Received: 24 March 2021 | Revised: 30 May 2021 | Accepted: 22 June 2021

IET Renewable Power Generation

DOI: 10.1049/rpg2.12243

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH PAPER

Model-agnostic online forecasting for PV power output

HyunYong Lee | **Jun-Gi Lee** | **Nac-Woo Kim** | **Byung-Tak Lee**

Honam Research Center (HRC), Electronics and Telecommunications Research Institute (ETRI), Buk-gu, Gwangju, Korea

Correspondence
HyunYong Lee, Honam Research Center (HRC), Electronics and Telecommunications Research Institute (ETRI), 176-11 Cheomdan Gwagi-ro, Buk-gu, Gwangju, Korea.
Email: hyunyonglee@etri.re.kr

Funding information
Korea Institute of Energy Technology Evaluation and Planning, Grant/Award Number: 20181210301570

Abstract

A reliable forecasting model is required for photovoltaic (PV) power output because solar energy is highly volatile. Another driver for the need of a reliable forecasting model is concept drift, which means that the statistical properties of the data change over time. In this paper, an online forecasting method to handle concept drift is proposed. First, the problem of forecasting in batch learning is transformed into a forecasting in online learning setting. Then, an online learning algorithm is applied, which is good for handling concept drift. Through experiments using the real-world data, it is shown that the method noticeably improves performance compared to the case where a trained model is used. Under various concept drift scenarios, the method improves performance by up to 87.3%. It is also shown that the re-training method (a representative existing method) has several limitations. This method requires several issues to be solved, such as selection of a proper window size, and this is evident through results showing different performance under different settings. In contrast, the method shows a reliable and desirable performance under various concept drift scenarios and thus outperforms the re-training method. The method improves performance by up to 79%.

1 | INTRODUCTION

A reliable forecasting model is necessary to enable highly volatile solar energy to be utilised. A reliable forecasting allows distribution system operators and transmission system operators to cope with the highly volatile nature of photovoltaic (PV) systems and thus to enhance the grid stability and reduce the cost of ancillary services [1]. In pursuing grid stability and unit commitment, the market regulations of several countries require day-ahead forecasting [2–4]. Another exemplary case is a microgrid that utilises PV systems as a generating source and is equipped with an energy storage system (ESS). In that microgrid, a reliable forecasting system is required to increase PV self-consumption, decrease curtailment losses and improve ESS revenues [5].

Considering that forecasting is a vital part of utilising solar energy, a forecasting model needs to be reliable, even under unexpected cases. From perspective of a forecasting model, which is typically a statistical model, unexpected cases indicate concept drift. Concept drift means statistical properties of data change over time in unforeseen ways. The relationship between inputs and desired outputs changes when concept drift happens which obviously impacts on the ability to forecast. Therefore, a forecasting model, trained with data before concept drift, loses its effectiveness when concept drift happens.

Concept drift can happen due to several reasons. PV faults can cause concept drift. In most cases, PV power output is heavily affected by weather conditions and so a forecasting model is typically trained including weather conditions. However, even with similar weather conditions, PV faults (e.g. hot-spotted PV string [7], partial shading [8], dust [9], and line-line faults [10]) likely cause concept drift by degrading PV power output [6]. Some features that are unknown to the forecasting model can also cause concept drift. In building a forecasting model, we use known features, which are allowed in terms of cost and technical aspect. However, unknown features heavily affect PV power output in some cases. For example, in Korea, which has four distinct seasons, it is common to install PV plants with typical meteorological sensors (e.g. temperature, irradiance, humidity, and wind speed). As a result, a forecasting model is unlikely to produce reliable forecast results when PV panels are covered by snow, while sky is clear.

A possible approach to deal with concept drift is to train a model with newly acquired data. Re-training a forecasting model

can be done periodically with recent data [13, 14] or when a forecasting error exceeds a pre-defined threshold [15–17]. However, this re-training approach often fails when similar inputs and different outputs are given (i.e. data before and after concept drift are mixed) or data after concept drift is limited. Another possible approach for dealing with concept drift is an ensemble-based approach. Individual forecasting models are dynamically combined, according to their forecasting errors [18–20] or a single forecasting model is selected among candidate models [21–23]. The ensemble-based approach may cause computational and memory overhead when using multiple models. Besides, the performance of an ensemble model is limited by each individual models' performance.

To create a reliable forecasting model, even with concept drift, and to overcome the limitations of existing approaches, we propose a model-agnostic online forecasting method. The basis of our method is to adjust the output of a single forecasting model in an online learning setting rather than through re-training or updating a trained model. Therefore, our approach is applicable to any forecasting models. We first transform the problem of forecasting with multivariate time series, in batch learning, into forecasting with univariate time series, in an online learning setting, to effectively adjust outputs of the forecasting model. The domain transformation is conducted through simple data transformation. Then, we apply an online learning algorithm, which is good for handling concept drift. We maintain and update a simple online model, while not changing the basis of the forecasting model.

To examine the feasibility of our model-agnostic online forecasting (i.e. MAOF), we conduct a series of experiments, using real-world data (i.e. 17,516 30-min interval data of 1-year downloaded from national renewable energy laboratory (NREL) [28, 29]). We use a light gradient boosting machine (LightGBM) [26] (as one of the popular machine learning models) and long short-term memory (LSTM) [27] (as one of the popular deep learning models) as the base forecasting models. We examine MAOF and existing methods under various concept drift scenarios.

We first examine how much MAOF improves performance compared to the case where a trained model is used as it is (i.e. Naive). Under no concept drift, Naive and MAOF show similar performances. In contrast, regardless of a base forecasting model, MAOF noticeably improves performance compared to the Naive model in concept drift scenarios. As the degree of concept drift increases, the degree of improvement increases. For example, with LightGBM, in a sudden concept drift scenario, the improvement in percentage increases from 27.2% to 86.7% as the ratio of degradation (used for realising concept drift) increases from 0.1 to 0.5.

We also compare MAOF with the re-training method (i.e. Retrain) as a representative of an existing method. MAOF shows reliable and desirable performance under various concept drift scenarios, regardless of the base model. MAOF shows the best performance in most cases. For example, MAOF improves performance by up to 79% and 78.5%, compared to Retrain with LightGBM and LSTM, respectively. In contrast, Retrain shows unreliable performance. In other words, Retrain requires different window sizes (i.e. the number of data used for re-training) to achieve its best performance under different concept drift scenarios. Therefore, several issues are required to be solved, including selection of a suitable window size to effectively utilise Retrain. Even with those solutions, Retrain shows fundamental limitations in some concept drift scenario. This is due to the fact that data likely includes cases before and after concept drift and that the mixed data results in poor performance. As a result, Retrain noticeably decreases in performance as the degradation ratio increases in those concept drift scenarios.

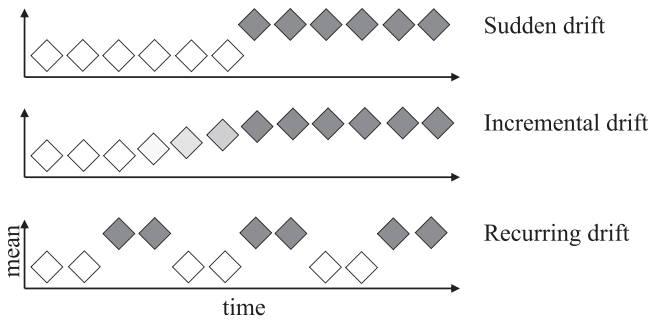


FIGURE 1 Types of concept drift

The rest of this paper is organised as follows. Section 2 describes backgrounds. Section 3 introduces model-agnostic online forecasting and Section 4 verifies its feasibility. Finally, Section 5 concludes this paper.

2 | BACKGROUND

2.1 | Concept drift

Training a model with training data creates a model that understands specific features or distributions of the training data. Therefore, as long as distributions of the training data and test data are the same, a trained model likely shows good performance during the test phase. However, when distributions of test data changes (i.e. concept drift), effectiveness of a trained model is lost. Figure 1 shows how concept drift may happen in time series. For simplicity, as shown in Figure 1, the distribution of data is represented by the mean. When concept drift happens, we may need to update or re-train a model or do something to sustain the effectiveness of a model for new data distributions.

Regarding PV power output forecasting, concept drift may happen when the desired outputs of a model for similar inputs (e.g. weather conditions) change. In other words, the relationship between inputs and desired outputs (learned by a model) is broken when concept drift happens. Various PV faults may cause concept drift. A forecasting model is typically trained with weather conditions because PV power output is heavily affected by weather conditions. However, various PV faults (e.g. hot-spotted PV strings [7], partial shading [8], dust [9], and line-line faults [10]) degrade PV power output [6]. In other words, PV faults likely result in degraded PV power out-

TABLE 1 Notations

Symbol	Notation
y_t	Forecasting target
$\hat{y}_t$	Forecasting output
F_t	Set of features for a forecasting model
h_θ	Forecasting model with trainable weights θ
l_t	Loss caused by the difference between y_t and $\hat{y}_t$

puts for similar weather conditions. Also, environmental factors that are not used for training a model may affect PV power output. For example, snow cover noticeably degrades PV power output, but it is not easy to use that information when training a model because there are costs associated. In other words, environmental factors that are unknown to a model may degrade PV power outputs for similar features known to a model.

The objective of this paper is to sustain the effectiveness of a trained model when concept drift happens in the field of PV power output forecasting.

2.2 | Online learning

Online learning considers cases in which data becomes available in a sequential order and is used to train or update a model at each step. Conversely, batch learning trains a model by using the entire data set at once. Online learning can be used in two different ways. First, online learning is used when it is not feasible to train a model over the entire dataset, because of limits in memory or computation power. Second, online learning is used in cases in which a model needs to dynamically adapt to new patterns in the data, or when the data is given sequentially. In this aspect, online learning is a good candidate for solving the issue of concept drift. In this paper, we adopt the method of online learning to achieve our objective.

More specifically, we consider the following online learning settings in this paper. First, a forecasting model is trained with training data in advance. Then, during a test phase, a trained model predicts $\hat{y}_t$ using $F_{t-N:t-1}$ and $y_{t-N:t-1}$, and the true y_t is disclosed. Then, the forecasting model suffers a loss $l_t = L(y_t, \hat{y}_t)$. Using l_t or $l_{t-(M-1):t}$, a forecasting model is updated. Updated forecasting model is used for prediction at next step. The above procedures are repeated during the test phase. For an explanation of the notation used within this paper, please refer to Table 1. $D_{a:b}$ indicates data D from time steps a to b, $\{D_a, D_{a+1}, \dots, D_b\}$.

2.3 | Related work

Most existing approaches for dealing with concept drift can be divided into two groups: re-training-based approach and ensemble-based approach.

2.3.1 | Re-training-based

Rationale of re-training-based approach is to adjust a model to dynamic environment by providing data of new environment. A simple and widely used approach for re-training is to re-train a model periodically with newly acquired data. In [11], the authors use adaptive linear time series models. The adjustment of a forecasting model is achieved by fitting the forecasting model with k-step recursive least squares with forgetting. The coefficients of the forecasting model are updated regularly, and newer values are weighted higher than old values, hence the models adapt over time to changing conditions. In [12], the authors propose a nonlinear autoregressive, with external input (NARX), network-based forecasting model. The forecasting model utilises clear-sky radiation, calculated from Hottel's radiation model and weather forecast data from public websites. They adjust the weights of the NARX network according to the forecast error. In [13], the authors utilise the knowledge-based library to provide the best training datasets for coming target days. By re-training a model using the best training datasets, they attempt to increase forecast accuracy. In [14], the authors propose an adaptive back-propagation neural network model that is re-trained periodically using the updated training data by scrolling time window (i.e. recent data). They also study how to find an optimised time window to guarantee forecast accuracy. Another method adopted for re-training a model is to re-train only when a forecasting error exceeds a pre-defined threshold [15–17]. In [15], a forecasting model is updated using a back-propagation neural network. In [17], the authors realise error-driven re-training through reinforcement learning.

Re-training approach has the following issues. First issue is that data relating to the new environment is likely to be minimal when concept drift just happens. Please note that any statistical model prefers to have sufficient data. This means that they need a longer period of time to adapt to a new environment. The second issue is that the re-training approach is likely to fail when the data of previous and new environment are mixed (e.g. similar inputs, but different outputs). Please note that a model learns a relationship between the inputs and the desired outputs. Thus, in this case, the model has a low probability of producing the desired outputs after concept drift. In contrast, we adjust a forecasting output without re-training a forecasting model.

2.3.2 | Ensemble-based

An ensemble-based approach is one available option for adapting to dynamic environments. One approach of the ensemble-based approach is to dynamically adjust the combination weight of individual models. In [18], the authors propose a meta-learning layer that arbitrates individual competing forecasting models used for global horizontal irradiance forecasting. Using the meta-learning layers, they aim to dynamically combine ensemble models, according to their aptitude in the input data. In [19, 20], the authors propose a dynamic ensemble of

ALGORITHM 1 Re-training a model

Input: A trained h_θ; N
while *TRUE* **do**
  Predict $\hat{y}_t = h_\theta(F_{t-N:t-1}, y_{t-N:t-1})$. Use $\hat{y}_t$ as a forecasting output at time step t;
  Receive y_t;
  Incur loss $l_t = L(y_t, \hat{y}_t)$;
  Update θ using l_t;
  t ← t+1;
end

neural networks. Through the dynamic ensembles, they keep track of the recent forecast accuracy of each model and adaptively weight the contribution of the ensemble models. Another approach of the ensemble method is to pick the best single model. In [21], the authors propose a way to intelligently switch between different types of base models, in an ensemble, to increase the predictive performance of online learning. In [22], the authors propose MetaStream, which conducts periodic algorithm selection in time-changing environments. MetaStream maps the features extracted, from the past and the current data, to the performance of forecasting models so as to choose between single learning algorithms or their combination. In [23], using meta-learning, the authors attempted to find evidence for features to be used for guiding the choice of whether to pick a single model or a combination of models while still focusing on which model to select. They exploit decision trees to find evidence for the existence of a link between time series features and the forecast accuracy of forecasting models

The ensemble method-based approach may cause computational and memory overhead when using multiple models. Besides, the performance of an ensemble model is limited by individual models' performance. On the other hand, we use a single forecasting model together with a simple online model. We deal with concept drift by adjusting forecasting outputs through an online model.

3 | MODEL-AGNOSTIC ONLINE FORECASTING

In achieving objective of this paper, we adjust the outputs of a forecasting model rather than updating the model itself. Therefore, our approach is applicable to any forecasting model (i.e. model-agnostic). Figure 2 compares a typical re-training-based approach and our proposed model-agnostic approach.

Our approach consists of two steps. We first transform the problem of forecasting, with multivariate time series in batch learning, into a forecasting with univariate time series in online learning. Then, we utilise an online autoregressive integrated moving average (ARIMA) [24] to handle the univariate time-series prediction. The detailed descriptions of each step will be given in the following subsections.

3.1 | Domain transformation

We realise the domain transformation (i.e. from batch learning to online learning) through simple data transformation. We assume a forecasting model that uses multivariate time series (i.e. weather conditions and PV power output history) as inputs and produces an expected PV power output of next time step. In adjusting the forecasting output to handle the issue of concept drift, we utilise online ARIMA (described in Section 3.2 in detail). Online ARIMA conducts prediction, using univariate time series, and is good at adapting to dynamic environments. To utilise online ARIMA, we conduct the following transformation.

$$z_{t-N:t-1} = y_{t-N:t-1} - \hat{y}_{t-N:t-1}. \quad (1)$$

Once we predict $\hat{z}_t$ correctly (i.e. close to z_t) using online ARIMA with $z_{t-N:t-1}$, we can adjust $\hat{y}_t$ (i.e. output of a forecasting model) through the following data de-transformation,

$$\hat{y}'_t = \hat{z}_t + \hat{y}_t. \quad (2)$$

$\hat{y}'_t$ is used as a final forecasting output.

In contrast, most existing approaches try to update θ of h_θ using l_t or $l_{t-(M-1):t}$. Figure 2(b) and Algorithm 1 describe this.

3.2 | Online ARIMA

In realising MAOF, we utilise online ARIMA. We first explain why we choose online ARIMA. Our idea for dealing with concept drift is to use an additional model for adjusting an output of a target model. The additional model should be simple and lightweight not to cause noticeable overhead. Moreover, the additional model should also be good at handling concept drift. In searching for a model that satisfies the two requirements above, we found that online ARIMA is suitable for our purpose. Online ARIMA is very simple to implement. Less than 10 lines are required to implement online ARIMA in Python language. Attractive features and performance of online ARIMA are

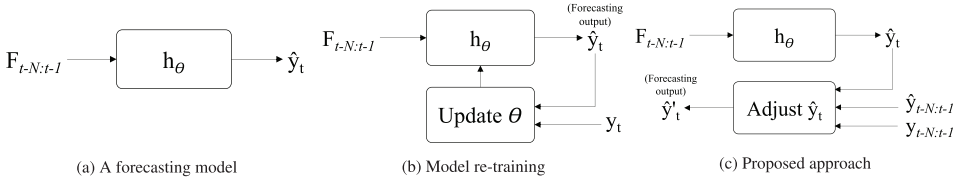


FIGURE 2 High-level illustration of methods

ALGORITHM 2 Model-agnostic Online Forecasting: Basic

Input: parameter k+m, d; learning rate η
Initialize γ;
while *TRUE* **do**
  # *Data transformation*
  Calculate $z_{t-(k+m):t-1} = y_{t-(k+m):t-1} - \hat{y}_{t-(k+m):t-1}$;
  # *ARIMA-based prediction*
  Predict $\hat{z}_t = \sum_{i=1}^{k+m} \gamma_i \nabla^d z_{t-i} + \sum_{i=0}^{d-1} \nabla^i z_{t-1}$.
  # *Data de-transformation*
  Receive $\hat{y}_t$ (from a forecasting model);
  Calculate $\hat{y}_t^{'} = \hat{z}_t + \hat{y}_t$;
  Use $\hat{y}_t^{'}$ as a forecasting output at time step t;
  # *ARIMA-OGD optimization*
  Receive y_t;
  Calculate $z_t = y_t - \hat{y}_t$;
  Incur loss $l_t = L(z_t, \hat{z}_t)$;
  Let $\nabla_t = \nabla l_t$;
  Update θ using l_t;
  Set $\gamma^{t+1} \leftarrow \prod_K(\gamma^t - \frac{1}{\eta}\nabla_t)$;
  t ← t+1;
end

ALGORITHM 3 Model-agnostic online forecasting: Enhanced

Input: parameter k+m, d; learning rate η
Initialize γ;
R = [];
while *TRUE* **do**
  # *Data transformation*
  Calculate ϕ = **Mean(R)**;
  Calculate $z_{t-(k+m):t-1} = y_{t-(k+m):t-1} - \hat{y}_{t-(k+m):t-1}\phi$;
  # *ARIMA-based prediction*
  Predict $\hat{z}_t = \sum_{i=1}^{k+m} \gamma_i \nabla^d z_{t-i} + \sum_{i=0}^{d-1} \nabla^i z_{t-1}$.
  # *Data de-transformation*
  Receive $\hat{y}_t$ (from a forecasting model);
  Calculate $\hat{y}_t^{'} = \hat{z}_t + \hat{y}_t\phi$;
  Use $\hat{y}_t^{'}$ as a forecasting output at time step t;
  # *ARIMA-OGD optimization*
  Receive y_t;
  R.insert($\frac{y_t}{\hat{y}_t}$**)**;
  Calculate $z_t = y_t - \hat{y}_t$;
  Incur loss $l_t = L(z_t, \hat{z}_t)$;
  Let $\nabla_t = \nabla l_t$;
  Update θ using l_t;
  Set $\gamma^{t+1} \leftarrow \prod_K(\gamma^t - \frac{1}{\eta}\nabla_t)$;
  t ← t+1;
end

studied in mathematical and experimental ways [24]. Regarding the other simple and lightweight classic time series prediction models, we could not find online versions of them. That is why we use online ARIMA for our purpose.

One of the simplest types of time series model is ARMA (autoregressive moving average). Let y_t and ϵ_t denote the observation and the zero-mean random noise term at time t, respectively. Then, ARMA(k, q) that combines AR(k) (autoregressive) and MA(q) (moving average) can be expressed as follows:

$$y_t = \sum_{i=1}^{q} \beta_i \epsilon_{t-i} + \sum_{i=1}^{k} \alpha_i y_{t-i} + \epsilon_t, \tag{3}$$

where α and β are coefficients for AR and MA, respectively.

Most time series are not stationary. However, ARMA is limited to stationary processes. ARIMA handles non-stationary time series using a differential method. For example, the first order differences of y_t is $\nabla y_t = y_t - y_{t-1}$, and the *dth* order differences of y_t is $\nabla^d y_t = \nabla^{d-1} y_t - \nabla^{d-1} y_{t-1}$. A time series y satisfies ARIMA(k, d, q) if $\nabla^d y$ satisfies ARMA(k, q). ARIMA(k, d, q) can be expressed as follows:

$$\nabla^d y_t = \sum_{i=1}^{q} \beta_i \epsilon_{t-i} + \sum_{i=1}^{k} \alpha_i \nabla^d y_{t-i} + \epsilon_t. \tag{4}$$

ARIMA(k, d, q) conducts prediction through a reversion of differential process. The prediction value from Equation (4) is

$$\hat{y}_t = \nabla^d \hat{y}_t + \sum_{i=0}^{d-1} \nabla^i y_{t-i}. \tag{5}$$

Given the basic description of ARIMA(k, d, q) above, we focus on online learning setting. At time t, the ARIMA(k, d, q) predicts $\hat{y}_t$, and then the true y_t is disclosed. As a result, a loss $l_t = L(y_t, \hat{y}_t)$ is given. Using Equations (4) and (5), the loss $l_t = L(y_t, \hat{y}_t)$ can be expressed as follows:

$$\begin{aligned} l_t &= L(y_t, \hat{y}_t) \\ &= L(y_t, \nabla^d \hat{y}_t + \sum_{i=0}^{d-1} \nabla^i y_{t-i}) \\ &= L(y_t, \sum_{i=1}^{q} \beta_i \epsilon_{t-i} + \sum_{i=1}^{k} \alpha_i \nabla^d y_{t-i} + \sum_{i=0}^{d-1} \nabla^i y_{t-i}). \end{aligned} \tag{6}$$

The goal of online ARIMA is to minimise Equation (6). One solution for this is an improper learning principle [25]. The idea is to approximate the original ARIMA(k, d, q) model with another ARIMA($k + m$, d, 0) model. $m \in N$ is a properly chosen constant, such that the new ARIMA model with $\gamma \in R^{m+k}$ is enough to approximate the original prediction as follows:

$$\hat{y}_t = \sum_{i=1}^{k+m} \gamma_i \nabla^d y_{t-i} + \sum_{i=0}^{d-1} \nabla^i y_{t-i}. \tag{7}$$

Then, the loss function described in Equation (6) can be rewritten as follows:

$$l_t = L(y_t, \hat{y}_t) = L(y_t, \sum_{i=1}^{k+m} \gamma_i \nabla^d y_{t-i} + \sum_{i=0}^{d-1} \nabla^i y_{t-i}). \tag{8}$$

3.3 | Online forecasting

Now, we are ready to explain our method of utilising online ARIMA for our purposes. For clear presentation of our approach, we first introduce the basic version of MAOF (Algorithm 2). We assume that k, d, and m for the ARIMA($k+m, d, 0$) model and learning rate η are given as input. The first step is to conduct data transformation as described in Section 3.1. Then, we conduct prediction with the modified ARIMA algorithm. Given the prediction of online ARIMA ($\hat{z}_t$) and prediction of a forecasting model ($\hat{y}_t$), we calculate an adjusted forecasting output ($\hat{y}'_t$) through data transformation. $\hat{y}'_t$ is used as the final forecasting output at time step t. Finally, we update ARIMA model using an OGD (online gradient descent) algorithm, which is an online convex optimisation solver. The above procedures are repeated during test phase. In Algorithm 2, K is a set of candidate $(m+k)$-dimensional coefficient vectors, that is, $K = (\gamma \in R^{m+k}, |\gamma_j| \leq 1, j = 1, ..., m)$. $\prod_K(\mathrm{f})$ indicates the Euclidean projection onto K, that is, $\prod_K(\mathrm{f}) = \mathrm{argmin}_{x \in K}||f - x||_2$.

To enhance the effectiveness of adjusting an output of a forecasting model, we first conduct a simple adjustment to an output of a forecasting model before online ARIMA is applied. Please note that our goal is to produce $\hat{y}'_t$ close to y_t. Therefore, using ϕ (i.e. the mean of $\frac{y_{t-N:t-1}}{\hat{y}_{t-N:t-1}}$), we adjust $\hat{y}_t$ as $\hat{y}_t\phi$. Algorithm 3 shows an enhanced version of our method. Compared to Algorithm 2, lines in bold indicate newly added operations.

4 | EXPERIMENTS

4.1 | Settings

To examine the feasibility of MAOF, we conduct a series of experiments using the following settings.

4.1.1 | Model

To show a general applicability of MAOF, we use LightGBM (as one of the popular machine learning models) and LSTM (as one of the popular deep learning models) as base forecasting models. We use Python library (lightgbm 2.3.3) for LightGBM and Tensorflow (2.0.0-beta1) for LSTM. We apply the basic version of MAOF (MAOF_Basic for short), the enhanced version of MAOF (MAOF_Enhanced for short), and existing methods to these two models.

4.1.2 | Data

As inputs to a forecasting model, we use historical weather data and PV power output data. We use NREL weather data (acquired from the national solar radiation database (NSRDB) data viewer [28]. The weather data is 30-min interval data. The weather data includes global horizontal irradiance (GHI), dew point, wind speed, relative humidity, and temperature. We use PV power output data (acquired from NREL Solar Power Data for Integration Studies [29]) of Texas, USA. More specifically, we use NREL data of the PV plant whose latitude is 33.45 and longitude is −94.35. Its capacity is 27 MW. The PV power output data is 5-min interval data. Thus, we transform 5-min interval data into 30-min interval data by cumulating corresponding values. The data from NREL are 1-year data (i.e. 2006). Number of data points is 17,516. We used 12,332 data points for training the model and 5,184 data points for testing. Figure 3 shows the data.

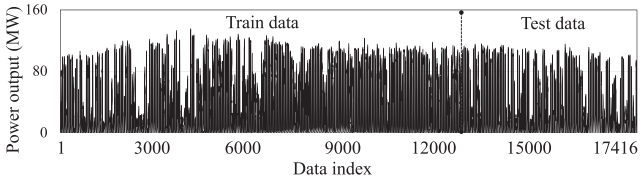


FIGURE 3 Train and test data (power output only)

4.1.3 | Test scenarios

Ideally, it is best to conduct experiments using datasets with real-world drifts. But, we could not find those datasets. Thus, our best option is to build test scenarios that mimic various PV faults. For this, we examine papers about performance degradation by PV faults while focusing on PV faults that commonly happen. As a result, we build 7 test scenarios, described in Table 2. The test scenarios may not represent all possible concept drifts that are caused by various PV faults. But, we believe that the test scenarios are enough for examining the feasibility of our method. Figure 4 shows some examples of test scenarios.

4.1.4 | Performance metrics

As performance metrics, we use the root mean square error (RMSE) and the mean absolute percentage error (MAPE), which are calculated as follows:

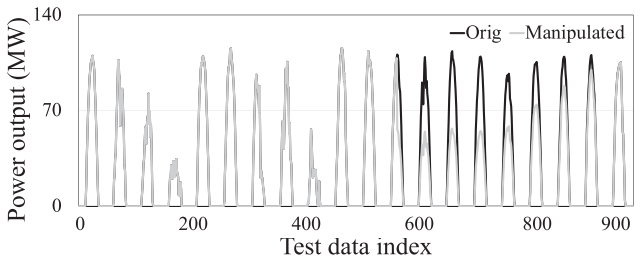
$$\mathrm{RMSE}(y, \hat{y}) = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}. \tag{9}$$

$$\mathrm{MAPE}(y, \hat{y}) = \frac{100}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{y_i}. \tag{10}$$

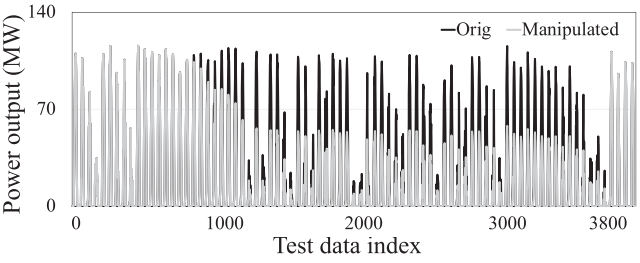
In Equations (9) and (10), n is the number of test data, y is the true value, and $\hat{y}$ is the predicted value. The following results are the average across 10 runs.

TABLE 2 Test

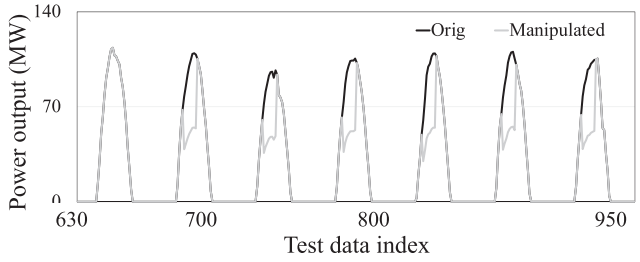
Concept drift type		Relevant PV faults [47, 48]	Configurations for test scenarios	Test code
Sudden drift	No recovery Manual recovery (perfect recovery)	**Environmental** Bird dropping [30, 31] **Electrical** Ground fault [32, 33] Line-line fault [32–34] Open-circuit fault [32, 33] **Physical** Degradation fault [32, 35–37]	Start of drop: random btw 500 and 1,000 (of 5,184 test data) Degree of drop: 10, 20, 30, 40, 50% **For manual recovery** Start of recovery: random btw 3,500 and 4,000	T1_SN T2_SM
	Incremental recovery (by sun)	**Environmental** Snow covering [38–40]	Start of drop: random btw 500 and 1,000 Degree of drop: 50, 60, 70, 80, 90% Start of recovery: 3 days after the drop Duration of recovery: 5 days	T3_SI
Incremental drift	No recovery Manual recovery (perfect recovery)	**Environmental** Dust accumulation [41–43] **Physical** Degradation fault [32, 35–37]	Start of drop: random btw 500 and 1,000 Max degree of drop: 10, 20, 30, 40, 50% Duration of incremental drop: 10 days For manual recovery Start of recovery: random btw 3,500 and 4,000	T4_IN T5_IM
Recurring drift	Daily repetition	**Environmental** Partial shading by objects (tree, cloud etc.) [44–46]	Start of drop: random btw 500 and 1,000 (of 5,184 test data) Duration of drop in a day: 09:00 - 12:00 (3H) Degree of drop: 10, 20, 30, 40, 50%	T6_RD
	Temporal drop		Days of drop events: random btw 500 and 1,000, 1,500 and 2,000, and 3,000 and 3,500 Duration of drop in a day drop: 3H (random btw 10:00 and 17:00) Degree of drop: 10, 20, 30, 40, 50%	T7_RT



(a) Sudden drift scenario (T3_SI)



(b) Incremental drift scenario (T5_IM)



(c) Recurring drift scenario (T6_RD)

FIGURE 4 Examples of test scenarios

4.2 | Performance study of MAOF

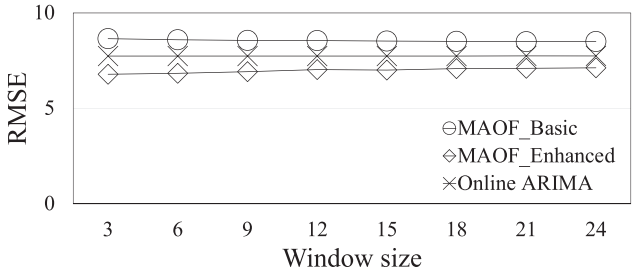
We first examine several aspects of MAOF in detail before we compare MAOF with existing methods.

4.2.1 | Hyper-parameters

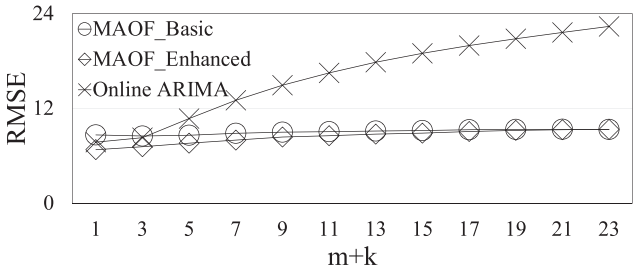
MAOF has two main hyper-parameters including the window size for a forecasting model (i.e. the number of recent data used for conducting current prediction) and $m + k$ for online ARIMA. To study the effects of the hyper-parameters, we use LightGBM as a base forecasting model. T1_SN scenario with 0.3 degradation ratio is used. A learning rate of online ARIMA is set to 1e-5. Figure 5(a) shows the effects of window size on forecasting accuracy. The effect of window size is marginal. Therefore, we use window size 3 because a small window size causes low operational overheads. Figure 5(b) shows the effects of $m + k$ on forecasting accuracy. Forecasting accuracy decreases as $m + k$ increases, particularly for online ARIMA. Interestingly, using just one recent data is enough for achieving the best performance in our experimental settings. In summary, in our current experimental settings, applying online ARIMA, with one recent data, to a forecasting model, with three recent data, leads to the best performance. Therefore, this setting is used for the following experiments.

4.2.2 | Improvements by MAOF

We first examine how much MAOF improves forecasting accuracy compared to **Naive** (i.e. no method for adjusting an



(a) Effects of window size ($m + k$ = 1 and LightGBM as a base model)



(b) Effects of $m + k$ (window size = 3 and LightGBM as a base model)

FIGURE 5 Effects of hyper-parameters

output nor a trained model). For this comparison, we use **MAOF_Basic**. Table 3 compares RMSE and MAPE. The first value and the second value of each table cell indicate RMSE and MAPE, respectively. When there is no concept drift, **Naive** and **MAOF_Basic** show similar performance. Conversely, **MAOF_Basic** shows noticeable improvement compared to **Naive** when concept drift happens. The degree of improvement increases as the degradation ratio increases. For example, in the case of LightGBM with the T1_SN scenario, the improvement percentage in RMSE increases from 15.2% to 72.1% as the degradation ratio increases from 0.1 to 0.5. This is because **MAOF_Basic** sustains the effectiveness of a base model while **Naive** does not. T6_RD and T7_RT scenarios are challenging cases to **MAOF_Basic**. Unlike the other cases, where the maximum improvement percentage in RMSE is over 70%, the maximum improvement percentages in RMSE in those cases are 59.7% and 30.9%, respectively. This result shows that adaptation to degradation during a short period (i.e. a drop during 3H including 6 data points) is not easy. However, **MAOF_Basic** still noticeably outperforms **Naive**. **MAOF_Basic** improves performance in both RMSE and MAPE compared to **Naive** regardless of a base model. This result shows that our approach is model-agnostic.

We also examine the improvements by **MAOF_Enhanced** compared to **MAOF_Basic**. When there is no concept drift, **MAOF_Enhanced** degrades performance slightly compared to **MAOF_Basic**, whereas **MAOF_Enhanced** improves performance compared to **MAOF_Basic** when concept drift happens. The degree of improvement increases as the degradation ratio increases. For example, in the case of LightGBM with the T1_SN scenario, the improvement percentage in RMSE increases from 14.2% to 52.2% as the degradation ratio increases from 0.1 to 0.5. In T6_RD and T7_RT scenarios, the maximum improvement percentages are 11.1% and 23.6%, while the maximum improvement percentage in the other cases are over 50% due to the same reason mentioned above. One interesting observation is that **MAOF_Enhanced** decreases RMSE and sustains MAPE as the degradation ratio increases in T1_SN, T2_SM, T4_IN, and T5_IM scenarios. This is because the target value is decreased in generating test concept drift scenarios and that **MAOF_Enhanced** follows the decreased target value quite well. The above results show that simple pre-adjustment (explained in Algorithm 3) leads to noticeable improvement.

4.3 | Comparison

We compare **MAOF_Enhanced** with the existing method. As a representative method of the existing approach, we re-train a model periodically. For the re-training-based method, we apply three different window sizes: 480 (i.e. recent 10 days), 1,440 (i.e. recent 30 days), and all previous data including the train data. After the model is trained using the train data, the trained model is re-trained whenever 48 new data points are acquired (i.e. every day). To re-train a LightGBM model, we re-train a model from scratch using the given data because the current Python library does not support the updating of a model. To re-train an LSTM model, we update a trained model using the given data. Throughout the following figures, **MAOF**, **RS**, **RM**, and **RL** indicate **MAOF_Enhanced**, re-training with recent 480 data, re-training with recent 1,440 data, and re-training with all previous data, respectively. Among the seven test scenarios, we show representative results (Figures 6 and 9).

4.3.1 | No concept drift

In both LightGBM and LSTM cases, **MAOF**, **RM**, and **RL** show similar performance (Figures 7 and 8), whereas **RL** shows the best performance. This means that using long history for re-training a model under no concept drift is desirable.

4.3.2 | T1_SN, T2_SM, T4_IN, and T5_IM

These four scenarios show similar results. Thus, we just discuss the results of T1_SN. **MAOF** sustains the effectiveness of a target model regardless of the degradation ratio in both LightGBM and LSTM. With LightGBM, **MAOF** decreases RMSE (MAPE) from 6.91 (14.68) to 4.5 (13.97) as the degradation ratio increases from 0.1 to 0.5 (Figure 6(a,b)). With LSTM, **MAOF** decreases RMSE (MAPE) from 10.61 (22.86) to 8.02 (23.59) as the degradation ratio increases from 0.1 to 0.5 (Figure 9(a,b)). **MAOF** shows the best performance in both LightGBM and LSTM. The re-training method shows different performance for different window sizes in both LightGBM and LSTM. **RL** shows the worst performance in both LightGBM and LSTM cases. This result shows that using long history for re-training a model, under these scenarios,

TABLE 3 Improvements by MAOF (in RMSE / MAPE)

	LightGBM			LSTM		
Degradation ratio	**Naive**	**MAOF_Basic**	**MAOF_Enhanced**	**Naive**	**MAOF_Basic**	**MAOF_Enhanced**
	No concept drift					
0	7.1 / 14.8	7.16 / 14.46	7.53 / 14.46	12.3 / 22.6	12.9 / 22.9	13.3 / 23.41
	Sudden degradation without recovery (T1_SN)					
0.1	9.51 / 20.87	8.06 / 17.48	6.91 / 14.68	12.24 / 26.38	12.01 / 24.43	11.66 / 22.82
0.2	14.71 / 32.48	8.63 / 20.35	6.38 / 14.88	15.61 / 36.17	12.55 / 27.03	10.61 / 22.86
0.3	20.68 / 48.97	8.9 / 22.83	5.84 / 14.96	20.68 / 49.67	13.34 / 30.81	10.02 / 23.06
0.4	27.29 / 71.31	9.05 / 25.31	5.13 / 14.13	26.51 / 70.19	13.98 / 35.78	9.15 / 24.89
0.5	33.86 / 104.24	9.41 / 30.69	4.5 / 13.97	32.52 / 101.81	14.62 / 44.56	8.02 / 27.17
	Sudden degradation with manual recovery (T2_SM)					
0.1	9.53 / 19.93	7.9 / 16.22	6.65 / 13.18	12.59 / 25.05	12.3 / 23.07	11.85 / 21.5
0.2	15.04 / 31.88	8.49 / 18.59	6.03 / 13.18	15.78 / 34.23	12.44 / 25.59	10.52 / 21.43
0.3	21.32 / 48.99	8.92 / 22.14	5.54 / 13.61	21.19 / 49.42	13.69 / 30.39	10.12 / 22.41
0.4	27.88 / 72.05	9.12 / 25.01	4.9 / 13.16	27.16 / 70.31	14.58 / 36.24	9.66 / 23.28
0.5	34.63 / 105.57	9.66 / 31.23	4.41 / 13.08	33.86 / 104.36	14.84 / 44.29	8.06 / 22.68
	Sudden degradation with incremental recovery (T3_SI)					
0.5	30.38 / 61.34	11.92 / 22.34	5.84 / 8.47	29.11 / 60.39	13.92 / 27.13	9.46 / 16.76
0.6	36.44 / 88.74	12.48 / 29.06	5.83 / 11.05	36.71 / 90.47	16.29 / 36.94	10.09 / 19.17
0.7	43.40 / 134.44	13.07 / 36.71	6.88 / 13.87	41.44 / 127.28	16.45 / 45.44	9.39 / 20.35
0.8	49.94 / 213.13	14.19 / 53.69	9.10 / 23.84	48.1 / 203.13	18.07 / 66.27	10.06 / 25.39
0.9	57.16 / 398.16	14.99 / 83.04	14.74 / 63.07	55.99 / 391.56	19.44 / 105.44	14.37 / 54.6
	Incremental degradation without recovery (T4_IN)					
0.1	9.2 / 20.09	7.93 / 17.19	6.89 / 14.57	11.8 / 25.68	11.57 / 23.97	11.18 / 22.33
0.2	14.06 / 31.31	8.65 / 20.43	6.48 / 15.25	15.02 / 34.75	12.4 / 27.3	10.78 / 23.4
0.3	19.7 / 45.73	8.67 / 22	5.71 / 14.41	19.98 / 47.86	13.25 / 30.35	10.18 / 23.27
0.4	25.64 / 65.95	8.75 / 24.11	5 / 13.78	25.07 / 65.9	13.94 / 35.35	9.59 / 23.53
0.5	31.86 / 95.67	9.24 / 29.41	4.27 / 13.53	31.38 / 96.33	14.2 / 42.01	8.13 / 22.82
	Incremental degradation with manual recovery (T5_IM)					
0.1	9.14 / 19.04	7.74 / 15.86	6.66 / 13.16	12.04 / 24.29	11.89 / 22.73	11.56 / 21.41
0.2	14.04 / 30.28	8.55 / 19.09	6.19 / 13.75	15.43 / 32.93	12.78 / 25.48	11.08 / 21.64
0.3	19.55 / 44.86	8.83 / 21.79	5.6 / 13.67	20.18 / 45.93	13.53 / 29.26	11.08 / 21.9
0.4	25.86 / 64.81	8.81 / 23.54	4.75 / 12.52	24.71 / 64.55	14.16 / 35.18	9.7 / 23.1
0.5	31.87 / 93.74	9.22 / 28.58	4.05 / 12.43	30.82 / 92.79	13.92 / 39.83	7.9 / 21.1
	Recurring degradation with daily repetition (T6_RD)					
0.1	10.51 / 20.94	9.19 / 18.2	8.86 / 17.32	13.26 / 26.34	13.08 / 24.83	13.3 / 24.23
0.2	16.69 / 32.85	10.93 / 22.87	9.92 / 20.82	17.49 / 36.53	15.44 / 31.47	14.62 / 29.02
0.3	23.39 / 48.66	12.2 / 27.06	10.94 / 24.17	23.17 / 50.19	16.51 / 34.89	14.78 / 30.7
0.4	30.79 / 71.82	13.72 / 32.66	12.24 / 28.61	29.91 / 71.24	19.01 / 44.36	16.37 / 37.49
0.5	38.27 / 105.04	15.4 / 41.97	13.68 / 35.98	36.78 / 105.5	20.54 / 56.88	17.15 / 45.83
	Recurring degradation with temporal drop (T7_RT)					
0.1	9.72 / 14.97	8.23 / 13.05	7.59 / 12.11	13.95 / 17.58	13.56 / 16.47	13.78 / 16.31
0.2	15.77 / 36.96	12.55 / 29.73	9.46 / 22.64	17.96 / 37.96	15.81 / 32.29	14.35 / 27.26
0.3	25.16 / 45.22	18.87 / 34.86	13.8 / 25.51	22.65 / 47.62	17.1 / 34.58	14.54 / 28.88
0.4	33.22 / 69.79	23.4 / 49.67	17.74 / 36.79	31.53 / 71.05	22.79 / 51.14	18.44 / 42.08
0.5	38.06 / 103.03	26.27 / 71.25	20.06 / 54.23	38.52 / 105.55	25.02 / 67.59	19.56 / 51.47

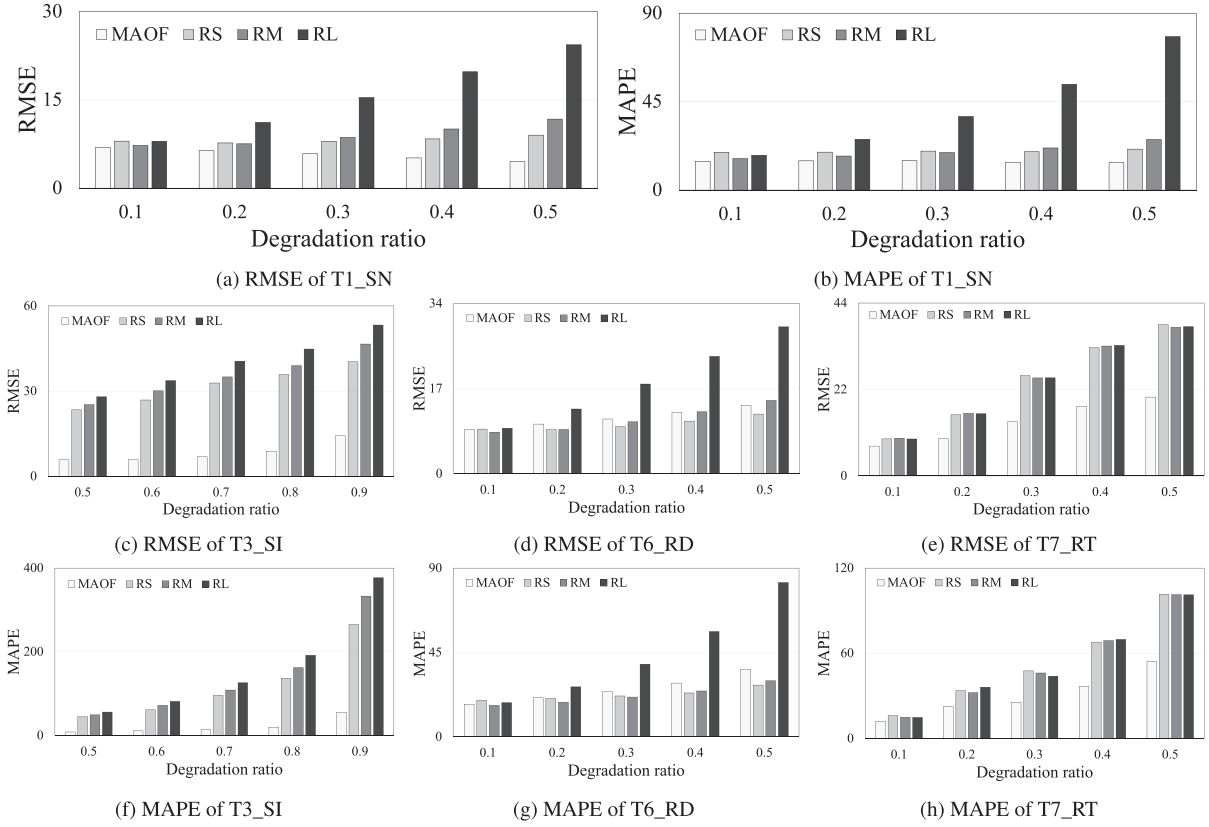


FIGURE 6 Comparison of results (LightGBM as a base model)

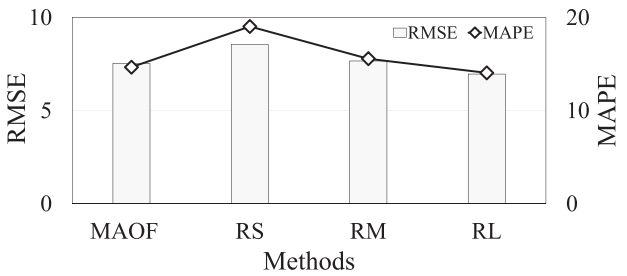


FIGURE 7 Performance comparison under no concept drift (LightGBM)

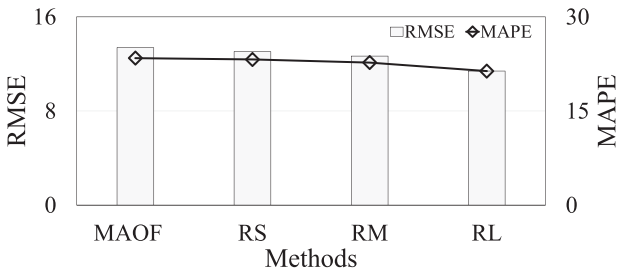


FIGURE 8 Performance comparison under no concept drift (LSTM)

leads to poor performance. Among the three re-training methods, **RS** shows the best performance in both LightGBM and LSTM cases.

4.3.3 | T3_SI

This scenario is challenging to both **MAOF** and the re-training methods. But, **MAOF** still noticeably outperforms the re-training methods (i.e. the maximum improvement in percentage is 79%). With LightGBM, **MAOF** increases RMSE (MAPE) from 5.84 (8.47) to 14.29 (55.84) as the degradation ratio increases from 0.5 to 0.9 (Figure 6(c,f)). With LSTM, **MAOF** increases RMSE (MAPE) from 9.46 (16.76) to 14.37 (54.6) as the degradation ratio increases from 0.1 to 0.5 (Figure 9(c,f)). The re-training methods show worse performance. With LightGBM, **RS** (the best among the three re-training methods) increases RMSE (MAPE) from 23.44 (44.88) to 40.33 (264.89) as the degradation ratio increases from 0.1 to 0.5. This result shows that the re-training methods are not helpful in this scenario.

4.3.4 | T6_RD

In this scenario, the re-training methods show good performance in both LightGBM (Figure 6(d,g)) and LSTM (Figure 9(d,g)). **RS** shows the best performance. This is because the degradation is repeated in the same pattern and thus re-training is useful to handle a concept drift. **RL** shows the worst performance because the data used for re-training includes data before concept drift. In contrast, **MAOF** does not show the best performance in this case. The degradation happens during a short

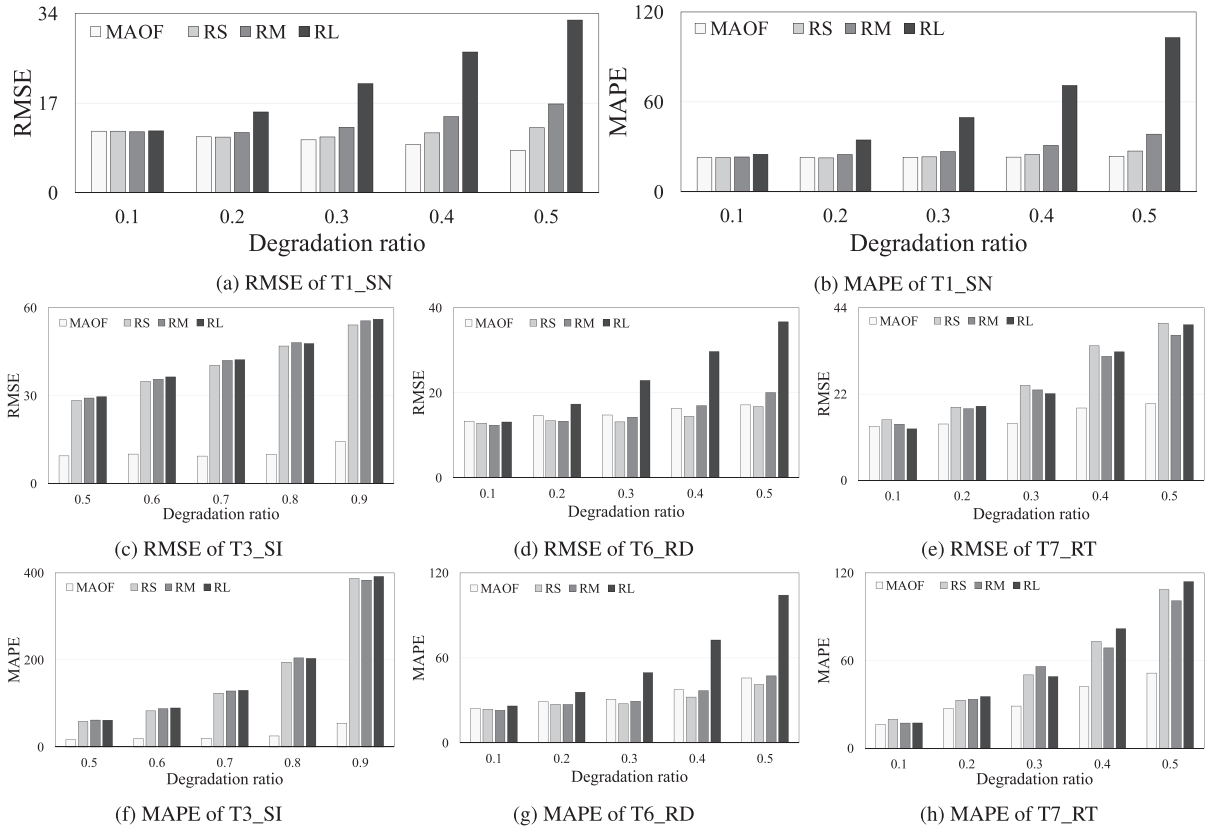


FIGURE 9 Comparison of results (LSTM as a base model)

period (i.e. 3H, including 6 data points). It is not easy for **MAOF** to adapt to the degradation during a short period without using degradation patterns previously observed, which are used by the re-training methods. However, even with this difficulty, **MAOF** shows competitive results.

4.3.5 | T7_RT

Unlike T6_RD, where the degradation is repeated in the same pattern, this scenario is challenging to the re-training methods in both LightGBM (Figure 6(e,h)) and LSTM (Figure 9(e,h)). This is because no useful history is available for re-training a model. As a result, **RS**, **RM**, and **RL** show similar and bad performance. This scenario is also challenging to **MAOF** due to the same reason mentioned before (i.e. degradation during a short period). As a result, **MAOF** increases RMSE and MAPE as the degradation ratio increases. However, **MAOF** still shows better performance than the re-training methods (i.e. the maximum improvement percentage in RMSE is 47.8%).

4.3.6 | Summary

MAOF shows reliable and desirable performance under various concept drift scenarios regardless of a base model, whereas the re-training method shows unreliable performance. In utilising re-training methods, we need to solve several issues such as determining type and degree of concept drift, and selection of window size. Even with those solutions, some cases like T3_SI and T7_RT are still very challenging to them.

5 | CONCLUSION

Even though solar energy is considered as one of the most promising alternatives to fossil fuels, building a reliable forecasting model for PV power output is still a challenge faced by those in this area of study. In this paper, we explore a successful method of handling concept drift in the field of PV power output forecasting. To realise a reliable forecasting model under concept drift scenarios, we propose a model-agnostic online forecasting that utilises an online learning algorithm with effective domain transformation. Experiments using the real-world data show that our method achieves reliable and desirable performance under various concept drift scenarios. As future work, we plan to study an ensemble method, using multiple data transformations, in applying an online learning algorithm. It would also be interesting to study a method of selecting a proper online optimisation solver adaptively, under dynamic environments.

ACKNOWLEDGEMENT

This work was supported by the Korea Institute of Energy Technology Evaluation and Planning (KETEP) and the Ministry of Trade, Industry & Energy (MOTIE) of the Republic of Korea (grant no. 20181210301570).

ORCID

HyunYong Lee https://orcid.org/0000-0002-0615-4241

REFERENCES

1. Sobri, S., K-Kamali, S., Rahim, N.A.: Solar photovoltaic generation forecasting methods: A review. Energy Convers. Manage. 156, 459–497 (2018)
2. Larson, D.P., Nonnenmacher, L., Coimbra, C.F.M.: Day-ahead forecasting of solar power output from photovoltaic plants in the American southwest. Renewable Energy 91, 11–20 (2016)
3. Nonnemacher, L., Kaur, A., Coimbra, C.F.M.: Day-ahead resource forecasting for concentrated solar power integration. Renewable Energy 86, 866–876 (2016)
4. Luoma, J., Mathiesen, P., Kleissl, J.: Forecast value considering energy pricing in California. Elsevier Appl. Energy 125, 230–237 (2014)
5. Litjens, G.B.M.A., Worrell, E., Van Sark, W.G.J.H.M.: Assessment of forecasting methods on performance of photovoltaic-battery systems. Appl. Energy 221, 358–373 (2018)
6. Pillai, D.S., Rajasekar, N.: A comprehensive review on protection challenges and fault diagnosis in pv systems. Renewable Sustainable Energy Rev. 91, 18–40 (2018)
7. Dhimish, M., Mather, P., Homes, V.: Evaluating power loss and performance ratio of hot-spotted photovoltaic modules. IEEE Trans. Electron Devices 65(12), 5419–5427 (2018)
8. Tur, M.R., Colak, I., Bayindir, R.: Effect of faults in solar panels on production rate and efficiency. In: Proc. IEEE Int. Conf. Smart Grid, pp. 287–293. IEEE, Piscataway, NJ (2018)
9. Maghami, M.R., et al.: Power loss due to soiling on solar panel: A review. Elsevier Renewable Sustainable Energy Rev. 59, 1307–1316 (2016)
10. Dhoke, A., Sharma, R., Saha, T.K.: An approach for fault detection and location in solar pv systems. Solar Energy 194, 197–208, (2019)
11. Bacher, P., Madsen, H., Nielsen, H. A.: Online short-term solar power forecasting. Solar Energy 83(10), 1772–1783 (2009)
12. Chupong, C., Plangklang, B.: Forecasting power output of PV grid connected system in Thailand without using solar radiation measurement. Energy Procedia 9, 230–237 (2011)
13. Manjili, Y.S., Vega, R., Jamshidi, M.M.: Data-analytic-based adaptive solar energy forecasting framework. IEEE Systems J. 12(1), 285–296 (2018)
14. Zhu, H., et al.: An improved forecasting method for photovoltaic power based on adaptive BP neural network with a scrolling time window. Energies 10, 1542–1559 (2017)
15. Wang, Y., et al.: Adaptive learning hybrid model for solar intensity forecasting. Trans. Ind. Inform. 14(4), 1635–1645 (2018)
16. Wang, Y., et al.: Adaptive solar power forecasting based on machine learning methods. Appl. Sci. 8, 2224–2241 (2017)
17. Wee, C.K., Nayak, R.: Adaptive load forecasting using reinforcement learning with database technology. J. Inf. Telecommun. 3(3), 381–399 (2019)
18. Cerqueria, V., Torgo, L., Soares, C.: Arbitrated ensemble for solar radiation forecasting. In: Rojas I., Joya G., Catala A. (eds.) Advances in Computational Intelligence, pp. 720–732. Springer, Cham (2017)
19. Wang, H., et al.: An adaptive ensemble model of extreme learning machine for time series prediction. In: Proc. Int. Comput. Conf. Wavelet Act. Media Technol. Inf. Process. (ICCWAMTIP), pp. 80–85. IEEE, Piscataway, NJ (2016)
20. Koprinska, Z.W.I., et al.: Static and dynamic ensembles of neural networks for solar power forecasting. In: Proc. Int. Joint Conf. Neural Netw. (IJCNN), pp. 1–8. IEEE, Piscataway, NJ (2018)
21. Idrees, M.M., et al.: A heterogeneous online learning ensemble for non-stationary environments. Knowl.-Based Syst. 188, 104983 (2019)
22. Rossi, A.L.D., et al.: MetaStream: A meta-learning based method for periodic algorithm selection in time-changing data. Neurocomputing 127(15), 52–64 (2014)
23. Lemke, C., Gabrys, B.: Meta-learning for time series forecasting and forecast combination. Neurocomputing 73(10), 2006–2016 (2010)
24. Liu, C., et al.: Online ARIMA algorithms for time series prediction. In: Proc. Assoc. Advance. Artif. Intell. (AAAI), pp. 1867–1873. ACM Press, New York (2016)
25. Anava, O., et al.: Online learning for time series prediction. In: Proc. Conf. Learn. Theory, pp. 1–13. Microtome Publishing, Brookline, MA (2013)
26. Ke, G., et al.: LightGBM: A highly efficient gradient boosting decision tree. In: Proc. Adv. Neural Inf. Process. Syst. (NIPS), pp. 3149–3157. ACM Press, New York (2017)
27. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. 9(8), 1735–1780 (1997)
28. NSRDB Data Viewer. https://maps.nrel.gov/nsrdb-viewer/, Retrieved: January 10, 2021
29. NREL Solar Power Data for Integration Studies. https://www.nrel.gov/grid/solar-power-data.html/, Retrieved: January 10, 2021
30. Solorzano, J., Egido, M.A.: Hot-spot mitigation in PV arrays with distributed MPPT (DMPPT). Solar Energy 101, 131–137 (2014)
31. Ghazi, S., Ip, K.: The effect of weather conditions on the efficiency of PV panels in the southeast of UK. Renewable Energy 69, 50–59 (2014)
32. Pei, T., Hao, X.: A fault detection method for photovoltaic systems based on voltage and current observation and evaluation. Energies 12(9), 1712 (2019)
33. Zhao, Y.: Fault analysis in solar photovoltaic arrays. Master's Thesis, Northeastern University (2010)
34. Zhao, Y., et al.: Line?line Fault analysis and protection challenges in solar photovoltaic arrays. IEEE Trans. Ind. Electron. 60, 3784–3795 (2013)
35. Lin, X., et al.: Online fault detection and tolerance for photovoltaic energy harvesting systems. In: Proc. IEEE/ACM Int. Conf. Computer-Aided Design, pp. 1–6. ACM Press, New York (2012)
36. Dhimish, M., et al.: The impact of cracks on photovoltaic power performance. J. Sci.: Adv. Mater. Devices 2, 199–209 (2017)
37. Arani, M. S., Hejazi, M. A.: The comprehensive study of electrical faults in PV arrays. Hindawi J. Elect. Comput. Eng. 2016, 8712960 (2016)
38. Bashir, N., Irwin, D., Shenoy, P.: DeepSnow: Modeling the impact of snow on solar generation. In: Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, pp. 11–20. ACM Press, New York (2020)
39. Odashima, J., Shinoda, Y., Takeda, H.: Estimation method of photovoltaic power output using extended Lambert–Beer law. IEEJ Trans. Power Energy 140, 42–48 (2020)
40. Sandell, M.: The effect of snowfall on the power output of photovoltaic solar panels in Halifax, NS. Dalhousie Univ. (2012), https://cdn.dal.ca/content/dam/dalhousie/pdf/science/environmental-science-program/Honours%20Theses/2012/HonoursThesisMarniSandell.pdf, Retrieved May 27, 2021
41. Chen, J., et al.: Study on impacts of dust accumulation and rainfall on PV power reduction in East China. Energy 194, (2020)
42. Hussain, A., Batra, A., Pachauri, R.: An experimental study on effect of dust on power loss in solar photovoltaic module. Renewables 4(9), (2017)
43. Andrea, Y., Pogrebnaya, T., Kichonge, B.: Effect of industrial dust deposition on photovoltaic module performance: Experimental measurements in the tropical region. Int. J. Photoenergy 2019, 1892148 (2019)
44. Chouder, A., Silvestre, S.: Automatic supervision and fault detection of PV systems based on power losses analysis. Energy Convers. Manage. 51, 1929–1937 (2010)
45. Dolara, A., Cristian, L.G., Ogliari, E.: Efficiency analysis of PV power plants shaded by MV overhead lines. Int. J. Energy Environ. Eng. 7, 115–123 (2016)

46. Leva, S., Mussetta, M., Ogliari, E.: PV module fault diagnosis based on microconverters and day-ahead forecast. IEEE Trans. Ind. Electron. 66, 3928–3937 (2019)
47. Alam, M. K., et al.: A comprehensive review of catastrophic faults in PV arrays: Types, detection, and mitigation techniques. IEEE J. Photovoltaics 5, 982–997 (2015)
48. Skomedal, A. F., et al.: Robust and fast detection of small power losses in large-scale PV systems. IEEE J. Photovoltaics 11, 819–826 (2021)

How to cite this article: Lee H, Lee Jun-Gi, Kim Nac-Woo, Lee Byung-Tak. Model-agnostic online forecasting for PV power output. IET Renew. Power Gener.. 2021;1–13.
https://doi.org/10.1049/rpg2.12243