ISA Transactions xxx (xxxx) xxx

Contents lists available at ScienceDirect

ISA Transactions

journal homepage: www.elsevier.com/locate/isatrans

Research article

A novel recurrent neural network approach in forecasting short term solar irradiance

Mustafa Jaihuni, Jayanta Kumar Basak, Fawad Khan, Frank Gyan Okyere, Thavisak Sihalath, Anil Bhujel, Jihoon Park, Deog Hyun Lee, Hyeon Tae Kim *

Department of Bio-systems Engineering, Gyeongsang National University (Institute of Smart Farm), Jinju 52828, Republic of Korea

ARTICLE INFO

Article history:
Received 6 August 2019
Received in revised form 30 January 2021
Accepted 26 March 2021
Available online xxxx

Keywords:
Unidirectional
Bidirectional
LSTM
GRU
Solar irradiance
Short term forecasting

ABSTRACT

Forecasting solar irradiance is of utmost importance in supplying renewable energy efficiently and timely. This paper aims to experiment five variants of recurrent neural networks (RNN), and develop effective and reliable 5-minute short term solar irradiance prediction models. The 5 RNN classes are long–short term memory (LSTM), gated recurrent unit (GRU), Simple RNN, bidirectional LSTM (Bi-LSTM), and bidirectional GRU (Bi-GRU); the first 3 classes are unidirectional and the last two are bidirectional RNN models. The 26 months data under consideration, exhibits extremely volatile weather conditions in Jinju city, South Korea. Therefore, after different experimental processes, 5 hyper-parameters were selected for each model cautiously. In each model, different levels of depth and width were tested; moreover, a 9-fold cross validation was applied to distinguish them against high variability in the seasonal time-series dataset. Generally the deeper architectures of the aforementioned models had significant outcomes; meanwhile, the Bi-LSTM and Bi-GRU provided more accurate predictions as compared to the unidirectional ones. The Bi-GRU model provided the lowest RMSE and highest R^2 values of 46.1 and 0.958; additionally, it required $5.25*10^{-5}$ seconds per trainable parameter per epoch, the lowest incurred computational cost among the mentioned models. All 5 models performed differently over the four seasons in the 9-fold cross validation test. On average, the bidirectional RNNs and the simple RNN model showed high robustness with less data and high temporal data variability; although, the stronger architectures of the bidirectional models, deems their results more reliable.

1. Introduction

The global population rise, climate change calamities, limited existing fossil fuels, and the ever increasing need for food production, comprise some of the main factors in an unprecedented demand for solar renewable energy (SRE) [1]. On the other hand, abundant solar energy received on earth, outnumbers other types of energy sources, almost by a factor of 2700 [2]; thus, making SRE an attractive, direct and indirect, substantial solution to many imminent crisis lurking over the horizon. On the other hand, cost reduction in solar panel technologies and governmental pro-solar energy policies have sped up the solar power integration into smart-grids and micro-grids, [3,4]. Hence, in many countries numerous mainstream applications of SRE are found in residential, commercial and industrial levels [5].

Discussion of contemporary works

SRE generation is irregular due to the intermittency and uncertainty in daily solar radiation levels, seasonal weather changes, and the diurnal–nocturnal cycle; it poses a big difficulty in energy supply scheduling, planning and operations [3,6]. Therefore, several forecasting methodologies and models, considering the above mentioned problems, have been developed for accurate solar irradiance prediction [2,6–8]. These prediction models can be classified into statistical, machine learning, empirical, and physical models [9,10]. Generally, machine learning forecasting models, such as neural networks, have outperformed other aforementioned methods [8,11,12]. Therefore, in this paper artificial neural network (ANN) forecasting models, related to solar irradiance, were reviewed.

Increasingly over the last years, many types of ANN models have been developed to predict solar irradiance. In the works of [12], hourly day-ahead LSTM-RNN and back-propagation neural network (BPNN) solar irradiance forecasting models were developed. As a result of different datasets used for training, the developed LSTM model was very robust, predicting a moderate RMSE level of 60.31 W m^{-2}. The writers in [13], also trained LSTM and feed forward neural network (FFNN) models by performing three experiments with more than 11 years of weather

* Corresponding author.
E-mail addresses: mjaihu@gmail.com (M. Jaihuni), basak.jkb@gmail.com (J.K. Basak), fawad.pid@gmail.com (F. Khan), okyerefrank200@gmail.com (F.G. Okyere), max7set@gmail.com (T. Sihalath), anil.bhujel@gmail.com (A. Bhujel), qkrwlgndl@naver.com (J. Park), elqr134@naver.com (D.H. Lee), bioani@gnu.ac.kr (H.T. Kim).

https://doi.org/10.1016/j.isatra.2021.03.043
0019-0578/

and solar irradiance data. An RMSE value of 76.245 W m^{-2} was obtained from the LSTM model, outperforming the latter model. In [14], LSTM and FFNN models are thoroughly studied and implemented. Their studies resulted in a more resilient LSTM and FFNN models that were validated to predict solar irradiance in various regions. The RMSE values between 23.60–37.78 W m^{-2} of the LSTM model were better than FFNN. The authors in [15], studied Elman neural network (ENN), radial basis function neural network (RBFNN), and BPNN under various seasons in a year. The study sheds light upon the volatility of solar radiation levels throughout different seasons. On average, the ENN model had an 8% mean absolute percentage error (MAPE), the lowest average error among the three models. In the works of [16], wavelet neural network (WNN) and ANN models were studied, in which, WNN had been better performant in modeling the nonlinearity in the one year exogenous data, as compared to other ANN models. In order to build a robust model, the data was divided into six segments, resulting in effective cross validation of the models. Meanwhile, [11] had an interesting approach by dividing weather data according to cloud characteristics, like optical depth or cloud dissipation rate. An ANN (FFNN) model was developed in two different ways, one with k-means clustering and one without. The first approach gave more accurate forecasts, with mean absolute error (MAE) of 0.08, as compared to the second one. Moreover, it had a specific ANN model built for each cloud type separately, capturing more abstract features and characteristics of it.

In general, most of the above mentioned works are devoid of the application of different RNN classes, such as Bi-LSTM or GRU. These variants are stronger than simple RNN models [17]. In various works such as [9,11,13–15,18], the developed models were shallower in deepness, because they consisted few neurons or limited to one hidden layer. ANN models with variable depth and width, may have resulted in better performance in predictions [19]. Nevertheless, such simple models would be weaker in capturing more abstract features of time series data, as neural networks are supposed to. On the other hand, in the works of [12,15] using only 3 variables as input decreased the reliability of the model, as the predictand may have been affected by more than 3 weather variables.

The application of RNN models in forecasting solar irradiance, is relatively new subject. As mentioned above, RNN has shown comparable results in modeling time-series data. This paper has targeted in testing the strength of 5 different types of RNN model in 5-minute short-term forecasting of solar irradiance, namely Simple RNN, LSTM, GRU, Bidirectional LSTM, and Bidirectional GRU. To the best of our knowledge, it is the first time such an approach is undertaken by developing 5 different variations of RNNs. It is the first application of Bi-LSTM and Bi-GRU to forecast solar irradiance. In order to develop the models, trial and error processes were carried out to select optimal hyper-parameters. These networks' different levels of architectural depth and width were tested, as well. The objectives were to determine the best variant of RNN model in 5 min short-term solar irradiance predictions; consequently, selecting a suitable architecture, i.e. shallow or deep RNN model. Additionally, each model was experimented by a k-fold cross validation; further aiming to establish each model's robustness against high seasonal variability. Lastly, it was intended to find out a computationally efficient model among the RNN variants as well. Hence, we intend to introduce two novel statistical ratios to categorize deep learning models based on their incurred temporal costs while training, hyper-parameters, and basic architectures.

The rest of the paper is arranged as follows: In Section 2, RNN concepts are detailed, as well as data engineering and experimental methodologies are explained. Section 3 contains a thorough discussion and analysis of experimental results and findings. Section 4 provides the conclusion part and final remarks about this paper.

Structure of paper

2. Material and methodology

2.1. Formulation of RNN models

RNN provides a vast plethora of tools for time series problems, speech recognition, sentiment analysis etc. Simple RNN, LSTM and GRU are some types of unidirectional RNN models; while, bidirectional LSTM and Bidirectional GRU are different types of the bidirectional RNN [20,21].

2.1.1. Simple RNN model

A simple RNN is a recursive model with an input sequence, x, which tries to predict a state, s_t at time t, considering prior state s_{t-1}, by applying a differentiable function, f. S_{t-1}, does not carry information solely from the previous time step, but can be thought as an extract of all the past states. As illustrated in Fig. 1.a and b, in an RNN architecture the weight parameters, here denoted as U, V, and W, are shared across all layers. Their corresponding relationships are depicted in formulas (1)–(2) [19].

$$s_t = s_{t-1} * W + x_t * U \tag{1}$$

$$y_t = s_t * V \tag{2}$$

Simple RNN formula

2.1.1.1. Back propagation through time. An RNN model is trained as far as to minimize a loss function, the error in the predictions as compared to the actual values. In training, firstly, the RNN model is unfolded into recurrent steps, as shown in Fig. 1b. Then, $\frac{\partial J}{\partial s}$ is calculated; it is the gradient of the loss function J, with respect to the output state at time t. The calculated gradient is propagated backward multiple time steps throughout the network. The following formulas depict these repeating relationships and the accumulated gradients of parameters, respectively [19]:

$$\frac{\partial J}{\partial s_{t-1}} = \frac{\partial J}{\partial s_t} W \tag{3}$$

$$\frac{\partial J}{\partial U} = \sum_{t=0}^{n} \frac{\partial J}{\partial s_t} x_t \tag{4}$$

$$\frac{\partial J}{\partial W} = \sum_{t=0}^{n} \frac{\partial J}{\partial s_t} s_{t-1} \tag{5}$$

2.1.2. Challenges in training an RNN

Vanishing and exploding gradients are two big challenges in training a time-layered RNN model. In the case of a deep RNN, in which the length of sequence can be high, when propagating the gradients backward through time, if $|W| > 1$, successive multiplications of weight parameters may increase gradients exponentially and lead to the so-called exploding gradients. While, for values of $|W| < 1$, in the same way, vanishing gradients problem occurs. Both cases are depicted in the following formula [20,21]:

$$\frac{\partial s_t}{\partial s_{t-m}} = \frac{\frac{\partial s_t}{\partial s_{t-1}} * \cdots * \partial s_{t-m+1}}{\partial s_{t-m}} = W^m \tag{6}$$

There are various solutions for vanishing and exploding gradients, like dropout, parameter regularization or layer normalization. Additionally, an effective solution for the vanishing problem, is by adding internal memory cells or gated architecture to RNN model, i.e. LSTM and GRU.

2.1.3. Gated RNNs: LSTM and GRU

GRU and LSTM are different strong variants of RNN, and have shown strong performances in learning long term dependencies [22]. As shown in Fig. 2, an LSTM memory cell can store information as long as the input/write gate is closed, it will be changed by new information, only when it is open. [23,24]. In

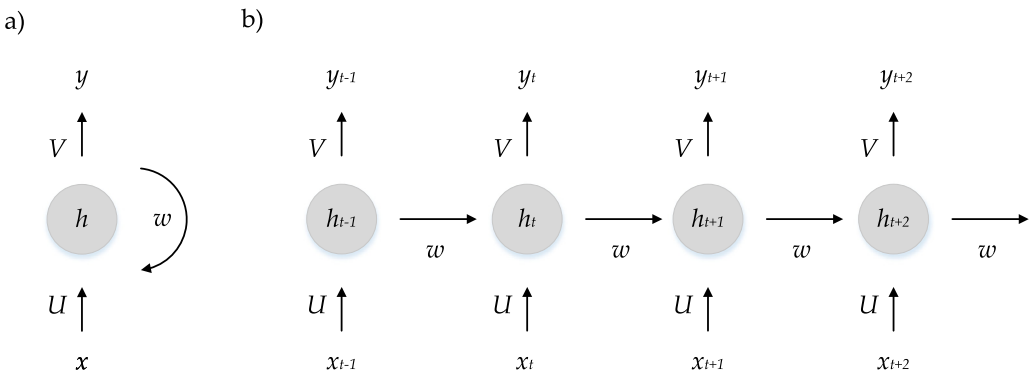

Fig. 1. (a) RNN unit (b) Unfolded RNN unit showing the different time-steps state.

other words, the gates depend on the input x at time t and output y at time $t-1$. Stored information from previous state $h[t-1]$ is updated by the *forget gate*; while, the *input gate* decides to what extent state h at time t would be amended with a new state $\tilde{h}[t]$ [20].

Equations for RNN LSTM & GRU

$$forget\ gate: \sigma_{ft} = \sigma(W_f x_t + R_f y_{t-1} + b_f) \quad (7)$$

$$candidate\ state: \tilde{h}_t = g_1(W_h x_t + R_h y_{t-1} + b_h) \quad (8)$$

$$update\ gate: \sigma_{ut} = \sigma\ (W_u x_t + R_u y_{t-1} + b_u) \quad (9)$$

$$cell\ state: h_t = \sigma_{ut} \odot \tilde{h}_t + \sigma_{ft} \odot h_{t-1} \quad (10)$$

$$output\ gate: \sigma_{ot} = \sigma\ (W_o x_t + R_o y_{t-1} + b_o) \quad (11)$$

$$output: y_t = \sigma_{ot} \odot g_2\ (h_t) \quad (12)$$

$$reset\ gate: r_t = \sigma\ (W_r y_t + R_r x_t + b_r) \quad (13)$$

$$current\ state: \acute{h}_t = h_{t-1} \odot r\ [t] \quad (14)$$

$$candidate\ state: z_t = g\left(W_z h\,\acute{[}t] + R_z x_t + b_z\right) \quad (15)$$

$$update\ gate: u_t = \sigma\ (W_u h_{t-1} + R_u x_t + b_u) \quad (16)$$

$$new\ state: h_t = (1-u_t) \odot h_{t-1} + u_t \odot z_t \quad (17)$$

LSTM specific

The relationships of the parameters in Fig. 2a, are shown in the formulas (7) to (12). In these formulas, b_f, b_h, b_u, b_o are the bias vectors of each gate, while $W_f, W_h, W_u, W_o, R_f, R_h, R_u, R_o$ are weight parameters' matrices. Moreover, g_1 *and* g_2 are nonlinear activation functions and $\sigma_f, \sigma_u, \sigma$ are sigmoid functions.

On the other hand, GRU is a simpler, though strong, version of LSTM; in GRU, a new gating mechanism with less parameters were introduced in [25]. Computationally, GRU converges faster, giving almost similar results as LSTM. As shown below in Fig. 2b, a GRU has two gates, for resetting and updating, as compared to 3 gates in LSTM. The formulas (13) to (17) are listed above and the architecture is shown in Fig. 2b [20]. Here, b_f, b_z, b_u represents the bias feeds in the network, while $W_r, W_z, W_u, R_r, R_z, R_u$ are weight parameters' matrices. There is no consensus on preferring GRU over LSTM or vice versa. However, LSTM is a more tested model and more related material can be found. While, comparatively, GRU is a new tool and the corresponding papers are fewer. LSTM has a general architecture but GRU's architecture is simpler. However, GRU can be preferred when there are computational concerns and time limitations for building a model.

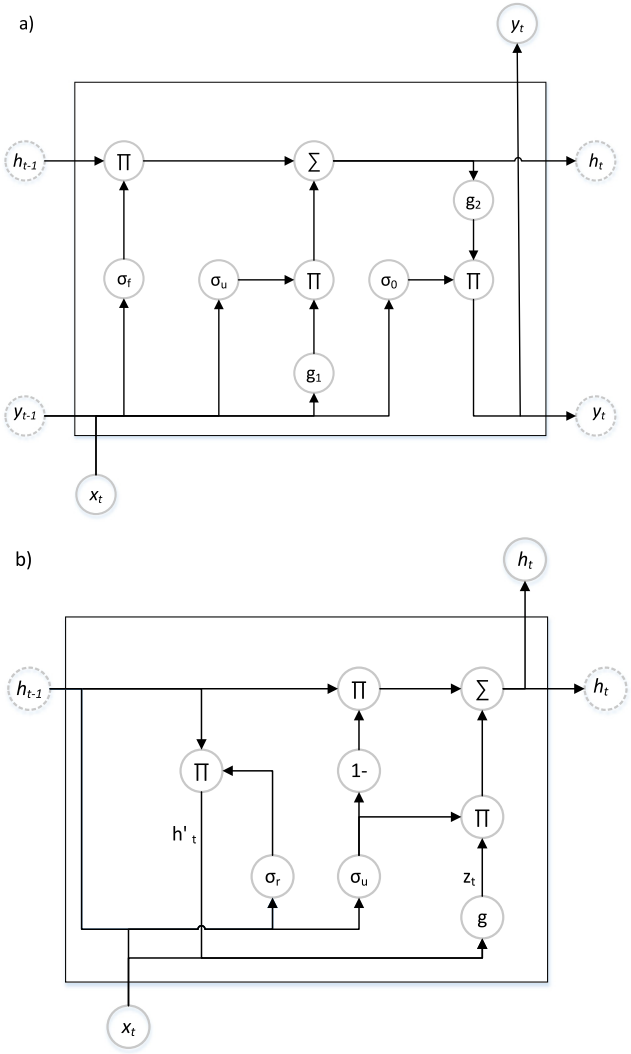

Fig. 2. (a) An LSTM cell (b) A GRU cell.

2.1.4. Bidirectional LSTM and GRU

If there are entire sequences available at the time of training, Bidirectional RNNs (BRNN) can be developed [22]. In some applications this method can give better results [26]. Moreover, for applications with high variability and less causal relationship, for example weather forecasts, bidirectional RNNs can be effective [21]. A BRNN has two separate recurrent hidden layers connected to same input and output, each of which are fed with forwards and backwards training sequences, as depicted in Fig. 3a. Hence, while training, for every prediction, future and past information are provided, simultaneously [24]. This process is quite similar to a unidirectional RNN, except that they are performed twice; once from the start to the end and second time from the end to the starting sequence, shown in Fig. 3b. The hidden states are separately computed twice for backwards and

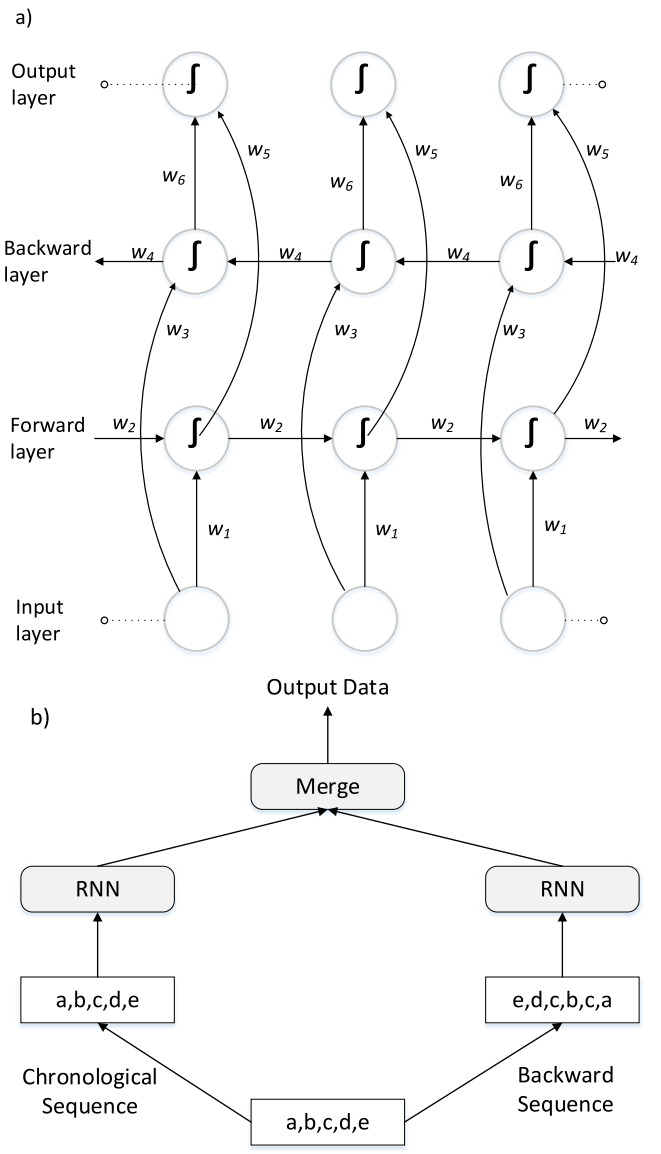

Fig. 3. (a) Bidirectional processes with individual weights, w_i (b) Two separate RNN processes in a BRNN (sample sequence *a,b,c,d,e*).

forwards direction; hence, it increases computational burden. But, for the sake of higher accuracy, the mentioned extra burden can be accepted.

2.2. Experimental designs

Five different variants of RNN, namely Simple-RNN, LSTM, GRU, Bi-LSTM, and Bi-GRU, will be trained and tested. Different architectural hyper-parameters will be used to find optimal architectures for the solar radiation forecasts. Various structures, with different deepness and width, will be tried. Moreover, the strength and robustness of these models will be experimented by applying a k-fold cross validation. As time-series weather data are quite variable over the course of seasons, it will be seen how these models will learn such a characteristic and provide predictions, accordingly. All of these tests will be carried out in Python and Keras library [14].

2.2.1. Selection of transfer function

Transfer functions are used to determine the output of neurons. They are one of the most crucial hyper-parameters in any neural network. In this paper, five transfer functions will be tested on each model; namely they are linear, tanh, sigmoid, relu, and sofmax activation functions. The sigmoid function is useful for a normalized data and it is easily interpretable as a probability measure. The data used in this paper, is normalized between 0 and 1, and solar irradiance has sinusoidal type variability. It is expected that sigmoid and tanh transfer functions might be suitable for a given RNN architecture in this paper. The formulas for each of the above mentioned functions are stated below [22].

$$Tanh: \psi(x) = \frac{\sinh x}{\cosh x} \tag{18}$$

$$Sigmoid: \psi(x) = \frac{1}{1+e^{-x}} \tag{19}$$

$$ReLU: \psi(x) = \max(0, x) \tag{20}$$

$$Linear: \psi(x) = x \tag{21}$$

$$Softmax: \psi(x) = \frac{1}{\sum e^x} e^x \tag{22}$$

2.2.2. Finding the optimal optimizers

An optimizer significantly affects the learning speed and updating of a model, hence helps in converging or exploding of a neural network [21]. Hence, it is important to select optimal optimizers for every model. Optimizers are useful in finding the combination of weight values that gives a minimum loss function value [19]. In this paper five optimizers are tested in each RNN model; each of them have different approach in handling the partial derivatives for updating model weights. These optimizers are adam, adadelta, adagrad, sgd, and rmsprop.

2.2.3. Depth and Width determination

The prediction accuracy of a neural network model differs as the network architecture is changed with respect to the number of neurons in a layer, the width, and number of layers, the depth [27]. In this paper, shallow and deep architectures of the above mentioned RNN models are tested. We expect that as the RNN model goes deeper, the accuracy and reliability of predictions would rise; as with a deeper RNN model more abstract features of data can be modeled.

2.2.4. Setting a value for Epochs

In this part, suitable number of epochs that would give better convergence would be determined. This hyper-parameter shows the number of times all the training data is used for training a model, iteratively [28]; after each iteration the learning rate are changed as far as a convergence is obtained [29]. We would examine 100, 150, 200, 250, and 300 epochs on each RNN model.

2.2.5. K-fold Cross validation

As explained, the time series data shows high seasonal variability, we expect that a single RNN model will not provide better predictions in all of the seasons. Therefore, K-fold cross validation is an effective technique to find out the robustness of models for each season and avoid over-fitting or under-fitting in the models [29]. Hence, data has been divided into seasons and applied on each model will be applied. It will be seen which RNN variants shows robustness against high seasonal variability. The 26 months data coincides with 3 times the spring season and 2 times each of the remaining seasons. Hence, a total of 9 fold cross validation will be applied, and the behaviors of each model will be experimented.

2.2.6. Computational costs

The number of trainable parameters increases in each model as the number of neurons, hidden layers, outputs and inputs changed, as shown in formulas (2) and (3). Consequently, as deeper networks are applied, it is expected that computational costs will go higher as well [21]. The configuration of each model will be developed as stated in Sections 2.2.1 to 2.2.5. As a result, at the last stage the training time for the final state of each model would be recorded.

We think that computational costs would not be solely determined by comparing the total training time of each model; it is required to consider other inclusive factors as well. Here, we propose two statistical ratios which are comprised of three factors such as the total consumed time in training, trainable parameters, and epoch, each of which encompasses different aspects of deep learning models. For instance, the total training time is directly affected by the selected hyper parameters like activation function and optimizer, the computer processor capacity, and the dataset size [21,22]. Meanwhile, the trainable parameter quantity demonstrates the complexity of the models and is a firm indication of the number of neurons, hidden layers and basic structure of each one [24,27]. On the other hand, the epoch hyper-parameter shows a deep learning model's convergence rate and is closely related to the training dataset size [28,29]. Therefore, in order to statistically determine the efficiency of models the following novel ratios are introduced in this paper.

1. Ratio of Training Time consumed per trainable parameter
2. Ratio of Training Time consumed per trainable parameter per epoch

These ratios demonstrate different structural, hyper-parametric, and data specific aspects of deep learning models and can provide practical information when deciding on a computationally efficient deep learning model.

Meanwhile the computer specifications used for running the Python software is as follows:

- Windows 10 Pro
- Processor: AMD Ryzen 3 2200G with Radeon Vega Graphics 3.50 GHz
- RAM: 4.00 GB
- 64-bit Operating System

2.2.7. Performance metrics

Root mean square error (RMSE) and R squared (R^2) metrics will be used to calculate model accuracies and error in predictions. The former metric shows high sensitivity to bigger errors while it is more tolerant towards small deviations. On the other hand, the latter one demonstrates how much a model is near to the predicted non-linear function. In other words, the bias and variance in the predictions are depicted by these two metrics, respectively [30]. While both metrics are important, the first metric would be focused more in this paper, as for solar irradiance applications, the variances in the predictions are important to be low for effective energy scheduling [3]. The formulas for both metrics are given below.

$$\boldsymbol{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_{t\_actual} - y_{t\_predicted})^2} \tag{23}$$

$$\boldsymbol{R^2} = 1 - \frac{\sum_{t=1}^{n}(y_{t\_actual} - y_{t\_predicted})}{\sum_{t=1}^{n}(y_{t\_actual} - y_{t\_mean})} \tag{24}$$

2.3. Data engineering and experimental design

The effectiveness of any neural network is highly correlated to the availability of abundant data, as well as computational resources [31]. Fortunately, computational resources are not a luxury anymore; developments in chip technologies, have equipped every lab with highly capable computers. On the other hand, data is the oil of neural networks; hence, efficient data preparation is very critical for obtaining reliable results.

2.3.1. Data preparation

In our experiments, exogenous weather data and endogenous solar irradiance data are used. The data was obtained from a weather station, depicted in Fig. 4, located in the Gyeongsang national University, Jinju city, South Korea. The variables included are solar irradiance, outside temperature, relative humidity, rain amount, and wind speed and direction. The collected data, averaged over five minute intervals, covered a total of 26 months records, from March 2017 to April 2019. The mentioned data was reduced to instances captured from 06:00 to 18:00, since solar irradiance values are mostly non-zero during this period of any day in a year.

2.3.2. The weather station

The weather station (Campbell Scientific, Inc., Logan, USA), was installed at the 35°9′6.7″N, 128°5′44.52″E location, at a 44 m height from the sea level. The global solar radiation and direct solar radiation were measured by the pyrometer (LI200X, LI-COR, LI-COR, Lincoln, USA) and sunshine duration sensors (CSD3, Kipp & Zones, Delft, Netherlands), respectively. While the amount of convertible solar energy was calculated and measured from the solar panel. On the other hand, the weather sensors (Vantaa, VI-ASALA, and WXT520, Finland) recorded outside air temperature, relative humidity, the speed and direction of wind, respectively. The data logger recorded the data from sensors in 5 min intervals.

2.3.3. Data preprocessing

Raw data gathered from sensors may contain abnormalities, such as outliers or missing values. Such instances of data may come from lapses of recording in electronic device or energy discontinuity may result service disruption. Data needs to be cleaned off them before applying on neural networks, as such values may affect the final results. Hence, the data was checked for such abnormalities, and few outliers and missing values were found. The outliers were amended to the average values of the respective variables or those records were completely deleted. Moreover, the missing values were treated by values of the previous or following records' related values. In the end, there were 114 621 records ready for the experiments. Table 2 shows the general descriptions of the mentioned data. For example, the distribution of percentile values in every variable are increasing normally. This table is important in showing the overall health of the data (see Table 1).

2.3.4. Treating seasonality in data

The weather variables have seasonal characteristics, which may cause problems like local optima selection, while training a model. As seen in Fig. 6, the solar irradiance, air temperature, relative humidity and wind speed show seasonal variations. Hence, it is important to remove seasonality from data. Normalization technique is one of the solutions for seasonality, which has been applied on solar irradiance data. The data was normalized between (0, 1) and then applied on training and testing the neural networks.

Fig. 4. The weather stations and various components depicted in this picture.

Table 1
Total data various descriptions, showing how normal are the values. Dataset description

	Solar irradiance (W m^{-2})	Air temperature (°C)	Rain amount (mm)	Wind direction (°)	Wind speed (ms^{-1})	Relative humidity (%)
Count	114621	114621	114621	114621	114621	114621
Mean	325.55	16.27	0.01	191.00	0.95	55.69
Standard deviation	263.99	10.34	0.08	65.72	0.51	23.52
Minimum	0.00	−13.50	0.00	21.70	0.11	7.55
25%	83.80	8.28	0.00	136.30	0.56	36.24
50%	277.70	16.87	0.00	200.00	0.85	53.28
75%	523.40	24.45	0.00	245.30	1.23	75.87
Maximum	1172.00	38.93	5.20	342.00	4.67	96.90

Table 2
Correlation ratios of weather factors with solar irradiance.

	Solar irradiance
Air temperature	0.415
Rain amount	−0.084
Wind direction	−0.065
Wind speed	0.403
Relative humidity	−0.537

2.3.5. Correlation analysis

The correlation between 5 variables and solar irradiance were calculated, shown in Table 2. As seen, solar irradiance has the highest positive correlation with outside air temperature and the highest negative correlation with relative humidity. While, the wind direction and rain amount have little negative correlation with solar irradiance absorption by the solar panels. Nevertheless, all of these variables are included in the analysis (see Fig. 5).

2.3.6. Seasonal deviations of the target variables

As depicted in the previous section, the target variables have mixed correlations. Moreover, it is important to find out these factors' monthly and seasonal trends, which directly affect solar radiation predictions. As seen in Fig. 6, these variables have different drifts over the four seasons. Therefore, solar irradiation forecasts would have different accuracy levels in each season. For instance, the relative humidity level is high during summers with a very low standard deviation, while, during the same season, the rain amount has very high deviations. Henceforth, it would be harder for the models to learn, and eventually, they would have opposing effect on solar irradiance forecasts. We expect that a single model will not be able to learn and predict with high reliability over the course of four seasons.

3. Results and discussions

3.1. Developing the RNN variants

Extensive trial and error processes were followed to select optimal hyper-parameters for individual RNN models. While experimenting on a target hyper-parameter, the remaining hyper-parameters were kept constant. Henceforth, the 5 variants of RNN models were built step by step, in order obtain reliable models for 5-minute solar irradiance forecasts. During the experiments many interesting results have been obtained, as well as some difficulties were faced. In this part, the implementation and results of these experiments would be discussed in detail.

3.1.1. Transfer function selection

As mentioned, 5 different transfer functions have been tested on all of the 5 RNN models for predicting the solar irradiance. Hence, the transfer functions giving the best prediction accuracies were selected for each model. The list of constant and variable hyper-parameters are given in Table 3b. As depicted in the below

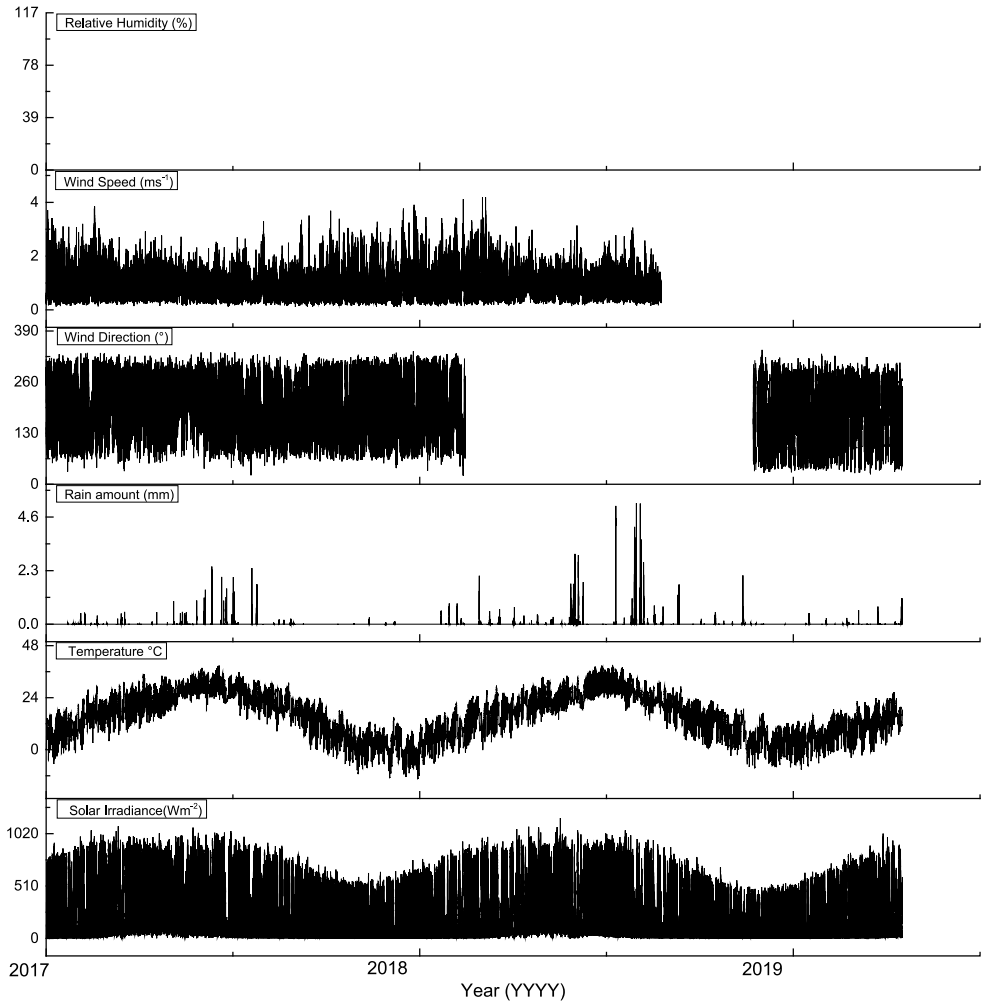

Fig. 5. Seasonal trends of weather and daily solar irradiance variables.

tables, the results are in favor of the more new variants of RNN. The bidirectional GRU has outperformed all of the other models, as listed in Table 3a; while, the most suitable transfer functions for individual models are listed in Table 3c. Surprisingly, the linear transfer function has given better results with 2 models. It might stem from the property of normalized time-series data that can give results very near to 0 or 1, in which case linear passing does not skew the inputs, hence developing higher accuracies in Bi-GRU and Simple RNN models. Meanwhile, as expected the application of sigmoid activation function has been effective too. It confirms the fact that sigmoid functions are effective with normalized data between 0 and 1, and with neural networks trained by back-propagation method [32,33].

3.1.2. Optimum optimizer

In this section, all the hyper-parameters listed in Table 3b, were constant, except the optimizers. While searching for suitable optimizers for each model, remarkable results were obtained. Bidirectional GRU has again surpassed all other models in accuracy metrics, as seen in Table 7. Meanwhile, the adam optimizer has been more convergent in training almost all of the models. It confirms the works of [32], as it has shown better results with huge temporal data and outperformed RMSProp and AdaGrad optimizers in applications (see Table 4).

3.1.3. Determining depth and width of the models

It was very crucial to determine an optimum level of depth and width in neural networks. In this paper, it has been avoided to rely on shallower networks, which are inclined to over-fitting, as well as computationally exhaustive deeper ones. Over numerous iterations, significant improvements have been achieved as well as suitable configurations were selected. Here, all the hyper-parameters were constant, except the number of hidden layers and neurons. As seen in Table 5, increasing the number of neurons and layers have had mixed results; firstly it had adverse effect on RMSE but then has changed the opposite way. Overall, the top five accurate values for individual models have come with higher configurations. As expected and in alignment with the works of [30], with deeper RNN models more abstract features of the time series model can be learnt, hence providing more reliable and better predictions. Additionally, the bidirectional GRU and LSTM have provided highest accuracies in short term solar irradiance forecasts.

3.1.4. Epoch levels

The number of epochs tested were selected between 100 and 300. Epochs are closely related to the learning speed of each model [21]; hence, as seen in Table 6b, each RNN model has shown different learning rates. Bidirectional GRU has shown slower convergence and given the best result with a comparatively higher epoch number, while bidirectional LSTM has shown the fastest learning rate with a lower epoch iterations. Therefore, one may be tempted to select a Bi-LSTM model due to its higher convergence rate, while Bi-GRU may be chosen for higher accuracies of short term solar irradiance forecasts.

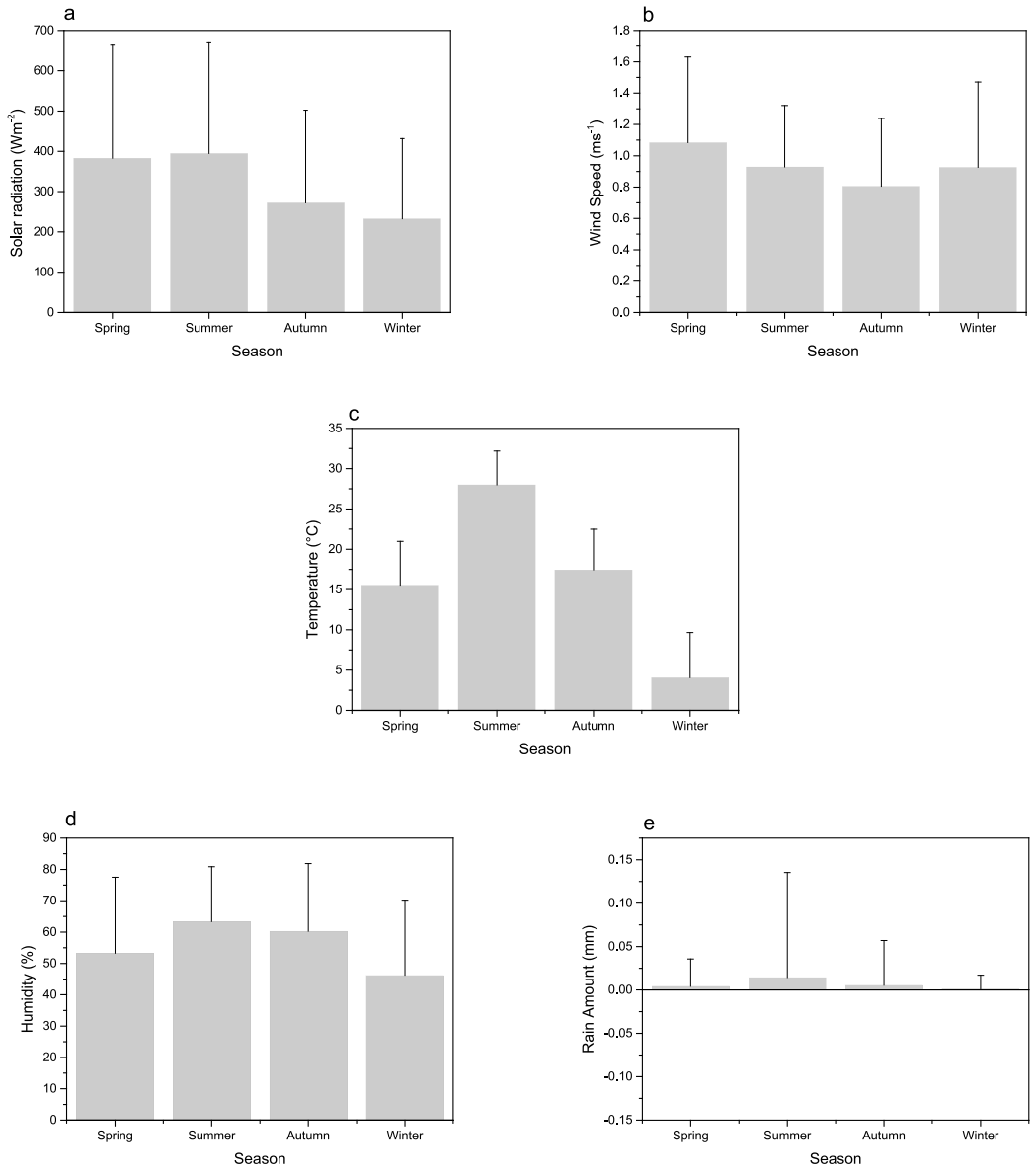

Fig. 6. Seasonal trends of the weather variables and the corresponding standard deviations.

Table 3
Transfer Functions selection process.

(a) The highest 5 accuracies				(b) Constant and Variable parameters		(c) The transfer function for each model			
Network type	Transfer function	R^2	RMSE (W m^{-2})	Hyper-parameters	Testing-Constant	Network type	Transfer function	R^2	RMSE (W m^{-2})
Bi-GRU	Linear	0.9573	46.3278	Transfer function	Testing	Bi-GRU	Linear	0.9573	46.3278
Bi-GRU	Tanh	0.9569	46.5208	Hidden layers	Constants	LSTM	Sigmoid	0.9564	46.8179
LSTM	Sigmoid	0.9564	46.8179	# Neurons	Constants	Bi-LSTM	Sigmoid	0.9563	46.8893
Bi-LSTM	Sigmoid	0.9563	46.8893	Optimizer	Constants	Simple RNN	Linear	0.9561	46.9608
Elman	Linear	0.9561	46.9608	Epochs	Constants	GRU	Sigmoid	0.9557	47.1724

Table 4
Finding the best optimizers.

(a) The highest 5 accuracies				(b) Optimal optimizer for each model			
Network type	Optimizer	R^2	RMSE (W m^{-2})	Network type	Optimizer	R^2	RMSE (W m^{-2})
Bi-GRU	Adam	0.9573	46.3278	Bi-GRU	Adam	0.9573	46.3278
Bi-GRU	Adam	0.9569	46.5208	Bi-LSTM	Adam	0.9563	46.8893
Bi-GRU	SGD	0.9569	46.5302	Elman	Adagrad	0.9566	46.7119
Bi-GRU	Adadelta	0.9567	46.6671	GRU	Adam	0.9557	47.1724
Bi-GRU	Adagrad	0.9567	46.6721	LSTM	Adam	0.9564	46.8179

Table 5
Testing results of each RNN model with different number of neurons and hidden layers; values in bold demonstrate the best predictions of the corresponding model and configuration.

Network	R^2	RMSE (W m^{-2})	Total neurons	Total hidden layers	Neuron distribution in hidden layers
	0.9572	***46.3937***	***160***	***2***	***32 + 128***
	0.957	46.4934	128	3	32 + 64 + 32
Bi-GRU	0.957	46.4966	224	3	32 + 64 + 128
	0.9569	46.5528	288	2	2: 32 + 256
	0.9568	46.5828	96	2	2: 32 + 64
	0.9571	***46.4137***	***96***	***2***	***32 + 64***
	0.9571	46.4354	160	2	32 + 128
Bi-LSTM	0.957	46.5102	80	3	3:32 + 32 + 16
	0.9568	46.5912	192	3	3:32 + 128 + 32
	0.9568	46.6214	64	2	2:32 + 32
	0.956	***47.0531***	***64***	***2***	***2 : 32 + 32***
	0.9559	47.0712	32	1	1:32
GRU	0.9555	47.3114	192	2	2:64 + 128
	0.9554	47.366	128	2	2:64 + 64
	0.9551	47.5243	48	2	2:16 + 32
	0.9558	***47.1504***	***128***	***2***	***2 : 64 + 64***
	0.9556	47.2593	32	1	1:32
LSTM	0.9552	47.4629	48	2	2:16 + 32
	0.9551	47.517	64	2	2:32 + 32
	0.9549	47.616	80	2	2:16 + 64
	0.9566	***46.7137***	***64***	***1***	***1 : 64***
	0.9566	46.7305	128	2	2:64 + 64
Simple RNN	0.9565	46.741	128	1	1:128
	0.9565	46.7441	32	1	1:32
	0.9565	46.7468	256	1	1:256

Table 6
Epoch level effects on the perfomance of the RNN models.

(a) Top five RMSE values				(b) Appropriate number of epochs for each model			
Epoch	Model	R2	RMSE (W m^{-2})	Epoch	Model	R2	RMSE (W m^{-2})
250	Bi-GRU	0.9577	46.1373	250	Bi-GRU	0.9577	46.1373
300	Bi-GRU	0.9575	46.1983	100	Bi-LSTM	0.9571	46.4137
100	Bi-GRU	0.9572	46.3937	200	Elman	0.9568	46.6198
100	Bi-LSTM	0.9571	46.4137	200	GRU	0.9563	46.8728
150	Bi-GRU	0.9571	46.4326	100	LSTM	0.9558	47.1504

3.2. Testing the strength and efficiency of the models

3.2.1. K-fold Cross validation results

With the application of K-fold cross validation, interesting results were obtained, some of which show the intricate relationships that the weather factors have. It is clearly seen in Fig. 7, the models have given varying short term solar irradiance predictions' accuracies in different seasons. Overall, the models have shown the best RMSE values in the winter season, while the worst results were obtained in the summer season. As shown in Fig. 6, during summers, the temperature and the solar irradiance levels are higher as compared to other seasons. But why the models were unable to predict better with lower error rates? In order to find the underlying factors, the weather data was further studied. The following graphs, Fig. 8a–d, show monthly variations of mean and standard deviations of solar irradiance, relative humidity, temperature, and wind speed variables. The solar irradiance levels demonstrate high standard deviations during summer seasons while the mean levels show large difference between corresponding months.

The mentioned high variability could hinder the mapping ability of RNN models; as in this process, the models will not be familiar with all of the characteristics of time series data during one season. Moreover, RNN models need more data to learn all abstract characteristics of high variable temporal data. Therefore, the short term solar irradiance prediction results in the summer season could be a result of under-fitting, while the best results in the winter season, could be termed as over-fitted. But in general, the main aim of cross validation is to find out an average accuracy level of any model, hence eliminating the under-fitting or over-fitting [29]. The average RMSE values calculated for the Bi-GRU, Bi-LSTM, Simple RNN, GRU, and LSTM models were 65.5, 64.5, 64.2, 70.2, and 68.9, respectively. Henceforth, it can be deduced that Bi-LSTM and Bi-GRU models have shown better results as compared to unidirectional LSTM and GRU; therefore, with smaller datasets the former two models are more robust than the latter two. Meanwhile, the highest average RMSE has been obtained from the simple RNN model; it shows that with small dataset simple RNN models behaves better than the more complex RNN variants.

3.2.2. Remarks on optimal RNN architectures for solar irradiance predictions

So far, 5 optimum hyper-parameters were selected for each RNN model. The best activation functions as well as the optimizers were set, while the most suitable depth and width of the models were determined. In the following table a summary of these configurations are shown, and in Fig. 9, an architecture, demonstrating general building blocks of RNN models, is depicted. Generally, in our experiments, Bidirectional GRU and Bidirectional LSTM have outperformed all the other variants of RNN in short term predictions of solar irradiance. As mentioned before, bidirectional training of neural networks utilizes the entire datasets twice more than the unidirectional counterparts; hence, they are more effective in learning the highly volatile characteristics of weather time series as shown in Fig. 8a–d. Moreover, the final architecture of the bidirectional GRU model contain the highest number of neurons and hidden layers as depicted

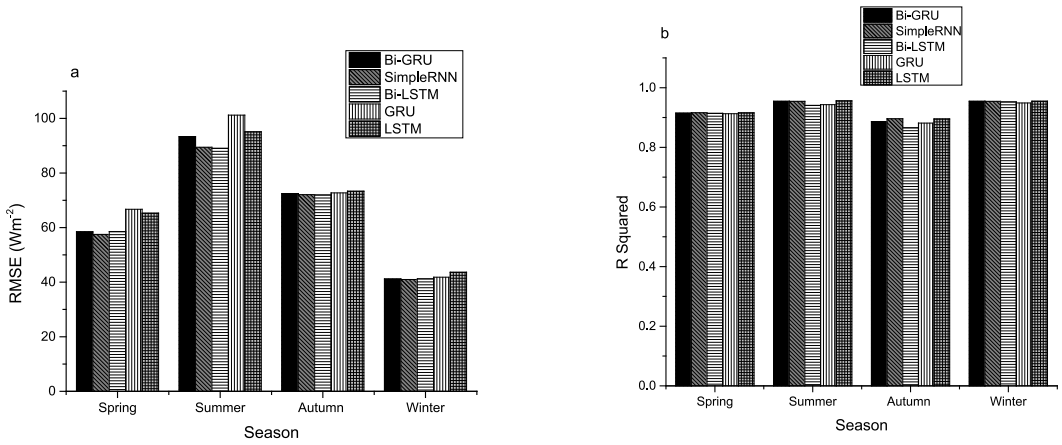

Fig. 7. 9-fold cross validation results showing the robustness levels of RNN models in each season.

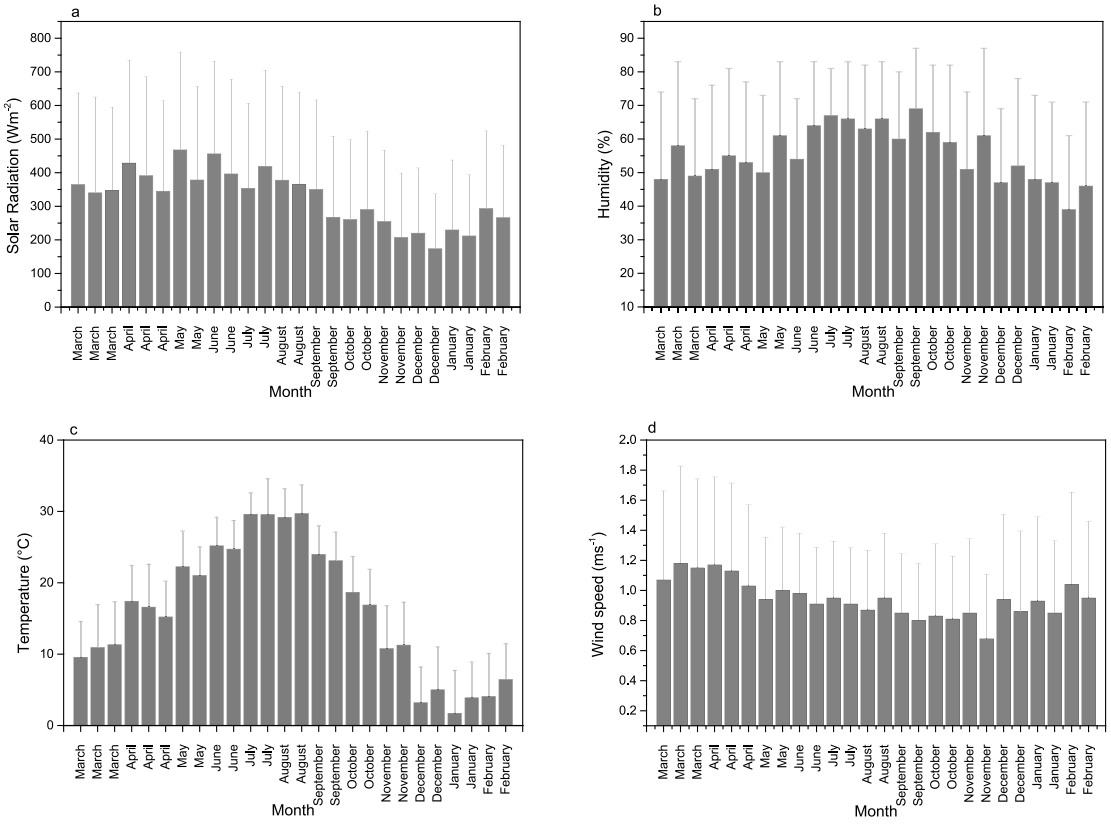

Fig. 8. Standard deviation and Mean distribution of the weather variables over months of same season.

in the following table. Therefore, it could learn and imitate far more aspects of the time series datasets than the other models. On the other hand, the epoch level of the Bi-GRU model was the highest; it shows the slower convergence rate of this model which is partly due to its high number of trainable parameters. It reveals the robustness of the model as it could learn deeper features of the dataset and update the corresponding weights of the model without falling into the exploding or vanishing gradient issues [20].

With the configurations shown in Table 7, the number of trainable parameters or weights and the time consumed for training each model was different, as shown in Table 8. Consequently each model had specific computational costs to train the corresponding number of weights. The unidirectional models consumed the least quantities of training time, but their efficiencies with respect to training time spent per trainable parameter per epoch were very low as compared to the bidirectional models. Additionally, the training time ratio to the trainable parameter statistics were also more in favor of the latter models. The Bi-GRU and Bi-LSTM had the highest numbers of trainable parameters, hence their respective training times were highest. But, it can be said that, in a given time, bidirectional models were more reliable due to their capacity to learn more abstract features of the time-series dataset. The Bi-GRU model performed better than all 5 models in terms of the training time to trainable parameters to epoch ratio as shown in Table 8. In terms of the second ratio, the LSTM model ranked higher than the rest but with a lower accuracy level as shown in Tables 7 and 8. Therefore, arguably, the Bi-GRU model was the most efficient and effective model in terms of computational costs, accuracy level, architectural rigidity.

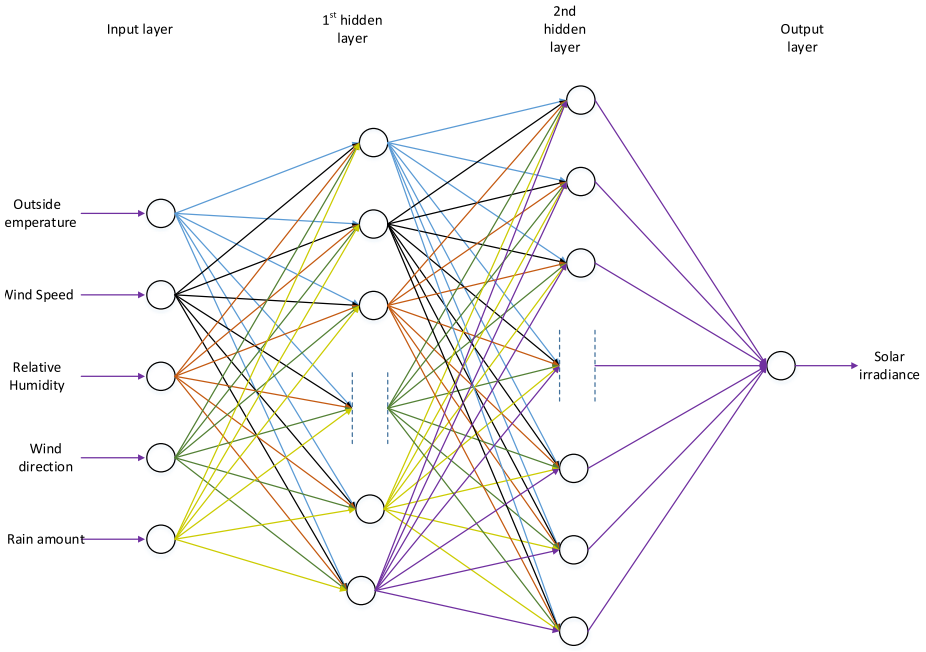

Fig. 9. General structure of an RNN model; output and input layers have constant number neurons.

Table 7
Final hyper-parameter selection for all the RNN models.

Network type	Transfer function	Optimizer	Neurons	Hidden layers	Neuron distribution in hidden layers	Epoch	R^2	RMSE (W m^{-2})
Bi-GRU	Linear	Adam	160	2	32 + 128	250	0.9577	46.1373
Bi-LSTM	Sigmoid	Adam	96	2	32 + 64	100	0.9571	46.4137
Simple RNN	Linear	Adagrad	64	1	1:64	200	0.9568	46.6198
GRU	Sigmoid	Adam	64	2	2:32 + 32	200	0.9563	46.8728
LSTM	Sigmoid	Adam	128	2	2:64 + 64	100	0.9558	47.1504

Table 8
Computational costs relative to time spent for training and model configurations.

Model	Trainable parameters-TP	Epoch	Training time-TT (s)	TT by TP by E ratio	TT by TP ratio
Simple RNN	4609	200	222	2.41E−04	4.82E−02
GRU	10 017	200	599	2.99E−04	5.98E−02
LSTM	51 265	100	401	7.82E−05	7.80E−03
Bi-LSTM	76 161	100	715	9.39E−05	9.40E−03
Bi-GRU	155 969	250	2047	5.25E−05	1.31E−02

4. Conclusion

Predicting solar irradiance has become very crucial to resolve the looming global energy and environmental problems. In this paper 5 RNN variants were developed for 5-minute short-term solar irradiance forecasts. RNN architectures with optimal hyper-parameters and considerable depth and width were developed. Moreover, each model was tested for their robustness against high levels of seasonal weather variability. After extensive processes, bidirectional gated RNNs, i.e. Bi-LSTM and Bi-GRU, outperformed the unidirectional counterparts. The best RMSE and R^2 values of 46.1 and 0.958 respectively, were obtained from the Bi-GRU model. Additionally, the Bi-GRU model had the lowest computational cost of $5.25^{*}10^{-5}$ s per weight per epoch, among all of 5 models. In this paper, it was established that these two models were data hungry and better predictions were viable with large datasets and deeper configurations. In this work, RNN variants were used for 5 min short term solar predictions; solar irradiance predictions for longer intervals, such 30 min or 1 h, is also possible with these models. Ensemble models such as autoregressive integrated moving average (ARIMA) and RNN, poses a good question worth studying. The weather data used in this paper belonged to one region, which can be a hindrance for application of the developed neural network on different regions. Real life application of these models, for example in micro-grids, would show how efficient and effective they are in providing real life economic gains. In the future, the mentioned points will be considered to prolong the merits of this study and add new extensions to the subject.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by the Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry (IPET) through Agriculture, Food and Rural Affairs Research Center Support Program, funded by the Ministry of Agriculture, Food and Rural Affairs (MAFRA) (Project No. 717001-7)

References

[1] United nations, sustainable development goals. 2015, https://www.un.org/sustainabledevelopment/sustainable-development-goals/ (accessed 01 2019).
[2] Kopp G, Krivova N, Wu CJ, Lean J. The impact of the revised sunspot record on solar irradiance reconstructions. Sol Phys 2016;291:2951–65. http://dx.doi.org/10.1007/s11207-016-0853-x.
[3] Torres JF, Troncoso AB, Koprinska I, Wang Z. International Joint Conference SOCO'18- CISIS'18-ICEUTE'18 771. 2019, p. 123–33. http://dx.doi.org/10.1007/978-3-319-94120-2.
[4] Renewables 2017: Analysis and Forecast to 2022. IEA: Paris, France: International Energy Agency; 2017, https://www.iea.org/renewables2017/#section1-2 (accessed 02 2019).
[5] Sobri S, Koohi-Kamali S, Rahim NA. Solar photovoltaic generation forecasting methods: A review. Energy Convers Manage 2018;156:459–97. http://dx.doi.org/10.1016/j.enconman.2017.11.019.
[6] Inman RH, Pedro HTC, Coimbra CFM. Solar forecasting methods for renewable energy integration. Prog Energy Combust Sci 2013;39:535–76. http://dx.doi.org/10.1016/j.pecs.2013.06.002.
[7] Das UK, Tey KS, Seyedmahmoudian M, Mekhilef S, Idris MYI, Van Deventer W, et al. Forecasting of photovoltaic power generation and model optimization: A review. Renew Sustain Energy Rev 2018;81:912–28. http://dx.doi.org/10.1016/j.rser.2017.08.017.
[8] Voyant C, Notton G, Kalogirou S, Nivet ML, Paoli C, Motte F, et al. Machine learning methods for solar radiation forecasting: A review. Renew Energy 2017;105:569–82. http://dx.doi.org/10.1016/j.renene.2016.12.095.
[9] Alzahrani A, Shamsi P, Dagli C, Ferdowsi M. Solar irradiance forecasting using deep neural networks. Procedia Comput Sci 2017;114:304–13. http://dx.doi.org/10.1016/j.procs.2017.09.045.
[10] Wan C, Zhao J, Song Y, Xu Z, Lin J, Hu Z. Photovoltaic and solar power forecasting for smart grid energy management. CSEE J Power Energy Syst 2016;1:38–46. http://dx.doi.org/10.17775/cseejpes.2015.00046.
[11] McCandless TC, Haupt SE, Young GS. A regime-dependent artificial neural network technique for short-range solar irradiance forecasting. Renew Energy 2016;89:351–9. http://dx.doi.org/10.1016/j.renene.2015.12.030.
[12] Husein Chung. Day-ahead solar irradiance forecasting for microgrids using a long short-term memory recurrent neural network: A deep learning approach. Energies 2019;12:1856. http://dx.doi.org/10.3390/en12101856.
[13] Qing X, Niu Y. Hourly day-ahead solar irradiance prediction using weather forecasts by LSTM. Energy 2018;148:461–8. http://dx.doi.org/10.1016/j.energy.2018.01.177.
[14] Srivastava S, Lessmann S. A comparative study of LSTM neural networks in forecasting day-ahead global horizontal irradiance with satellite data. Sol Energy 2018;162:232–47. http://dx.doi.org/10.1016/j.solener.2018.01.005.
[15] Yu Y, Cao J, Wan X, Zeng F, Xin J, Ji Q. Comparison of short-term solar irradiance forecasting methods when weather conditions are complicated. J Renew Sustain Energy 2018;10. http://dx.doi.org/10.1063/1.5041905.
[16] Sharma V, Yang D, Walsh W, Reindl T. Short term solar irradiance forecasting using a mixed wavelet neural network. Renew Energy 2016;90:481–92. http://dx.doi.org/10.1016/j.renene.2016.01.020.
[17] Chen J, Jing H, Chang Y, Liu Q. Gated recurrent unit based recurrent neural network for remaining useful life prediction of nonlinear deterioration process. Reliab Eng Syst Saf 2019;185:372–82. http://dx.doi.org/10.1016/j.ress.2019.01.006.
[18] Di Piazza A, Di Piazza MC, Vitale G. Solar and wind forecasting by NARX neural networks. Renew Energy Environ Sustain 2016;1:39. http://dx.doi.org/10.1051/rees/2016047.
[19] Spacagna G, Slater D, Zocca V, Roelants P, Vasilev I. Computer vision with convolutional networks. In: Dhandre P, Deokar Y, Dias N, Shingote K, Safis, editors. Python Deep Learning. Birmingham: Packt Publishing; 2019, p. 93–121.
[20] Bianchi FM, Maiorino E, Kampffmeyer MC, Rizzi A, Jenssen R. An overview and comparative analysis of recurrent neural networks for short term load forecasting. 2017, http://dx.doi.org/10.1007/978-3-319-70338-1.
[21] Aggarwal CC. Recurrent neural networks. In: Neural Networks Deep Learn. A Textb. Cham: Springer International Publishing; 2018, p. 271–313. http://dx.doi.org/10.1007/978-3-319-94463-0_7.
[22] Caterini AL, Chang DE. Deep neural networks in a mathematical framework. 2018, http://dx.doi.org/10.1007/978-3-319-75304-1.
[23] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997;9:1735–80. http://dx.doi.org/10.1162/neco.1997.9.8.1735.
[24] Graves A. Long short-term memory. In: Supervised Sequence Labelling with Recurrent Neural Networks. Berlin, Heidelberg: Springer Berlin Heidelberg; 2012, p. 37–45. http://dx.doi.org/10.1007/978-3-642-24797-2_4.
[25] Cho K, van Merrienboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. 2014.
[26] Zhu S. Bidirectional Recurrent Neural Networks as Generative models 45 (1997) 1–10.
[27] Elsheikh AH, Sharshir SW, Abd Elaziz M, Kabeel AE, Guilan W, Haiou Z. Modeling of solar energy systems using artificial neural network: A comprehensive review. Sol Energy 2019;180:622–39. http://dx.doi.org/10.1016/j.solener.2019.01.037.
[28] Goh GB, Siegel C, Vishnu A, Hodas NO, Baker N. Chemception: A deep neural network with minimal chemistry knowledge matches the performance of expert-developed QSAR/QSPR models. 2017, p. 1–38.
[29] Chollet F. Getting started with neural networks. In: Arritola T, Gaines J, Dragosavljevic´ A, Taylor T, editors. Deep Learning with Python. 1st ed.. Greenwich, CT, USA: Manning Publications Co.; 2017, p. 56–92.
[30] Neal B, Mittal S, Baratin A, Tantia V, Scicluna M, Lacoste-Julien S, et al. A modern take on the bias-variance tradeoff in neural networks. 2018.
[31] Lee KF. AI Superpowers: China, Silicon Valley, and the New World Order. first ed.. New York: Houghton Mifflin Harcourt; 2018.
[32] Makarenkov V, Rokach L, Shapira B. Choosing the right word: Using bidirectional LSTM tagger for writing support systems. Eng Appl Artif Intell 2019;84:1–10. http://dx.doi.org/10.1016/J.ENGAPPAI.2019.05.003.
[33] Ramachandran P, Zoph B, Le QV. Searching for activation functions. 2017, p. 1–13.