Machine Learning Algorithms in Forecasting of Photovoltaic Power Generation

Di Su, Efstratios Batzelis, Bikash Pal
Electrical and Electronic Engineering
Imperial College London
London, UK
{di.su17, e.batzelis, b.pal}@imperial.ac.uk

Abstract — **Due to the intrinsic intermittency and stochastic nature of solar power, accurate forecasting of the photovoltaic (PV) generation is crucial for the operation and planning of PV-intensive power systems. Several PV forecasting methods based on machine learning algorithms have recently emerged, but a complete assessment of their performance on a common framework is still missing from the literature. In this paper, a comprehensive comparative analysis is performed, evaluating ten recent neural networks and intelligent algorithms of the literature in short-term PV forecasting. All methods are properly fine-tuned and assessed on a one-year dataset of a 406 MWp PV plant in the UK. Furthermore, a new hybrid prediction strategy is proposed and evaluated, derived as an aggregation of the most well-performing forecasting models. Simulation results in MATLAB show that the season of the year affects the accuracy of all methods, the proposed hybrid one performing most favorably overall.**

Keywords—Forecasting, photovoltaic, machine learning, neural networks, intelligent algorithms.

I. Introduction

The UK targets for very high photovoltaic (PV) integration into the power system necessitates reliable forecasting of the stochastic and highly uncertain PV power generation. This is important for the power system stability and for keeping the PV power curtailments low. Recently, machine learning algorithms have emerged as powerful tools in predicting the PV power generation, as they avoid modelling of complex atmospheric phenomena but focus on the actual operation data.

Artificial Neural Networks (ANN) are widely used in this context; some of the recent forecasting methods are discussed in the following. A Back-Propagation Neural Network (BPNN) is adopted in [1] for 24 hours ahead solar power forecasting, while the study in [2] explores a Non-linear Auto Regressive Neural Network with Exogenous Inputs (NARXNN) to predict the PV generation power at a standalone micro grid on a remote island. The authors of [3] achieve a 72-hour ahead PV power forecasting using an Elman Neural Network (ENN) and [4] presents a Generalized Regression Neural Network (GRNN) combined with Wavelet Transform (WT) for short-term PV power forecasting. A Fuzzy Neural Network (FNN) for PV power estimation is proposed in [5].

Another large class of solar power forecasting methods are based on *Intelligent Algorithms* (IA). Extreme Learning Machine (ELM) is used in [6] to predict the PV power output in multiple steps ahead, while a Random Forest (RF) model is adopted in [7] for day-ahead hourly PV power forecasting. The study in [8] estimates the PV power output of a 1 MW plant based on Support Vector Regression (SVR) and investigates the effect of cloudiness on the forecasting performance. SVR is also employed in [9], proposing a selection method of the SVR's parameters for minimum estimation error. A comparison of the K-Nearest-Neighbours (KNN) and SVR methods on actual measurements and Numerical Weather Prediction (NWP) data is given in [10]; a feature extraction is attempted, resulting in the ten best features to be used as the model's inputs.

A literature review reveals that the machine learning approaches are generally superior to the conventional statistical methods due to their inherent ability to model any non-linear, complex and dynamic process. However, training of ANN or IA is complicated and there is still no commonly accepted way to construct the perfect model; this is why selecting and optimizing the model' parameters is usually a trial and error process. Most of the relevant studies in the literature examine only a few machine learning methodologies, focusing on short-term (up to three days ahead) forecasting and not providing sufficient details on how the model's parameters are found; a comprehensive comparative analysis to account for all relevant methods and longer look-ahead times is still missing from the literature. Furthermore, the various studies consider different real-world installations with dissimilar plant specifications, locations, time periods, weather conditions and datasets, while there is no consistent way to select the model training variables and error metrics. To this day, these methods have not been assessed on a common evaluation framework simultaneously.

In this paper, *ten* different machine learning algorithms for *six-day ahead* PV power forecasting are implemented and compared; these include six ANN and four IA methods. A brief discussion is provided on the parameters tuning and performance evaluation for each method. Furthermore, a *new hybrid prediction strategy* is proposed, based on some of the most well-performing models, and is included in the comparison to evaluate its effectiveness. All simulations are curried out in MATLAB, using a dataset of one-year hourly measurements from a 406 MW PV park in the UK. This is the first study in the literature to perform such an assessment and performance comparison on a common evaluation framework and for medium-term horizons of six days.

The rest of the paper is organized as follows. The dataset used is described in Section II, while the ten forecasting methods and the proposed hybrid approach are presented in Section III. The overall performance is discussed in Section IV, followed by the conclusions in Section V.

II. Case Study and Dataset

A. Plant Specifications

The selected PV power plant has an installed capacity of 406 MWp and is connected to the Norwich Main Substation (Norfolk, England, UK). As shown in Fig. 1, this plant has a favourable position in terms of solar radiation and can generate more electrical power than the majority of other PV stations in the UK. The original training dataset is jointly provided by Sheffield Solar [11], Copernicus Atmosphere Monitoring Service (CAMS) [12] and MERRA-2 [13].

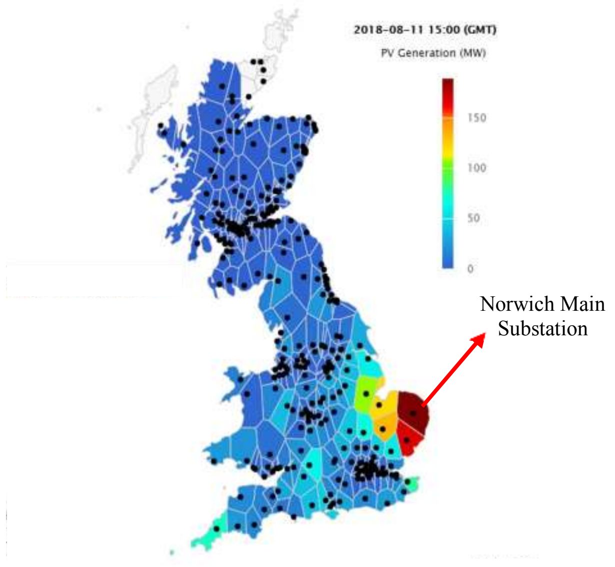


Fig.1. Regional PV generation across the UK. The black dots indicate PV stations [11].

Sheffield Solar is a collaborative PV live service between the University of Sheffield and UK National Grid. This service models nationwide live PV plants and can provide reliable time series data of power generation of all solar PV systems connected to UK transmission network. Solar irradiation data (including global horizontal, beam horizontal, diffuse horizontal and beam normal irradiation) is provided by CAMS. MERRA-2 can deliver time series data of some weather variables such as temperature, humidity, pressure, wind speed, wind direction, rainfall, snowfall and snow depth. The time period under investigation is from 1 March 2017 to 28 February 2018 covering all four seasons. To study the seasonal effect on the PV power forecasting, the collected dataset is divided into four parts, one for each season of the year, as shown in Fig. 2. The night data is excluded, as the PV generation during the night is zero.

B. *Training and Validation Datasets*

Table I shows the training and validation datasets for each of the four seasons. The former set is used only for training purposes, while typical season days from the rest of the year are randomly selected for the evaluation, making sure that training

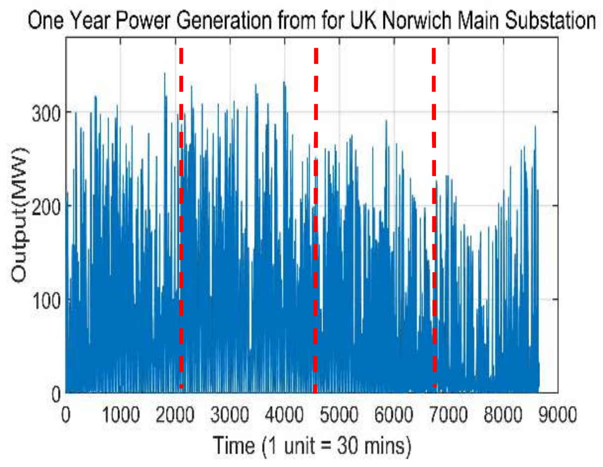


Fig. 2. One-year data is divided into four parts.

TABLE I. TRAINING AND VALIDATION DATASETS

	Training data	Validation data
Spring	1-3-2017 to 31-5-2017	1-4-2016 to 6-4-2016
Summer	1-6-2017 to 31-8-2017	26-7-2016 to 31-7-2016
Autumn	1-9-2017 to 30-11-2017	15-10-2016 to 20-10-2016
Winter	1-12-2017 to 28-2-2018	15-1-2017 to 20-1-2017

data and corresponding validation data belong to the same season. The separation into the four seasons is made to investigate the anticipated strong seasonal effect in the PV generation forecasting.

C. *Performance Metric*

To evaluate the performance of the forecasting methods, the normalized root mean square error (nRMSE) is adopted here, as widely done in the literature [8]:

$$nRMSE = 100\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{\hat{P}(i)-P(i)}{P_{ins}}\right)^2} \tag{1}$$

N is the number of samples; $\hat{P}(i)$ and $P(i)$ are the predicted and measured power at the time i; P_{ins} is the installed capacity. Furthermore, the models' performance can be also compared by calculating the skill score for a given metric:

$$skill\ score = \frac{Metric_{base}-Metric_{forecast}}{Metric_{base}} \times 100 \tag{2}$$

The skill score is obtained by comparing a specific method against a base method. Generally, a model with the least satisfying performance is chosen as the base model. In this paper, nRMSE is used as the metric since it gives more weight to large errors and therefore can be treated as a suitable indicator of the cost caused by unbalance between supply and demand.

III. METHODOLOGIES

This paper explores ten machine learning methodologies for solar power forecasting. Since these methods are well established, the focus of this section is on case-specific description rather than general theory representation. The training dataset is normalized between 0 and 1 to eliminate scale differences. All methods are implemented in MATLAB R2018a.

A. *Back Propagation Neural Network (BPNN)*

In this paper, a static feed-forward network with a single hidden layer is adopted [1]. The neurons in the input and output layers can be automatically determined through dimensions of input and output vectors. The values of weights and thresholds are randomly initialized. As for the hidden layer neuron number, an initial number (4 to 14 in this paper) is determined empirically and then 10-fold cross validation (10-CV) is employed to choose a specific desired number. We can observe the error performance as hidden layer neurons increase and select the neuron number associating with the best error performance. We decide to change the network structure according to the season, thus finding a near-perfect model for each season.

B. *BPNN with Genetic Algorithm Optimization*

The weights and thresholds of BPNN are arbitrarily initialized, thus exhibiting randomness on the training phase. To enhance the model performance, Genetic Algorithm (GA) [14] is proposed to optimize the network weights and thresholds (GABPNN). The main function of GA is to implement selection, crossover and mutation on the number strand and generate optimal weights and thresholds which outperforms previous generation. Specifically, we extract the weights and thresholds from the previous realised model and encode them into a real number strand. For example, the structure of Spring BPNN is 9-11-1. Therefore, the number of weights is $9 \times 11 + 11 \times 1 = 110$ and the number of thresholds is $11 + 1 = 12$. The total individual length is $110 + 12 = 122$. During the iteration, we consider the sum of absolute error between forecasted value and expected value as fitness, thus the smaller the fitness, the better the individual. GA is implemented with MATLAB. The population size is set to 10, the evolution time to 20 generations, the crossover probability to 0.3 and the mutation probability to 0.2. We observed that the fitness function decreases with the iterations, which indicates better forecasting over the standard BPNN.

C. Elman Neural Network (ENN)

The ENN is a multi-layered recurrent neural network [3]; it is a typical dynamic neural network which can store a hidden layer output and feed it back into the input layer through a delay operator. Due to the feedback layer, the network ability for handling nonlinear and dynamic processes is enhanced. Similar to the BPNN, the hidden layer neurons number, the delay steps and the training algorithm need to be tuned through a trial and error process. In this paper, the ENN model of spring, summer, autumn and winter have 8, 7, 7 and 4 hidden layer neurons respectively. The delay step is set to 1:2.

D. Generalized Regression Neural Network (GRNN)

GRNN was first proposed in [15]. The main difference with BPNN is that GRNN has additional layers: the pattern layer and summation layer. In the pattern layer, the neurons number is equal to the number of total observations of input data so that there is no concern about the hidden layer neuro number. Both the mathematical summation and weighted summation are computed at the summation layer. As a final term, the two sums are divided for the forecasting output. MATLAB provides a built-in function to implement GRNN. The GRNN implementation is superior to the BPNN in terms of computation efficiency, since only one parameter, i.e. Spread, needs to be tuned during the training phase. Therefore, GRNN is generally faster than BPNN. Spread is a significant factor of the model performance; a small Spread value may lead to overfitting, while a large Spread value may increase the forecasting error. Here, 10-CV is adopted to select an optimal value of Spread within a predetermined range. The values for the spring, summer, autumn and winter are 0.15, 0.11, 0.1 and 0.24 respectively.

E. Adaptive Network based Fuzzy Inference System (ANFIS)

ANFIS was firstly proposed in [16]. The essence of ANFIS is mathematical logic operation. The typical procedure of implementing ANFIS involves fuzzification, i.e. converting normal input series into fuzzy series. This is achieved by computing the membership degree of each input through a membership function. Membership degree is an indicator quantifying how well the given input satisfies the linguistic condition. Membership degree is generally in the range between 0 (low degree) and 1 (significant degree). The algorithm of ANFIS is similar to BPNN. Providing the computed output, the error between forecasted value and expected value is back propagated and parameter set is updated based on error gradient descent method. The number of membership function is arbitrarily initialized. The model training is implemented through trial and error. However, since the parameter set of ANFIS is randomly initialized, the network convergence is not guarantee: during the training process, the output of the ANFIS model may converge to Not-a-Number (NaN) if the initial setting is not appropriate. Improvement in parameter initialization can potentially increase ANFIS model forecasting accuracy.

F. Nonlinear Autoregressive Neural Network with Exogenous Inputs (NARXNN)

NARXNN is a typical dynamic neural network which consists of static neurons and output feedback [2]. The standard architecture of NARXNN is parallel, according to which the model output is directly fed back into the model input. However, since the expected output is available during the training, the series-parallel architecture is preferable, and the expected output can be directly adopted for model training. The series-parallel architecture is able to convert a feedback network into a feedforward network, resulting in a static neural network. Both input and feedback delays are set to $1:2$ and the hidden layer neurons number is set to 10 for all models. The spring model used the Levenberg-Marquardt algorithm; the summer one used the Bayesian Regulation algorithm; the autumn and winter models employed a Gradient descent with momentum and adaptive learning rate. Initially, the open-loop network is trained on the training dataset; then, the network is changed to close-loop for multi-step prediction.

G. K Nearest Neighbours (KNN)

The concept of KNN was introduced in [17]. KNN assumes that similar weather conditions could possibly result in similar PV power generation. Therefore, the historical dataset can be regarded as a set of pairs of cause and result. The task of forecasting future PV power generation can be converted into searching in the past database for K feature vectors that are the neatest neighbours to those of the time of interest. Then, the PV power generation associated with the selected neighbours are combined as a single forecasting in a closeness-weighted approach. In this study, the closeness is quantified by Euclidean distance. Note that, if more than one sample point has the same distance from a query point, the observation with the smallest index is selected among all candidates. The search method is configured as a k-d tree, which is a built-in function in MATLAB. Regarding the neighbours K, a predetermined range is set between 0 and 500, and K can be selected in a trial and error manner. Here, 230, 230, 27 and 64 are the values for the spring, summer, autumn and winter respectively. Compared to Neural Network, KNN is more intuitive and deterministic, and the forecasting result is reproducible.

H. Extreme Learning Machine (ELM)

The motivation behind ELM [18] is to optimize a single layer feedforward network (SLFN) that generally suffers from slow training speed, local minimum convergence and learning rate sensitivity. ELM is superior to the conventional Feedforward neural network in terms of learning speed and generalization performance. The weights connecting the input layer and hidden layer, as well as the hidden neurons thresholds, can be randomly initialized. During the training phase, these randomly generated weights and thresholds do not need to be adjusted; the only free parameter is the number of hidden layer neurons. ELM can be described in the following steps: (i) the input weights W_{in} and thresholds b are randomly initialized; (ii) choose a proper number of hidden layer neurons; (iii) choose an infinite-differentiable activation function $g(*)$, and compute the hidden layer output $H = g(wx + b)$; (iv) compute the output weights W_{out} via $W_{out} = H^{\wedge} + Y$, where $H^{\wedge}$ is the Moore-Penrose inverse of H. A predetermined range of hidden layer neurons number is between 4 and 20. We also used 10-CV to optimize the network structure. The neurons number is 20 for the spring and summer models, and 18 for the autumn and winter ones. It is worth noting that ELM takes 3.05s to complete the model test (hidden layer neurons number between 4 and 20), whereas BPNN needs 56.852s for the same neurons number; this is an improvement of the execution time by around 19 times.

I. Random Forest (RF)

RF is an ensemble model that integrates the classification or regression outputs from several uncorrelated subsystems denoted as decision trees [7]. Decision tree is a statistical tool which employs a tree-like architecture to depict potential outputs for a given input. The procedure of implementing an RF is briefly described as follows: (i) Randomly generate T training sets $S_1, S_2, \ldots, S_T$ from the original training dataset S_N based on the Bootstrap method; an observation may appear more than once in a training set. (ii) For every single training set, there is a corresponding decision tree C with added node split mechanism, which randomly selects m features from the total M features as decision tree inputs at each node. The RF algorithm keeps the split proceeding in the best possible way. During the growth process, the value of m remains unchanged. (iii) Feed the validation dataset X into each grown decision tree and produce multiple outputs $C_1(X)$, $C_2(X)$, …, $C_T(X)$. (iv) The aggregation result is obtained by averaging all the outputs from these trees. In this paper, the split number is set to $\sqrt{M}$. As for the number of decision trees, the RF model is repeatedly tested from 50 to 1000 trees and the number associated with a minimum forecasting error is chosen as the final number: 850 for spring, 400 for summer, 600 for autumn and 150 for winter. MATLAB does not have a built-in function for RF, therefore the RF Toolbox developed by Abhishek Jaiantilal from University of Colorado Boulder was employed in this research [19].

J. Support Vector Regression (SVM)

In SVR, a so-called Kernel function is adopted to map input patterns X_i into a higher dimension space where output patterns become linearly separable, permitting pattern extraction via linear fitting. The objective is to find the best fit, which can be converted into an optimization problem [8]. Among many existing SVR Toolboxes, LIBSVM software library and Epsilon-SVR [20] were applied in this paper since it provides many default values and has built-in function of cross validation. The penalty parameter C and Kernel function parameter γ are the most important parameters; these are tuned by the K-CV method. The grid search algorithm is used to select the most appropriate values of (C, γ) in a predefined range: first discretize C and γ, then search for the desired parameter set (C, γ) by walking along a path which is set as exponentially growing sequences of C and γ(for example, $C = 2^{-8}, 2^{-6}, \ldots, 2^{8}$ and $\gamma = 2^{-8}, 2^{-6}, \ldots, 2^{8}$). The grid path is set as exponent minimum forecasting error, then select the set in which C has the smallest value because large could possibly lead to overfitting. Based on this tuning, the Spring model adopts $C = 0.44$ and $\gamma = 0.09$; the summer model $C = 0.02$ and $\gamma = 0.5$; the autumn model $C = 0.5$ and $\gamma = 0.5$; the winter model $C = 0.09$ and $\gamma = 0.09$. It is worth noting that a large grid usually requires long search times; GA and Particle Swarm Optimization (PSO) may be used instead for shorter computation times.

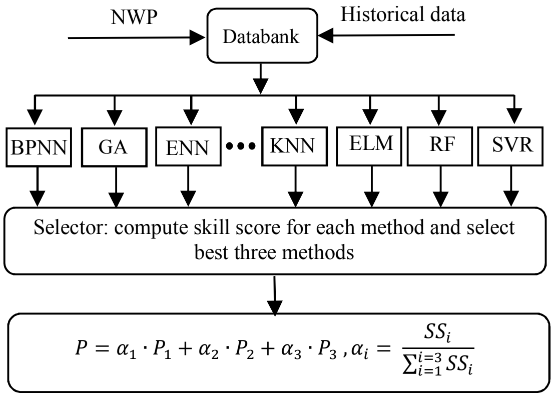

NWP
Historical data
Databank
BPNN
GA
ENN
KNN
ELM
RF
SVR
Selector: compute skill score for each method and select best three methods
$P = \alpha_1 \cdot P_1 + \alpha_2 \cdot P_2 + \alpha_3 \cdot P_3, \alpha_i = \frac{SS_i}{\sum_{i=1}^{i=3} SS_i}$

Fig. 3. Flow chart of the hybrid method.

K. Proposed Hybrid Method

The objective of the proposed strategy is to implement some of the most well-performing uncorrelated techniques and combine their results to a weighted average as a single forecast. First, all ten forecasting models are implemented and evaluated separately based on a common databank. Then, a selector chooses the best three methodologies according to a skill score associated with each method. Finally, the prediction results of the chosen methods are combined to a weighted average using the respective weight coefficients as computed by the Skill Score (SS). A calculation flowchart of this hybrid model is presented in Fig. 3. We used the nRMSE to compute the Skill Score and then normalised the coefficients to have a unit sum.

The concept of the Hybrid model is based on the observation that none of the ten individual methods was superior in all four seasons: some methods perform better in spring, while others produce the most accurate prediction in autumn. The proposed approach essentially takes into account several different opinions before making the final decision. Instead of improving the data quality (sometimes we do not even have complete datasets) or dedicating to building a perfect model, combination of results from several uncorrelated outstanding techniques is more robust and exhibits better transferability from one case to another.

While there are other method-combinations in the literature, this is the first study to integrate several forecasting models based on their skill score. Skill-score-computed coefficients are more computationally efficient as compared to MoorePenrose pseudoinverse or GA adopted elsewhere. In this paper, the coefficients computation is intuitive, as more importance is given to the most accurate methods. The theory of the hybrid method is simple, but the forecasting performance is expected to generally outperform the aforementioned techniques. In the next section, the findings verify that a simple combination of several good methods normally generates more reliable predictions than any single method on its own. This may be a useful technique, especially when the collected data is not complete and building a single perfect model is nearly impossible.

IV. NUMERICAL PERFORMANCE

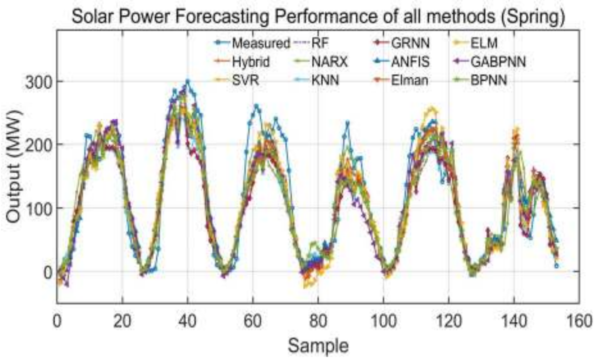
Fig. 4 presents the measured and estimated PV power by all methods under consideration for six days: all techniques exhibit generally satisfying performance in the four seasons. The typical "bell shape" pattern is apparent, according to which the PV plant provides the maximum power at mid-day and near-zero output before the sunrise and after the sunset. It is worth noting that most models successfully detect fast irradiance fluctuations (e.g. last day of the spring - Fig. 4(a)).

Several interesting aspects could be observed from Fig. 4. First, the forecasting performance in the winter is the worst among four seasons. As shown in Fig. 4(d), negative values are produced and most methods do not have a good agreement with the expected power generation. This is expected, as the short sunshine duration in the UK's winter results in insufficient data for model training. Furthermore, another conclusion from the same figure is that SLFN (typically ELM and BPNN) is more likely to be affected by insufficient dataset compared to other methods, as both of them produce non-realistic negative estimations even after parameter tuning.

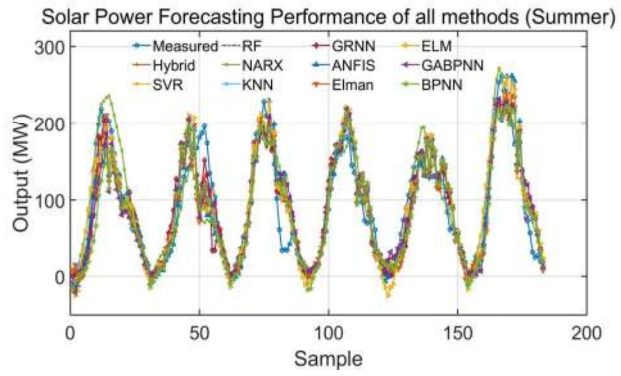
On the other hand, there is a very good match between measurements and predictions in the remaining seasons, as Fig. 4(a)-(c) illustrate. Some fluctuations at mid-day that are not detected properly are mainly due to sudden change of weather condition. The UK has significant wind resource that leads to stochastic cloud motion and uncertainty to the power forecasting.

More elaborate numerical results are shown in Tab. II. An expected conclusion is that the forecasting performance is significantly affected by the specific season of the year. Summer and autumn are found to generally perform better than spring and winter. For example, in terms of $nRMSE$, average errors of all models are 7.21% in summer and 6.92% in autumn while these numbers increase to 8.62% in spring and 9.37% in winter. Spring and winter in the UK are two seasons in which there is insufficient irradiation, strong wind and frequent snow; these factors hinder the successful forecasting. Autumn, rather than summer, is surprisingly found to be the most easily forecasted season. This seems reasonable when the typical UK summer is considered, which is a rainy season with unstable weather. Conversely, autumn combines strong radiation and a more stable weather.

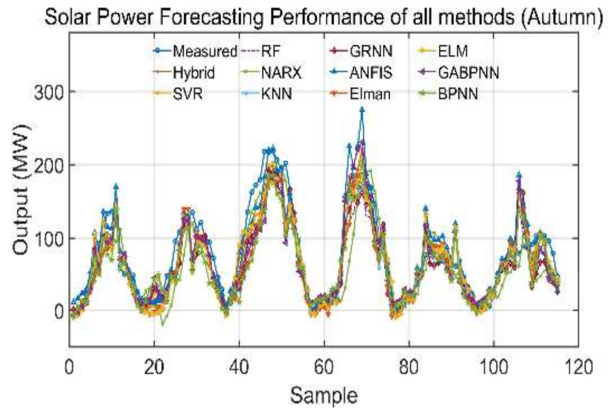
Among the six neural networks, NARXNN exhibits noteworthy superiority, producing the best overall prediction at a yearly error of only 7.09%. This is mainly due to its dynamic feedback mechanism which allows for the network to memorize past-time data. ENN, which also has the feedback function, performs better than most other approaches as well, as it ranks within the top three (excluding the hybrid model) in terms of yearly error shown in Tab. II. As a rule, we have found that a neural network with a feedback mechanism or adopting delayed data as model input improves the forecasting accuracy. BPNN performance is the overall least-satisfying, but it can be enhanced with GA optimization: the negative values are not apparent in GABPNN in Fig. 4(d). ANFIS usually provides more accurate forecasts than GRNN.



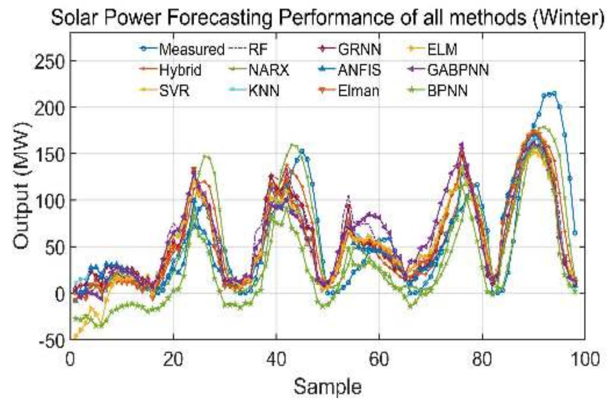
(a)



(b)



(c)



(d)

Fig. 4. Aggregated comparison among all methods. (a) Spring; (b) Summer; (c) Autumn; (d) Winter.

TABLE II. NUMERICAL PERFORMANCE OF ALL METHODS (NRMSE)

Model	BPNN	GABPNN	ENN	GRNN	ANFIS	NARXNN	KNN	ELM	RF	SVR	Hybrid	Season Avg
Spring	8.76	8.52	8.39	9.84	8.32	8.00	9.75	8.39	8.49	8.79	7.55	8.62
Summer	8.02	7.80	7.46	6.79	7.84	7.34	6.47	7.63	6.29	7.33	6.31	7.21
Autumn	7.85	7.52	6.86	7.20	6.64	6.88	6.91	6.43	6.91	6.86	6.61	6.92
Winter	11.57	10.52	9.45	9.73	9.43	6.14	9.56	9.98	10.12	9.57	6.98	9.37
Year	9.05	8.59	8.04	8.39	8.06	7.09	8.17	8.11	7.95	8.14	6.74	

In a conclusion, neural networks manage to extract the nonlinear dynamic relation quite well (better with a feedback mechanism), but the training process exhibits great randomness. At every simulation run, different results may be obtained; it is not certain whether the next simulation would be better or not. On the contrary, intelligent algorithms are more robust. RF, as the best candidate of IAs, is a combination of several decision trees and exhibits immunity to irrelevant data.

Another observation is that in the winter, the numbers of hidden layer neurons or decision trees are the lowest, as discussed in Section III. This is because the effective irradiation is insufficient in the winter, resulting in incomplete historical databank for the model training. It seems that it is better to resort to a simple well-trained model, rather than a more complex albeit poorly-trained model. For accurate forecasting in a season like UK winter, it is suggested to collect more key influencing factors.

KNN, ELM and SVR exhibit similar in this paper. Table II shows that the proposed Hybrid model achieves the best overall performance among all methods, as it obtains a yearly nRMSE of as low as 6.74%. An overall comparison based on the average skill score is shown in Fig. 5; the Hybrid method exhibits the highest score.

V. CONCLUSION

In this paper, a comprehensive performance assessment among some of the most popular PV power forecasting methods is performed on a common dataset. NARXNN is found to be superior over other neural networks due to its dynamic feedback mechanism. RF performs the best among the intelligence algorithms, since it is a combination of uncorrelated decision trees that exhibits bad data tolerance.

There is a seasonal effect on the forecasting problem; summer and autumn are easier to forecast than spring and winter. The training process of a neural network exhibits great randomness, while intelligent algorithms are generally more robust. The proposed Hybrid method performs most favourably among all methods, correcting erroneous fluctuations and negative forecasting. In fact, a major conclusion from this investigation is that simple combination of several good models can generate a more reliable prediction than any single method on its own. This may be found useful especially when there is no complete data for model training.

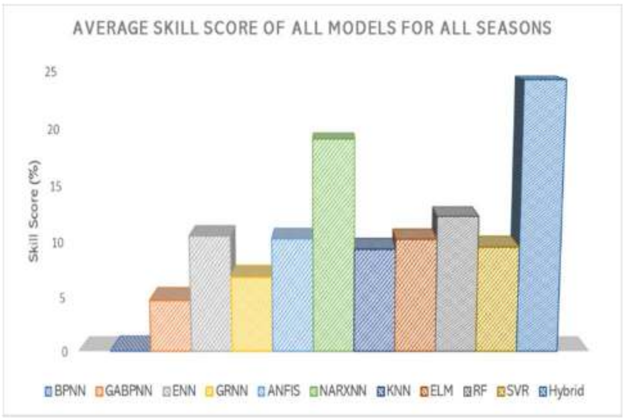


Fig. 5. Average skill score of all methods.

REFERENCES

[1] L. Liu, D. Liu, Q. Sun, H. Li and R. Wennersten, “Forecasting Power Output of Photovoltaic System Using A BP Network Method,” *Energy Procedia*, vol. 142, pp. 780-786, Aug. 2017.

[2] W. W. Anderson and O. A. Yakimenko, “Using neural networks to model and forecast solar PV power generation at Isle of Eigg,” *in Proc. CPE-POWERENG 2018*, Doha, Qatar, Apr. 2018.

[3] I. Khan, H. Zhu, J. Yao and D. Khan, “Photovoltaic power forecasting based on Elman Neural Network software engineering method,” *in Proc. ICSESS*, Beijing, China, pp. 747-750, Nov. 2017.

[4] P. Mandal, A. U. Haque, S. T. S. Madhira and D. I. Al-Hakeem, “Applying wavelets to predict solar PV output power using generalized regression neural network,” *in Proc. NAPS*, Manhattan, KS, USA, Sep. 2013.

[5] Wang. Fei, et al, “Short-Term Solar Irradiance Forecasting Model Based on Artificial Neural Network Using Statistical Feature Parameters,” *Energies*, vol. 5, pp. 1355-1370, 2012.

[6] I. Jayawardene and G. K. Venayagamoorthy, “Comparison of echo state network and extreme learning machine for PV power prediction,” *in Proc. CIASG*, Orlando, FL, Dec. 2014.

[7] A. Lahouar, A. Mejri and J. Ben Hadj Slama, “Importance based selection method for day-ahead photovoltaic power forecast using random forests,” *in Proc. GECS*, Hammamet, Tunisia, Mar. 2017.

[8] Fonseca, Joao Gari Da Silva, et al. “Use of support vector regression and numerically predicted cloudiness to forecast power output of a photovoltaic power plant in Kitakyushu, Japan,” *Progress in Photovoltaics,* vol. 20, pp. 874-882, July. 2011.

[9] M. Abuella and B. Chowdhury, “Solar Power Forecasting Using Support Vector Regression,” *in Proc. American Society for Engineering Management 2016 International Annual Conference*, 2016.

[10] Wolff. B, Lorenz. E, Kramer. E “Statistical Learning for Short-Term Pgotovoltaic Power Predictions,” *Computational Sustainability. Studies in Computational Intelligence*, vol. 645, pp. 31-45, April. 2016.

[11] Sheffield Solar. Regional PV Live Service. Available from: https://www.solar.sheffield.ac.uk/pvlive/regional/

[12] Schroedterhomscheidt, Marion, et al. "The Copernicus Atmosphere Monitoring Service (CAMS) Radiation Service in a nutshell", *in Proc. SolarPACES16*, Abu Dhabi, United Arab Emirates, pp. 11-14, Oct. 2016.

[13] SoDa Portal, www.soda-pro.com/web-services/meteo-data/merra.

[14] Y. Tao and Y. Chen, "Distributed PV power forecasting using genetic algorithm based neural network approach," *in Proc. AMS14*, Kumamoto, Japan, pp. 557-560, Aug. 2014.

[15] D. F. Specht, "A general regression neural network," in *IEEE Transactions on Neural Networks*, vol. 2, no. 6, pp. 568-576, Nov. 1991.

[16] J. -. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 665-685, May-June 1993.

[17] Yakowitz S., “Nearest-neighbour methods for time series analysis,” *Journal of Time Series Analysis*, vol. 8, pp. 235-247, Mar. 1987.

[18] Li. Zhongwen, et al. "Day-ahead hourly photovoltaic generation forecasting using extreme learning machine," *in Proc. CYBER,* Shenyang, China, pp. 779-783, Jun. 2015.

[19] *Source:* https://code.google.com/archive/p/randomforest-matlab/.

[20] C. Chang and C. Lin, “LIBSVM: a library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 1-27, April. 2011.