IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 25, 2018, accepted November 12, 2018, date of publication November 26, 2018,
date of current version December 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2883330

Forecasting Solar Power Using Long-Short Term Memory and Convolutional Neural Networks

WOONGHEE LEE, KEONWOO KIM, JUNSEP PARK, JINHEE KIM, AND YOUNGHOON KIM
Department of Computer Science, Hanyang University, Ansan 15588, South Korea

Corresponding author: Younghoon Kim (yhkim7951@gmail.com)

This work was supported in part by the Korea Institute of Energy Technology Evaluation and Planning (KETEP) granted financial resource from the Ministry of Trade, Industry & Energy, South Korea, under Grant 20153010011980, in part by the Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning under Grant 2017M3C4A7063570, and in part by the research fund of Hanyang University under Grant HY-2014-N.

ABSTRACT As solar photovoltaic (PV) generation becomes cost-effective, solar power comes into its own as the alternative energy with the potential to make up a larger share of growing energy needs. Consequently, operations and maintenance cost now have a large impact on the profit of managing power modules, and the energy market participants need to estimate the solar power in short or long terms of future. In this paper, we propose a solar power forecasting technique by utilizing convolutional neural networks and long–short-term memory networks recently developed for analyzing time series data in the deep learning communities. Considering that weather information may not be always available for the location where PV modules are installed and sensors are often damaged, we empirically confirm that the proposed method predicts the solar power well with roughly estimated weather data obtained from national weather centers as well as it works robustly without sophisticatedly preprocessed input to remove outliers.

INDEX TERMS Solar power forecasting, deep learning, convolutional neural networks, long-short term memory.

I. INTRODUCTION

As rapidly growing interest in renewable energy and gradually decreasing costs of power generation, alternative power generation has received a lot of attention over the last decade. Especially, solar power has the potential to make up a larger share of growing energy needs as it becomes more and more cost-effective. Reportedly, photovoltaic (PV) module costs have fallen by about four-fifths, making residential solar PV systems as much as two-thirds cheaper than in 2010 [1]. As the installation of PV modules becomes cheaper, operations and maintenance (O&M) cost gradually takes a large portion of the power generation cost.

Maintaining PV operations may typically be considered simpler and cheaper than the other alternative energy sources, such as wind power and natural gas; however, due to the power supply being largely dependent on the changing weather, deciding power price and managing the budget of production for system operators and power market participants become the essential issues to keep power plants commercially profitable [2].

Solar PV system operators are using long and short term solar power forecasts to schedule generation, estimating operating reserves and ensuring the system stability by fluctuation in output. Market participants also manage their generation portfolios based on forecasts. Considering that 38 GW of solar capacity is traded on the energy market in 2014 in Germany for example [3], solar power forecasting influences the market price and cost-efficiency of power generation substantially. Hence, solar power forecasting is playing an important role in the management of PV systems nowadays.

Solar power forecasting techniques have been extensively studied not only in solar power industry but also in the academic communities (See recent surveys for an overview [3]–[6]). The list of traditional regression techniques used for solar power prediction includes multiple linear regression [7], [8], random forest regression [9], support vector regressor [10]–[12], k-nearest neighbor [11], autoregressive moving average (ARMA) and autoregressive integrated moving average (ARIMA) [13]–[16].

These traditional forecasting techniques have been generally relied on physical models calculating solar power based on irradiation or a simple linear/non-linear regression model. However, due to not only the non-linear dependence of the efficiency of solar power generation on meteorological variables such as irradiation and temperature [4], but also irregular errors in input data exhibited by inaccurate sensors [17], they require heavy preprocessing to refine input data and also suffer from poor accuracy with incomplete inputs. Thus, deep learning, which is the-state-of-the-art machine learning technique based on artificial neural networks that has been achieved a great performance improvement in various prediction problems, is recently introduced for forecasting solar power [8], [12].

Our contributions: In this paper, we propose an effective solar power forecasting technique by introducing convolutional neural networks (CNNs) and long-short term memory (LSTM) networks recently developed for analyzing time series data in the deep learning communities. Considering that weather information may not be always available for each site where PV modules are installed if weather station is not near that area or sensors are damaged, we predict the solar power generated in the next day based on the records measured in each PV inverter as well as the weather data observed with a large granularity by the national meteorological organization.

As we mentioned in the above, almost recent techniques either depend on exact and clear data inputs or require some unusual data costing a high price to measure them. In [8], the propose method uses optical devices called Clear Sky camera filters that can estimate the irradiation under cloudless sky. In [18], they analyze sky images using deep learning techniques for estimating the weather and solar irradiation more accurately. The method proposed in [19] requires data measured from weather satellites. Furthermore, the techniques proposed in [20] and [21], which predict solar power based on deep learning, resort in heavy preprocessing to obtain clear training data without outliers. Even if exploiting such unconventional data and preprocessing with great effort may improve the accuracy of forecasting, it significantly raises the cost of maintenance and possibly increases the cost of power generation.

Despite of the rapid growth in deep learning techniques and its flexibility for developing various networks suitable for applications, it has not been extensively applied to solar power forecasting. Furthermore, most of traditional and recent works have been assumed clean preprocessed training datasets without outliers. Our contributions are summarized as follows:

- We develop a novel deep neural network to estimate the next day's solar power; in our network, we select appropriate layers very suitable for regressing a target value (i.e., the next day's solar power) from a time series data observed in the previous day such as temperature and solar irradiation measured every 10 minutes.
- We extensively evaluate the performance of the state-of-the-art related work and our proposed algorithm, and confirm that our algorithm predicts the solar power accurately and robustly even we do not refine the raw data with great effort. Furthermore, we show that our model improves the performance using weather data roughly estimated with data obtained from national meteorological centers, which was not observed exactly at the location where PV modules are installed.

We organize the paper as follows: In Section II, we extensively investigate the solar power forecasting techniques developed so far. In Section III, we formulate the problem of solar power prediction with the definition of notations for input and output data sets, and briefly discuss the recent technologies of deep learning to be used in our method. We present our proposed network in Section IV and the empirical performance evaluation of the proposed method in Section V.

II. RELATED WORK

Regression is an analytical technique that predicts a target value by modeling the relationship between input variables. To predict solar power of near future based on various measures such as PV output power, temperature, humidity and precipitation, regression has been thus widely used. In this section, we examine the techniques developed for forecasting solar power and categorize them by the regression algorithms used. We provide the summary of some important work selected from each category in Table 1.

Multivariate linear regression (MLR) models the relationship between multi-attributed input and output using an affine function, and is utilized for solar power prediction in [7]. Due to the linearity assumption, it however suffers from poor performance. Hence, support vector regression (SVR) is developed to represent the relationship between two variables nonlinearly. In [10] and [23], SVR is exploited to predict solar power by including weather information such as cloudiness, atmosphere transmissivity and sun duration in the input attributes.

Autoregressive moving average (ARMA) and autoregressive integrated moving average (ARIMA) models are widely used in regression when the input is time series data. ARMA models use the past n input values to predict the current output based on linear regression model, and ARIMA, a generalized model of ARMA, considers the difference between two outputs in a row in the past as well. Many works have dealt with solar power forecasting using these methods in [13]–[16]. However, since ARMA and ARIMA are still based on linear regression, they inevitably result in bad performance.

In [9], an ensemble method called bagging [26] is utilized to achieve accurate predictions by aggregating the results calculated by heterogeneous regression models such as decision tree and k-nearest neighbors (KNN). In [12], a simple ensemble using both deep belief networks and SVR is proposed. These works have obtained a slightly better performance compared to using a single technique individually. However, they fail to overcome the limitation of the traditional regressors used in the ensemble methods.

Traditional artificial neural networks (ANNs), which have a basic structure called fully-connected networks, have also been widely applied to solar power prediction. In [24], a generalized ARMA model by using a fully-connected network,

TABLE 1. A summary of related work for prediction of renewable energy.

Ref.	Year	Forecast horizon	Method	Description
[15]	2009	5 mins∼1 hour ahead	ARIMA	They evaluate various regressior such as ARIMA, Transfer functions, neural networks, and hybrid models to predict solar radiation over short time horizons. It confirms that ARIMA is the best performer since it can capture the diurnal cycle better than the other methods.
[22]	2011	24 hours ahead	ANN	It utilizes ANN for estimating the solar power of the next 24 hours in a hourly granularity. To amplify the training data, it suggests to use the day showing similar weather condition as virtual inputs for predicting the next day's power.
[23]	2012	1 day ahead	SVR	The data is first classified into four categories by the weather condition: clear, cloudy, foggy, and rainy. The conditions are then encoded as binary attributes and by using SVR, they predict power based on the weather condition and the past power generation.
[13]	2012	1 hour ahead	ARMA	Implementing the two-phase realization, the autoregressive moving average (ARMA) model has better performance predicting the future values on a micro-grid level based on solar radiation data from SolarAnywhere. Also, the solar generation data produced by System Advisor Model (SAM) supports the decision making process.
[10]	2013	1 & 3 hours ahead	SVR	Support vector regressor (SVR) is used to forecast solar power in this work; its input data contains recent records of atmospheric transmissivity and some other meteorological variables.
[14]	2013	1 hour ahead	ARIMA	This work suggests to use multi-year observation of a horizontal irradiation dataset for solar power predictions to capture the seasonal patterns; it utilizes ARIMA and the dataset is collected from a high-end device measuring meteorological values called MeteoLab.
[24]	2014	3 hours∼3 days ahead	ANN	They introduce an ANN model called NARX, in which the outputs are fed back to the network with delays, for forecasting solar power with different targeting terms from 3 hours to 3 days.
[12]	2014	24 hours ahead	Ensemble	An ensemble of traditional regression and deep learning algorithms was proposed; they connect a deep belief network (DBN) and SVR by using the output of DBN as the input of SVR.
[25]	2015	24 hours ahead	ANN	Aerosol Index (AI), which measures transmissivity affected by dust in the air such as desert dust, biomass burning, volcano smoke and power plant emission, is newly considered in this work as input for solar power prediction. While the accuracy of prediction is improved slightly on cloudy days, it is rather degraded in sunny days. Unlike their argues, the benefit of using AI is appeared to be very little.
[7]	2015	1 month ahead	MLR	Analysis of variance(ANOVA) was used for multiple linear regression(MLR) analysis to forecast solar power based on European Centre for Medium-Range Weather Forecasts(ECMWF) data.
[9]	2015	1 month ahead	Ensemble	Ensemble methods was proposed obtained from the individual models such as decision tree, random forest, k-nearest neighbors (KNN), ridge regression, lasso regression, and gradient boosting. The ensemble model shows significant improvement on probabilistic forecasts compared to the individual models.
[8]	2016	1 day ahead	AE & LSTM	Experiments for solar power forecasting has conducted by various method such as multiple linear regression (MLR), deep belief network(DBN), auto encoder, and long short-term memory (LSTM). The proposed deep learning algorithms show better forecasting performance than other reference models.

called non-linear auto regressive models with exogenous input (NARX), is introduced. Furthermore, many solar power forecasting systems have been developed based on ANNs in [22], [24], and [25] using various inputs such as Aerosol Index [27] that are known to affect power generation. However, due to the large computational cost and memory requirements of the traditional ANNs, it is hard to obtain a regressor accurate enough to use practically.

Recently, on the progress of parallel computing using GPUs and the availability of large datasets, various architectures of ANNs are developed by deepening the networks such as long short term memory (LSTM) [28] and convolutional neural networks (CNNs) [29]. By the adoption of such techniques, solar power prediction has been significantly improved. In [8], a deep learning network for forecasting solar power is proposed by combining auto-encoder (AE) [30] and LSTM, and achieved a large improvement of performance; AE reduces the dimensionality of input vectors for each time slot and LSTM (we will study about it later in detail since we also adopt it in our model) is the state-of-the-art recurrent neural network developed to capture the features in a time series efficiently.

III. PRELIMINARY

In this section, we define some notations required to formulate our problem and give our problem definition. To tackle the problem, since we will use the state-of-the-art technique called deep learning, we provide backgrounds about the technique.

Let $\mathbf{D} = \{D_1, \ldots, D_T\}$ be the dataset collected for T days. Let $D_j = (X_{j,0}, \ldots, X_{j,N-1}, y_j)$ be a tuple recorded in the j-th day and it contains a sequence of vectors X_{ji}, where X_{ji} consists of M attributes (i.e., $X_{ji} = \langle A_1, \ldots, A_M \rangle_{ij}$), and the

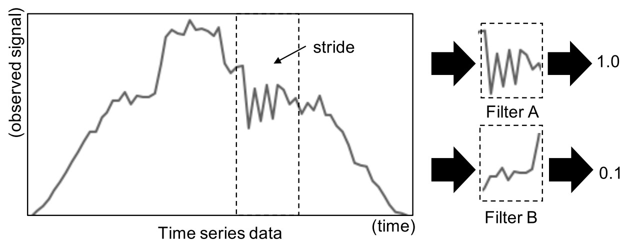


FIGURE 1. Features are extracted with filter from data with element-wise multiplication.

corresponding output data y_j. In this paper, X_{ji} represents a set of values collected from a photovoltaic inverter for every 10 minutes, including voltage, electric capacity, the amount of power generation, irradiation and temperature, as well as weather attributes obtained from national weather centers such as wind speed, humidity and precipitation. Since the sensors measure those values a day long, N is then 144. Furthermore, y_j denotes power generation of the j-th day.

Given D_j collected from photovoltaic inverter and meteorological observation, our goal is to predict the amount of solar power y_{j+1} for the next day $j + 1$.

A. CONVOLUTIONAL NEURAL NETWORK FOR TIME SERIES DATA

Convolutional neural network (CNN) was originally proposed in [31] for automatic classification of digit images and has been achieved great successes in a number of image recognition applications [32]–[34]. Compared to the traditional fully connected neural networks which may discover patterns scattered over all coordinates, CNN is designed to focus on finding local patterns (i.e., patterns located in adjacent dimensions) that frequently occur in 2-dimensional images. Thus, CNN is naturally well adopted with 1-dimensional time series data for various classification as introduced in [35].

CNNs also successfully optimize the performance with a less memory requirement; fully connected neural networks have connection between all neurons in the adjacent layers. Therefore, the fully connected neural networks have a tremendously large number of model parameters. However, CNNs share model parameters named filters and the number of parameters used in CNNs can be efficiently reduced [36]. Furthermore, CNNs are capable to well extract features with various lengths [37] The following example shows how filters work in a CNN to estimate the outputs.

Example 1: Suppose that we have a tuple D_j with only a single attribute which is irradiation recorded every 10 minutes as shown in Figure 1, and want to predict the power generation of next day. For example, the solar power of a cloudy day when the irradiation shows the pattern similar to the signals between two dashed line in Figure 1 probably decreases. Let assume that filter A and B represent the features positively correlated with cloudy and sunny days respectively. The convolution of the signals between two dashed line and the filter A results in a large value (e.g., 1.0) indicating that it will be cloudy, while that of the signals and the filter B outputs a small value (e.g., 0.1) implying that it will not be sunny.

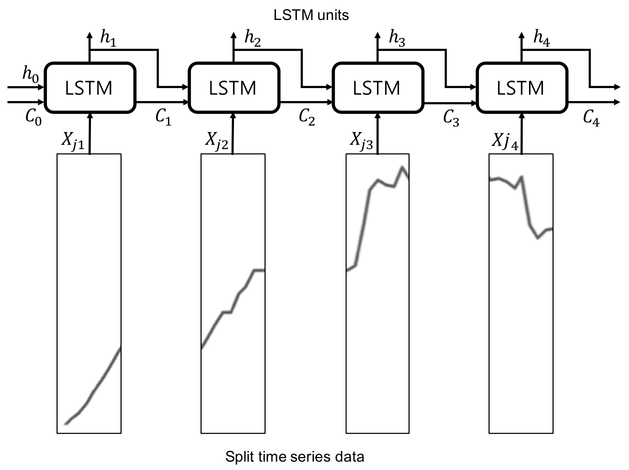


FIGURE 2. LSTM conveys information using input data with cell state and hidden state from previous unit.

B. LONG SHORT TERM MEMORY FOR TIME SERIES DATA

Long short term memory (LSTM) is one of recurrent neural network (RNN) that has successively dealt with a vanishing gradient problem of the traditional RNNs, which makes it impossible to find useful features occurring in the early position in input sequences. By introducing cell status which is specially designed to keep old information well, LSTM has alleviated the problem effectively. In Figure 2, we illustrate the process the forward propagation with a sample signal. The LSTM network depicted in a box takes a vector X_{ji} as well as hidden $h_{j,i-1}$ and cell $C_{j,i-1}$ statuses, which are output by the previous run of LSTM, as input. Then, it outputs hidden h_i and cell C_i statuses, which is recurrently input to the next step with the next input vector $X_{j,i+1}$. Note that even if the LSTM networks in boxes are presented separately in Figure 2, we use a single LSTM network repeatedly. In the following example, we discuss why LSTM tends to capture the patterns over time series well.

Example 2: Suppose that we have a time series where a signal gradually increases in its early stage but decreases in the last moment. Assume that we want to find the point that the signal shows such a pattern, that is, a sudden decrement after a long increment. With feeding signals that grows gradually, LSTM may recognize a feature which describes the pattern of increasing signal and such feature will be encoded in the hidden status. Then, if the signal is decreased, LSTM may discover the moment we want to find.

IV. POWER GENERATION PREDICTION USING DEEP NETWORKS

We exploit CNNs and LSTMs in our model to extract useful features for the solar power prediction of the next day. Since solar energy is converted to electrical energy directly by semi-conductors materials, solar radiation level and temperature are the most important factors. It is known that solar radiation level has a positive correlation on solar power, on the other hand, there is a negative correlation between temperature and panel power. In this section, we describe the details of our model.

We first define a training dataset as follows: For each j-th day, the input is a sequence of multi-attributed vectors $X_{j,0}, \ldots, X_{j,N-1}$ where X_{ji} is a vector with M values collected from various sensors every 10 minutes as defined in Preliminaries. The output is then the solar power y_{j+1} of the next day.

A. ADOPTION OF RECURRENT NEURAL NETWORKS (RNNS)

To predict the power generation of the next day based on the input such as irradiation, temperature and power output collected all day long on the previous day, the most simple way is to represent the previous day's input as a vector with the average of each attribute, and use linear or non-linear regression algorithms, such as logistic regression or SVR, that calculate the next day's output. However, it is hard to exactly predict the power without considering the trends of changes in the input attributes over time because we cannot utilize the momentum in the input signals; for example, the trend of temperatures in a day may give us a useful clue to consider the next day's weather. We discuss in detail that the trend of changes has a correlation to the next day's power output in the following.

Example 3: Suppose that we have collected temperature, precipitation, irradiation and DC current by every 10 minutes in a day as shown in Figure 3. Figures 3(a)–(d) show four patterns of changes in each attribute, which represent (a) sunny day, (b) rainy day, (c) sunny morning and rainy afternoon and, (d) sunny day with a system failure respectively.

With a high chance, the next day of a sunny day may be a sunny day too and the next day of a rainy day may become another rainy day rather than a sunny day. In such days, the solar power of the next day may not largely differ from that of the previous day. In a day when it is sunny in the morning but rains from the afternoon as Figure 3(c), however, it probably rains in the next day and thus, the changing patterns of the observed values over time should be considered in the prediction of the next day's power output.

It is not limited to the values related to weather only; changing patterns of other attributes such as current and voltage may indicate the performance of power generation systems. For example, a sunny day when DC current suddenly drops by a system failure (see Figure 3(d)) may be followed by a day with a low power output.

Note that we do not aim at forecasting the next day's weather explicitly in our model. We illustrate some cases using precipitation and temperature in the above, however, it is simply to show that we can find useful patterns in those attributes by our model.

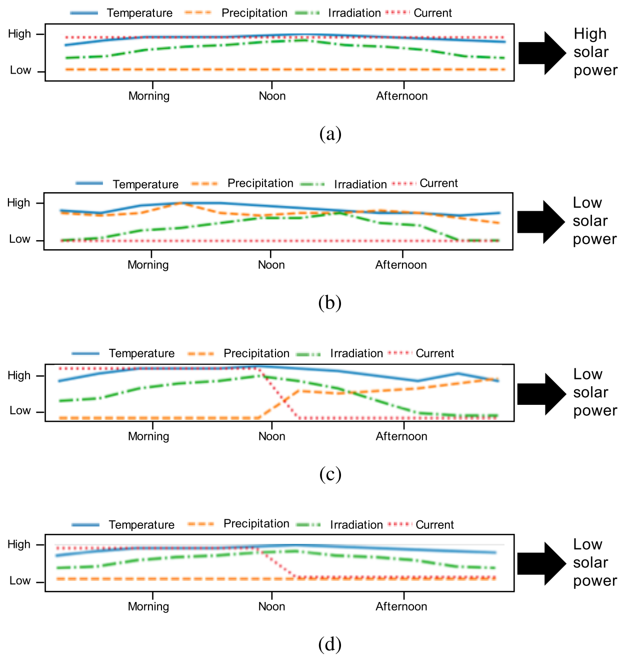


FIGURE 3. Weather condition affects solar power generation. (a) Sunny day. (b) Rainy day. (c) Sunny morning and rainy afternoon. (d) Sunny day with PV inverter malfunction.

The above examples show that considering a time series of data is crucial to improving the performance of prediction, even if it increases computational cost due to the high dimensionality by using the time series as input. As we have introduced in Preliminaries, RNNs are especially designed to handle the time series data, where the correlation between inputs with a time distance get weaker as the distance increases, that is, patterns typically have short-term local dependencies. Furthermore, LSTM networks [38] have shown the best performance among RNNs, we exploit LSTM in our model.

B. EXPLOITING CNNS FOR SHORT-TIME LOCAL FEATURES

The straightforward way to exploit LSTM for finding local patterns from time series data is to simply use the raw vector in each time slot as input to LSTM networks. To learn local features from time series data frequently measured between short terms, the performance is, however, significantly degraded since RNNs unfortunately tend to forget early patterns in a time series. Thus, in [8], Auto-encoder [30] is introduced for dimension reduction; an encoder is trained in advance to the actual learning of regressor for power prediction and then, in the phase of training a LSTM network, the latent factor vector of a smaller dimensionality output by the pre-trained encoder is input to the network. However, we hardly expect that the encoder can find relevant latent factors for the power prediction since the encoder is independently trained regardless of the next day's solar power.

As we have discussed in Preliminaries, CNN [29], which is originally proposed to capture local features in 2-dimensional images, has been well adopted for finding short-time local features from time series data [35]. Theoretically, using a large filter in CNNs may enable us to discover long-range features as well as short-time local features if there exist some useful ones with a strong correlation with the output. However, a large filter increases the computational cost significantly and moreover, training data also should be very large enough to cover all size features with using a single large filter. In [39], a doubled CNN (D-CNN) by concatenating the output from two CNNs with different sizes is proposed to adaptively extract both long-range and short-time local features. As some recent works such as [40] and [41] have confirmed the efficiency of D-CNN in the applications that take time series data as input, we also utilize D-CNN in our model. The architecture of D-CNN used in our model is shown in Figure 4.

C. SETTINGS OF OUR NETWORKS

We describe the architecture of our proposed networks by presenting the computation of forward propagation. First, we present the setting of D-CNN depicted in Figure 4.

Let n and s denote the interval and stride sizes respectively, which are constants for slicing the time series. D-CNN takes a sequence of $6n$ vectors as input representing n-hours observations measured between every 10 minutes. With $k = 0, 1, \ldots,$ a subsequence $\langle X_{j,k\cdot s}, X_{j,k\cdot s+1}, \ldots, X_{j,k\cdot s+6n-1}\rangle$ of the time series D_j is input to two separate CNNs both of which have 128 filters but, whose filter sizes are 2 and 4 respectively; since the input signal contains M channels, each filter of the first convolutional layers in two CNNs is then represented by $(M \times 2)$ and $(M \times 4)$-dimensional vectors respectively. The forward propagation is computed following the layers as shown in Figure 4 through two CNNs, and the last outputs by the batch normalization layer in both CNNs are concatenated and fed into a fully-connected network after flattening. Finally, the 1024-dimensional vector output by the fully-connected layer is input to LSTM network illustrated in Figure 5.

For our LSTM network, we set both of the hidden state and the cell state sizes to 1024. A 1024-dimensional vector, calculated by D-CNN with a slice of input sequence of length $6n$, is input to LSTM network and the hidden state output by the network is recurrently fed into the network with the next output of D-CNN. Finally, the hidden state computed with the final input is used for forecasting the next day's solar power by going through two fully-connected layers whose output sizes are 1024 and 1 respectively. Note that the *batch normalization layers* in Figures 4 and 5 indicate optimization layers which are used in the training phase only; it is developed in [42] to boost learning efficiency.

With all convolutional and fully-connected layers, the activation function is set to the *rectifier function* [43], except the last full-connected layer in Figure 5 that is using a linear function.

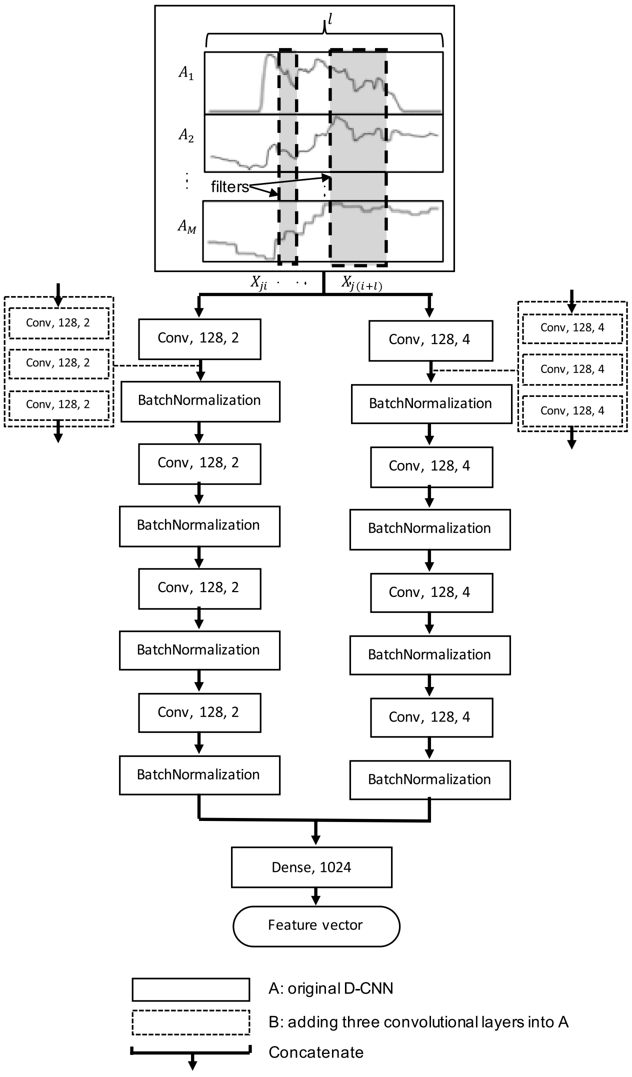


FIGURE 4. Convolutional neural network layers with small and large sized filters.

D. VARYING THE SETTING FOR THE OPTIMAL NETWORKS

To find the best setting of networks such as the architecture of layers (e.g., the number of convolutional layers used) and filter sizes in D-CNN, we perform experiments with varying the networks. To see the influence of increasing the depth of convolutional layers in D-CNNs, we have inserted three convolutional layers in both CNNs as enclosed in the dashed-line boxes in Figure 4. Furthermore, we also vary the filter sizes in D-CNN with (2, 4), (4, 8) and (8, 16) using input time series sliced by 3 hours with setting stride size to 3 hours.

In Table 2, we present the results of varying the settings in terms of MAPE, which will be defined later in Experiments. The result shows that using more convolutional layers rather degrades the performance by increasing the complexity of networks. With varying the filter sizes in D-CNN, larger filters usually result in better prediction. The filter sizes (8, 16) are valid with 3 hours-long sequences (i.e., length of 18),

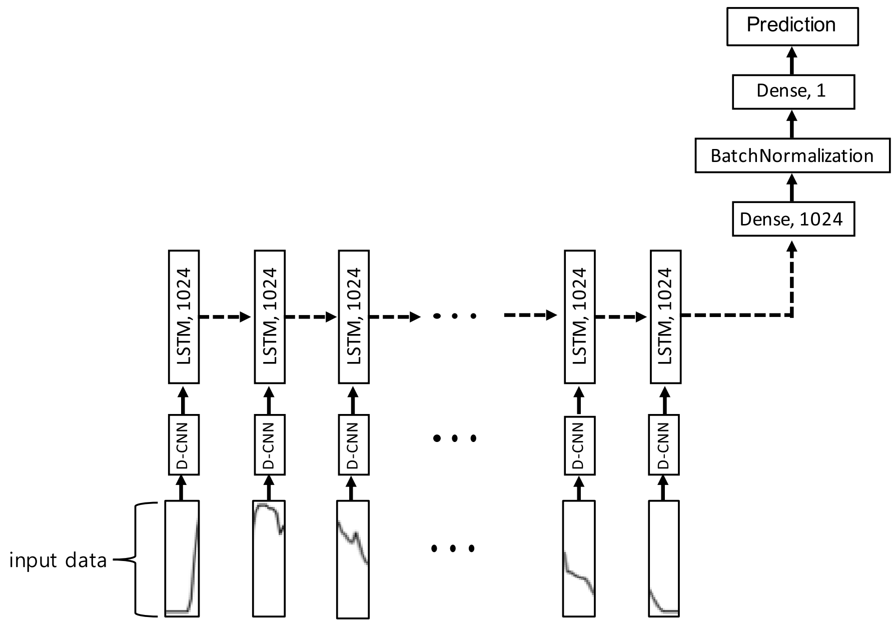


FIGURE 5. Sequence model with feature extraction unit.

TABLE 2. Parameters search for CNNs.

	FS: 2, 4	FS: 4, 8	FS: 8, 16.
A	104.20	89.18	65.16
B	181.80	136.11	180.51

A: original CNNs of two different sized filters
B: adding three convolutional layers into A
FS: filter size

however, it does not when we use input sequences with a shorter size. Thus, in our experiments in the next section, we use the largest valid filter sizes among (2, 4), (4, 8) and (8, 16) according to the input sequence size.

Furthermore, we have also tested the performance of power prediction with changing the hidden state size of LSTM network in Figure 5. With size 1024, we have obtained the good enough performance and using larger hidden states does not improve the performance much while the training time grows largely.

V. EXPERIMENTS

We empirically demonstrated the accuracy of the proposed algorithm forecasting solar power of the next day. The experiments are done on a workstation with Intel(R) Core(TM) i7-6850K CPU at 3.60GHz and 125GB of main memory. In the machine, three NVIDIA GeForce GTX 1080 Ti graphics cards are installed. We implemented all algorithms with Python 3.x and used opensource machine learning packages such as Scikit-learn 0.19.2 as well as Keras 2.0.8 using TensorFlow 1.4.0 as backend.

A. DATASETS

We have collected data from 71 photovoltaic (PV) inverters in 14 sites where solar power generators are installed in South Korea as shown in Table 3; the data is collected from 10 cities between Feb. 29, 2012 and Jan. 6, 2016 while the terms for data collection with each inverter are slightly different. The records obtained from all inverters contain the same list of attributes, which are inverter ID, site ID, date, time, power generation, irradiation and temperature. Table 4 summaries the attributes recorded from inverters.

TABLE 3. The term of data collection and the number of inverters in each site.

Location	Term	The number of inverters
Busan 1	2013 - 2015	9
Busan 2	2013 - 2015	12
Chuncheon	2013 - 2015	3
Gunsan	2012 - 2015	8
Haenam 1	2013 - 2015	2
Haenam 2	2014 - 2015	10
Haenam 3	2013 - 2015	6
Incheon 1	2012 - 2015	6
Incheon 2	2013 - 2015	2
Incheon 3	2013 - 2015	2
Sangju	2013 - 2015	2
Seoul	2013 - 2015	4
Suwon	2013 - 2015	3
Yeoncheon	2012 - 2015	2

We also obtained the weather records from Korea Meteorological Administration[1] for those locations where the solar power generators are installed. In the dataset, weather conditions such as temperature, wind speed and humidity are recorded at intervals of an hour. In Table 4, we provide the list of weather attributes used in our experiments.

Preprocessing: We performed the minimal preprocessing and used as it is as close as possible to the raw data collected

[1]https://web.kma.go.kr/eng/

TABLE 4. Recorded attributes from inverters and weather center.

No.	Notation	Description	Unit
1	ddate	Date	
2	dtime	Time	
3	site	Location name	
14	devseq	Inverter ID	
5	wh	Power generation	kWh
6	trad	Slope irradiation	W/m^2
7	hrad	Horizontal surface irradiation	W/m^2
8	atemp	Air temperature	$^\circ C$
9	mtemp	Inverter temperature	$^\circ C$
10	ddate	Date	
11	dtime	Time	
12	site	Location name	
13	wtime	Time	
14	watemp	Air temperature	$^\circ C$
15	wspeed	Wind speed	m/s
16	whumid	Humidity	%
17	wgtemp	Ground temperature	$^\circ C$

from PV inverters. For preprocessing, we have substituted the negative solar power values, which are obviously erroneous values, with zero. The categorical attributes such as *inverter ID* (0∼70) and *date* (0∼365) are replaced by one-hot encoded vectors; for example, *date* is represented by a 366-dimensional vector whose all entries are 0 but only the corresponding coordinate has 1. We also simply normalized all numerical attributes to be between 0 and 1. Furthermore, since our problem is to forecast the next day's solar power based on the data observed in the previous day, we selected the records collected from a pair of adjacent days only; the records of a whole day are often omitted because some inverters have been power-off due to maintenance operations.

We join the records from inverters, consisting of the attributes from No. 1 to No. 9, and those from the national weather center, including the attributed No. 10 to No. 17, using date, time and location name as joining keys. Finally, we obtained a data set with 18,620 pairs of input time series and output value (i.e., input records of 18,620 days and the next day's power output). Then, we split the data set into a training and test data sets by the ratio of 75:25. Furthermore, to obtain an unbiased evaluation of a model fit by avoiding overfitting, we shuffled the training data set and sampled 10 percentages of the data for validation. The remainder 90 percentages of training data set is then used for fitting model parameters, and by assessing the loss function with the validation data set, we examine if the model is biased or not and determine to stop the training epochs; we stop training networks if the loss is gradually increased or does not change for the last 500 steps.

In Figure 6, we plotted the losses of training and validation data sets with every epoch until training stops; the graph shows the losses calculated while training our model *CNN+LSTM 1h* which will be defined later in this section, and the other models trained in our experiments show similar trends. In the graph, it takes about 40,000 epochs for training until the stop condition is satisfied (i.e., the loss with validation data set does not change for 500 epochs).

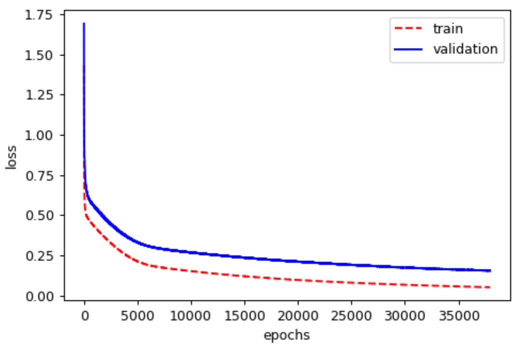


FIGURE 6. Loss of training and validation data sets in the training phase.

B. PERFORMANCE MEASUREMENT

Mean absolute percentage error (MAPE), root mean square error (RMSE), and mean absolute error(MAE) are frequently used to assess the difference between the prediction of a model and the actually observed value.

MAPE calculates the average error ratio in percentage to the correct values as

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|,$$

where n represents the size of test data set, A_t is the actual solar power and F_t is the estimated one for the t-th day in the test data set.

RMSE is the average of the root mean of squared error between the predicted value and the observed value, and calculated as

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n}(A_t - F_t)^2}{n}}.$$

MAE denotes the mean of absolute difference between the predicted solar power and the observed one, which is computed as

$$MAE = \frac{\sum_{i=1}^{n}|A_t - F_t|}{n}.$$

Considering that the purpose of forecasting solar power in many applications is usually to estimate energy supply in advance for trading in a market, RMSE and MAE are suitable quality measures to evaluate prediction algorithms. However, since RMSE and MAE assess the errors of prediction without considering the scale of error or observed solar power, the error in a small scale is simply considered as the same as that in a large scale, and thus, MAPE is also useful to identify the method that predicts the trend well. In particular, because the solar power value often gets close to zero in cloudy or raining days, considering RMSE and MAE only may mislead for evaluating the algorithms.

C. IMPLEMENTED ALGORITHMS

We implement 9 traditional regressors for comparative performance evaluation. For traditional regression algorithms, we simply flatten each input sequence of vectors to be a 1-dimensional vector.

For solar power prediction using deep learning, we implemented the state-of-the-art deep learning method which consists of autoencoder and LSTM proposed in [8]. Note that it is reported that the network outperforms deep learning methods such as fully-connected neural network, LSTM as well as deep belief net for forecasting solar power in [8].

The implemented algorithms are summarized as follows:

- **LR:** This is an implementation of a linear regression that discovers the relationship between input variables and an output response by fitting a linear function such as $y = \beta_0 + \sum_i \beta_i X_i$ where X_i is an input variable, β_i is the model parameter and y is output.
- **RFR:** It indicates the implementation of a random forest regressor [44]. This method ensembles multiple decision trees and chooses the mean value of their predictions for an answer. In our implementation using scikit-learn package, we set to use 10 decision trees and all the other parameters are left to default values.
- **SVR:** This denotes a support vector regressor [45] implemented using scikit-learn. To represent the relationship between two variables non-linearly, we select RBF for kernel.
- **EN:** It implements Elastic Net [46]. Elastic Net is a technique that regularizes coefficients in a linear regression; in the linear function shown in the above, it finds β_i minimizing the l_1-norm and l_2-norm together as $\arg\min_\beta(||y - X\beta||^2 + \lambda_2\|\beta\|^2 + \lambda_1\|\beta\|_1)$.
- **SGDR:** It is based on the identical model and regularization to EN, but utilizes a stochastic gradient descent for optimizing model parameters, which is also provided in the scikit-learn package.
- **BR:** This is Bayesian ridge algorithm [47] that improves linear regression by using probability distribution instead of individually estimated values for objective function.
- **LL:** It denotes the implementation of lasso that is basically a linear regression using l_1-norm for regularization.
- **PAR:** It is a passive agressive regressor [48]. This can optimize the coefficients in a linear regression model incrementally with streaming input.
- **OMP:** This implements an orthogonal matching pursuit [49] which is developed to handle high dimensional inputs well in a linear regression and minimizes the sum of coefficient in l_0-norm for regularization.
- **AE+LSTM nh:** It denotes the solar power prediction method based on a deep neural network combining autoencoder and LSTM proposed in LSTM [8]. Note that n indicates the length of sliced sequences to be fed into LSTM in hours.
- **CNN+LSTM nh:** This is our deep neural network based algorithm presented in Section IV, which combines D-CNN and LSTM for estimate the next day's solar power with the previous day's observation in a time series.

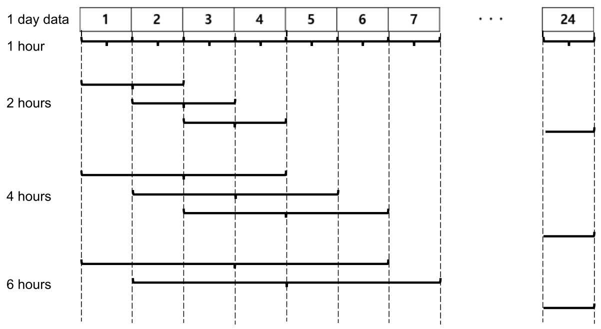


FIGURE 7. Splitting 1 day record into 1, 2, 4 and 6 hours using window slide.

TABLE 5. Performance (MAPE, RMSE, MAE, standard deviation) for algorithms with weather data.

Method	MAPE	RMSE	MAE	err. STD
LR	$> 10^9$	$> 10^9$	$> 10^7$	$> 10^7$
RFR	67.58	0.1789	0.1378	0.1141
SVR	40.03	0.1221	0.0994	0.0709
EN	98.52	0.2256	0.1835	0.1313
SGDR	39.99	0.1803	0.1348	0.1199
BR	46.77	0.1453	0.1082	0.0970
LL	98.52	0.2256	0.1835	0.1313
PAR	51.02	0.1609	0.1186	0.1089
OMP	33.23	0.1370	0.0950	0.0987
AE+LSTM 1h	22.20	0.1041	0.0661	0.0805
AE+LSTM 2h	26.70	0.1230	0.0940	0.0793
AE+LSTM 4h	23.95	0.1000	0.0736	0.0677
AE+LSTM 6h	37.59	0.1436	0.1148	0.0862
CNN+LSTM 1h	**13.42**	**0.0987**	**0.0506**	0.0848
CNN+LSTM 2h	16.49	0.1238	0.0605	0.1081
CNN+LSTM 4h	25.17	0.1644	0.0779	0.1448
CNN+LSTM 6h	37.83	0.2298	0.1334	0.1872

Since both our network using CNN and the model using auto-encoder utilizes LSTM that takes a time series sequence as input, we split the input sequence observed in a day by intervals of 1, 2, 4 and 6 hours, and each slice is fed into CNN and auto-encoder networks in CNN+LSTM and AE+LSTM respectively. Figure 7 shows how we split the sequence by 1, 2, 4 and 6 hours.

D. PERFORMANCE EVALUATION

We compute MAPE, RMSE and MAE for all algorithms to evaluate the performance and present the result in Table 5. It is clear that the traditional methods based on linear regression fail to forecast solar power better than AE+LSTM and CNN+LSTM in terms of all quality measures. With those two algorithms based on deep learning, Table 5 confirms that our proposed CNN+LSTM has achieved the best performance with all quality measures. Furthermore, the results show that CNN+LSTM best predicts the next day's solar power when we divide the input sequence of a day by 1 hour and feed them into D-CNN, rather than using longer intervals. It implies that long-time features during a day, which is expectedly detected by LSTM, has a stronger correlation on the next day's solar power than the short-time local features that might be discovered by D-CNNs.

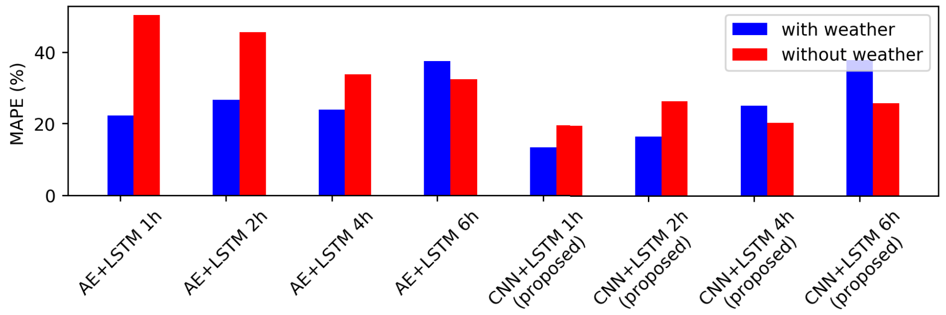


FIGURE 8. Performance (MAPE) for deep learning methods.

TABLE 6. Performance (MAPE, RMSE, MAE, standard deviation) for algorithms without weather data.

Method	MAPE	RMSE	MAE	err. STD
LR	$> 10^7$	$> 10^7$	$> 10^6$	$> 10^7$
RFR	97.87	0.1918	0.1482	0.1219
SVR	47.32	0.1260	0.0980	0.0791
EN	136.49	0.2295	0.1871	0.1329
SGDR	58.49	0.1726	0.1286	0.1153
BR	67.08	0.1522	0.1150	0.0998
LL	136.49	0.2295	0.1871	0.1329
PAR	56.94	0.1992	0.1455	0.1361
OMP	41.57	0.1375	0.0984	0.0961
AE+LSTM 1h	50.38	0.1374	0.1030	0.0910
AE+LSTM 2h	45.66	0.1417	0.1101	0.0892
AE+LSTM 4h	33.86	0.1250	0.0959	0.0803
AE+LSTM 6h	32.47	**0.1069**	0.0831	0.0673
CNN+LSTM 1h	**19.57**	0.1409	**0.0585**	0.1283
CNN+LSTM 2h	26.39	0.1182	0.0677	0.0969
CNN+LSTM 4h	20.27	0.1197	0.0635	0.1015
CNN+LSTM 6h	25.85	0.2520	0.0947	0.2337

We also test the performance of all implemented algorithms with the data obtained from PV inverters only without merging with the data obtained from the national meteorological center. In Table 6, we can see that the performance in most algorithms generally is degraded compared to that using weather data together. In Figure 8, we plotted MAPE of deep neural network based algorithms obtained with and without the weather information. As we have argued, it confirms that utilizing a coarsely estimated weather information can improve the accuracy of solar power prediction significantly even if we do not have a high-end sophisticated device.

Next, we have evaluated the performance of CNN+LSTM with each PV inverter individually. Table 7 shows three quality measures calculated with the test inputs from some selected inverters that has enough number of samples in the test data set. It reveals that the prediction for the inverter No. 2674 is unusually inaccurate; MAPE is 1692.40 % while that with the other inverters is at most 26.35 %. With a close investigation of the data from inverter No. 2674, we found that in a large number of records, power output is positive while the irradiation recorded from the inverter is zero. It is likely that irradiation sensors installed in the inverter were

TABLE 7. Prediction accuracy with data from 7 inverters.

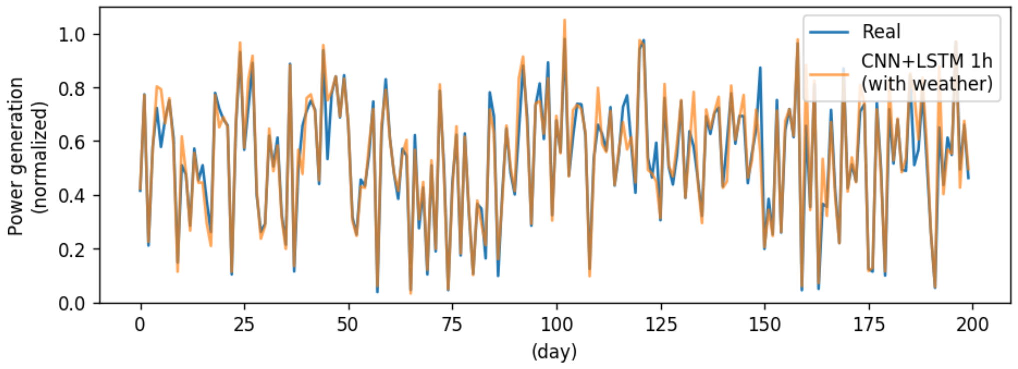
PV Inv. Num.	# of train	# of test	MAPE(%)	RMSE	MAE
2674	580	194	1692.40	0.3518	0.2689
8641	306	103	26.35	0.1423	0.0974
8902	390	131	8.42	0.0654	0.0366
9006	378	127	7.99	0.0551	0.0346
9093	385	129	6.62	0.0587	0.0304
9509	374	125	13.13	0.0662	0.0399
9813	397	133	8.72	0.0664	0.0367

damaged. However, even if there still exist outliers in the training data set due to the minimum preprocessing, we discovered that deep neural network based solar power prediction can work robustly.

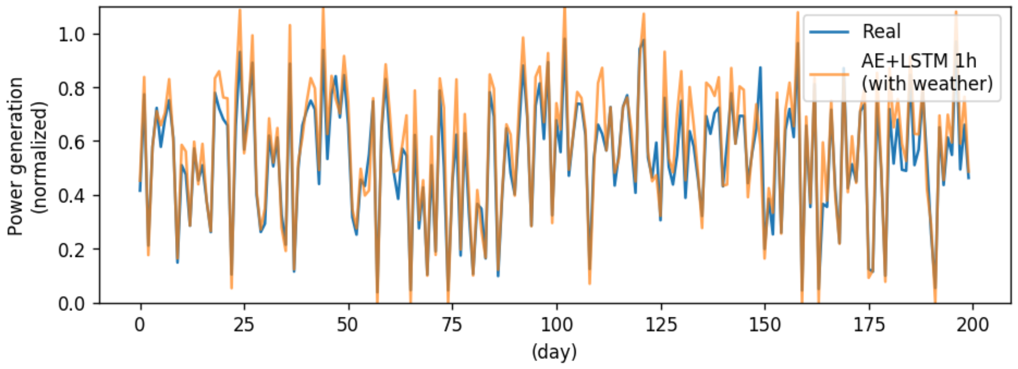
E. CASE STUDIES

To show the actual solar power output and the estimated one together, we selected a PV inverter installed in Haenam, South Korea, and plotted the power output and predicted value for 200 days in the test data set in Figures 9(a) and 9(b), which are respectively obtained using CNN+LSTM 1h and AE+LSTM 1h. In both graphs, the actual power output is drawn with a blue line, and the prediction is shown with a orange line. Plotting the prediction with the correct answer clearly shows that the prediction by CNN+LSTM is closer to the line of actual solar power output in Figure 9(a) while the prediction line of AE+LSTM often falls off from the line of actual output in Figure 9(b).

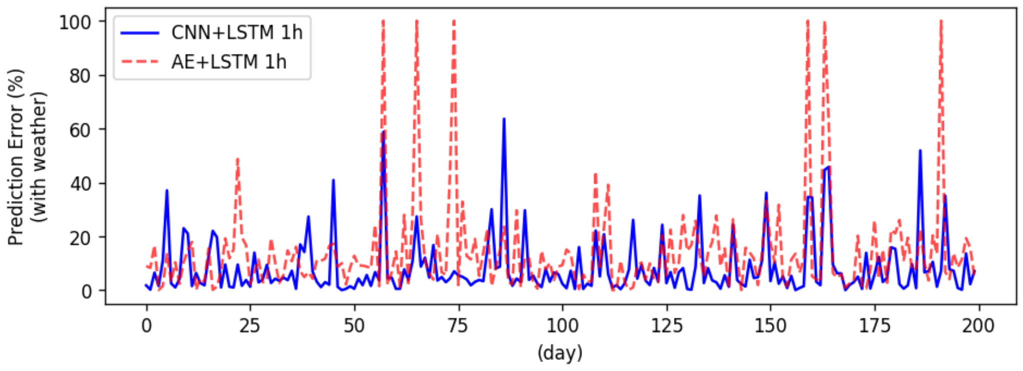
To test the relative precision of both algorithms, we also plotted the absolute prediction error in percentage (APE) (i.e., $|A - F|/A * 100$ with an actual output A and prediction F) in Figure 9(c) with the data from the same inverter. The APEs of CNN+LSTM and AE+LSTM are depicted with a blue solid line and red dashed line respectively. The result also shows that CNN+LSTM estimates the solar power more precisely than AE+LSTM. Furthermore, while AE+LSTM often predict inaccurately in the days generating a pitch (See the red dashed line at Day 74 in Figure 9(c)) when solar power is very low (See the blue solid line at Day 74 in Figure 9(b)), CNN+LSTM relatively estimates the solar power well without exceeding 60% of APE.



(a)



(b)



(c)

FIGURE 9. Estimated solar power for 200 days. (a) Proposed prediction and ground truth with weather data. (b) Prediction of AE with LSTM and ground truth with weather data. (c) Prediction errors between proposed model and AE plus LSTM with weather data.

VI. CONCLUSION

As forecasting solar power is important for solar power grid operators and energy market participants, we introduced a novel deep neural network to be trained to predict the next day's solar power using time series data collected from photovoltaic inverters and national weather centers. We combined two CNNs with filters of different sizes to extract short-time local features well and LSTMs to capture long-time features efficiently. By extensive experiments with real-life data sets, we showed that our network outperforms not only several traditional regressors but also another state-of-the-art deep neural network based solar power prediction algorithm.

REFERENCES

[1] B. Stiller, T. Bocek, F. Hecht, G. Machado, P. Racz, and M. Waldburger, "Renewable power generation costs in 2017," Int. Renew. Energy Agency, Sedalia, CO, USA, Tech. Rep. 01, 2018.

[2] T. Ehara, "Overcoming PV grid issues in urban areas," *IEA PVPS Task*, vol. 10, 2009.

[3] A. Tuohy *et al.*, "Solar forecasting: Methods, challenges, and performance," *IEEE Power Energy Mag.*, vol. 13, no. 6, pp. 50–59, Nov./Dec. 2015.

[4] D. K. Chaturvedi and I. Isha, "Solar power forecasting: A review," *Int. J. Comput. Appl.*, vol. 145, no. 6, pp. 28–50, 2016.
[5] J. Antonanzas *et al.*, "Review of photovoltaic power forecasting," *Sol. Energy*, vol. 136, pp. 78–111, Oct. 2016.
[6] U. K. Das *et al.*, "Forecasting of photovoltaic power generation and model optimization: A review," *Renew. Sustain. Energy Rev.*, vol. 81, pp. 912–928, Jan. 2018.
[7] M. Abuella and B. Chowdhury, "Solar power probabilistic forecasting by using multiple linear regression analysis," in *Proc. SoutheastCon*, 2015, pp. 1–5.
[8] A. Gensler, J. Henze, B. Sick, and N. Raabe, "Deep learning for solar power forecasting—An approach using AutoEncoder and LSTM neural networks," in *Proc. IEEE Int. Conf. Syst., Man, (SMC)*, Oct. 2016, pp. 002858–002865.
[9] A. A. Mohammed, W. Yaqub, and Z. Aung, "Probabilistic forecasting of solar power: An ensemble learning approach," in *Intelligent Decision Technologies*. Cham, Switzerland: Springer, 2015, pp. 449–458.
[10] J. Zeng and W. Qiao, "Short-term solar power prediction using a support vector machine," *Renew. Energy*, vol. 52, pp. 118–127, Apr. 2013.
[11] R. Li, H.-N. Wang, H. He, Y.-M. Cui, and Z.-L. Du, "Support vector machine combined with k-nearest neighbors for solar flare forecasting," *Chin. J. Astron. Astrophys.*, vol. 7, no. 3, p. 441, 2007.
[12] X. Qiu, L. Zhang, Y. Ren, P. N. Suganthan, and G. Amaratunga, "Ensemble deep learning for regression and time series forecasting," in *Proc. IEEE Symp. Comput. Intell. Ensemble Learn. (CIEL)*, Dec. 2014, pp. 1–6.
[13] R. Huang, T. Huang, R. Gadh, and N. Li, "Solar generation prediction using the ARMA model in a laboratory-level micro-grid," in *Proc. IEEE 3rd Int. Conf. Smart Grid Commun. (SmartGridComm)*, Nov. 2012, pp. 528–533.
[14] S. Ferrari, M. Lazzaroni, V. Piuri, L. Cristaldi, and M. Faifer, "Statistical models approach for solar radiation prediction," in *Proc. IEEE Int. Instrum. Meas. Technol. Conf. (I2MTC)*, May 2013, pp. 1734–1739.
[15] G. Reikard, "Predicting solar radiation at high resolutions: A comparison of time series forecasts," *Solar Energy*, vol. 83, no. 3, pp. 342–349, Mar. 2009.
[16] I. Colak, M. Yesilbudak, N. Genc, and R. Bayindir, "Multi-period prediction of solar radiation using ARMA and ARIMA models," in *Proc. IEEE 14th Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2015, pp. 1045–1049.
[17] M. Diagne, M. David, P. Lauret, J. Boland, and N. Schmutz, "Review of solar irradiance forecasting methods and a proposition for small-scale insular grids," *Renew. Sustain. Energy Rev.*, vol. 27, pp. 65–76, Nov. 2013.
[18] D. Anagnostos, T. Schmidt, S. Cavadias, D. Soudris, J. Poortmans, and F. Catthoor, "A method for detailed, short-term energy yield forecasting of photovoltaic installations," *Renew. Energy*, vol. 130, pp. 122–129, 2019.
[19] L. Gigoni *et al.*, "Day-ahead hourly forecasting of power generation from photovoltaic plants," *IEEE Trans. Sustain. Energy*, vol. 9, no. 2, pp. 831–842, Apr. 2018.
[20] M. Abdel-Nasser and K. Mahmoud, "Accurate photovoltaic power forecasting models using deep LSTM-RNN," *Neural Comput. Appl.*, pp. 1–14, Apr. 2017, doi: 10.1007/s00521-017-3225-z.
[21] G. Graditi, S. Ferlito, G. Adinolfi, G. M. Tina, and C. Ventura, "Energy yield estimation of thin-film photovoltaic plants by using physical approach and artificial neural networks," *Solar Energy*, vol. 130, pp. 232–243, Jun. 2016.
[22] M. Ding, L. Wang, and R. Bi, "An ann-based approach for forecasting the power output of photovoltaic system," *Procedia Environ. Sci.*, vol. 11, no. Part C, pp. 1308–1315, 2011.
[23] J. Shi, W.-J. Lee, Y. Liu, Y. Yang, and P. Wang, "Forecasting power output of photovoltaic systems based on weather classification and support vector machines," *IEEE Trans. Ind. Appl.*, vol. 48, no. 3, pp. 1064–1069, May/Jun. 2012.
[24] I. Sansa, S. Missaoui, Z. Boussada, N. M. Bellaaj, E. M. Ahmed, and M. Orabi, "Pv power forecasting using different artificial neural networks strategies," in *Proc. Int. Conf. Green Energy*, Mar. 2014, pp. 54–59.
[25] J. Liu, W. Fang, X. Zhang, and C. Yang, "An improved photovoltaic power forecasting model with the assistance of aerosol index data," *IEEE Trans. Sustain. Energy*, vol. 6, no. 2, pp. 434–442, Apr. 2015.
[26] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.
[27] S. P. Ahmad, O. Torres, P. Bhartia, G. Leptoukh, and S. Kempler, "Aerosol index from toms and OMI measurements," in *Proc. 86th AMS Annu. Meeting*, 2006. [Online]. Available: https://ams.confex.com/ams/Annual2006/techprogram/paper_104496.htm
[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[30] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, vol. 1. Cambridge, MA, USA: MIT Press, 1986, pp. 318–362.
[31] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.
[32] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556
[33] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3431–3440.
[34] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 779–788.
[35] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu, "Convolutional neural networks for time series classification," *J. Syst. Eng. Electron.*, vol. 28, no. 1, pp. 162–169, 2017.
[36] H. H. Aghdam and E. J. Heravi, *Guide to Convolutional Neural Networks: A Practical Application to Traffic-Sign Detection and Classification*. Cham, Switzerland: Springer, 2017.
[37] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 818–833.
[38] M. Arjovsky, A. Shah, and Y. Bengio, "Unitary evolution recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1120–1128.
[39] M. X. Cohen, *Analyzing Neural Time Series Data: Theory and Practice*. Cambridge, MA, USA: MIT Press, 2014.
[40] A. Supratak, H. Dong, C. Wu, and Y. Guo, "DeepSleepNet: A model for automatic sleep stage scoring based on raw single-channel EEG," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, no. 11, pp. 1998–2008, Nov. 2017.
[41] W. Lee and Y. Kim, "Interactive sleep stage labelling tool for diagnosing sleep disorder using deep learning," in *Proc. 40th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Jul. 2018, pp. 183–186.
[42] S. Ioffe and C. Szegedy. (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift." [Online]. Available: https://arxiv.org/abs/1502.03167
[43] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, Haifa, Israel, Jun. 2010, pp. 807–814.
[44] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[45] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 1997, pp. 155–161.
[46] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *J. Roy. Statist. Soc. B, Stat. Methodol.*, vol. 67, no. 2, pp. 301–320, 2005.
[47] D. J. C. MacKay, "Bayesian interpolation," *Neural Comput.*, vol. 4, no. 3, pp. 415–447, 1992.
[48] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive-aggressive algorithms," *J. Mach. Learn. Res.*, vol. 7, pp. 551–585, Dec. 2006.
[49] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, 1993.

WOONGHEE LEE received the B.S. degree from the Department of Industrial and Management Engineering, Hanyang University at ERICA in 2015, where he is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering. His research interests include data mining and machine learning.

KEONWOO KIM received the B.Sc. degree from the Department of Computer Science and Engineering, Hanyang University, Ansan, South Korea, in 2015, and the M.Sc. degree with the Department of Computer Science and Engineering, Hanyang University, Seoul, South Korea, in 2017, where he is currently pursuing the Ph.D. degree.

JUNSEP PARK received the B.S degree in electrical and communication engineering from Hanyang University, Ansan, South Korea, in 2013. He is currently pursuing the M.S. degree in computer science and engineering with Hanyang University, Seoul, South Korea. He has studied in collecting and classifying topics spreading over social networks.

JINHEE KIM received the B.S. degree in computer engineering from the National Mokpo Maritime University of Korea. She is currently pursuing the M.S. degree in computer science and engineering with Hanyang University, Seoul, South Korea. She has worked in the recent topics, including deep learning for image classification in online shops.

YOUNGHOON KIM received the B.S. degree in computer science and engineering and the Ph.D. degree from Seoul National University in 2006 and 2013, respectively. He is currently an Assistant Professor at Hanyang University at ERICA, South Korea. He was a Visiting Scholar with the University of Illinois at Urbana–Champaign in 2014 invited by Prof. J. Han. His research interests include machine learning with stochastic modeling, substring query processing in database, and social network analysis focusing on text mining. He has presented his work in many top conferences on Computer Science known for their impact history and their rigorous review process, such as ICDE (2010, 2012), WWW (2013), SIGMOD (2013), and KDD (2015).