (RBFNN) [4], and support vector machine (SVM) [5]. In general, the AI and ML models can better handle non-linear relationship between input and output and are more flexible in forecast applications; however, they describe the relationship in implicit ways and sometimes are computationally intensive.

In recent years, several methods for solar power output forecasting have been proposed. In [4] the measured PV power and meteorological forecasts including solar irradiance, temperature, and relative humidity at the plant site are served as input. The 24-hour ahead local weather type then is trained and classified by a self-organized map provided by the online meteorological services to improve the forecast accuracy of the network. Reference [5] analyzes four-year historical satellite images and utilized such information to configure a large number of input and output data sets for the support vector machine (SVM) learning. In [6] it describes the historical power output and forecasted irradiance fed into an autoregressive with exogenous input model to generate six-hour look-ahead power output forecasts with a normalized root mean square error serving as

the accuracy index. Reference [7] presents using the historical PV power output and forecast temperature input to a recurrent neural network to predict 24-hour look-ahead power. In [8] the forecasted high, medium, and low temperatures are to classify historical power output. Three feedforward neural networks are adopted to generate 24-hour look-ahead forecasts. Reference [9] compares eleven machine learning-based methods applied to online and offline training methodologies to identify the most suitable training mode. Results show that SVM can assure the minimal prediction error and satisfying accuracy with the optimal dataset length. In [10] it shows a linear relationship between the global daily solar radiation on a horizontal surface and the power output of the sunshine hours at the clear sky with the daily maximum temperature variation is achieved based on one-day forecast. Similar conclusions are also reported in [11] for the forecast obtained by hybrid methods.

Among the aforementioned ANN-based and hybrid methods, the RBFNN has a wide range of applications for time series prediction [12]. Several advantages of RBFNN over other ANN-based methods include its capability of approximating highly nonlinear functions, the sequential manner of training process, and the use of local approximation [13], [14]. However, when applying RBFNN for forecasting, the challenge is that it is difficult to decide the initial values of parameters of the center and standard deviation for each Gaussian basis function of a hidden neuron. Some approaches have been proposed to determine such parameters [15].

With the advances of CPU and GPU computational capabilities and machine learning algorithms, many deep learning-based approaches have been proposed for forecasting purposes [16]–[19]. However, applications of deep learning for PV output forecasts still need further explorations. In addition, the hierarchical structure does not guarantee the good performance of PV time-series forecasting.

To improve the PV output forecasting accuracy and overcome the aforementioned shortcomings, this paper proposes a deep belief network (DBN)-based model that is combined with grey theory-based preprocessor for input data smoothing and the supervised method of the feed-forward neural network to fine-tune the training data. A grey system means that a system in which part of information is known and part of information is unknown. Grey theory (GT) has been shown to be useful to transform the irregular data series into the new data series with better regularity by using a data generation scheme. When trained on a set of data without supervision using restricted Boltzmann machine (RBM), a DBN can probabilistically reconstruct for input and learn the weights more accurately than traditional neural networks such as BPNN and RBFNN. The RBM is an undirected and generative energy-based model with a visible input layer and a hidden layer, where connections are between layers and not within layers. The proposed GT-DBN model can tackle the regression issue with a training algorithm that greedily trains one layer at a time, exploiting an unsupervised training algorithm for each layer and apply the feed-forward network (FFN) as the fine-tune method at the last layer to perform the supervised learning. The DBN has become well-recognized approach and widely applied in traditional AI application domains such as

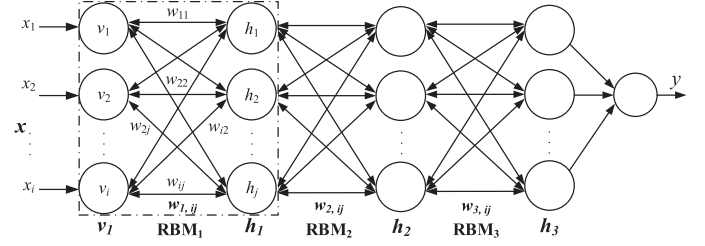


Fig. 1. A structure of DBN with three RBMs.

semantic parsing, natural language processing, and especially in computer vision tasks [20], [21].

The novelty of the proposed model is that the grey data preprocessor and the supervised model of DBN combined with FFN are advantageous in PV output forecasting applications, especially when the input data are strongly irregular. The overfitting problem also can be mitigated through the proposed grey data preprocessor. Forecasting results show that, under normal sunny day with stationary PV output, the proposed GT-DBN model is slightly better than the DBN model alone in forecasting accuracy and computational efficiency, and yields much higher forecasting accuracy when the PV output is non-stationary and is with more spikes due to other weather conditions. Test results also show that the proposed model is superior to those obtained by other compared models in forecast accuracy.

The organization of this paper is as follows. In Section II, it gives an overview of the deep belief network approach. In Section III and IV, the proposed method and the solution procedures are described. Section V presents the test results obtained by using the proposed and other methods under comparisons. It is followed by the conclusions in Section VI.

II. OVERVIEW OF STRUCTURE OF DEEP BELIEF NETWORK WITH RESTRICTED BOLTZMANN MACHINES

The deep learning has been extensively studied in the computer vision area and a good number of related approaches have emerged in recent years. The applications of convolutional neural network are commonly seen in feature classifications [22], while several deep learning approaches can be found in time series forecasting for weather conditions and traffic flows [16]–[19]. Among many available deep learning techniques, the DBN is a generative probabilistic model with an input and an output layers, and in between are several layers of hidden stochastic units trained by composing RBMs. The following briefly reviews the DBN structure with RBMs for time series forecast.

A. Restricted Boltzmann Machine

The RBM is a two-layer neural network served as major components of deep belief network, as shown in the dashed block of Fig. 1, where x_i is the i -th input data and w_{ij} is the bi-directional weight between the i -th node of the visible layer and the j -th node of the hidden layer. The RBM represents the data by mapping the raw features into latent binary factors [24], [25]. The first layer of a RBM is the input layer called visible

layer (indicated by v_i for its nodes) while the second layer is the hidden layer (indicated by h_i for its nodes). A RBM is further restricted to abandon visible-visible and hidden-hidden connections between different nodes in the same layer. Each node is a computational locus that performs input data processing and randomly determines the decisions for transmitting the data.

B. Deep Belief Network

The DBN is a feed-forward neural network with many hidden layers. It is one of the most commonly used deep learning methods. For example, the DBN shown in Fig. 1 is with three RBMs, where the input vector is \mathbf{x} , $\mathbf{w}_{l,i,j}$ is the weight vector for the l -th RBM, $l = 1, 2, 3$, and the single output is y . In Fig. 1 the hidden layer \mathbf{h}_1 of RBM₁ is served the visible layer, \mathbf{v}_2 , of RBM₂, and so forth. It is noted that the number of nodes at each hidden layer is not necessary the same as those of different hidden layers.

The following section further describes the theories regarding the RBM and DBN used in the proposed PV output forecast.

III. PROPOSED METHOD

This paper proposes a DBN forecasting model in which the grey theory is used as the input data preprocessor (GT-DBN) and a feed-forward neural network (FFN) after the DBN is to calculate the single forecasted output. The proposed forecasting model is firstly built by training a set of 24-hour actual measured input PV power output data samples. Then the trained parameters associated with the DBN are applied to forecast the 24-hour look-ahead power output. The proposed methods used for the PV output forecast are elaborated below.

A. Gray Theory

Grey theory has been applied in many areas of research [26], and it mainly concerns incomplete or not fully defined system. For instance, if the internal structures and features of a system are completely unknown, the system is usually denoted as “black”. On the contrary, “white” means that the internal features of a system are fully explored. Between the white and black, there is a grey system indicating that part of the information is clear, while another part is still unknown. The system was named by using grey as the color that indicates the system contains both known and unknown information.

This paper applies the accumulated generating operation (AGO) of grey theory to transform the irregular PV power output data into the new output data set with strong regularity by using a data generation scheme. The more the times of accumulation, the more evident the data series be described by the exponential function. If the randomness of the data obtained from a grey theory is smoothed, it is easier to derive the specific characteristics of that system. Then, by using the predicted values obtained in AGO, the inverse accumulating generation operator (IAGO) is applied to find the predicted values of the original data.

Consider a time sequence $\mathbf{q}^{(0)}$ of (1) that denotes the original discrete data series.

$$\mathbf{q}^{(0)} = \{q^{(0)}(1), q^{(0)}(2), \dots, q^{(0)}(p)\}, \quad p \geq 4 \quad (1)$$

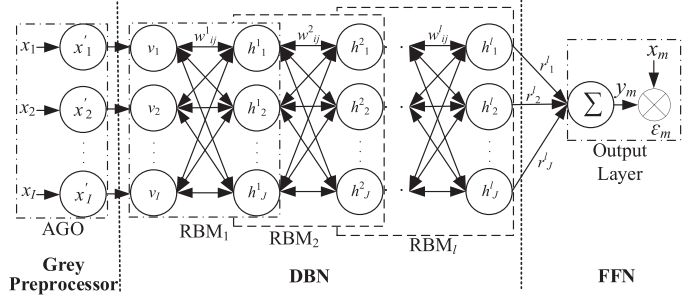


Fig. 2. Proposed DBN-based forecasting model with the input gray preprocessor and the output feedforward network for training.

where $\mathbf{q}^{(0)}$ is a non-negative sequence and p is the sample size of the data. When the sequence of (1) is subjected to the AGO, the following sequence, $\mathbf{q}^{(1)}$, is obtained.

$$\mathbf{q}^{(1)} = \{q^{(1)}(1), q^{(1)}(2), \dots, q^{(1)}(p)\}, \quad p \geq 4 \quad (2)$$

where $q^{(1)}(m) = \sum_{h=1}^m q^{(0)}(h)$, $m \leq p$.

To obtain the predicted value of the primitive data at time instant s , the IAGO can be used to reverse the forecasted values of (2) to obtain the forecasted data series from (3).

$$q^{(0)}(s) = q^{(1)}(s) - q^{(1)}(s-1), \quad s = 2, 3, \dots, p \quad (3)$$

B. Gray Theory-Based Deep Belief Network (GT-DBN)

Fig. 2 depicts the proposed DBN-based model with preprocessed input data obtained from the grey preprocessor for training [24], [25]. In the DBN, after a RBM has been trained, the follow-up activities of its hidden units can be applied as the input data for training the next level of RBM. As described in Section II-A, a RBM is an undirected graphical model in which visible variables, \mathbf{v} , are connected to stochastic hidden units, \mathbf{h} , using undirected weighted connections [27]. They are restricted that there are no connections within hidden variables or visible variables. The model defines a probability distribution over \mathbf{v} and \mathbf{h} via an energy function of (4).

$$\begin{aligned} E(\mathbf{v}, \mathbf{h}) &= -\mathbf{a}^T \mathbf{v} - \mathbf{b}^T \mathbf{h} - \mathbf{v}^T \mathbf{w} \mathbf{h} \\ &= -\sum_{i=1}^I a_i v_i - \sum_{j=1}^J b_j h_j - \sum_{i=1}^I \sum_{j=1}^J v_i w_{ij} h_j \end{aligned} \quad (4)$$

where w_{ij} is the bi-direction weight between the units of hidden layers; b_j and a_i are the corresponding bias; I and J stand for the numbers of v_i and of h_j , respectively. y_m and x_m are the m -th estimated and desired outputs, respectively, and the difference is ϵ_m , where $m = 1, \dots, M$, and where M is the total number of output data to be forecasted.

The joint probability over the visible and hidden units can be defined by

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

where Z is the partition function defined as the sum of $e^{-E(\mathbf{v}, \mathbf{h})}$ over all possible configurations. The two probability

distributions of (5) associated with \mathbf{v} and \mathbf{h} are given in (6).

$$P(\varphi) = \frac{1}{Z} \sum_{\varphi} e^{-E(\mathbf{v}, \mathbf{h})}, \quad \varphi = \mathbf{v} \text{ or } \mathbf{h} \quad (6)$$

Because of the specific structure of the RBM, the visible and hidden nodes are conditionally independent to each other. For binary state units (i.e., v_i and $h_j \in [0,1]$), the conditional probability distributions P are given in (7) and (8), respectively, when $v_i = 1$ or $h_j = 1$.

$$P(h_j = 1 | \mathbf{v}) = \sigma \left(b_j + \sum_{i=1}^I w_{ij} v_i \right) \quad (7)$$

$$P(v_i = 1 | \mathbf{h}) = \sigma \left(a_i + \sum_{j=1}^J w_{ij} h_j \right) \quad (8)$$

where $\sigma(\bullet)$ is the sigmoid function, $\sigma(u) = 1/(1+e^{-u})$.

To train the RBM, the maximum likelihood estimation is adopted. Given a training set, the log likelihood of the model for a single training sample is

$$\begin{aligned} L(\theta) &= \log P(\mathbf{v} | \theta) \\ &= \log \sum_{\mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})} - \log \sum_{\mathbf{v}, \mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})} \end{aligned} \quad (9)$$

where $\theta = \{\mathbf{a}, \mathbf{b}, \mathbf{w}\}$ is the parameters set to be estimated. The gradient of (9) thus becomes

$$\begin{aligned} \frac{\partial L}{\partial \theta} &= \frac{\partial}{\partial \theta} \left(\log \sum_{\mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})} - \log \sum_{\mathbf{v}, \mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})} \right) \\ &= \sum_{\mathbf{h}} P(\mathbf{h} | \mathbf{v}) \left(\frac{-\partial E(\mathbf{v}, \mathbf{h})}{\partial \theta} \right) \\ &\quad - \sum_{\mathbf{v}, \mathbf{h}} P(\mathbf{v}, \mathbf{h}) \left(\frac{-\partial E(\mathbf{v}, \mathbf{h})}{\partial \theta} \right) \\ &= \hat{\theta}_{data} - \hat{\theta}_{model} \end{aligned} \quad (10)$$

In (10) the first term can be easily calculated and the second term needs to traverse all possible value combinations of the visible and hidden units. The partial derivatives of the energy function to model parameters are given in (11)–(13).

$$-\frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial w_{ij}} = v_i h_j \quad (11)$$

$$-\frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial a_i} = v_i \quad (12)$$

$$-\frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial b_j} = h_j \quad (13)$$

To solve the log maximum likelihood problem of (9), the contrastive divergence (CD) algorithm in which $\hat{\theta}_{model}$ of (10) is obtained using Gibbs sampling method has been proposed [23], [28]. The Gibbs sampling starts with a training sample, and alternately samples the hidden and visible units using (7) and (8) by k steps (k is an integer). When $k \rightarrow \infty$, the accurate model distribution can be obtained and $\hat{\theta}_{model}$ can be calculated. Reference [23] indicates that the CD learning with $k = 1$ can provide adequate results to properly estimate the model gradient.

Therefore, it can be estimated using Gibbs sampling as

$$\hat{\theta}_{model} = \frac{1}{G} \sum_G \sum_h P(\mathbf{h} | \mathbf{v}) \left(\frac{-\partial E(\mathbf{v}^{(1)}, \mathbf{h}^{(1)})}{\partial \theta} \right) \quad (14)$$

where G is the number of samples used to estimate the model representing the distribution in Gibbs sampling.

In practice, the training data is divided into mini-batches to enhance the computing efficiency. A commonly used strategy is to set G being equal to the size of the mini-batch. The total gradient is also divided into “mini-batch” by the size of the data mini-batch to avoid changing the learning rate when the size of a mini-batch changes. Therefore, the updating rules of the parameters using stochastic gradient descent algorithm can be given by

$$\theta := \theta + \eta \Delta \theta = \theta + \eta (\hat{\theta}_{data} - \hat{\theta}_{model}) \quad (15)$$

where η is the learning rate. The gradient of the three parameter sets defined in θ for a mini-batch with the size of G can be expanded as

$$\Delta w_{ij} = \frac{\sum_{g=1}^G (v_{(g),i}^{(0)} h_{(g),j}^{(0)} - v_{(g),i}^{(k)} h_{(g),j}^{(k)})}{G} \quad (16)$$

$$\Delta a_i = \frac{\sum_{g=1}^G (v_{(g),i}^{(0)} - v_{(g),i}^{(k)})}{G} \quad (17)$$

$$\Delta b_j = \frac{\sum_{g=1}^G (h_{(g),j}^{(0)} - h_{(g),j}^{(k)})}{G} \quad (18)$$

where $v_{(g),i}^{(k)}$ and $h_{(g),i}^{(k)}$ denote the i -th element of the g -th training sample corresponding to \mathbf{v} and \mathbf{h} , respectively, and k implies the sample obtained after k -step Gibbs sampling. In (16) and (17), $k = 1$ according to [23].

The structure of DBN is trained layer by layer with a set of input data on a series of RBM units. After all RBMs are sequentially trained, the parameters of the energy function associated with the last RBM are then saved. The output of the DBN (i.e., the output of the last RBM) is served as the input to the feedforward network (i.e., FFN, the output layer shown in Fig. 2). The output layer then computes a linear weighted sum of the hidden neuron outputs using (19).

$$y_m = \sum_{j=1}^J r_j^l E(\mathbf{v}^l, \mathbf{h}^l) \quad (19)$$

where r_j^l is the weight of the hidden layer connecting to the output layer between the j -th hidden neuron and the output neuron and l indicates the last number of RBMs. The learning process of the network can be viewed as a search for free parameters by minimizing a predefined criterion function, such as the mean squared error. The square of the difference between the target output, x_m , and the inferred output, y_m , forms the error function of (20) that is to be minimized.

$$\xi(n) = \frac{1}{2} (x_m - y_m)^2 = \frac{1}{2} \varepsilon_m^2(n), \quad m = 1, \dots, M \quad (20)$$

where n is the iteration index.

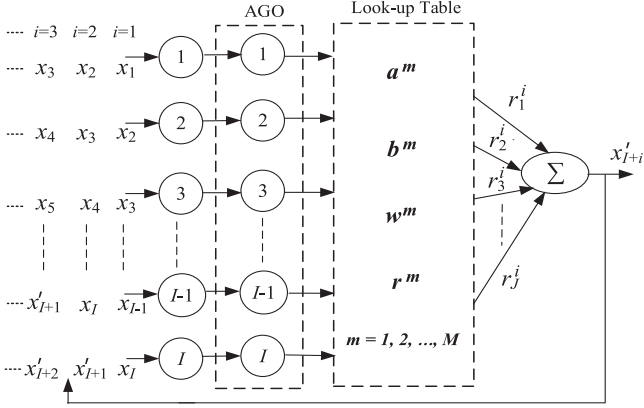


Fig. 3. Recalling (forecasting) structure prior to performing IAGO based on the proposed model.

C. Proposed Procedure for PV Power Output Forecast

The proposed solution procedure combines DBN with grey data preprocessor to model the non-linear components of the forecasted PV output more efficiently. At the first stage in the learning process, the raw PV output power data are transformed to the new PV output power data by AGO. The preprocessed PV data are then input to the RBM model. As the learning progresses, the DBN model will more closely approximate the true deterministic component and hence the feedforward network output after the last RBM approaches the more accurate forecast.

As shown in Fig. 2, it illustrates the proposed GT-DBN forecasting scheme for training purpose. The GT-DBN processes the time series of I neurons to predict the output. In addition, the proposed model of Fig. 2 includes the PV data input layer, x_1 through x_I , to the visible layer, v , and to several hidden layers, h^1 through h^l , where l is the number of RBMs. Then, the output neuron in the feedforward network forecasts the PV power output by the supervised learning method (i.e., $\varepsilon_m = x_m - y_m$, $m = 1, 2, \dots, M$, and M is the total number of forecasted PV output).

In the proposed solution procedure for PV power output forecast, the first stage is to transform the continuous input of I power output data of PV by AGO, and then to train the I continuous AGO output data points, x'_1 through x'_I , by DBN. During the training process, the output of the last RBM is to train the weight connecting to the output layer, r_j , by the gradient descent method. A look-up table for 24-hour look-ahead forecasts at one-minute resolution with the size of $60 \times 24 \times (I + J + I \times J + J)$ is built to save the parameters of a_i , b_j , w_{ij} , and r_j for the energy function of (20) of the last RBM and the feedforward network weights, where $i = 1, 2, \dots, I$ and $j = 1, 2, \dots, J$. After the proposed network of Fig. 2 is trained, the look-up table is used in the recalling process to obtain the forecasted PV power output by providing I points of new (untrained) input data to the forecasting process shown in Fig. 3, where the recalling structure before performing IAGO to produce the PV output forecast is illustrated.

C1. Solution Procedure for Training the DBN With Actual Measured PV Power Output Data (Learning Process)

Listed below describes the major steps for training the proposed method for PV power output forecast.

1. Assign the number of learning cycles of the feedforward network, the learning rates, the number of RBMs, the numbers of neurons of the input, hidden, and output layers, the number of G , and the convergence tolerance of (20).
2. Take consecutive 24-hour of measured PV power output data set plus I more data points at one-minute resolution (i.e., $M + I = 60 \times 24 + I$ data points) and obtain the new data set of PV output after preprocessed by AGO of (2).
3. Initialize the look-up table for storing the parameters of the last RBM and the weights connecting to the feedforward output layer (i.e., a , b , w , and r), as well as the corresponding input data set obtained by AGO. Randomly assign the initial values of a , b , w , and r .
4. Enter a set of I consecutive samples pre-processed by AGO obtained at step 2 as the input vector.
5. Calculate the RBM energy function of a , b , and w of (4).
6. Calculate the three parameters by using (16)–(18) based on the k -step ($k = 1$) Gibbs sampling and update the values.
7. Check if the maximum number of RBMs has been reached. If yes, proceed to next step. Otherwise, return to step 5 for next RBM.
8. Calculate the target output value of the feedforward network by (19).
9. Check if (20) is within the tolerance. If yes, proceed to next step. Otherwise, update the weights by (21) and (22) until the convergence criteria of (20) is met.

$$\frac{\partial \xi(n)}{\partial r_j^l} = 0, j = 1, 2, \dots, J \quad (21)$$

$$r_j^l : = r_j^l + \eta \xi(n) E(v^l, h^l), j = 1, 2, \dots, J \quad (22)$$

10. Check if the number of learning cycles, $n = N_{max}$, is reached. If yes, proceed to next step. Otherwise, return to step 8.
11. Check if the last input sample has been trained. If yes, stop the training and output the updated parameter sets associated with the M forecasted output data points. If not, enter the next set of I consecutive training samples that include the second through the last one of present set of training samples and one additional adjacent input data as the new sample.

Fig. 4 depicts the flowchart of the training procedure based on the proposed GT-DBN model for PV output forecast.

C2. Solution Procedure for Forecasting PV Power Output (Recalling Process)

After the GT-DBN has been trained in the learning process, a new set of I measured PV output are input to the model to forecast the output up to 24-hour (i.e., $M = 60 \times 24$ output data points). Listed below are major steps of the recalling process of the proposed solution procedure.

1. Read the parameters of GT-DBN (i.e., the number of RBMs, the number of neurons at each layer) and the

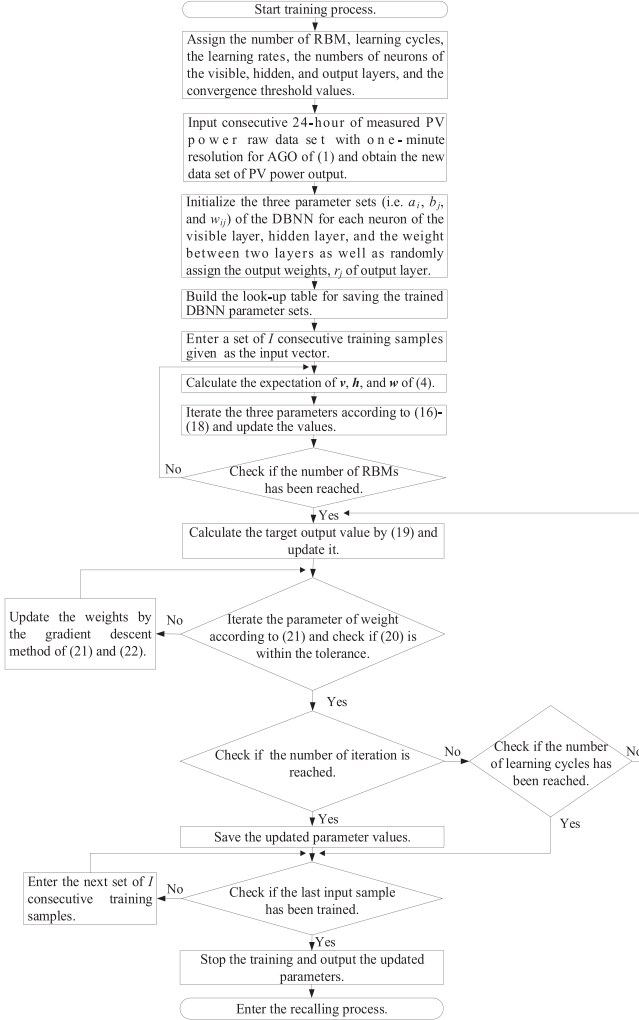


Fig. 4. Flowchart of the training procedure based on the proposed GT-DBN model for PV output forecast.

maximum number of PV power output data to be forecasted. Initialize the input vector count, $h = 1$.

2. Input an untrained set of I actual measured PV output data one after another to serve as initial input vector to be transformed by AGO of (2).
3. Obtain the new data set by AGO of (2) to serve as input vector, x_h , to the model.
4. Perform the data similarity check by using the minimum value of the differences between the input data set and each trained data set used in the learning process. The minimum difference occurred at a specific training data set indicates that the associated saved parameters are candidates used for forecasting PV output.
5. Identify the parameters saved in the look-up table corresponding to the data set obtained at step 2 to calculate the forecasted output by (19).
6. Check if the maximum number (i.e., M) of output samples forecasted has been reached. If yes, stop and proceed to next step. Otherwise, increase the input data vector count h by 1 and enter the next set of I consecutive input samples that include the second through the last one of present

TABLE I
MAJOR PARAMETERS OF COMPARED METHODS

| Model | Parameters |
|--------------|--|
| ARIMA(p,d,q) | p=6, d=1, q=5 |
| BPNN | The number of learning cycles, the learning rate, η , the convergence threshold value, ε , are 1200, 0.06, and 10^{-6} , respectively. The numbers of neurons of input, hidden, and output layers are 6, 6, and 1, respectively. Initial value of each network weights, w , is randomly selected. |
| RBFNN | The number of learning cycles, the learning rate, η , the convergence threshold value, ε , are 1200, 0.06, and 10^{-6} , respectively. The numbers of neurons of input, hidden, and output layers are 6, 6, and 1, respectively. Initial values of the Gaussian basis function in each hidden neuron, c and σ , and each network weight, w , are randomly selected. |
| SVR | The maximum iteration number is 1200, $M=6$, and $b=5$. Initial values of w and C are 0.5. Initial values of ε and ε^* are 0.5 and -0.5, respectively. The threshold value is 10^{-6} . α , γ , α' , and γ' are selected between 0.1 and 0.5. |
| DBN/GT-DBN | The number of learning cycles of the feedforward network, the learning rate, η , the number of RBMs, the numbers of neurons of the input, hidden, and output layers, the number of G , and the convergence tolerance are 1200, 0.06, 6, 6, 6, 1, 6, and 10^{-6} , respectively. Initial values of both corresponding biases, a and b , the bi-direction weights, w , and the weights of FFN, r , are randomly selected. |

input data vector and one additional adjacent input data sample obtained at step 5. Then, return to step 4.

7. Perform IAGO of (3) to obtain the actual forecasted PV power output data series.

IV. TEST RESULTS

In the study, a 5-MW PV power plant in central Taiwan connecting to a 161-kV substation is served as the measurement target. Test cases for 12 months in 2016 and a look-ahead time up to 24-hour with one-minute resolution have been performed. In the study there are two cases representing both typical sunny and non-typical (i.e., with more drastic output changes) daily PV generation patterns of the four months corresponding to four seasons in the area where the PV plant located are presented. The actual measured 24-hour power output data of the PV power plant at one-minute time interval are used to train the proposed model. In the study, the number of data input to the model is $I = 6$, according to simulation experiences by assigning I from 5 to 10. The parameter G of (14) is set to be 6 after testing its value ranging from 6 to 12. The number of the hidden neurons of the RBM is $J = 6$ and the number of RBMs is selected to be 6. The number of training days are selected from one to three days for day-ahead forecasting. Simulation results also show that data input of one training day is sufficient for the study. To show the effectiveness of the proposed model for forecasting PV power output, results obtained are also compared with those obtained by other models including ARIMA, BPNN, RBFNN, SVM-Regression (i.e., SVR) [29], and DBN alone (without grey data preprocessor). Table I lists major parameters used by the compared methods according to simulation experiences. The convergence threshold value between two consecutive iterations

for each parameter is 10^{-6} . Simulations are performed on a PC with Intel Core i7-2600 3.4-GHz processor with the developed Matlab codes.

A. Case I (January and April)

In this study case, two sets of 24-hour (i.e., 24×60 data points per set) consecutive measured PV power data at one-minute resolution taken in January 10 and April 15, as shown in Figs. 5(a) and 5(b), respectively, are used to train the proposed and other compared models. Two other sets of 24-hour measured data taken in January 11 and April 16, as shown in Figs. 5(c) and 5(j), respectively, are then used to validate the forecasting results. It is observed that the actual PV output of Fig. 5(c) varies more drastically than that of Fig. 5(j). Figs. 5(d)–5(i) display the forecasted PV power output obtained by the six models for 24-hour look-ahead power output on January 11. Figs. 5(k)–5(p) illustrate forecasted PV power output obtained by the six models for 24-hour look-ahead power output on April 16.

Table II (January 11) and Table III (April 16) list the errors of forecasted PV output in terms of the mean absolute percentage error (MAPE) and the root mean squared percentage error (RMSPE) of (23) and (24), the iteration number to convergence, and the CPU time for all methods under comparisons, where the CPU time is in per unit value based on the CPU time required to perform forecasting by BPNN (for Case I, the time required are 10.071 seconds for January case and 10.115 seconds for April case). As shown in Figs. 5(c), Figs. 5(d)–5(i), and Table II, the proposed GT-DBN model gives superior forecast results for the non-typical day case in January than the DBN model alone and other methods.

$$MAPE = \frac{1}{M} \sum_{m=1}^M \left| \frac{x_m - x_{m,f}}{S} \right| \times 100\% \quad (23)$$

$$RMSPE = \frac{1}{M} \sqrt{\sum_{m=1}^M \left(\frac{x_m - x_{m,f}}{S} \right)^2} \times 100\% \quad (24)$$

where x_m is the m -th actual measured data, the $x_{m,f}$ is the m -th forecasted data, M (i.e., 60×24) is the total forecasted data points, and S is the total PV MW capacity.

B. Case II (July and October)

In this study case, two sets of 24-hour (i.e., 24×60) consecutive measured PV power data at one-minute resolution taken in July 5 and October 15, as shown in Figs. 6(a) and 6(b), respectively, are used to train the proposed and compared models. Two other sets of 24-hour measured data taken in July 6 and October 16, as shown in Figs. 6(c) and 6(j), respectively, are then used to validate the forecast results. It is seen that the actual PV output of Fig. 6(c) is with more spikes than that of Fig. 6(j). Figs. 6(d)–6(i) depict the forecasted PV power output of six compared models for the 24-hour look-ahead power output on July 6. Figs. 6(k)–6(p) illustrate forecasted PV power output of six models for 24-hour look-ahead power output on October 16.

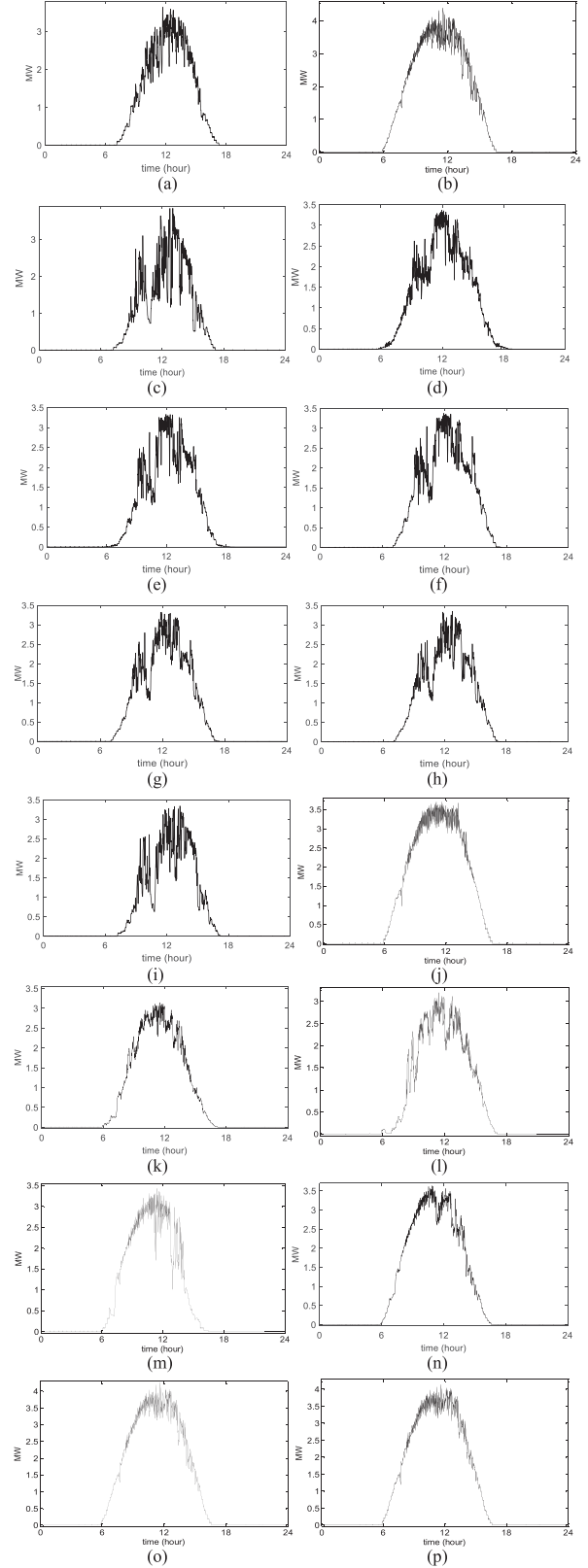


Fig. 5. 24-hour measured data of PV output for training in (a) winter (January) and (b) spring (April); (c) 24-hour measured data of PV output in winter (January) for comparisons and forecasting results obtained by (d) ARIMA, (e) BPNN, (f) RBFNN, (g) SVR, (h) DBN, and (i) GT-DBN; (j) 24-hour measured data of PV output in spring (April) for comparisons and forecasting results obtained by (k) ARIMA, (l) BPNN, (m) RBFNN, (n) SVR, (o) DBN, and (p) GT-DBN.

TABLE II
PV POWER OUTPUT FORECAST ERRORS OF COMPARED
METHODS (JANUARY 11, WINTER)

| Method | MAPE (%) | RMSPE (%) | Iteration Number | CPU Time (pu) |
|--------|----------|-----------|------------------|---------------|
| ARIMA | 27.995 | 29.014 | 993 | 0.771 |
| BPNN | 20.624 | 21.991 | | 1.000 |
| RBFNN | 17.572 | 18.964 | | 0.989 |
| SVR | 10.874 | 12.131 | 842 | 0.991 |
| DBN | 6.354 | 7.959 | 741 | 1.482 |
| GT-DBN | 4.937 | 5.328 | 608 | 1.324 |

TABLE III
PV POWER OUTPUT FORECAST ERRORS OF COMPARED
METHODS (APRIL 16, SPRING)

| Method | MAPE (%) | RMSPE (%) | Iteration Number | CPU Time (pu) |
|--------|----------|-----------|------------------|---------------|
| ARIMA | 17.849 | 18.773 | 934 | 0.773 |
| BPNN | 15.573 | 16.638 | | 1.000 |
| RBFNN | 8.775 | 9.814 | | 0.978 |
| SVR | 6.016 | 7.417 | 776 | 0.992 |
| DBN | 3.878 | 4.731 | 620 | 1.400 |
| GT-DBN | 3.877 | 4.729 | 557 | 1.386 |

Table IV (July 6) and Table V (October 16) list the errors of forecasted PV output in terms of MAPE, RMSPE of (23) and (24), the iteration number to convergence, and the CPU time (for Case II, the required time performed by BPNN are 10.201 seconds for July case and 10.025 seconds for October case) for the methods under comparisons. As shown in Fig. 6(c), Figs. 6(d)–6(i), and Table IV, the proposed model produces more accurate forecast results for the case in July 6 than the DBN model alone and other models.

C. Case III (PV Output Forecast for Twelve Months)

In the study representative test cases in each month are performed. For convenience of comparison, each data set for test includes two consecutive typical sunny days in a month of PV output data. The data set in the first day is for training and the other set of the second day is for 24-hour look-ahead forecasting comparison. Table VI lists the least errors of output forecasts in terms of MAPE of (23) obtained from the six methods under comparisons. Table VII lists the least errors of output forecasts in terms of RMSPE of (24).

By observing Fig. 5, Fig. 6, and Tables II–VII obtained by the six compared methods, it is found that the two DBN-based methods lead to more accurate forecasts. Under typical sunny days in Case III (including April and October of Cases I and II), the proposed GT-DBN model provides slightly better computational efficiency and solution accuracy comparing to DBN model alone. When the PV daily output is more drastically varying, as shown in test days for January and July of Cases I and II, and Tables II and IV, the proposed GT-DBN model leads to much higher solution accuracy than the DBN alone and other models because the grey data preprocessor smooths the non-stationary behavior of the primitive input data. The solution accuracy is improved by 1.4% to 2.6% for both MAPE and RMSPE and the CPU time is reduced about 12% compared to

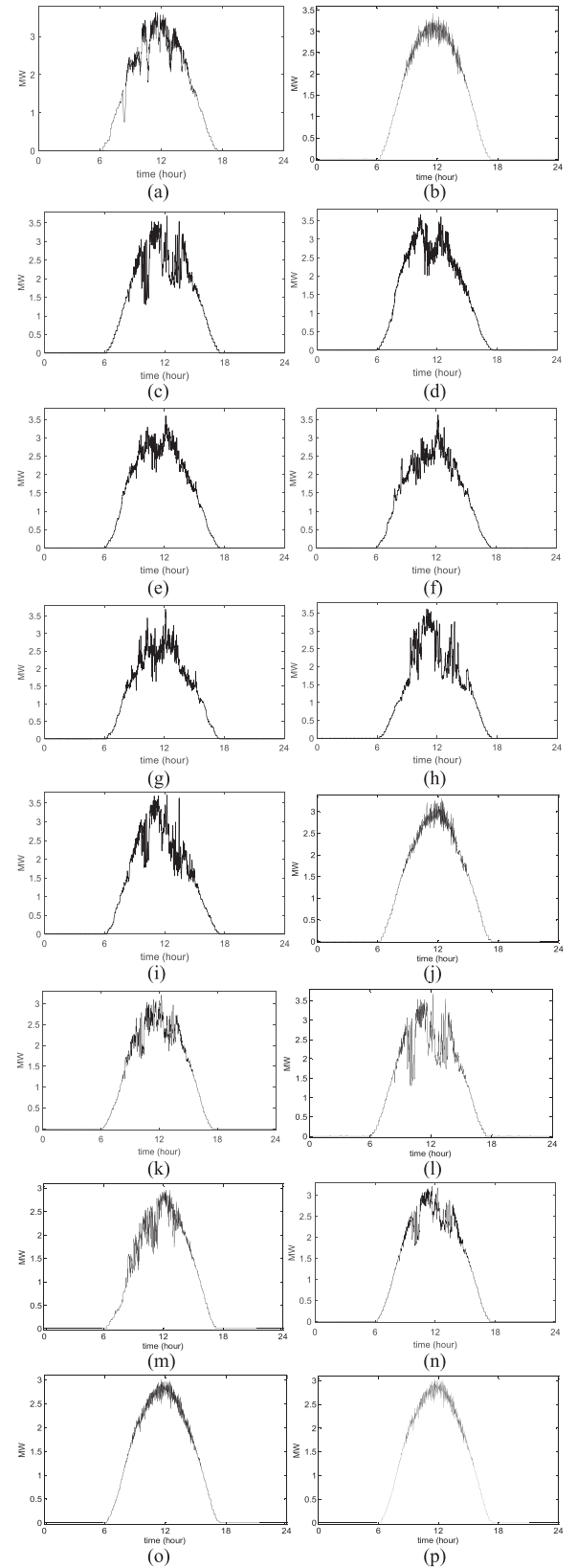


Fig. 6. 24-hour measured data of PV output for training in (a) summer (July) and (b) fall (October); (c) 24-hour measured data of PV output in summer (July) for comparisons and forecasting results obtained by (d) ARIMA, (e) BPNN, (f) RBFNN, (g) SVR, (h) DBN, and (i) GT-DBN; (j) 24-hour measured data of PV output in fall (October) for comparisons and forecasting results obtained by (k) ARIMA, (l) BPNN, (m) RBFNN, (n) SVR, (o) DBN, and (p) GT-DBN.

TABLE IV
PV POWER OUTPUT FORECAST ERRORS OF COMPARED
METHODS (JULY 6, SUMMER)

| Method | MAPE (%) | RMSPE (%) | Iteration Number | CPU Time (pu) |
|--------|----------|-----------|------------------|---------------|
| ARIMA | 29.361 | 30.982 | 1003 | 0.771 |
| BPNN | 22.583 | 24.017 | | 1.000 |
| RBFNN | 18.297 | 20.032 | 913 | 0.992 |
| SVR | 12.363 | 13.966 | 885 | 0.986 |
| DBN | 7.535 | 8.535 | 749 | 1.487 |
| GT-DBN | 5.133 | 6.029 | 613 | 1.319 |

TABLE V
PV POWER OUTPUT FORECAST ERRORS OF COMPARED
METHODS (OCTOBER 16, FALL)

| Method | MAPE (%) | RMSPE (%) | Iteration Number | CPU Time (pu) |
|--------|----------|-----------|------------------|---------------|
| ARIMA | 16.575 | 17.591 | 867 | 0.751 |
| BPNN | 14.992 | 16.013 | | 1.000 |
| RBFNN | 8.002 | 9.025 | 744 | 0.966 |
| SVR | 6.134 | 7.066 | 702 | 0.974 |
| DBN | 3.511 | 4.550 | 599 | 1.395 |
| GT-DBN | 3.507 | 4.447 | 508 | 1.378 |

TABLE VI
24-HOUR PV OUTPUT FORECAST ERRORS BASED ON MAPE OF COMPARED
MODELS OVER 12 MONTHS

| | ARIMA | BPNN | RBFNN | SVR | DBN | GT-DBN |
|---------|--------|--------|--------|-------|-------|--------|
| Jan. | 18.318 | 16.877 | 9.349 | 6.432 | 3.921 | 3.921 |
| Feb. | 18.993 | 17.019 | 9.553 | 6.551 | 3.982 | 3.981 |
| Mar. | 18.014 | 16.431 | 8.853 | 6.318 | 3.717 | 3.715 |
| Apr. | 17.849 | 15.573 | 8.775 | 6.016 | 3.878 | 3.877 |
| May | 18.003 | 16.551 | 7.839 | 5.874 | 3.550 | 3.548 |
| Jun. | 18.308 | 16.862 | 8.017 | 5.914 | 3.678 | 3.677 |
| Jul. | 18.994 | 17.341 | 10.934 | 6.992 | 4.108 | 4.104 |
| Aug. | 18.532 | 16.993 | 10.393 | 6.399 | 3.746 | 3.746 |
| Sep. | 18.245 | 16.832 | 10.278 | 6.381 | 3.697 | 3.695 |
| Oct. | 16.575 | 14.992 | 8.002 | 6.134 | 3.511 | 3.507 |
| Nov. | 17.552 | 15.722 | 8.603 | 6.195 | 3.577 | 3.576 |
| Dec. | 18.027 | 16.381 | 8.794 | 6.402 | 3.779 | 3.777 |
| Average | 18.118 | 16.465 | 9.116 | 6.300 | 3.762 | 3.760 |

TABLE VII
24-HOUR PV OUTPUT FORECAST ERRORS BASED ON RMSPE OF COMPARED
MODELS OVER 12 MONTHS

| | ARIMA | BPNN | RBFNN | SVR | DBN | GT-DBN |
|---------|--------|--------|--------|-------|-------|--------|
| Jan. | 19.252 | 17.961 | 10.294 | 7.518 | 4.898 | 4.897 |
| Feb. | 19.715 | 18.032 | 10.412 | 7.527 | 4.929 | 4.927 |
| Mar. | 19.012 | 17.543 | 9.997 | 7.292 | 4.631 | 4.630 |
| Apr. | 18.773 | 16.638 | 9.814 | 7.417 | 4.731 | 4.729 |
| May | 18.571 | 17.659 | 8.997 | 7.154 | 4.466 | 4.466 |
| Jun. | 19.349 | 17.978 | 9.136 | 7.189 | 4.560 | 4.599 |
| Jul. | 19.832 | 18.747 | 12.071 | 7.854 | 5.007 | 5.004 |
| Aug. | 19.115 | 18.082 | 11.502 | 7.333 | 4.663 | 4.662 |
| Sep. | 19.047 | 17.965 | 11.487 | 7.352 | 4.689 | 4.689 |
| Oct. | 17.591 | 16.013 | 9.025 | 7.066 | 4.550 | 4.447 |
| Nov. | 18.496 | 16.876 | 9.624 | 7.155 | 4.567 | 4.565 |
| Dec. | 19.005 | 17.435 | 9.729 | 7.431 | 4.771 | 4.770 |
| Average | 18.980 | 17.577 | 10.174 | 7.357 | 4.705 | 4.699 |

the DBN model alone. Though ARIMA, BPNN, RBFNN, and SVR methods require approximately 35% to 45% less CPU time than the two DBN-based methods, their solution accuracies are unsatisfactory. As indicated in Tables VI and VII, the average MAPE error of the proposed method is 3.760% and the average

RMSPE error of the proposed method is 4.699% for PV output forecast under typical sunny days over 12 months of test cases, respectively. The GT-DBN model gives the least forecasting errors among the compared methods.

V. CONCLUSIONS

The increasing penetration of PV plants in the grid as a source of electricity has drawn much attention in forecasting their power output over different look-ahead time horizons. However, due to the intermittent natures of the PV generation, it is difficult to precisely forecast the PV output in short term for operations planning. This paper has proposed a method that integrates the data preprocessor based on grey theory and a DBN with a supervised learning scheme to forecast PV output over 24-hour look-ahead time horizon. By observing the test results, the forecast accuracy of the proposed model is superior to that of the DBN model alone, especially when the PV output varies more drastically. Results also show that the forecast accuracy and computational efficiency of the proposed model are better than the other compared models.

The major contributions and advantages of the proposed model for day-ahead solar PV output forecast are as follows. (1) There is no need of longer dataset for training the proposed model. Simulation experiences show that only one day of training dataset is satisfactory. (2) The adoption of GT-based data preprocessor can substantially smoothing the irregular input data series and mitigate the over-fitting problem. (3) The inclusion of DBN and the FFN as the fine-tune method at the last layer in the proposed GT-DBN model enhances the effectiveness to time series forecast with high accuracy through learning feature representations of data. (4) By combining GT, DBN, and FFN, as shown in Fig. 2, the proposed model has successfully determined the parameters that best fit the training data to obtain the least forecasting error among the compared models for solar PV output. The data preprocessor and the supervised model are useful in many forecasting or classification cases, especially when input data are incomplete or irregular.

To extend the applications of the proposed model, future works will focus on exploring new strategies that combine the DBN for multi-area large-scale PV power output forecasts and taking into account PV output fluctuations due to meteorological and geographical conditions. The total PV output thus can be more accurately managed to enhance the system operations planning and maintain the reliability requirement.

REFERENCES

- [1] A. Kaur, H. T. C. Pedro, and C. F. M. Coimbra, "Impact of onsite solar generation on system load demand forecast," *Energy Convers. Manage.*, vol. 75, pp. 701–709, Nov. 2013.
- [2] J. Liu, W. Fang, X. Zhang, and C. Yang, "An improved photovoltaic power forecasting model with the assistance of aerosol index data," *IEEE Trans. Sustain. Energy*, vol. 6, no. 2, pp. 434–442, Apr. 2015.
- [3] G. Capizzi, C. Napoli, and F. Bonanno, "Innovative second-generation wavelets construction with recurrent neural networks for solar radiation forecasting," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 11, pp. 1805–1815, Nov. 2012.
- [4] C. Chen, S. Duan, T. Cai, and B. Liu, "Online 24-h solar power forecasting based on weather type classification using artificial neural network," *Sol. Energy*, vol. 85, no. 11, pp. 2856–2870, Nov. 2011.

- [5] H. S. Jang, K. Y. Bae, H. S. Park, and D. K. Sung, "Solar power prediction based on satellite images and support vector machine," *IEEE Trans. Sustain. Energy*, vol. 7, no. 3, pp. 1255–1263, Jul. 2016.
- [6] P. Bacher, H. Madsen, and H. A. Nielsen, "Online short-term solar power forecasting," *Sol. Energy*, vol. 83, no. 10, pp. 1772–1783, Oct. 2009.
- [7] C. Chupong and B. Plangklang, "Forecasting power output of PV grid connected system in Thailand without using solar radiation measurement," *Energy Procedia*, vol. 9, pp. 230–237, 2011.
- [8] M. Ding, L. Wang, and R. Bi, "An ANN-based approach for forecasting the power output of photovoltaic system," *Procedia Environ. Sci.*, vol. 11, pp. 1308–1315, 2011.
- [9] S. Ferlito, G. Adinolfi, and G. Graditi, "Comparative analysis of data-driven methods online and offline trained to the forecasting of grid-connected photovoltaic plant production," *Appl. Energy*, vol. 205, no. 1, pp. 116–129, Nov. 2017.
- [10] A. Dumas *et al.*, "A new correlation between global solar energy radiation and daily temperature variations," *Sol. Energy*, vol. 116, pp. 117–124, Jun. 2015.
- [11] G. Graditi, S. Ferlito, and G. Adinolfi, "Comparison of photovoltaic plant power production prediction methods using a large measured dataset," *Renewable Energy*, vol. 90, pp. 513–519, May 2016.
- [12] S. O. Haykin, *Neural Networks and Learning Machines*, 3rd ed., Englewood Cliffs, NJ, USA: Prentice-Hall, 2009.
- [13] S. S. Haykin, *1931- Neural Networks: A Comprehensive Foundation*, Upper Saddle River, NJ, USA: Prentice-Hall, 1999.
- [14] N. K. Kasabov, *Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering*, Cambridge, MA, USA: MIT Press, 1996.
- [15] S. Chen, K. Labib, and L. Hanzo, "Clustering-based symmetric radial basis function beamforming," *IEEE Signal Process. Lett.*, vol. 14, no. 9, pp. 589–592, Sep. 2007.
- [16] C. Y. Zhang, C. L. P. Chen, M. Gan, and L. Chen, "Predictive deep Boltzmann machine for multiperiod wind speed forecasting," *IEEE Trans. Sustain. Energy*, vol. 6, no. 4, pp. 1416–1425, Oct. 2015.
- [17] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, and J. Liu, "LSTM network: A deep learning approach for short-term traffic forecast," *IET Intell. Transp. Syst.*, vol. 11, no. 2, pp. 68–75, Mar. 2017.
- [18] H. F. Yang, T. S. Dillon, and Y. P. P. Chen, "Optimized structure of the traffic flow forecasting model with a deep learning approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2371–2381, Oct. 2017.
- [19] M. Khodayar, O. Kaynak, and M. E. Khodayar, "Rough deep neural architecture for short-term wind speed forecasting," *IEEE Trans. Ind. Inform.*, vol. 13, no. 6, pp. 2770–2779, Dec. 2017.
- [20] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.
- [21] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
- [22] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.
- [23] M. A. Carreira-Perpinan and G. E. Hinton, "On contrastive divergence learning," in *Proc. 10th Int. Workshop Artif. Intell. Statist.*, Bridgetown, Barbados, 2005, pp. 33–40.
- [24] G. Hinton, "A practical guide to training restricted Boltzmann machines," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer-Verlag, 2012, pp. 599–619.
- [25] G. Hinton, "A practical guide to training restricted Boltzmann machines," Mach. Learn. Group, Univ. Toronto, Toronto, ON, Canada, Tech. Rep. TR 2010–003, 2010.
- [26] T. H. M. El-Fouly, E. F. El-Saadany, and M. M. A. Salama, "Grey predictor for wind energy conversion systems output power prediction," *IEEE Trans. Power Syst.*, vol. 21, no. 3, pp. 1450–1452, Aug. 2006.
- [27] Y. W. Teh and G. E. Hinton, "Rate-coded restricted Boltzmann machines for face recognition," in *Advances in Neural Information Processing Systems*, MIT Press, Cambridge, MA, USA, 2001, pp. 908–914.
- [28] G. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, pp. 1771–1800, Aug. 2002.
- [29] V. Kecman, *Support Vector Machines—An Introduction*, New York, NY, USA: Springer-Verlag, 2005.

Gary W. Chang (M'94-SM'01-F'10) received the Ph.D. degree from the University of Texas at Austin, USA, in 1994. He is currently working as a Professor with the Department of Electrical Engineering at National Chung Cheng University, Chia-Yi, Taiwan. His areas of research interests include power quality, renewable energy, and smart grid. Dr. Chang is a Registered Professional Engineer in the State of Minnesota. He serves as the Chair of IEEE PES Transmission and Distribution Committee (2019–2020).

Heng-Jiu Lu (M'16) received the B.S.E.E., M.S.E.E., and Ph.D. degrees from Cheng Shiu University, Tainan, Taiwan, in 2007, 2009, and 2016, respectively. He is currently working with the Green Energy and Environment Research Laboratories, Industrial Technology Research Institute, Hsin-Chu, Taiwan. His areas of research interests include renewable energy forecasting and fuzzy control.