Ensemble Methods for Solar Power Forecasting

Zezhou Chen
School of Computer Science
University of Sydney
Sydney, NSW, Australia
zche7701@uni.sydney.edu.au

Irena Koprinska
School of Computer Science
University of Sydney
Sydney, NSW, Australia
irena.koprinska@sydney.edu.au

***Abstract*—We consider the task of predicting the solar power generated by a photovoltaic system, one-step ahead, from previous half-hourly photovoltaic power data. We propose a range of strategies for constructing static and dynamic heterogeneous ensembles and conduct an extensive evaluation using data for two years from two Australian solar power plants. We analyse the performance of the proposed static and dynamic ensembles and compare them with classical ensemble methods (bagging, boosting and random forest) and a baseline, showing an improved performance. The best result was achieved by the dynamic ensembles DEPast and DEPast+Future, in conjunction with the Peer Check algorithm for selecting base learners for inclusion in the dynamic ensembles.**

Keywords— solar power forecasting, ensembles of prediction models, dynamic ensembles, time series forecasting

I. INTRODUCTION

Solar power produced by PhotoVoltaic (PV) systems is one of the most promising renewable energy options. It is rapidly growing due to lower cost and improved efficiency of PV panels, the increasing number of large scale solar farms and the government policies encouraging the use of renewable energy. However, solar power is highly variable as it depends on weather conditions. This creates challenges for its large-scale integration into the power grid, and requires solar power forecasting to ensure efficient and reliable grid operation.

In this paper we consider univariate, one-step ahead solar power forecasting. Specifically, given: a time series of half-hourly solar power outputs till time t: $P_1, P_2, P_3, \ldots, P_t$, the goal is to forecast the next value P_{t+1}. While solar power depends on weather conditions, this information is not always available for the location of the PV panels. Recent studies [1-4] have shown promising results using only previous PV data, especially for very-short term prediction such as 30-min ahead.

Different approaches for PV power forecasting have been proposed - using statistical methods such as Linear Regression (LR) and autoregressive moving average [1, 5-8] or machine learning such as Neural Networks (NNs) [1, 6-12], Support Vector Regression (SVR) [7, 8, 13, 14] and nearest neighbors [1, 6-8, 15].

While the majority of previous work on time series forecasting has focused on using single prediction models, ensembles for time series forecasting tasks, combining the predictions of several base learners, have also been investigated and shown excellent results [16, 17]. Ensembles can be classified as static and dynamic. Static ensembles combine the predictions of the base learners in the same way for all new examples, regardless of the characteristics of the new example and changes in the time series. Static ensembles of NNs for solar power forecasting have been investigated in [3, 18, 19], demonstrating promising results. On the other hand, dynamic ensembles calculate a separate prediction for each new example, by adapting the combination of base learners to the new example and the changes in the time series. For instance, this can be done by tracking the error on recent data or predicting the error on future data, and weighting the predictions of the base learners accordingly. Dynamic ensembles for solar power forecasting, mainly employing NNs as base learners, have been investigated in [3, 4, 20] showing competitive results.

In this paper, we propose novel static and dynamic ensembles for time series forecasting, with a case study in solar power forecasting. We consider heterogeneous ensembles – ensembles combining the predictions of different types of base learners. The different assumptions and inductive bias of these models introduce inherent diversity which can lead to improved predictive performance.

Our contribution can be summarized as follows:

1. We propose two novel static ensembles, SE2 and SE3, and the application of trimming and winsorizing for combining the predictions in static ensembles.
2. We propose the Peer Check algorithm for selecting base learners for inclusion in dynamic ensembles.
3. We propose the MetaSelector algorithm for dynamic ensembles, which uses meta-learning to predict the best single learner for the new example.
4. We propose a new error tracking function for dynamic ensembles which weights the contribution of the recent error and predicted future error. We also propose a weighting function based on reciprocal error and show its effectiveness for both static and dynamic ensembles.
5. We conduct an extensive evaluation using data for two years from two Australian PV solar plants. We analyse the performance of the proposed static and dynamic ensembles and compare them with classical ensembles

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

(bagging, boosting and random forest) and a baseline, showing an improved performance. We assess the effect of the different types of base learner selection and weighting methods. The best result on both datasets was achieved by the dynamic ensembles, in conjunction with the Peer Check algorithm.

II. Problem Statement

We consider one-step ahead, univariate prediction of PV solar power output. Given a time series of half-hourly solar power outputs till time t: $P_1, P_2, P_3, \ldots, P_t$, our goal is to forecast the next value P_{t+1}.

III. Data and Data Preprocessing

A. Data

We used PV solar data for two years, from 1 January 2015 to 31 December 2016 (731 days), collected from two Australian PV plants: a rooftop PV plant located at the University of Queensland in Brisbane and the Sanyo PV plant located in Alice Springs. These PV plants are about 2600 km apart, in different climate zones, with more sunny days and less seasonal variability for the Alice Spring's plant. The data is available from [21] and [22] respectively. Only data for the 10 hours during the daylight period between 7am and 5pm was selected.

B. Data Preprocessing

The raw PV data was measured at 1-minute intervals for the UQ dataset and 5-minute intervals for the Sanyo dataset. It was aggregated to 30-minute by taking the average value of every 30 minute interval. There was a small percentage of missing values (0.82% for UQ and 1.98% for Sanyo), which were filled by a nearest neighbor approach. The data was also normalized to the range of [0,1]. Hence, each dataset contains 731 days x 20 half-hourly measurements = 14,620 data points.

IV. Static Ensembles

Static ensembles combine the predictions of the base learners in the same way for all new examples, without adapting the combination to the new example and the changes in the time series. We investigated three types of static ensembles (SE1, SE2 and SE3), each with three variations to calculate the final prediction (with/without trimming and winsorizing).

A. SE1: Average of All Base Learners' Predictions

This is the standard and widely used static ensemble method. To make a prediction for a new example t+1, it takes the average of the base learners' predictions for t+1.

B. SE2: Weighting Based on Previous Performance (Negative Error)

SE2 uses a weighed combination of the base learners' predictions. It assigns a weight to each learner based on its performance on year 1 data (training and validation set). Thus, it assumes that previous performance is an indicator for future performance. Higher weights are assigned to the more accurate base learners and lower to less accurate.

Specifically, it uses the softmax function of the negative of the error to calculate the weight matrix: $W = softmax(-E)$. The weight of a base learner i for predicting example t+1 is calculated as: $w_{i,t+1} = \frac{exp(-E_i)}{\sum_{j=1}^{S} exp(-E_j)}$, where E_i is the error of base learner i on the year 1 data and j is over all S base learners. The denominator ensures that all weights sum up to 1. For the error E we used the RMSE.

C. SE3: Weighting Based on Previous Performance (Reciprocal Error)

SE3 is similar to SE2 but uses the reciprocal error instead of the negative error: $W = softmax(1/E)$. This function increases the difference between the weights of the best and worst base learners. Hence, it increases the influence of the more accurate base learners and decreases the influence of the less accurate ones in the final prediction.

D. Trimming and Winsorizing

Trimming and winsorizing are statistical methods for limiting the extreme values when calculating a mean. Trimming of order k discards the k smallest and largest values, while winsorizing replaces them with the highest and lowest values from the remaining values of the distribution. These methods were shown to be more accurate than the standard mean, reducing the risk of high errors outliers [23].

We applied trimming and winsorizing to the predictions of the base learners and tested them in conjunction with all static ensembles.

V. Dynamic Ensembles

Dynamic ensembles adapt the combination of base learners' predictions to the new example, e.g. by re-calculating the weighing of the individual predictions for every new example. The main idea is that different base learners have different areas of expertise. We can match this expertise to the new example by selecting the most appropriate weighted combination of base learners or a single base learner, using suitable criteria.

We consider three types of dynamic ensembles: DEPast, DEFuture and DEPast+Future.

A. DEPast: Tracking Past Performance

DEPast assumes that performance on the most recent past examples is a good indicator for the performance on the next example. Specifically, it tracks the error E_i of each base learner L_i on the most recent m examples and assigns higher weights to the more accurate base learners. To predict a new example t+1, each base learner L_i is assigned a weight $w_{i,t+1}$ calculated based on E_i. We used $w_{i,t+1} = softmax(1/E_i)$ with winsorizing, as this was the best weighting function for static ensembles.

B. DEFuture: Estimating Future Performance

DEFuture predicts the error of the base learners for the new example and converts this error into weights for the final prediction. Base learners that are predicted to be more accurate will be assigned higher weights.

To do this, every base learner L_i is assigned an associated meta-learner ML_i, trained to predict the error E_i (we used

RMSE) of L_i for the new time point. Specifically, ML_i, takes as an input previous PV data from the forecasting window w and predicts the error of E_i for time point t+1. This error is then converted into a weight for L_i and used in the weighted average to combine the predictions of all base learners.

In our implementation, the weights were calculated again using the $softmax(1/E)$ function with winsorizing. As meta-learners we used LR models.

C. DEPast+Future: Considering Both Past and Estimated Future Performance

This ensemble method uses both error components (previous error and predicted future error) to determine the weights of the base learners for the new example. Specifically, for a base learner L_i, it firstly calculates the past error PE_i as in DEPast and the predicted future error FE_i as in DEFuture, then combines them: $E_i = \alpha_1 PE_i + \alpha_2 FE_{i_i}$, where the coefficients $\alpha_1$and α_2 are determined empirically ($\alpha_1 + \alpha_2 = 1$). Then, E_i is converted to a weight for L_i using $softmax(1/E)$ with winsorizing as in DEPast and DEFuture.

In our experiments we used $\alpha_1 = 0.4$ and $\alpha_2 = 0.6$ for the UQ dataset and $\alpha_1 = 0.2$ and $\alpha_2 = 0.8$ for the Sanyo datasets.

D. The Peer Check Algorithm

Effective ensembles combine diverse learners, which are experts in different areas. Depending on their expertise, the base learners will perform better on some examples and worse on others. The Peer Check algorithm aims to estimate which base learners are not likely to perform well on the new example and exclude them from the ensemble.

The base learners are firstly divided into two groups – "good" and "bad", based on the error tracking strategy of the dynamic ensemble, e.g. previous error, predicted future error or both. Then, the predictions of the base learners on the new example t+1 are obtained and compared. If a good learner G has a similar prediction as a bad learner B, and none of the other good learners supports G, i.e. has a similar prediction to G, then G will be excluded from the ensemble for t+1. If at least one other good learner supports G, then G will be included in the ensemble. Fig. 2 illustrates the exclusion and inclusion conditions. The final ensemble is formed by all good learners that are confirmed by the Peer Check algorithm.

The Peer Check algorithm has three parameters: *sp* - used to divide the base learners into good and bad, *ck1* and *ck2* - used to determine if two predictions are similar (between the current good learner and any of the other learners). For a given new example, Peer Check splits the base learners into two groups, obtains the predictions of each base learner, and then iterates through the predictions of the good learners checking if the inclusion criterion is satisfied or the learner should be excluded from the ensemble for this example.

In summary, Peer Check is an algorithm for base learner selection, for use with dynamic ensembles. We tested it in conjunction with the dynamic ensembles DEPast, DEFuture and DEPast+Future.

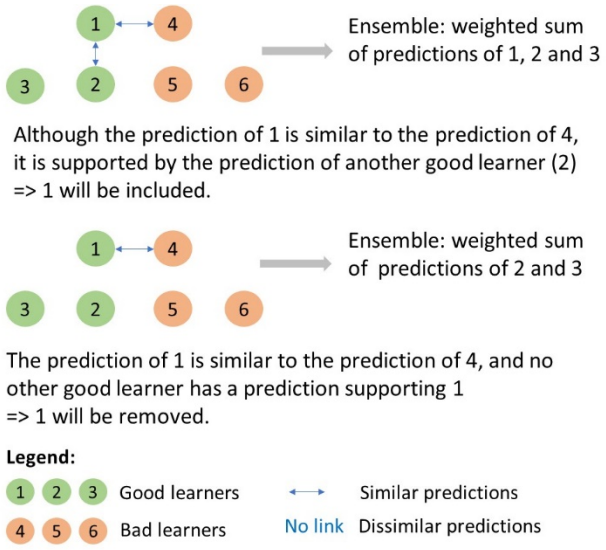

Fig. 1. The Peer Check algorithm

E. BL: Select the Best Base Learner

Instead of combining the predictions of all base learners, this method selects a single best base learner - the one with the lowest error E_i, calculated according to the dynamic ensemble type. It was used with all three dynamic ensembles.

F. MetaSelector: Predict the Best Base Learner Using Meta-Learning

We employed again meta-learning, but this time for a different task: predicting the best base learner for the new example. Using the training data, we built a classifier (called MetaSelector); its input is a sequence of PV values and its target class corresponds to one of the seven base learners.

For the meta-learners, we experimented with different types of classifiers, evaluating performance on the validation set. The selected classifiers were SVM for the UQ and LDA for the Sanyo dataset.

G. Base Learners

Successful ensembles include diverse individual learners. We chose to combine seven base learners representing different machine learning paradigms – NN (multi-layer feedforward networks), SVR, LR, Polynomial Regression (PR), Long Short Term Memory (LSTM) networks, Regression Tree (RT) and ARIMA. NN and SVR are also widely used in solar power forecasting, demonstrating good results.

VI. Experiment Setup

A. Data Sets

Each dataset was divided into two sets: year 1 (2015 data) and year 2 (2016 data). The year 1 data was further split into training (the first 70%), used for building the prediction models, and validation (the remaining 30%), used for parameter tuning. The year 2 data was used as a test set, to evaluate the performance of the trained prediction models.

B. Evaluation Measures

We used two standard performance measures: Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE).

C. Methods Used for Comparison

We compare the performance of the proposed ensembles with three groups of methods: 1) the single prediction models comprising the ensembles (SVR, LR, PR, NN, LSTM, ARIMA and RT), 2) classic static ensembles: bagging, boosting (2 versions – AdaBoost and XGBoost gradient boosting) and Random Forest (RF) and 3) a persistence model which outputs the previous value (for time t) as the prediction for time t+1.

D. Parameter Tuning

The parameters of all models were tuned using grid search with evaluation on the validation set. All models except NN, LSTM and ARIMA were implemented using the *scikit-learn* library; NN and LSTM were implemented using *keras* and ARIMA using *statsmodels*.

VII. Results and Discussion

A. Base Learners

Table I shows the accuracy of all base learners for the two datasets, and Fig. 2 presents the RMSE in sorted order. For both datasets, the best performing base learner is LSTM, closely followed by SVR and NN, and the least accurate is ARIMA.

Although overall ARIMA performed significantly worse than the other base learners, after examining its half-hourly performance, we found that there are times during the day when it is the most accurate base learner, e.g. in the middle of the day. Hence, it contributes to the diversity of the ensemble and was included.

The static and dynamic ensembles discussed in the next sections include all seven base learners: SVR, LR, PR, NN, LSTM and RT.

TABLE I. Accuracy of Base Learners

Base learner	RMSE [kW]	MAE [kW]
UQ dataset		
SVR	110.09	83.24
LR	118.06	89.81
PR	131.48	102.31
NN	112.22	83.15
LSTM	109.44	78.89
ARIMA	152.59	118.98
RT	123.24	92.96
Sanyo dataset		
SVR	0.6815	0.4925
LR	0.8338	0.5768
PR	0.8278	0.5418
NN	0.6948	0.4582
LSTM	0.6797	0.4371
ARIMA	1.1878	0.9085
RT	0.8116	0.5316

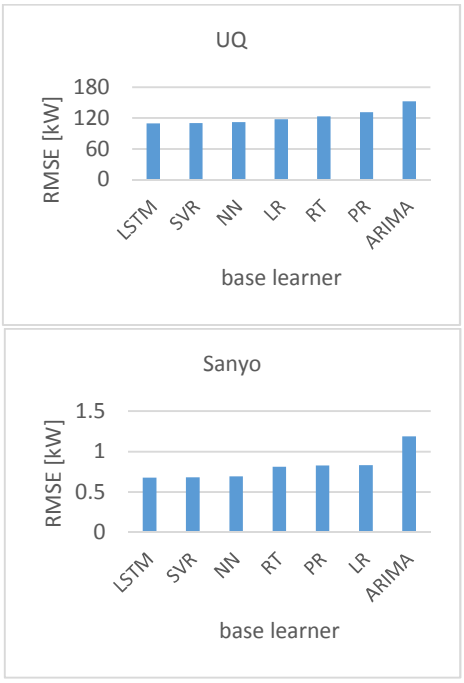

Fig. 2. Comparison of base learners (RMSE)

B. Static Ensembles

Table II shows the accuracy results of the three static ensembles SE1, SE2 and SE3, and their variations using winsorizing and trimming.

TABLE II. Accuracy of Static Ensembles

Static ensembles	RMSE [kW]	MAE [kW]
UQ dataset		
SE1	110.28	83.52
SE1-W	108.98	81.94
SE1-T	107.69	80.19
SE2	110.19	83.33
SE2-W	108.98	81.94
SE2-T	108.98	81.94
SE3	107.04	79.91
SE3-W	107.04	79.91
SE3-T	107.78	80.65
Sanyo dataset		
SE1	0.6924	0.4786
SE1-W	0.6689	0.4443
SE1-T	0.6689	0.4431
SE2	0.6893	0.4756
SE2-W	0.6689	0.4437
SE2-T	0.6623	0.4431
SE3	0.6610	0.4443
SE3-W	0.6604	0.4407
SE3-T	0.6604	0.4437

We firstly compare the performance of SE1, SE2 and SE3. Recall that SE1 takes the average of the base learners' predictions, while SE2 and SE3 weight the base learners' predictions depending on their error on the training and validation set (the year 1 data) using two different ways: softmax of the negative of the error or softmax of the reciprocal error.

The results show that SE3 is the best static ensemble, while SE1 and SE2 performed similarly. To gain a better understanding, we examined the difference between SE2 and SE3. Table III shows the error on year 1 and the weights based on this error for the UQ dataset. We can see that the SE2's weighting method calculates very similar weights for all base learners. This means assigning equal weights to all base learners in the final prediction as in SE1 and explains why SE1 and SE2 perform similarly. On the other hand, we can see that the SE3's weighting method calculates different weights by increasing the difference between the best and worst performing base learners, which proved beneficial for our datasets.

TABLE III. SE2 AND SE3 WEIGHTS FOR UQ DATASET

Base learner	RMSE [kW]	SE2 weight	SE3 weight
SVR	113.15	0.1455	0.2943
LR	126.76	0.1434	0.1225
PR	142.59	0.1409	0.0543
NN	121.59	0.1442	0.1653
LSTM	117.50	0.1448	0.2181
ARIMA	163.43	0.1378	0.0237
RT	126.85	0.1434	0.1217

Now we compare the performance of the static ensembles with the single base learners (Tables I and II, and also Fig. 3 for visual comparison in sorted order). We can see that SE3 outperforms all base learners, although the differences with the best performing base learner, LSTM, are small. On the other hand, SE1 and SE2 perform similarly to LSTM, SVR and NN.

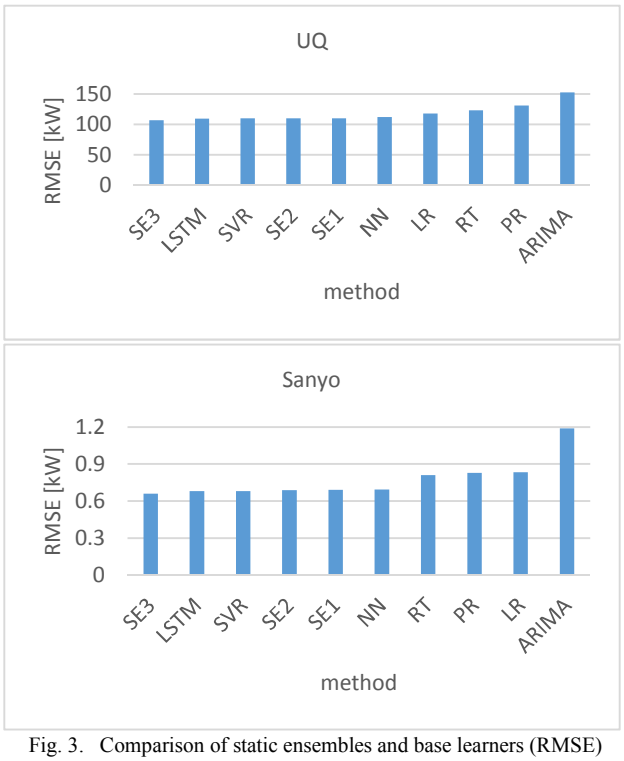

Fig. 3. Comparison of static ensembles and base learners (RMSE)

Finally, we assess the effect of winsorizing and trimming on the accuracy of the static ensembles. The purpose of these methods is to reduce the effect of extreme prediction values when combining the predictions. As shown in Fig. 4, winsorizing and trimming were effective for SE1 and SE2 on both datasets and resulted in considerably improved accuracy.

However, for SE3 winsorizing and trimming did not improve the accuracy. In all but one case, the accuracy of SE3+W was the same as SE3 or negligibly higher. A possible explanation is that the extreme prediction values (that were removed or soften by applying trimming and winsorizing) had low weights, i.e. were produced by less accurate base learners; modifying low-weight predictions did not affect the overall performance. However, in one case – SE3+T on the UQ dataset, trimming had a negative impact compared to winsorizing and not using these methods. Trimming is more aggressive than winsorizing as it removes the extreme values; a possible explanation is that the extreme values that were removed were produced by highly accurate base learners with high weights.

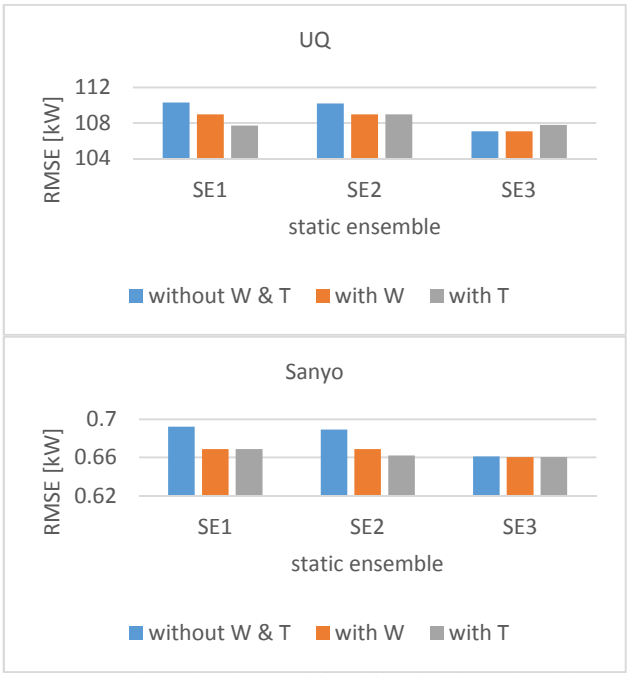

Fig. 4. Comparison of static ensembles with and without winsorizing and trimming (RMSE)

In summary, we found SE3 to be the most effective static ensemble outperforming SE1 and SE2 and all single base learners. The use of winsorizing and trimming was effective for SE1 and SE2 but not for SE3.

C. *Dynamic Ensembles With and Without Peer Check*

Tables IV shows the performance of the three dynamic ensembles (DEPast, DEFuture and DEPast+Future) with and without Peer Check.

We firstly compare the dynamic ensemble versions without Peer Check. On both datasets the best results were achieved by DEPast+Future, which shows the importance of using both past performance and predicted future performance when calculating the weights for the base learners' predictions. For the UQ dataset the second best method was DEPast followed by DEFuture while for the Sanyo dataset this order was reversed.

TABLE IV. ACCURACY OF DYNAMIC ENSEMBLES WITH AND WITHOUT PEER CHECK

Dynamic Ensemble	RMSE [kW]	MAE [kW]
UQ dataset		
DEPast	105.65	75.37
DEPast with PC	104.07	75.56
DEFuture	109.72	82.96
DEFuture with PC	107.69	82.98
DEPast+Future	101.76	73.98
DEPast+Future with PC	101.20	73.89
Meta Selector	111.30	80.46
Sanyo dataset		
DEPast	0.7056	0.4371
DEPast with PC	0.6394	0.4112
DEFuture	0.6689	0.4431
DEFuture with PC	0.6647	0.4612
DEPast+Future	0.6532	0.4184
DEPast+Future with PC	0.6460	0.4088
Meta Selector	0.7435	0.4497

We now turn to assessing the impact of the Peer Check method. From Fig. 5, we can see that Peer Check improved the results for all three ensembles on both datasets. The biggest improvement was achieved for DEPast on the Sanyo dataset and DEFuture on the UQ dataset.

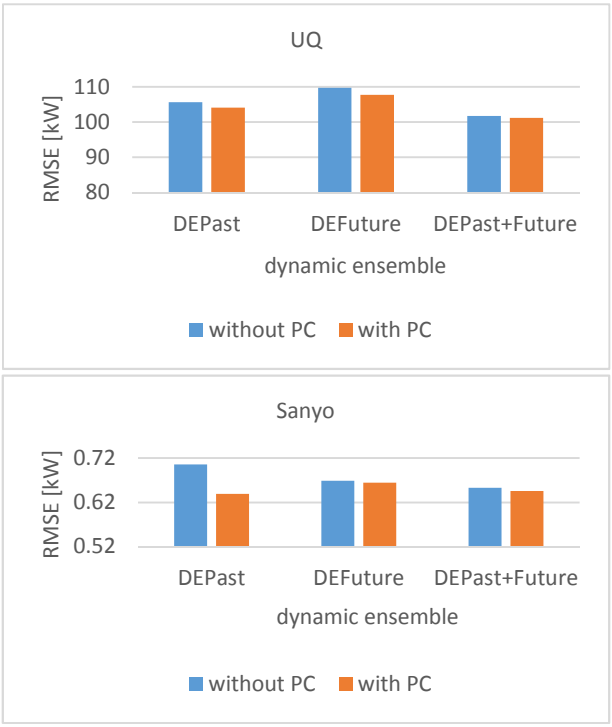

Fig. 5. Comparison of dynamic ensembles with and withour Peer Check (RMSE)

In the other cases, the improvement was small and we investigated the reasons for this. We found that the application of Peer Check was triggered in about 1/4 of the test examples. For instance, for DEPast+Future, Peer Check intervened in the prediction procedure 1799 times, which is 24.6% of all 7299 predictions on the test data. Peer Check also depends on the tracking error used to evaluate the performance of the base learners and the threshold parameters sp, ck1 and ck2. A more reliable error estimate and better parameter tuning for each dataset can further improve the results.

D. Dynamic Ensembles With Best Learner Selection

This method selects a single best base learner instead of using a weighted combination of the predictions of all base learners. It can be used together with DEPast, DEFuture and DePast+Future by selecting the base learner with the lowest error based on the respective strategy. The results are shown in Table V, where BL denotes "Best Learner".

TABLE V. ACCURACY OF DYNAMIC ENSEMBLES WITH BEST LEARNER SELECTION

Dynamic Ensemble	RMSE [kW]	MAE [kW]
UQ dataset		
DEPast with BL	107.59	75.65
DEFuture with BL	109.81	85.96
DEPast+Future with BL	102.94	74.35
Sanyo dataset		
DEPast with BL	0.7050	0.4323
DEFuture with BL	0.7700	0.5129
DEPast+Future with BL	0.7104	0.4612

Fig. 6 presents a comparison with the ensemble versions which use a weighted combination of the predictions of all base learners. The single best learner method achieved similar results to the weighted combination method in two cases (DEFuture for UQ and DEPast for Sanyo) but had lower accuracy in the other four cases, especially on the Sanyo dataset. Hence, for our data, combining all base learners and weighting their contribution gives better accuracy than selecting a single best learner.

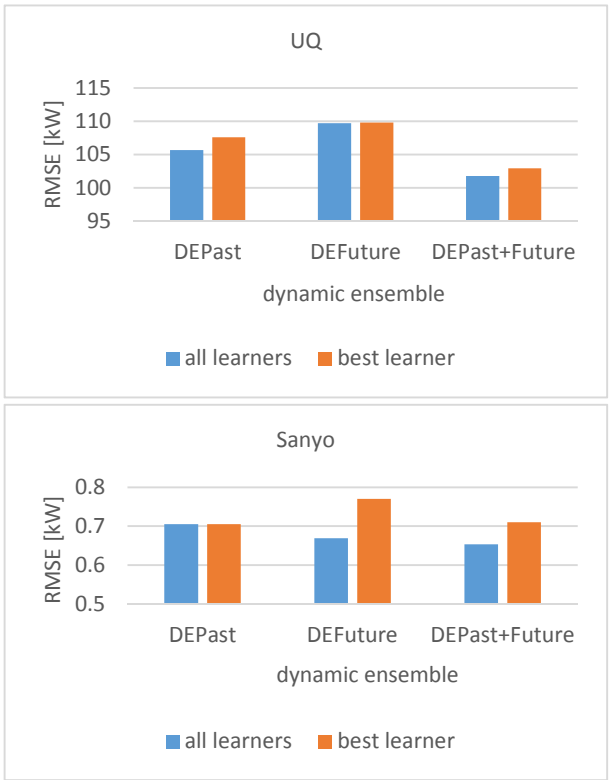

Fig. 6. Comparison of dynamic ensembles – combining all learners vs selecting a single best learner (RMSE)

E. *Meta Learning for Best Learner Selection*

The results of the MetaSelector method are presented in Table VI. Compared to the other dynamic ensembles, its accuracy is the lowest.

TABLE VI. ACCURACY OF METASELECTOR FOR BEST LEARNER SELECTION

Meta-Learner	RMSE [kW]	MAE [kW]
UQ dataset		
MetaSelector	111.30	80.46
Sanyo dataset		
MetaSelector	0.7827	0.4756

Hence, both methods for selecting a single best learner - BL and Meta Selector - produced similar or inferior results compared to the dynamic ensembles using a weighted combination of all base learners.

However, it is possible to achieve an improved performance with a single learner. We conducted additional experiments to assess the performance of a "perfect" MetaSelector, which correctly selects the best base learner for each test example. The performance improves greatly – e.g. the RMSE on the UQ dataset for the "perfect" MetaSelector is 67.87 kW, a considerable improvement compared to 111.30 kW (Table VI).

We also investigated the performance of DEPast+Future with a "perfect" meta-learning component, i.e. estimating correctly the error of all base learners on the test set. We achieved RMSE=73.98 kW on the UQ dataset, which is also a significant improvement compared to 101.76 KW (Table IV).

This shows the potential for improving the accuracy. In future work we plan to investigate more sophisticated meta-learning components for both methods and devise a better method for error tracking and estimation.

F. *Classical Ensembles and Persistence Model*

Table VII shows the accuracy results of the classical ensemble methods and the persistence model used as a baseline.

TABLE VII. ACCURACY OF CLASSICAL ENSEMBLES AND PERSISTENCE BASELINE

Method	RMSE [kW]	MAE [kW]
UQ datset		
Persistent	184.29	124.80
XGBoost	109.91	81.20
AdaBoost	125.93	98.98
Bagging	108.15	79.07
RF	110.00	81.02
Sanyo dataset		
Persistent	0.7498	0.5841
XGBoost	0.7321	0.4859
AdaBoost	0.6893	0.4371
Bagging	0.8296	0.6394
RF	0.6845	0.4335

The classical ensembles show a very competitive performance. Different methods perform best on the two datasets - on the UQ dataset, Bagging is the most accurate method and AdaBoost is the least accurate, while on the Sanyo dataset RF is the most accurate and Bagging is the least accurate.

G. *Comparison of All Methods*

Fig. 7 presents a comparison of the best method from each group - base learner, static ensemble, dynamic ensemble, dynamic ensemble with PC, dynamic ensemble with BL, MetaSelector, classic ensemble and persistence baseline. (Note that the persistent baseline is not shown on the UQ graph in Fig.7 since it is disproportionally less accurate than the other methods.)

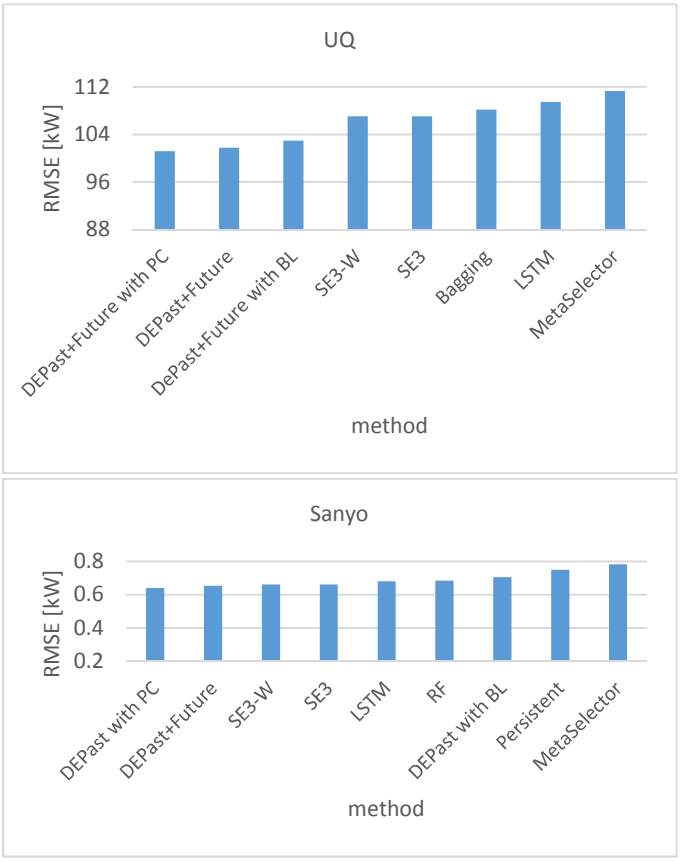

Fig. 7. Comparison of the proposed ensemble methods (best from each group) with the classical ensembles

We can summarize the results as follows:

- *Best model:* The most accurate models for both datasets are dynamic ensembles with Peer Check – on the UQ dataset: DEPast+Future with PC, on the Sanyo dataset: DEPast with PC, closely followed by DEPast+Future with PC. This shows the potential of the proposed dynamic ensembles with Peer Check.
- *Best dynamic ensemble without Peer Check:* DEPast+Future was the most accurate dynamic ensemble without Peer Check on both datasets. It uses a novel error tracking function which weights the contribution of the previous error and predicted future error. DEPast+Future outperformed the dynamic ensemble versions which use only one of these error components, all static ensembles, the single models comprising the ensemble and the persistent baseline.

- *All base learners vs a single base learner:* The dynamic ensembles combining all base learners performed better than the ensembles selecting a single best base learner. From the two strategies for best base learner selection – BL and MetaSelector, BL was more successful.
- *Static ensembles:* The proposed static ensemble SE3, which weights the base learners' predictions depending on their previous performance using the reciprocal error, was the best static ensemble. It outperformed the classic static ensemble SE1 which takes the average of the base learners' predictions, SE2 which is similar to SE3 but uses the negative error and also all base learners. The use of winsorizing and trimming was very effective for SE1 and SE2 but not for SE3.
- *Best base learner:* The most accurate base learner was LSTM. It showed competitive performance and outperformed the classic tree-based ensembles RF, AdaBoost and XGBoost on both datasets.

VIII. CONCLUSION

In this paper we considered one-step ahead forecasting of PV power output, from a time series of previous half-hourly PV power data. We proposed a range of strategies for constructing static and dynamic heterogeneous ensembles. In particular, we proposed the Peer Check algorithm for selecting base learners for inclusion in dynamic ensembles, the MetaSelector algorithm which predicts the best single learner for the new example using meta learning, the novel error tracking function for the dynamic ensemble DEPast+Future, and several novel strategies for building static ensembles that consider previous performance, winsorizing and trimming.

We conducted an extensive evaluation using data for two years from two Australian PV plants, located in Brisbane and Alice Springs. We analysed the performance of the static and dynamic ensembles and compared them with classical ensembles (bagging, boosting and random forest) and a baseline, showing an improved performance. We also assessed the effect of the different types of base learner selection and weighting methods. The best result on both datasets was achieved by using dynamic ensembles with Peer Check.

REFERENCES

[1] H. T. C. Pedro and C. F. M. Coimbra, "Assessment of forecasting techniques for solar power production with no exogenous inputs," Solar Energy, vol. 86, pp. 2017-2028, 2012.

[2] M. Rana, I. Koprinska, and V. Agelidis, "Univariate and multivariate methods for very short-term solar photovoltaic power forecasting," Energy Conversion and Management, vol. 121, pp. 380-390, 2016.

[3] Z. Wang, I. Koprinska, A. Troncoso, and F. Martinez-Alvarez, "Static and dynamic ensembles of neural networks for solar power forecasting," in Proceedings of the International Joint Conference on Neural Networks (IJCNN), 2018.

[4] Z. Wang and I. Koprinska, "Solar Power Forecasting Using Dynamic Meta-Learning Ensemble of Neural Networks," in Proceedings of the International Conference on Artificial Neural Networks (ICANN), 2018.

[5] P. Bacher, H. Madsen, and H. A. Nielsen, "Online short-term solr power forecasting," *Solar Energy,* vol. 83, pp. 1772-1783, 2009.

[6] Y. Chu, B. Urquhart, S. M. I. Gohari, H. T. C. Pedro, J. Kleissl, and C. F. M. Coimbra, "Short-term reforecasting of power output from a 48 MWe solar PV plant," *Solar Energy,* vol. 112, pp. 68-77, 2015.

[7] H. Long, Z. Zhang, and Y. Su, "Analysis of daily solar power prediction with data-driven approaches," *Applied Energy,* vol. 128, pp. 29-37, 2014.

[8] Z. Wang, I. Koprinska, and M. Rana, "Solar power prediction using weather type pair patterns," in Proceedings of the International Joint Conference on Neural Networks (IJCNN), 2017.

[9] S. K. Chow, E. W. Lee, and D. H. Li, "Short-term prediction of photovoltaic energy generation by intelligent approach," *Energy and Buildings,* vol. 55, pp. 660-667, 2012.

[10] C. Chen, S. Duan, T. Cai, and B. Liu, "Online 24-h solar power forecasting based on weather type classification using artificial neural networks," *Solar Energy,* vol. 85, pp. 2856-2870, 2011.

[11] I. Koprinska, D. Wu, and Z. Wang, "Convolutional Neural Networks for Energy Time Series Forecasting," in Proceedings of the International Joint Conference on Neural Networks (IJCNN), 2018.

[12] J. F. Torres, A. Troncoso, I. Koprinska, Z. Wang, and F. Martínez‐Álvarez, "Big data solar power forecasting based on deep learning and multiple data sources," *Expert Systems,* vol. 36, no. 4, p. e12394.

[13] M. Rana, I. Koprinska, and V. G. Agelidis, "2D-interval forecasts for solar power production," *Solar Energy,* vol. 122, pp. 191-203, 2015.

[14] J. Shi, W.-J. Lee, Y. Lin, Y. Yang, and P. Wang, "Forecasting power output of photovoltaic systems based on weather classification and support vector machines," *IEEE Transactions on Industry Applications,* vol. 48, no. 3, pp. 1064-1069, 2012.

[15] Z. Wang and I. Koprinska, "Solar power prediction with data source weighted nearest neighbours," in Proceedings of the International Joint Conference on Neural Networks (IJCNN), 2017.

[16] M. Oliveira and L. Torgo, "Ensembles for time series forecasting," in Proceedings of the Sixth Asian Conference on Machine Learning, 2014.

[17] V. Cerqueira, L. Torgo, F. Pinto, and C. Soares, "Arbitrated ensemble for time series forecasting," in Proceedings of the European Conference on Machine Learning and Principles of Knowledge Discovery in Databases (ECML-PKDD), 2017.

[18] M. Rana, I. Koprinska, and V. G. Agelidis, "Forecasting solar power generated by grid connected PV systems using ensembles of neural networks," in Proceedings of the International Joint Conference on Neural Networks (IJCNN), 2015.

[19] M. Rana, I. Koprinska, and V. G. Agelidis, "Solar power forecasting using weather type clustering and ensembles of neural networks," in Proceedings of the International Joint Conference on Neural Networks (IJCNN), 2016.

[20] I. Koprinska, M. Rana, and A. Rahman, "Dynamic ensemble using previous and predicted future performance for multi-step-ahead solar power forecasting," in Proceedings of the International Conference on Artificial Neural Networks (ICANN), 2019.

[21] *University of Queensland Solar Data*. http://www.uq.edu.au/solarenergy/

[22] *Sanyo Solar Data*. Available: http://dkasolarcentre.com.au/source/alice-springs/dka-m4-b-phase

[23] V. R. R. Jose and R. L. Winkler, "Simple robust averages of forecasts: some empirical results," *International Journal of Forecasting,* vol. 24, no. 1, pp. 163-169, 2008.