POLITECNICO DI TORINO
Repository ISTITUZIONALE

A compound of feature selection techniques to improve solar radiation forecasting

Original
A compound of feature selection techniques to improve solar radiation forecasting / Castangia, M., Aliberti, A., Bottaccioli, L., Macii, E., Patti, E.. - In: EXPERT SYSTEMS WITH APPLICATIONS. - ISSN 0957-4174. - 178:(2021). [10.1016/j.eswa.2021.114979]

Availability:
This version is available at: 11583/2884619 since: 2021-04-20T10:57:55Z

Publisher:
Elsevier

Published
DOI:10.1016/j.eswa.2021.114979

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
Elsevier postprint/Author's Accepted Manuscript

© 2021. This manuscript version is made available under the CC-BY-NC-ND 4.0 license
http://creativecommons.org/licenses/by-nc-nd/4.0/.The final authenticated version is available online at:
http://dx.doi.org/10.1016/j.eswa.2021.114979

(Article begins on next page)

13 June 2026

A compound of feature selection techniques to improve solar radiation forecasting

Marco Castangia[a,*], Alessandro Aliberti[a], Lorenzo Bottacioli[a,b], Enrico Macii[c], Edoardo Patti[a,b]

[a] *Department of Control and Computer Engineering, Politecnico di Torino, 10129 Torino, Italy, email: {name.surname}@polito.it*
[b] *Energy Center Lab, Politecnico di Torino, 10129 Torino, Italy, email: {name.surname}@polito.it*
[c] *Interuniversity Department of Regional and Urban Studies and Planning, Politecnico di Torino, 10129 Torino, Italy, email: {name.surname}@polito.it*

Abstract

The Global Horizontal Solar Irradiance prediction (GHI) allows estimating in advance the future energy production of photovoltaic systems, thus ensuring their full integration into the electricity grids. This paper investigates the effectiveness of using exogenous inputs in performing short-term GHI forecasting. Thus, we identified a subset of relevant input variables for predicting GHI by applying different feature selection techniques. The results revealed that the most significant input variables for predicting GHI are ultraviolet index, cloud cover, air temperature, relative humidity, dew point, wind bearing, sunshine duration and hour-of-the-day. The predictive performance of the selected features was evaluated by feeding them into five different machine learning models based on Feedforward, Echo State, 1D-Convolutional, Long Short-Term Memory neural networks and Random Forest, respectively. Our Long Short-Term Memory solution presents the best prediction performance among the five models, predicting GHI up to 4 h ahead with a Mean Absolute Deviation (MAD) of 24.51%. Then, to demonstrate the effectiveness of using exogenous inputs for short-term GHI forecasting, we compare the multivariate models against their univariate counterparts. The results show that exogenous inputs significantly improve the forecasting performance for prediction horizons greater than 15 min, reducing errors by more than 22% in 4 h ahead predictions, while for very short prediction horizons (i.e. 15 min) the improvements are negligible.

Keywords: Solar radiation forecast, Photovoltaic system, Renewable energy, ANN, LSTM, 1D-CNN

2010 MSC: 00-01, 99-00

*marco.castangia@polito.it

1. Introduction

The critical depletion of fossil fuels and the global climate change stimulated many countries to employ ever larger volumes of renewable energy sources to gradually substitute the conventional carbon based technologies. The International Energy Agency (IEA) in its Renewables 2019 report (IEA, 2019) estimates that in the next 5 years solar photovoltaic (PV) systems will show the fastest growth among other renewables in the electricity sector, driven by supportive government policies and favourable market conditions. Indeed, solar PV is a very popular renewable energy source and it constitutes a viable method to directly convert solar energy into electricity (Palage et al., 2019). The modularity of the PV technology permits a wide range of applications, from large utility-scale power generation facilities to smaller off-grid residential systems. Solar energy contributes also to stabilise the cost of electricity generation, thanks to its independence from the price volatility of fossil fuels. Furthermore, in the following years the manufacturing cost of modules is expected to continue its downtrend, thanks to the economies of scale. In light of these considerations, it is reasonable to expect that solar PV systems will have an increasingly important role in the future energy mix.

The solar power belongs to the class of variable renewable energy (VRE) sources. Indeed, the power output of PV systems presents a certain variability during the day and strictly depends on the actual exposition of solar panels to the sunlight, which can be partially absorbed by atmospheric components like clouds and aerosols (Vindel & Polo, 2014). The intermittent nature of solar power represents the major challenge for the full integration of solar PV systems into the power grid (Albadi, 2016). In the 21^{st} century, the increasing employment of renewable energy sources led several countries to develop a more sophisticated kind of power grids called smart grids. In particular, smart grids can leverage information from the network to provide a better network management and stability (Berger & Iniewski, 2012). However, with the larger penetration of VREs into the energy mix, smart grids require further adaptations to face the challenges posed by these kind of energy sources. In fact, the uncertainty of power supply negatively affects the stability of power grids and makes difficult the participation in the energy market by solar power producers (Mills et al., 2009). Recent findings suggest that the problem of fluctuating power outputs can be solved

by adopting smart system operating procedures such as demand-response (Siano, 2014) and real-time forecasting (Sugihara et al., 2017). In detail, the prediction of future energy production can significantly improve the energy management by power grid operators, allowing to take actions in advance upon power supply variations. In order to estimate the power output of PV systems, researchers take advantage of sophisticated PV energy simulators that are able to compute the expected power generation given a certain amount of solar radiation as input (Bottaccioli et al., 2017). For this purpose, a measure of primary interest is the Global Horizontal Irradiance (GHI), which is defined as the total irradiance received from the sun measured on a horizontal surface on the Earth. The dynamics of GHI depends upon several factors. The contribution of deterministic factors like the solar position during the day and the extraterrestrial radiation outside the Earth's atmosphere has been already successfully described by several clear sky GHI models with different degrees of complexity (Reno et al., 2014). On the other hand, the effects of stochastic factors like weather conditions are more difficult to forecast and require the adoption of complex physical and statistical models. Indeed, more efforts are needed in this way in order to obtain accurate solar radiation predictions that can be used effectively in the energy sector.

In this work, we propose an innovative methodology for short-term solar radiation forecasts (i.e. from 15 min to 6 h ahead) based on machine learning approaches. In detail, we investigate the effectiveness of using exogenous inputs for GHI prediction by identifying a subset of relevant input variables correlated to the solar radiation phenomena, by applying and comparing different feature selection techniques. For this purpose, we collected a total of 6 years of GHI observations, together with various potentially useful meteorological indicators. Then, to find the most significant input variables for predicting solar radiation, we perform a novel approach based on multiple feature selection methods. Therefore, to evaluate the prediction performance we design and optimize five different machine learning models based on Feedforward Neural Network (FNN), Echo State Network (ESN), 1D Convolutional Neural Network (1D-CNN), Long Short-Term Memory (LSTM) neural network and Random Forest (RF), respectively. Finally, the performances of multivariate models are compared with their univariate implementations, in order to demonstrate the effectiveness of using exogenous inputs for short-term solar radiation forecasts.

The remaining sections of this work are organized as follows: Section 2 reviews the most significant machine learning methods in literature for GHI forecasts and presents the main contributions of our work. Section 3 gives a thorough description of the dataset used. Section 4 details the main

steps of our innovative methodology. Then, Section 5 discusses the prediction results by exploiting different analytical indexes. Finally, Section 6 presents the relevant findings of this work, providing some indications for future works.

2. Related works

Statistical models have been widely applied for solar radiation forecasting. Indeed, they proved to be very effective on predicting short-term solar radiation, from 5 min up to 6 h ahead (Diagne et al., 2013). Early applications mostly exploited conventional time series analysis tools such as Autoregressive Integrated Moving Average (ARIMA) models (Reikard, 2009). However, nowadays these models have been clearly outperformed by more sophisticated techniques belonging to the field of machine learning and artificial intelligence in many forecasting domains (Bontempi et al., 2012). Indeed, researchers have witnessed that solar radiation time series can be better characterized by performing a non-linear mapping between past and future solar radiation values (Lauret et al., 2015a). In this scenario, the Artificial Neural Network (ANN) is certainly the most extensively applied machine learning model in the field of solar radiation forecasting (Voyant et al., 2017). In (Aliberti et al., 2018a,b), authors perform short-term solar radiation forecasting, demonstrating that ANNs can be successfully applied for predicting solar radiation. Other studies devised various ways to improve the prediction performance of ANNs. In (McCandless et al., 2016), the authors showed that the prediction errors can be significantly reduced by detrending the solar radiation time series through the *clear sky index transformation*. Pedro and Coimbra (Pedro & Coimbra, 2012) proved that substantial improvements can be obtained by optimizing the neural network hyperparameters by using genetic algorithms. McCandless, in his research work (Paoli et al., 2010), demonstrated that the prediction performance can be further enhanced by training a single neural network for each specific cloud regime pattern.

In recent years, researchers started to employ even more sophisticated neural network techniques characterized by using an higher number of layers, which are commonly referred to as deep learning. In the context of solar radiation forecasting there are still very few applications of deep learning techniques, but these methods may easily outperform conventional methods in the future, as is already the case in other application domains (Voyant et al., 2017). Qing et al. (Qing & Niu, 2018) compared a Long Short-Term Memory (LSTM) neural network against a multilayered Feedforward Neural Network (FNN) for hourly day-ahead solar radiation forecasting. The LSTM algorithm

showed an impressive improvement of 42.3% in terms of Root-Mean-Square Error (RMSE) against its simpler counterpart, demonstrating the effectiveness of using deep learning techniques in solar radiation prediction. Other machine learning models such as Support Vector Regressors (SVR) and Random Forests have been rarely used in this field (Voyant et al., 2017). However, tree-based ensemble algorithms already exhibited promising results when applied (Benali et al., 2019; Hassan et al., 2017), representing a valuable alternative to neural network techniques.

The correlation between solar radiation and weather conditions inspired many researchers to adopt a multi-variable approach, taking into account also potential external factors influencing the solar radiation dynamics. Commonly used exogenous inputs are represented by meteorological variables measured on the site of interest, such as temperature, humidity, wind speed and cloud cover. In case ground-based meteorological data are not available, an alternative source of exogenous variables are Numerical Weather Prediction models, which provide weather forecasts for several worldwide locations with different scales and forecasting horizons. In (Voyant et al., 2011), the authors compared an ANN using only endogenous inputs and an ANN using both endogenous and exogenous inputs for forecasting daily solar radiation. The comparison between the univariate and the multivariate methods showed that the usage of exogenous inputs produced a performance gain between 0.5% and 1% in terms of Normalized Root-Mean-Square Error (nRMSE), providing a significant improvement specially during the winter season, when the solar radiation variations are greater. Rana et al. in (Rana et al., 2016) evaluated the effectiveness of using exogenous inputs in forecasting the electricity power generated by a PV system from 5 to 60 min ahead. The prediction results showed that the adoption of exogenous inputs does not provide any performance improvement in the short-term power forecasting. According to the authors, a possible reason for this is that PV power data already reflects the weather changes in the short period, without the need of additional input variables. The authors believe that meteorological data are more likely to be useful with longer forecasting horizons.

Even though in the literature there is some evidence of the usefulness of exogenous inputs in solar radiation forecasting, it is still unclear to what extent they provide an edge over the univariate case for the different forecasting horizons. Furthermore, we noticed that despite some works collected a large number of potentially useful input variables, they ended up using only a low number of those variables after feature selection. We believe that the reason for this lies in the adoption of too stringent feature selection methods, which overlook the majority of the collected features and

lead to an underutilization of the input variables. In detail, we think that the employment of the Pearson's correlation coefficient as a single feature selection criteria represents a serious limitation for the application of machine learning models, since it is only able to identify linear relationships between variables (Shannon, 1948). Some studies overcome this limitation by using the Mutual Information criteria (Monjoly et al., 2017; Lauret et al., 2015b), which is known for recognizing both linear and non-linear dependencies in the data. Nevertheless, the mutual information still presents the typical limitations of filter methods, which evaluate the usefulness of each variable independently of the context of others (Guyon & Elisseeff, 2003).

In this work, we propose an innovative methodology to evaluate the effectiveness of using exogenous inputs for short-term GHI forecasting. The novelty of our approach consists on applying multiple feature selection techniques chosen to counterbalance the limitations of each other, with the aim of achieving more robust results than those that could be achieved by using a single selection method. The predictive performance of the selected features are evaluated by feeding them into five different machine learning models: namely a Feedforward Neural Network (FNN), an Echo State Network (ESN), a 1D Convolutional Neural Network (1D-CNN), a Long Short-Term Memory (LSTM) neural network and a Random Forest (RF). Our work provides also a fair comparison between less conventional machine learning models, thus giving significant insights about their relative prediction performance. To conclude, we compare each multivariate model with its univariate counterpart trained by using only endogenous inputs, in order to evaluate the effectiveness of using exogenous inputs for short-term solar radiation forecasting.

3. Dataset

In this work, we aim at forecasting the short-term GHI by using both endogenous and exogenous inputs. For this purpose, we used a dataset of six years (i.e. observations from January 1^{st} 2010 to December 31^{st} 2015). It provides GHI values sampled every 15 min by the weather station in our University Campus. Moreover, in addition to GHI values, the weather station provided also a set of meteorological variables measured on site, namely cloud cover, air temperature, relative humidity, sea-level pressure and wind speed. To collect additional exogenous inputs, we exploit the weather forecasting and visualization service Dark Sky (Grossman & Turner, 2019). The Dark Sky API provides historical weather observations for worldwide locations by gathering information from multiple sources. The API responses come up with the distance of the nearest station that

contributed to the results, together with the list of sources used for the aggregation. The nearest station for our site of interest was located about at 1 km of distance. It is worth mentioning that Dark Sky data do not represent a binding requirement for the proposed methodology, which can easily generalize to other datasets given that the same input variables can be acquired from other sources such as ordinary weather stations on the ground.

During feature engineering, we added three additional variables containing time information: i) day of the year, ii) hour of the day and iii) minute of the hour. Furthermore, we decided to replace the variables sunrise time and sunset time derived from Dark Sky by using a more compact representation called *sunshine duration*, which is defined as the amount of time elapsed between the sunrise time and the sunset time.

To ensure data quality, we carefully examined each variable collected and fixed any irregularity present in the dataset. The GHI variable presented a single outlier during this period, consisting on a negative observation probably due to a temporary malfunction of the pyranometer, while other variables presented some missing values. Both the outlier and the missing values were replaced by using a simple linear interpolation. The comparison between the observed GHI values and the clear sky GHI values computed by using the Ineichen and Perez model (Ineichen & Perez, 2002) revealed that the measured GHI values exceeded several times the estimated GHI values in clear sky conditions. The discrepancies were probably due to some uncertainty in the measurements of the pyranometer. Therefore, since the clear sky GHI should be the maximum value that can be assumed by the observed GHI, we decided to round the exceeding GHI values to the corresponding clear sky GHI values. This rounding step is crucial for the computation of the clear sky index transformation that will be introduced in Section 4.

The final dataset obtained in this way was composed of 210333 records and 15 input variables. Table 1 provides a brief description of each variable present in the dataset. Finally, to evaluate the prediction performance of the proposed models, we divided the dataset into a training set and a test set. The training set consisted of the first 5 years of observations in the period 2010-2014, while the test set consisted of the last year of data, i.e. 2015. It is important to notice that all optimizations were performed by using only the training set in order to avoid any look-ahead bias. For this purpose, we used the last year (2014) of the training set as validation set.

Table 1: List of input variables

Variable	Unit	Typology	Source
Global Horizontal Irradiance (GHI)	W/m^2	endogenous	weather station
Ultraviolet (UV) index	-	exogenous	Dark Sky
Air temperature	°C	exogenous	weather station
Relative humidity	%	exogenous	weather station
Sea-level air pressure	hPa	exogenous	weather station
Cloud cover, i.e. the percentage of sky occluded by clouds	%	exogenous	weather station
Hourly precipitation intensity	mm/h	exogenous	Dark Sky
Hourly precipitation probability	%	exogenous	Dark Sky
Wind speed	m/s	exogenous	Dark Sky
Wind bearing, measured in degrees progressing clockwise from the true north	degrees	exogenous	Dark Sky
Dew point	°C	exogenous	Dark Sky
Sunshine duration	s	exogenous	Dark Sky
Day of the year	-	exogenous	feature engineering
Hour of the day	-	exogenous	feature engineering
Minute of the hour	-	exogenous	feature engineering

4. Methodology

Figure 1 shows the main phases of our methodology. Firstly, the training data are preprocessed by applying some normalization techniques. Then, the most significant input variables for predicting solar radiation are selected, while the remaining variables are discarded from the training set. Once the training data are prepared, the machine learning models can be trained to predict GHI values by using the previously selected input variables. During the test phase, the same preprocessing and feature selection steps used during the training phase are applied to the test set. Finally, the trained models are evaluated and compared based on their prediction performance on test data.

4.1. Preprocessing

To facilitate the analysis by statistical models, we deal with stationary time series. This means that a certain probabilistic regularity should exist over time in the behavior of the time series (Shumway & Stoffer, 2017). Therefore, we normalized the GHI values by using the *clear sky index* transformation, which is widely used in literature to introduce stationarity in solar radiation time series by removing the seasonal and daily trends. The formula of the clear sky index is the

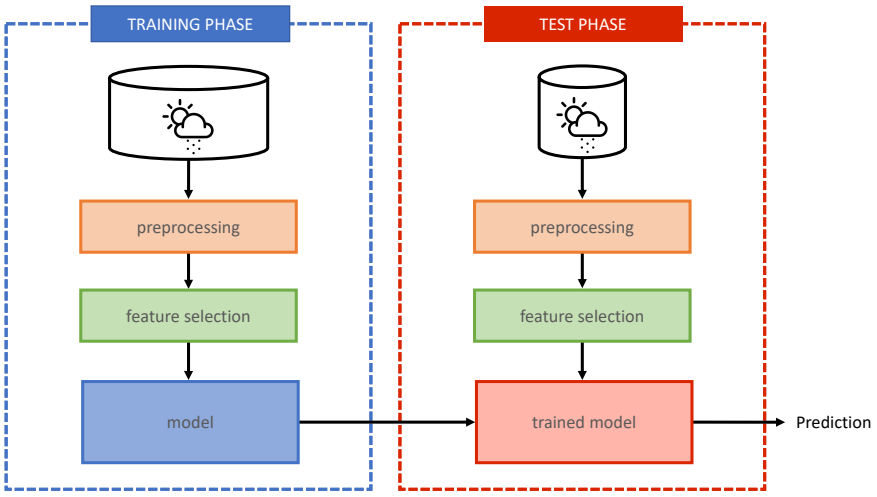


Figure 1: Pipeline with the main steps of our methodology

following:

$$K_{cs} = \frac{GHI_{measured}}{GHI_{clear\_sky}} \tag{1}$$

where the observed GHI values are divided by the clear sky GHI values computed by the Ineichen and Perez model (Ineichen & Perez, 2002). The clear sky index assumes values in the range between 0 and 1 and measures the ratio between the measured GHI values and the clear sky GHI values. The clear sky index computation is not possible during the nights, because the denominator assumes zero values. However, to keep night values in our analysis, we decided to set the clear sky index to 1 during the nights.

Feature scaling is necessary in order to give the same order of magnitude to each feature and to speed up the training process of machine learning models. Therefore, we decided to scale each input variable in the range between 0 and 1 by applying the min max normalization:

$$x_{scaled} = \frac{x - min(x)}{max(x) - min(x)} \tag{2}$$

where x is the vector of values to be scaled, $max(x)$ and $min(x)$ are the maximum and minimum values of that vector, respectively.

4.2. Feature selection

The selection of the most relevant features is aimed at removing unimportant inputs from machine learning models, thus reducing both model complexity and computational costs (Li et al., 2017). Guyon et al. in (Guyon & Elisseeff, 2003) classified the feature selection methods into three major categories: i) filter methods, ii) wrapper methods and iii) embedded methods. Filters rank features based on some correlation criteria. Wrappers aim to find the best subset of features based on their empirical prediction performance. Lastly, embedded methods employ learning machines that perform feature selection as part of their training process. In this work, we selected a couple of the most commonly used methods for each aforementioned category, employing a total of six feature selection techniques. In the following, we provide a brief description of the six feature selection methods adopted by our methodology, concluding with the final results of the feature selection process.

4.2.1. Correlation criteria

The Pearson's correlation coefficient measures the linear relationship between two variables. It takes values between -1 and 1, where 1 is positive linear correlation, 0 is lack of linear correlation and -1 is negative linear correlation. The Pearson's correlation estimate between two variables x and y is given by the following formula, where the bar notation stands for the sample mean:

$$R = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2(y_i - \overline{y})^2}} \tag{3}$$

Since the Pearson's correlation can assume both positive and negative values, we decided to adopt a different measure to compare the input variables. For this purpose, we used the *coefficient of determination* R^2, which is defined as the square of the correlation coefficient R and represents the fraction of variance in the target variable that is explained by individual variables. The coefficient of determination between the input variables and solar radiation constituted our ranking criteria for this method.

4.2.2. Information criteria

In information theory, the Mutual Information (MI) is a measure of the mutual dependence between two random variables (Shannon, 1948). More precisely, it is a measure of the statistical dependence between the density of a variable x and the density of a variable y. The mutual

information is given by the following formula:

$$\mathrm{MI} = \int_x \int_y p(x,y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right) dxdy \tag{4}$$

where $p(x)$ and $p(y)$ are the probability densities of x and y, and $p(x,y)$ is their joint probability density. In case of continuous variables, the densities $p(x)$, $p(y)$ and $p(x,y)$ are estimated by discretizing the variables or by approximating their densities with *Parzen windows*. For this method, the input variables were ranked based on their mutual information with solar radiation.

4.2.3. Sequential forward selection

The Sequential Forward Selection (SFS) algorithm (Pudil et al., 1994) is a feature selection method that heuristically searches for the best subset of variables to minimize the prediction error. The overall procedure is described in Algorithm 1. The SFS starts with an empty set of features (line 1) and at each iteration extends the previous set with the feature whose insertion gives the lowest prediction error (lines 3 and 4). The model used for evaluating the subsets was a simple linear regression model, trained on data from 2010 to 2013 and tested on the following year (2014). In order to evaluate the relevance of all features, we iterated the procedure until all input variables have been inserted. The order of insertion constitutes our ranking criterion, in which those variables inserted first are more important than those inserted last.

Algorithm 1 Sequential Forward Selection (SFS)

1: $Y \leftarrow \{\}$; $F \leftarrow features$
2: **while** $size(F) > 0$ **do**
3: $\quad f = argmin_{f \in F}[J(Y+f)]$
4: $\quad Y \leftarrow Y + f$
5: $\quad F \leftarrow F - f$

4.2.4. Sequential backward selection

The Sequential Backward Selection (SBS) algorithm (Pudil et al., 1994) is very similar to the SFS method. The procedure is described in Algorithm 2. The SBS starts with the complete set of features (line 1) and at each iteration remove from the previous set the feature whose removal gives the lowest prediction error (lines 3 and 4). As before, we used a simple linear regression model trained on data from 2010 to 2013 and tested on 2014. In order to evaluate the relevance of

all features, we iterated the procedure until all input variables have been removed. The order of removal constitutes our ranking criterion, in which those variables removed last are more important than those removed first.

Algorithm 2 Sequential Backward Selection (SBS)

1: $Y \leftarrow features$; $F \leftarrow features$
2: **while** $size(F) > 1$ **do**
3: $\quad f = argmin_{f \in F}[J(Y - f)]$
4: $\quad Y \leftarrow Y - f$
5: $\quad F \leftarrow F - f$

4.2.5. LASSO regression

Least Absolute Shrinkage and Selection Operator Regression (LASSO Regression) is a Linear Regression model that uses l_1 regularization (Tibshirani, 1996). In particular, a penalty term is added to the cost function by using the l_1 norm of the model parameters. The regularized version of the cost function is defined as:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (\theta^T x_i - y_i)^2 + \alpha \sum_{i=1}^{n} |\theta_i| \tag{5}$$

where m is the number of training instances, n is the number of input variables, x_i is the input vector, y_i is the target, θ is the parameter vector of the model and α is the regularization hyperparameter. The peculiarity of l_1 regularization is that it favors sparsity in model's weights by reducing the contribution of less important features. The regularization hyperparameter α controls the amount of penalization applied during the training. It is important to use the right value for α in order to avoid either too heavily sparse vectors or flat distributions of weights. To find the best value for α we trained the model on data from 2010 to 2013 by using the complete set of features, keeping the model coefficients presenting the lowest prediction error on the validation set (2014). The best value for α was found to be equal to 0.001. The magnitude of model coefficients represents the importance assigned to each input variable by the model itself.

4.2.6. Random forest

A Random Forest is an ensemble of *decision trees*, each one trained on a different random subset of the training set (Breiman, 2001). The training algorithm of decision trees is designed such that

the most important features are located closer to the root of the tree, while less important features are located closer to the leaves. In other words, the height of the tree at which a specific feature is evaluated determines its discriminative power for predicting the target variable. In this work we trained an ensemble of 100 decision trees on the whole training set from 2010 to 2014. No validation set is required for this method. Finally, the input variables are ranked based on the impurity-based feature importance estimated by the training algorithm of random forest.

4.2.7. Feature selection results and remarks

Table 2 shows the final results of the feature selection process. In detail, we reported the ranking position assigned to each variable for the six feature selection methods employed. The final ranking (reported in the rightmost column) was obtained by computing the mean position of each variable across the different feature selection methods. The threshold between the selected features and the discarded features was found empirically during the model optimization phase. In particular, we trained and tested our models on incremental subsets of features generated by iteratively adding the next most important feature from our final ranking. Again, notice that the entire procedure used solely data from the training set, thus completely ignoring the final test set. Figure 2 shows the mean validation error computed across all our models after each incremental variable insertion. As depicted in Figure 2, we noticed a performance improvement until the insertion of humidity variable, where the prediction error reached its global minimum. On the other hand, the variables day of the year, pressure, wind speed, precipitation probability, precipitation intensity and minute penalized prediction performance once inserted. In conclusion, we found that the most relevant features for solar radiation forecasting are ultraviolet index, temperature, sunshine duration, cloud cover, hour, wind bearing, dew point and humidity

Table 2: Ranking of variables based on feature selection results

	Features	Methods						Ranking
		R^2	MI	SFS	SBS	LASSO	RF	
Selected	ultraviolet index	1	1	1	1	1	1	1
	temperature	3	4	6	2	2	6	3
	sunshine duration	4	7	5	4	4	5	4
	cloud cover	9	8	3	6	6	2	5
	hour	12	2	7	5	8	4	6
	wind bearing	5	3	4	8	9	9	6
	dew point	7	10	9	3	3	7	6
	humidity	2	6	2	13	14	3	6
Discarded	day	11	5	8	7	10	11	8
	pressure	13	11	10	9	5	8	9
	wind speed	6	9	14	14	12	10	10
	precipitation probability	8	13	11	10	11	13	11
	precipitation intensity	10	12	12	11	7	14	11
	minute	14	14	13	12	13	12	13

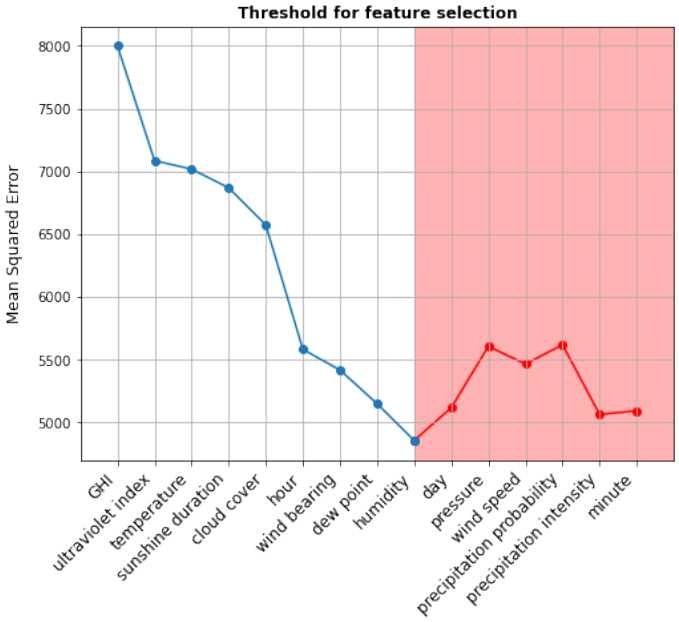


Figure 2: Validation errors obtained by incrementally expanding the set of inputs with the next most important feature from the final ranking

4.3. Machine learning models

In the following, we present the machine learning models that we exploited to predict short-term solar radiation. We design and optimize five different state-of-art machine learning models, namely a Feedforward Neural Network (FNN), an Echo State Network (ESN), a 1D Convolutional Neural Network (1D-CNN), a Long Short-Term Memory (LSTM) neural network and a Random Forest (RF). In order to evaluate how far ahead our models can still achieve acceptable forecasting errors, we adopted a Multiple-Input Multiple-Output (MIMO) approach (Ben Taieb et al., 2010), where a single model with multiple outputs allows to predict multiple values at once. For this purpose, we performed multi-step ahead predictions with a time step of 15 min up to 6 h ahead, corresponding to 24 outputs. The maximum prediction horizon was chosen arbitrarily, considering also the difficulty in predicting many time steps ahead with an acceptable forecasting error. The best architectures for the proposed models were found by using a simple grid search algorithm as an alternative to more complex methodologies (Esfe et al., 2017), evaluating as many configurations as possible. At each iteration of the grid search, models were trained by using the first four years of observations (2010-2013), while the following year (2014) was used for validation. In case of

comparable performance between two different configurations, the simpler one was chosen. Finally, the configuration with the lowest error on validation data was selected.

4.3.1. Feedforward Neural Network

The Feedforward Neural Network, also known as Multilayer Perceptron, is a neural network architecture where the signal can only flow in one direction (i.e. from input neurons to output neurons). The FNN is composed of one input layer, one or more hidden layers and one output layer. Each layer (except the output layer) is fully connected to the next layer by a set of weighted connections. The neurons in the input layer simply replicate on their output whatever they received in input, while the neurons in the next layers apply a non linear function to the weighted sum of their inputs. An additional bias term b is usually added to the weighted sum of inputs to further increase the model flexibility. The output of a single neuron is described by the following formula:

$$\hat{y} = f(\sum_{i=1}^{n} w_i x_i + b) \tag{6}$$

where x_i are the inputs, w_i are the input weights, b is the bias term and f is a non linear function that is commonly called *activation function*. The set of all weights and bias terms constitutes the network parameters to be optimized during the training process. The optimal weights are found through the backpropagation training algorithm, which iteratively updates the network parameters by using some optimization technique that minimizes the prediction error. In this work, all neural networks were trained by using the Adaptive Moment Estimation (Adam) algorithm with a learning rate of 0.001, which usually guarantees a good convergence. The loss function to be minimized during the training process was the *Mean Square Error* (MSE), which is defined as:

$$loss_{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{7}$$

where y_i are the observed values, $\hat{y}_i$ are the predicted values and n is the number of samples.

The number of epochs was set to 500, while the batch size was set to 200 samples. In order to prevent overfitting and reduce the training time, we adopted the early stopping criteria with a patience of 10 epochs and a validation split equal to 0.1 (10% of the training set). Thus, if the model does not show any improvement on the validation set for 10 epochs, the training is stopped. The training hyperparameters are summarized in Table 3 and are the same for all the following neural networks presented in this work.

Table 3: Training hyperparameters

hyperparameter	Value
Optimizer	Adam
Loss	Mean sqaured error
Learning rate	0.001
Epochs	500
Batch size	200
Validation split	0.1
Stopping criteria	early stopping with patience equal to 10

The FNN was implemented in Python by using the Keras library with Tensorflow backend (Chollet et al., 2015). Firstly, we investigated the number of past observations to be used as input by evaluating the model performance with different number of regressors. The FNN obtained the best prediction performance by using the past 3 observations for each input variable. The final architecture of the FNN was composed of two hidden layers of respectively 100 units and 50 units, and a final output layer of 24 units, corresponding to the number of prediction horizons of interest. The hidden layers used a hyperbolic tangent (tanh) activation function, while the output layer used a linear activation function. The FNN hyperparameters are described in Table 4.

Table 4: FNN hyperparameters

Layer type	Units	Activation
Dense	100	tanh
Dense	50	tanh
Dense	24	linear

4.3.2. Echo State Network

The Echo State Network is a Recurrent Neural Network (RNN) that belongs to the class of reservoir computing. The reservoir simply consists of a large set of randomly connected neurons that constitute the hidden layer of the network. Figure 3 provides a schematic representation of a

general ESN architecture.

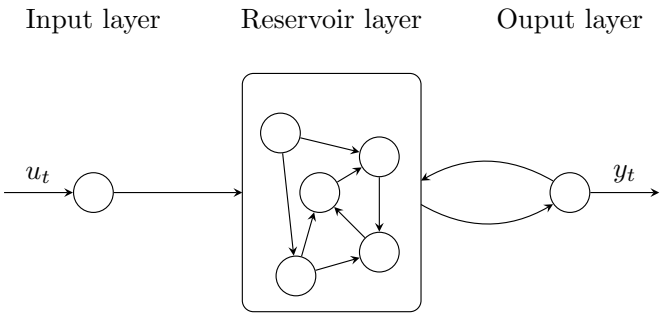


Figure 3: ESN architecture

The state update of a standard discrete-time ESN with N reservoir units, K inputs and L outputs is governed by the following equation:

$$x_{t+1} = f(Wx_t + W^{bf}y_t + W^{in}u_t) \tag{8}$$

where f is the state update activation function, $W \in R^{N\times N}$ is the reservoir weight matrix, x_t is the reservoir state, $W^{bf} \in R^{N\times L}$ is the feedback weight matrix, y_t is the output, $W^{in} \in R^{N\times K}$ is the input weight matrix and u_t is the input. The output of an Echo State Network is given by the following formula:

$$y_t = g(W^{out}[x_t, u_t]) \tag{9}$$

where g is the output activation function, $W^{out} \in R^{L\times(N+K)}$ is the output weight matrix and $[x_t, u_t]$ is the concatenation of the reservoir and the input states. The forward connections W^{out} from the reservoir nodes to the output units are the only parameters that are learned during the training process, while the other weights are randomly initialized at the creation of the network and remain fixed. In order for the ESN to work properly, the reservoir must respect the Echo State Property (ESP), which states that the effect of initial conditions x_0 should progressively vanish as the length of the input sequence goes to infinity. The necessary condition for the ESP to hold is that the spectral radius λ, which is defined as the largest absolute eigenvalue of the matrix W, must be less than 1. Pratically, it has been shown that the ESN can work properly also for spectral radius λ greater but close to 1.

In this work, the Echo State Network was implemented in Python by using the open source library easyesn (Zimmermann, 2017). As before, the best hyperparameters for the ESN were found

though a trail-and-error approach. Thus, we found that the best prediction performance for the ESN were obtained by using only 1 regressor for each input variable. The final ESN configuration used a reservoir of 100 units with a reservoir density of 0.1. The spectral radius was set to 0.9, while the leaking rate was set to 0.2. The set of all ESN hyperparameters with their values is described in Table 5.

Table 5: ESN hyperparameters

hyperparameter	Value
Number of reservoir units	100
Leaking rate	0.2
Spectral radius	0.9
Reservoir density	0.1
Transient time	100

4.3.3. 1D Convolutional Neural Network

The 1D Convolutional Neural Network is a special kind of CNN which was specifically designed for modelling 1-dimensional inputs. The 1D-CNN already achieved top performance in several signal processing applications (Kiranyaz et al., 2019), thanks to its great capability in extracting meaningful features from sequential data. The block responsible for the feature selection process is the *filter*, which is basically a feature detector that learns to recognize a specific pattern in the input data. The filter slides across the input sequence and activates the corresponding neuron in the feature map whenever a match for that pattern is found. The process of sliding a filter across the input data is called *convolution* and consists of a series of multiplications between the filter and the input values. Generally, convolutional layers are composed of several filters working over multiple input channels, enabling the network to recognize multiple patterns in the input data. Moreover, convolutional layers can be stacked together to form a deep architecture, where each layer builds upon the features detected by the previous layer in a hierarchical way. Convolutional layers may be followed by a pooling layer, which is responsible for reducing the amount of information received from the previous layer. Finally, one or more fully connected layers are added at the end of the network to interpret the features extracted.

We design and implement our 1D-CNN by exploiting the Keras library with Tensorflow backend (Chollet et al., 2015). The best configuration for the 1D-CNN was achieved by using the past 5 observations of each input variable. The convolutional part of our network used two 1-dimensional convolutional layers with 32 filters and a kernel size equal to 2. The activation function used for each convolutional layer was the hyperbolic tangent function. No pooling layer was inserted after the convolutional layers, because the number of features was quite small and did not show a limitation in terms of computational costs. Finally, a fully connected layer with 100 units was added at the end of the network, using a hyperbolic tangent activation function. Like in the FNN, the output layer was constituted of 24 output units with a linear activation function. The full description of the 1D-CNN hyperparameters is provided in Table 6. The hyperparameters for the training process were the same used for the FNN and are summarized in Table 3.

Table 6: 1D-CNN hyperparameters

Layer type	Units	Filters	Kernel size	Stride	Padding	Activation
Convolution 1d	-	32	2	1	valid	tanh
Convolution 1d	-	32	2	1	valid	tanh
Dense	100	-	-	-	-	tanh
Dense	24	-	-	-	-	linear

4.3.4. Long Short-Term Memory

The Long Short-Term Memory neural network is a Recurrent Neural Network that demonstrated particularly good performance in modelling sequential data. A Recurrent Neural Network differs from a Feedforward Neural Network because it contains also backward connections. Indeed, at each time step t a recurrent neuron receives the current input x_t plus its own output from the previous time step y_{t-1}, as shown in Figure 4.

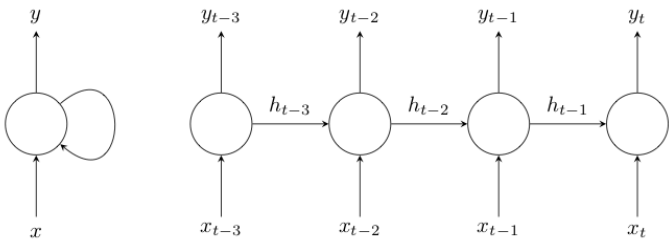


Figure 4: A recurrent neuron unrolled through time

Recurrent Neural Networks can be easily extended by adding more neurons in the hidden layer, forming what is called a *recurrent layer*. Furthermore, recurrent neural networks can be composed of multiple recurrent layers stacked together to form deeper architectures. A single recurrent neuron or a layer of recurrent neurons implement a basic form of memory cell, thanks to their capability of preserving some states across different time steps. However, basic memory cells present some limitations for longer input sequences and suffer from the vanishing gradient problem. Indeed, the training process may become very slow for longer input sequences and recurrent neurons tend to forget past states contributions as the number of time steps increases. Thus, the LSTM cells were introduced by Hochreiter and Schmidhuber in (Hochreiter & Schmidhuber, 1997) to overcome the vanishing gradient problem. In particular, LSTM cells provide faster convergence during the training process and are able to detect long-term dependencies in the input data.

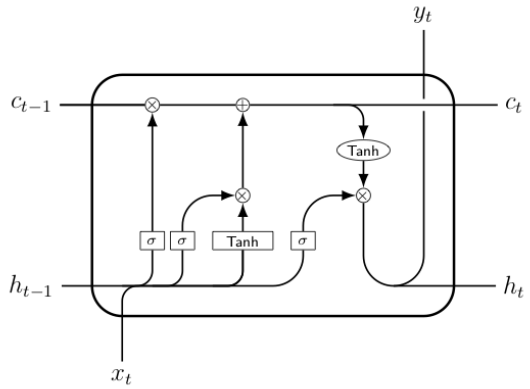


Figure 5: LSTM cell

Figure 5 shows the architecture of a basic LSTM cell. The LSTM state is split in two main components: the short-term state h_t and the long-term state c_t. The short-term state h_t replicates the output of the previous memory cell, while the long-term state c_t is responsible for transporting past states contributions. During the training, LSTM cells learn what information must be stored in the long-term, what information must be throw away and what information are needed in the current time step. The *forget gate* controls which information should be removed from the long-term state. The *input gate* controls which information should be added to the long-term state. Finally, the *output gate* controls which information should be read and output at the current time step. The following set of equations in vectorial form describes how the LSTM cell computes its short-term state h_t, its long-term state c_t and its output y_t at each time step t:

$$\begin{aligned} i_t &= \sigma(W^{xi}x_t + W^{hi}h_{t-1} + b_i) \\ f_t &= \sigma(W^{xf}x_t + W^{hf}h_{t-1} + b_f) \\ o_t &= \sigma(W^{xo}x_t + W^{ho}h_{t-1} + b_o) \\ g_t &= tanh(W^{xg}x_t + W^{hg}h_{t-1} + b_g) \\ c_t &= f_t \otimes c_{t-1} + i_t \otimes g_t \\ y_t &= h_t = o_t \otimes tanh(c_t) \end{aligned} \tag{10}$$

where W^{xi}, W^{xf}, W^{xo} and W^{xg} are the weight matrices of the connections to the input vector x_t, W^{hi}, W^{hf}, W^{ho} and W^{hg} are the weight matrices of the connections to the previous short-term state vector h_{t-1} and b_i, b_f, b_o and b_g are the bias terms.

In this work, the LSTM neural network was implemented in Python by using the Keras library with Tensorflow backend (Chollet et al., 2015). The best performance for the LSTM were obtained by using the past 3 observations for each input variable. The final LSTM architecture was composed of two recurrent layers with 50 units using a tangent hyperbolic activation function. Like in the previous networks, the output layer used 24 output units with a linear activation function. The full description of the LSTM hyperparametes is provided in Table 7, while the training hyperparameters are the same described in Table 3.

Table 7: LSTM hyperparameters

Layer type	Units	Activation
LSTM	50	tanh
LSTM	50	tanh
Dense	24	linear

4.3.5. Random Forest

A Random Forest is an ensemble of Decision Trees where each predictor is trained by using a different random subset of the training set, sampled via the bagging or the pasting methods. Generally, when Decision Trees perform regression tasks they are called *Regression Trees*, while the corresponding ensemble model is called *Random Forest regressor*. The prediction of a Regression Tree is simply given by the mean target value of the training data reaching the leaf node. The Regression Tree algorithm tries to iteratively split the training data at each node such that the MSE between the target values and mean of target values is as low as possible. The cost function that the training algorithm tries to minimize at each node is described by the following equation:

$$J(k, t_k) = \frac{m_{left}}{m} \sum_{i \in left} (\bar{y}_{left} - y_i)^2 + \frac{m_{right}}{m} \sum_{i \in right} (\bar{y}_{right} - y_i)^2 \tag{11}$$

where k is the feature chosen for the splitting, t_k is the splitting threshold value, m is the cardinality of the training subset reaching that node, m_{left} and m_{right} are the cardinalities of the next training subsets, y_i are the training subset target values and $\bar{y}$ is the mean target value. The prediction of a Random Forest regressor is obtained by averaging the predictions of individual trees.

The Random Forest regressor was implemented in Python by using the sklearn open source library (Pedregosa et al., 2011). The model showed the best prediction performance when using 10 regressors for each variable. The ensemble model was limited to 100 Decision Trees, since a greater number of estimators did not show substantial improvements. In oder to prevent overfitting, the minimum number of samples reaching a node was used as stopping criteria. By using a trial-and-error approach, we decided to stop the tree growth when the number of samples reaching a node goes below 100 instances. The Random Forest hyperparameters are summarized in Table 8

Table 8: Random Forest hyperparameters

hyperparameter	Value
Estimators	50
Minimum samples split	100

5. Results and discussion

In this section, we present the prediction results obtained by using the proposed methodology. For a deep analysis of our models, we exploit a set of statistical indicators widely used in the literature for time series analysis. In order to demonstrate the effectiveness of using exogenous inputs for short-term solar radiation forecasting, we compared the prediction performance of the five machine learning models (FNN, ESN, LSTM, 1D-CNN, RF) in two different scenarios: i) models are trained by using both endogenous and exogenous inputs and ii) models are trained by using only endogenous inputs. For the sake of clarity, we recall that endogenous inputs are represented by past solar radiation values (GHI), while exogenous inputs are represented by the most significant features elected by our feature selection methodology (UV index, temperature, sunshine duration, cloud cover, hour, wind bearing, dew point, humidity). For a fair comparison, the univariate and multivariate models are trained and tested by using the same dataset partition. The hyperparameters and the number of regressors of each model are kept the same between the multivariate and the univariate scenarios without applying further optimizations.

5.1. Performance metrics

To evaluate the prediction accuracy of models, we exploited a number of metrics that are widely used in descriptive statistics and in regression analysis to quantify the similarities between predicted and observed time series (Gueymard, 2014): i) the *Mean Absolute Difference* (MAD), ii) the *Root Mean Square Difference* (RMSD) and iii) the *Coefficient of determination* (R^2). MAD measures the absolute difference between the predicted and the observed values. RMSD measures the standard deviation of the difference between the predicted and the observed values. R^2 measures the fraction of variance in the observed values that is explained by the predicted values. The three statistical

indicators are described by the following equations:

$$MAD = \frac{100}{\bar{y}_{test}} \frac{\sum_{i=1}^{n} |y_{pred,i} - y_{test,i}|}{n} \tag{12}$$

$$RMSD = \frac{100}{\bar{y}_{test}} \sqrt{\frac{\sum_{i=1}^{n} (y_{pred,i} - y_{test,i})^2}{n}} \tag{13}$$

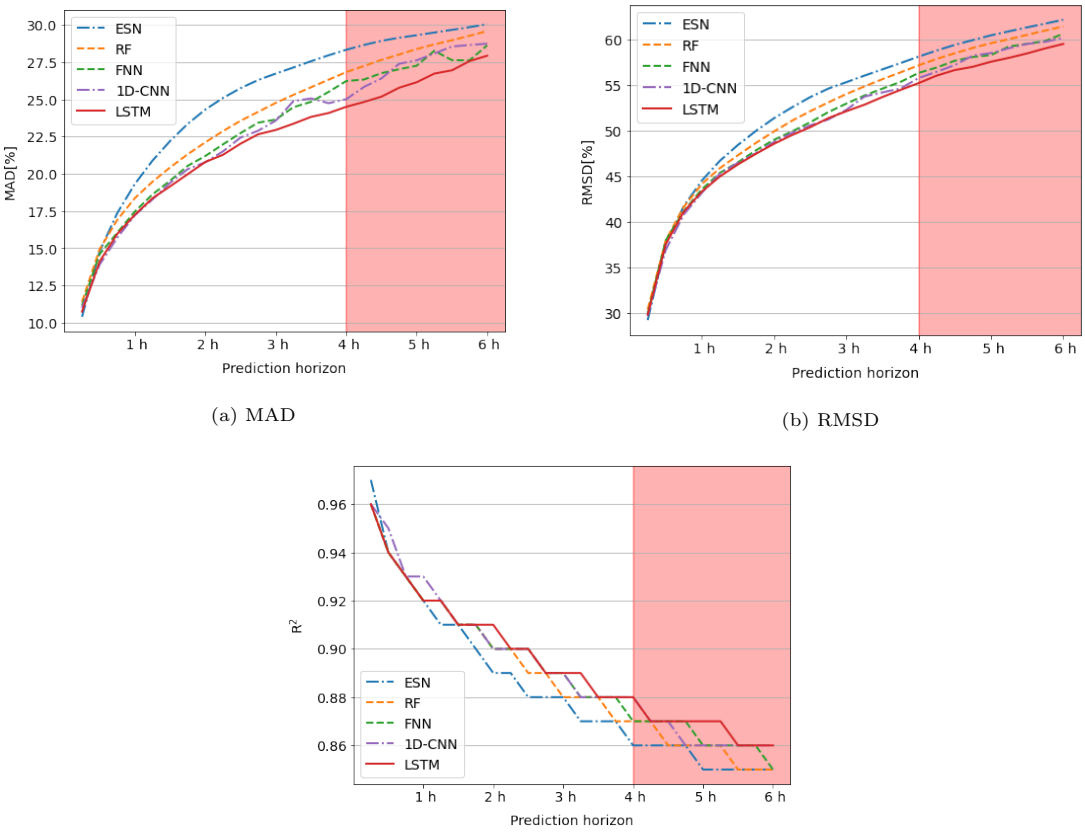
$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_{test,i} - y_{pred,i})^2}{\sum_{i=1}^{n} (y_{test,i} - \bar{y}_{test})^2} \tag{14}$$

where y_{pred} are the predicted values, y_{test} are the observed values, n is the number of predictions and the bar notation stands for the mean value. It is worth noticing that the models are trained to forecast the clear sky index values instead of directly predicting the solar radiation values. Therefore, the statistical indicators are computed by using the solar radiation values reconstructed by inverting the clear sky index transformation.

5.2. Model evaluation

In the following, we compare the five proposed machine learning models in the multivariate scenario, i.e. by using both endogenous and exogenous inputs. The models are compared based on their prediction performance on the test set. Figure 6 provides a visual comparison of the forecasting errors in terms of MAD, RMSD and R^2, respectively. The differences between the five models are more evident in the MAD, where the forecasting errors of different models are more distinguishable. The dissimilarities between models are less pronounced in the RMSD and R^2, but still appreciable. Overall, the plots show a general uptrend in the forecasting errors of all models. Indeed, the forecasting errors increase as the prediction horizon gets larger. In accordance with the results obtained in (Aliberti et al., 2018a), we fixed a threshold of 25% in terms of MAD to indicate the upper bound above which we consider an excessive degradation in the prediction performance. Based on these assumptions, we can state that the proposed methodology produced acceptable forecasting errors up to 4 h ahead, with a maximum error of 24.51% (MAD) achieved by our LSTM model. The red area in Figure 6 indicates the forecasting window where the prediction performance of all models started collapsing. Hence, we decided to focus our discussion only on the temporal window with acceptable forecasting errors, i.e. from 15 min up to 4 h ahead. The numerical results of the forecasting errors plotted in Figure 6 are clearly reported in Tables 7, 8

and 9 under the column name "*Endogenous + Exogenous*", to indicate that these results are solely referred to the multivariate models. Overall, the LSTM demonstrated the best prediction performance among the five models, outperforming other models in almost every prediction horizon. In particular, Tables 9 and 10 show that the LSTM presented better performance for prediction horizons greater than 1 h, ranging from 18.36% to 24.51% in terms of MAD and from 43.31% to 55.19% in terms of RMSD. Apparently, the 1D-CNN slightly surpassed the LSTM performance in some sporadic cases. For example, the 1D-CNN outperformed the LSTM in terms of MAD for prediction horizons between 30 min and 75 min ahead and between 30 min and 60 min ahead in terms of RMSD, as shown in Tables 9 and 10. However, we must consider that the LSTM provided a small advantage over the 1D-CNN at cost of higher computational resources, which definitely makes the 1D-CNN a valid competitor for the LSTM. The FNN demonstrated also very good prediction performance, but it remained steadily behind the LSTM and the 1D-CNN for every prediction horizon that we considered. Interestingly, the ESN presented the best results for very short-term predictions, achieving the top performance for 15 min ahead predictions. On the other hand, the ESN presented the worst performance for longer prediction horizons (> 15 min), revealing a general poor capability in modelling multivariate time series. The RF performed slightly better than the ESN, but presented lower prediction performance with respect to other neural network techniques.



(a) MAD

(b) RMSD

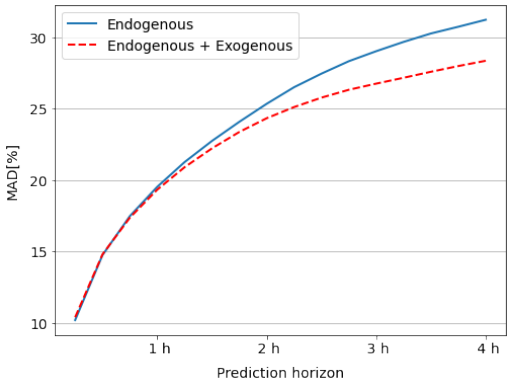
(c) R^2

Figure 6: Comparison of multivariate models based on MAD, RMSD and R^2

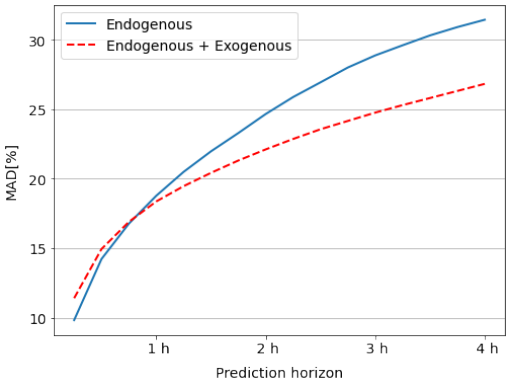
5.3. Improvement of exogenous inputs

To demonstrate the effectiveness of using exogenous inputs for short-term solar radiation forecasting, we compared the five multivariate models with their univariate counterparts trained by using only endogenous inputs. Figures 7, 8 and 9 provide a visual comparison between the univariate and multivariate models in terms of MAD, RMSD and R^2, respectively. According to the results, the models using both endogenous and exogenous inputs demonstrated better prediction performance with respect to the models using only endogenous inputs. In particular, the performance improvements due to exogenous inputs are more pronounced for longer prediction horizons, while for shorter prediction horizons the performance of the two approaches are very similar. Tables 9, 10 and 11 compare the models using exogenous inputs with the models using only endogenous

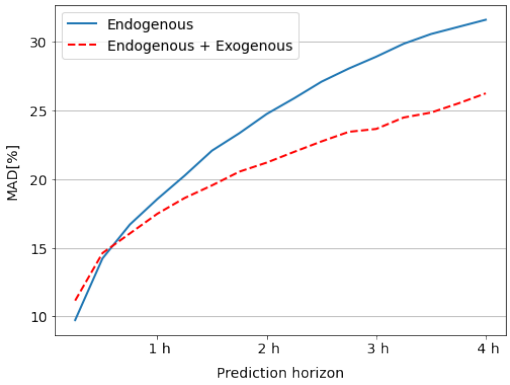
inputs in terms of MAD, RMSD and R^2, respectively. The underlined values indicate the model with the lowest error for that specific prediction horizon. In case of similar errors the model with less parameters was selected. In the third column we reported the percentage improvement of each multivariate model with respect to its univariate implementation, where a positive value indicates a beneficial impact of exogenous inputs in prediction performance. The results clearly show that the multivariate approach outperforms the univariate approach for almost every prediction horizon. Indeed, all statistical metrics present a certain improvement in the multivariate scenario from 30 min to 240 min ahead, showing higher improvements for longer forecasting horizons. For example, the LSTM trained by using both endogenous and exogenous inputs achieved an impressive performance improvement of 22.14% in terms of MAD for 4 h ahead predictions. The RSMD and the R^2 indices also show an outstanding performance improvement of 18.99% and 8.64% for 4 h ahead forecasts, respectively. On the other hand, the univariate approach surpassed the multivariate solution only for 15 min ahead forecasts, where the 1D-CNN reported the best results in terms of MAD with 9.61%. As a matter of fact, the MAD index shows that there are no advantages in the multivariate approach for 15 min ahead predictions, reporting only negative performance improvements for all models in that prediction horizon. To conclude, the results show that the adoption of exogenous inputs can significantly improve the forecasting performance for prediction horizons greater than 15 min, while for very short prediction horizons the performance improvement due to exogenous inputs can be considered as negligible. On these bases, we suggest to adopt a multivariate approach only for longer forecasting horizons, where the benefits provided by exogenous inputs give reasons for the higher computational costs required by more complex models. On the contrary, we believe that endogenous inputs can suffice for very short-term solar radiation predictions, since there is no evidence that exogenous inputs can provide performance improvements for very short prediction horizons.



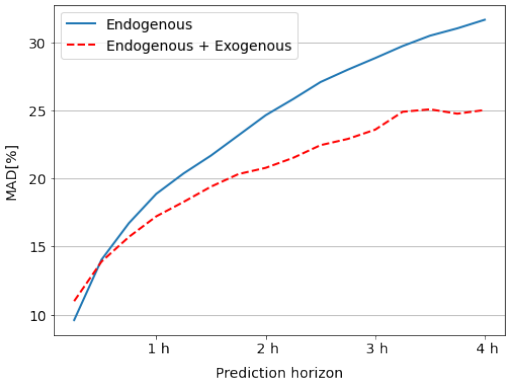
(a) ESN



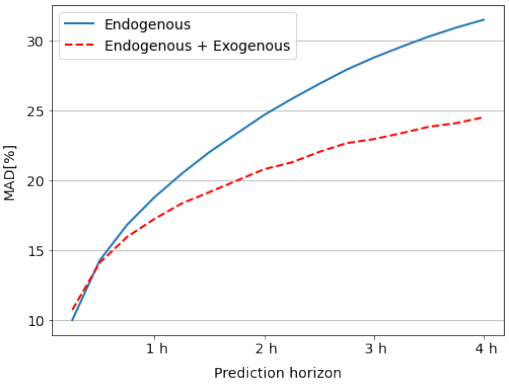
(b) RF



(c) FNN

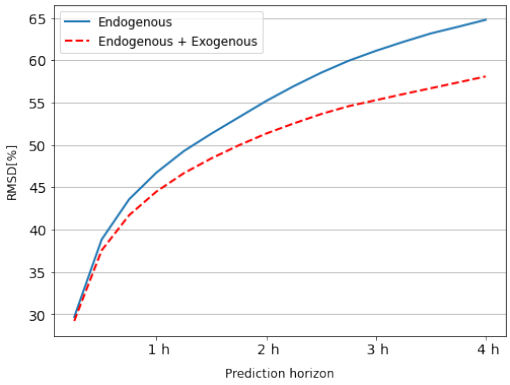


(d) 1D-CNN

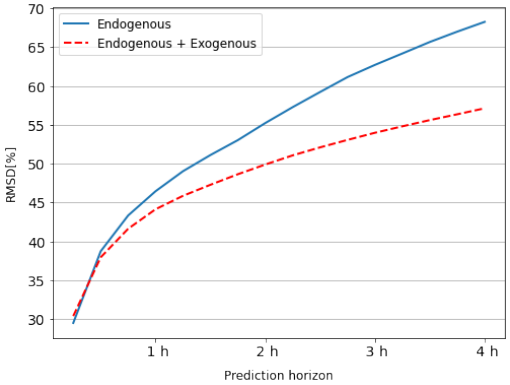


(e) LSTM

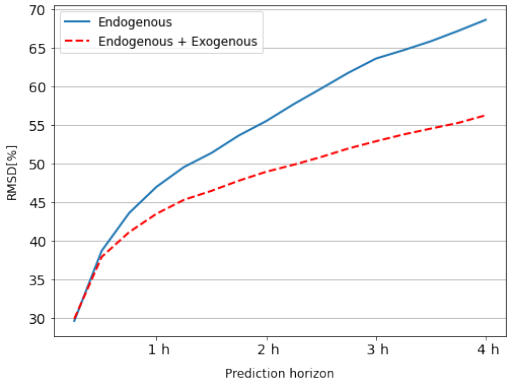
Figure 7: Improvement of exogenous inputs in terms of MAD



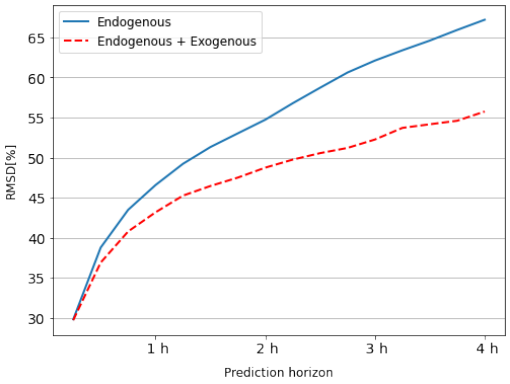
(a) ESN



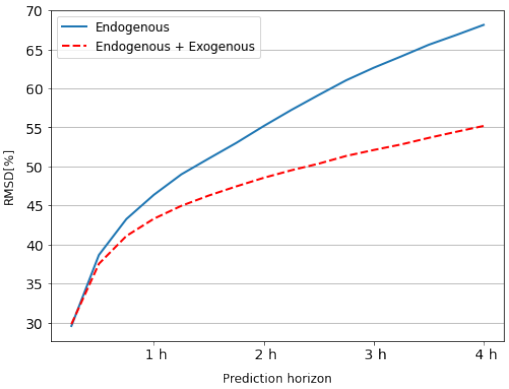
(b) RF



(c) FNN

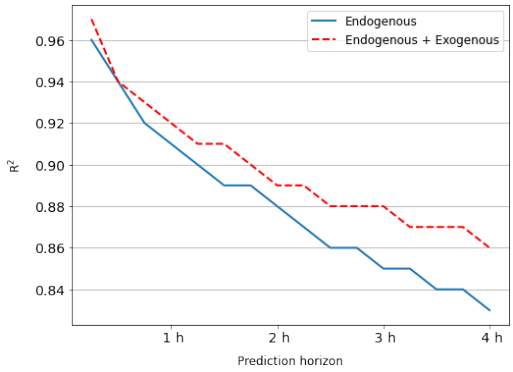


(d) 1D-CNN

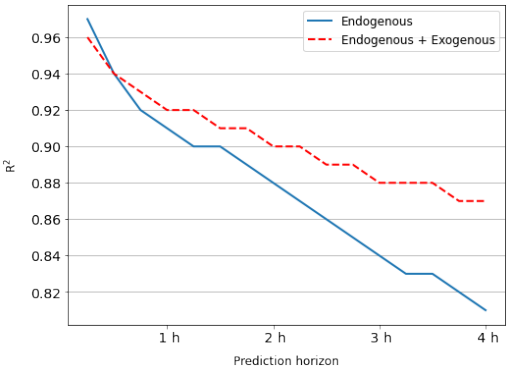


(e) LSTM

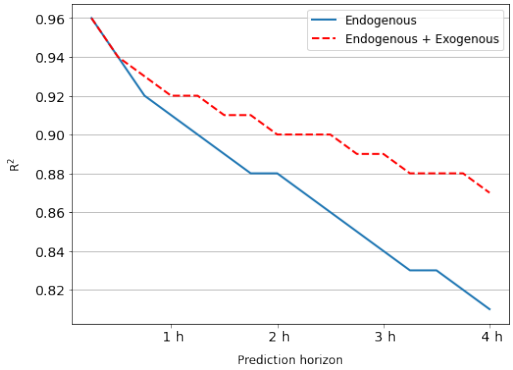
Figure 8: Improvement of exogenous inputs in terms of RMSD



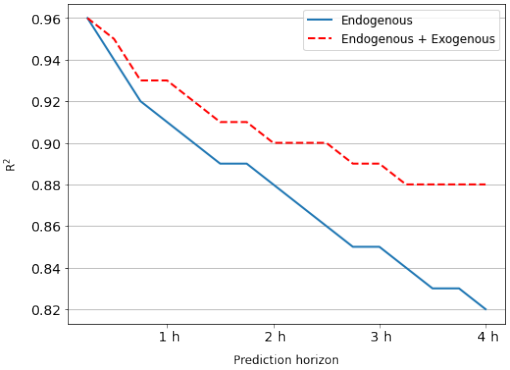
(a) ESN



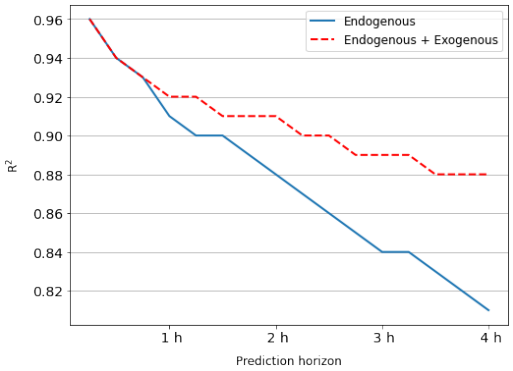
(b) RF



(c) FNN



(d) 1D-CNN



(e) LSTM

Figure 9: Improvement of exogenous inputs in terms of R^2

Table 9: Improvement of exogenous inputs in terms of MAD

	Endogenous + Exogenous					Endogenous					Improvements				
Pred. horizon	RF	FNN	ESN	LSTM	CNN	RF	FNN	ESN	LSTM	CNN	RF	FNN	ESN	LSTM	CNN
15 min	11.40	11.15	10.39	10.74	11.00	9.82	9.73	10.19	10.01	**9.61**	-16.09%	-14.59%	-1.96%	-7.29%	-14.46%
30 min	14.94	14.61	14.79	14.11	**13.90**	14.22	14.22	14.74	14.28	14.04	-5.06%	-2.74%	-0.34%	1.19%	1.00%
45 min	16.90	16.04	17.36	15.96	**15.71**	16.77	16.69	17.47	16.83	16.72	-0.78%	3.89%	0.63%	5.17%	6.04%
60 min	18.37	17.47	19.32	17.25	**17.22**	18.78	18.54	19.52	18.80	18.87	2.18%	5.77%	1.02%	8.24%	8.74%
75 min	19.47	18.62	20.89	18.36	**18.28**	20.50	20.25	21.25	20.49	20.37	5.02%	8.05%	1.69%	10.40%	10.26%
90 min	20.43	19.54	22.21	**19.16**	19.40	21.98	22.07	22.73	22.02	21.68	7.05%	11.46%	2.29%	12.99%	10.52%
105 min	21.32	20.54	23.36	**19.98**	20.32	23.30	23.34	24.07	23.36	23.16	8.50%	12.00%	2.95%	14.47%	12.26%
120 min	22.12	21.20	24.32	20.80	**20.78**	24.67	24.74	25.33	24.68	24.64	10.34%	14.31%	3.99%	15.72%	15.67%
135 min	22.87	21.98	25.10	**21.28**	21.51	25.89	25.88	26.49	25.82	25.82	11.66%	15.07%	5.25%	17.58%	16.69%
150 min	23.57	22.74	25.76	**22.03**	22.44	26.95	27.09	27.44	26.89	27.07	12.54%	16.06%	6.12%	18.07%	17.10%
165 min	24.17	23.44	26.31	**22.66**	22.90	28.02	28.05	28.31	27.91	27.98	13.74%	16.43%	7.06%	18.81%	18.16%
180 min	24.77	23.65	26.74	**22.95**	23.58	28.89	28.91	29.01	28.78	28.83	14.26%	18.19%	7.82%	20.26%	18.21%
195 min	25.31	24.48	27.15	**23.37**	24.89	29.62	29.85	29.66	29.55	29.71	14.55%	17.99%	8.46%	20.91%	16.22%
210 min	25.82	24.84	27.57	**23.83**	25.07	30.33	30.56	30.25	30.28	30.47	14.87%	18.72%	8.86%	21.30%	17.72%
225 min	26.33	25.51	27.97	**24.09**	24.75	30.93	31.08	30.72	30.92	31.01	14.87%	17.92%	8.95%	22.09%	20.19%
240 min	26.84	26.24	28.34	**24.51**	25.02	31.46	31.60	31.21	31.48	31.64	14.69%	16.96%	9.20%	22.14%	20.92%

Table 10: Improvement of exogenous inputs in terms of RMSD

	Endogenous + Exogenous					Endogenous					Improvements				
Pred. horizon	RF	FNN	ESN	LSTM	CNN	RF	FNN	ESN	LSTM	CNN	RF	FNN	ESN	LSTM	CNN
15 min	30.39	29.92	**29.22**	29.81	29.73	29.50	29.65	29.69	29.58	29.83	-3.02%	-0.91%	1.58%	-0.78%	0.34%
30 min	37.95	37.95	37.56	37.54	**36.91**	38.73	38.74	38.84	38.67	38.78	2.01%	2.04%	3.30%	2.92%	4.82%
45 min	41.63	41.16	41.71	41.10	**40.79**	43.34	43.64	43.59	43.27	43.49	3.95%	5.68%	4.31%	5.02%	6.21%
60 min	44.14	43.55	44.51	43.31	**43.18**	46.46	47.04	46.75	46.37	46.58	4.99%	7.42%	4.79%	6.60%	7.30%
75 min	45.87	45.35	46.68	**44.96**	45.26	49.04	49.60	49.29	48.99	49.23	6.46%	8.57%	5.30%	8.23%	8.06%
90 min	47.29	46.51	48.42	**46.28**	46.46	51.12	51.42	51.34	51.02	51.32	7.49%	9.55%	5.69%	9.29%	9.47%
105 min	48.65	47.83	49.96	**47.45**	47.51	53.04	53.69	53.27	53.02	53.03	8.28%	10.91%	6.21%	10.51%	10.41%
120 min	49.90	48.99	51.35	**48.55**	48.75	55.24	55.56	55.20	55.18	54.73	9.67%	11.83%	6.97%	12.02%	10.93%
135 min	51.08	49.90	52.54	**49.51**	49.75	57.29	57.75	56.94	57.23	56.78	10.84%	13.59%	7.73%	13.49%	12.38%
150 min	52.11	50.90	53.65	**50.36**	50.56	59.24	59.77	58.53	59.16	58.72	12.04%	14.84%	8.34%	14.87%	13.90%
165 min	53.07	52.01	54.58	51.35	**51.21**	61.16	61.83	59.94	61.04	60.62	13.23%	15.88%	8.94%	15.87%	15.52%
180 min	53.99	52.94	55.29	**52.11**	52.26	62.73	63.66	61.12	62.64	62.11	13.93%	16.84%	9.54%	16.81%	15.86%
195 min	54.82	53.82	56.01	**52.82**	53.71	64.20	64.74	62.18	64.09	63.39	14.61%	16.87%	9.92%	17.58%	15.27%
210 min	55.62	54.58	56.70	**53.66**	54.16	65.68	65.90	63.17	65.58	64.60	15.32%	17.18%	10.24%	18.18%	16.16%
225 min	56.37	55.32	57.39	**54.44**	54.60	67.01	67.25	63.96	66.83	65.93	15.88%	17.74%	10.27%	18.54%	17.18%
240 min	57.14	56.31	58.09	**55.19**	55.75	68.27	68.68	64.78	68.13	67.20	16.30%	18.01%	10.33%	18.99%	17.04%

Table 11: Improvement of exogenous inputs in terms of R^2

	Endogenous + Exogenous					Endogenous					Improvements				
Pred. horizon	RF	FNN	ESN	LSTM	CNN	RF	FNN	ESN	LSTM	CNN	RF	FNN	ESN	LSTM	CNN
15 min	0.96	0.96	0.97	0.96	0.96	**0.97**	0.96	0.96	0.96	0.96	-1.03%	0.00%	1.04%	0.00%	0.00%
30 min	0.94	0.94	0.94	0.94	**0.95**	0.94	0.94	0.94	0.94	0.94	0.00%	0.00%	0.00%	0.00%	1.06%
45 min	0.93	0.93	0.93	0.93	0.93	0.92	0.92	0.92	**0.93**	0.92	1.09%	1.09%	1.09%	0.00%	1.09%
60 min	0.92	0.92	0.92	0.92	**0.93**	0.91	0.91	0.91	0.91	0.91	1.10%	1.10%	1.10%	1.10%	2.20%
75 min	**0.92**	0.92	0.91	0.92	0.92	0.90	0.90	0.90	0.90	0.90	2.22%	2.22%	1.11%	2.22%	2.22%
90 min	**0.91**	0.91	0.91	0.91	0.91	0.90	0.89	0.89	0.90	0.89	1.11%	2.25%	2.25%	1.11%	2.25%
105 min	**0.91**	0.91	0.90	0.91	0.91	0.89	0.88	0.89	0.89	0.89	2.25%	3.41%	1.12%	2.25%	2.25%
120 min	0.90	0.90	0.89	**0.91**	0.90	0.88	0.88	0.88	0.88	0.88	2.27%	2.27%	1.14%	3.41%	2.27%
135 min	**0.90**	0.90	0.89	0.90	0.90	0.87	0.87	0.87	0.87	0.87	3.45%	3.45%	2.30%	3.45%	3.45%
150 min	0.89	**0.90**	0.88	0.90	0.90	0.86	0.86	0.86	0.86	0.86	3.49%	4.65%	2.33%	4.65%	4.65%
165 min	**0.89**	0.89	0.88	0.89	0.89	0.85	0.85	0.86	0.85	0.85	4.71%	4.71%	2.33%	4.71%	4.71%
180 min	0.88	**0.89**	0.88	0.89	0.89	0.84	0.84	0.85	0.84	0.85	4.76%	5.95%	3.53%	5.95%	4.71%
195 min	0.88	0.88	0.87	**0.89**	0.88	0.83	0.83	0.85	0.84	0.84	6.02%	6.02%	2.35%	5.95%	4.76%
210 min	**0.88**	0.88	0.87	0.88	0.88	0.83	0.83	0.84	0.83	0.83	6.02%	6.02%	3.57%	6.02%	6.02%
225 min	0.87	**0.88**	0.87	0.88	0.88	0.82	0.82	0.84	0.82	0.83	6.10%	7.32%	3.57%	7.32%	6.02%
240 min	0.87	0.87	0.86	0.88	**0.88**	0.81	0.81	0.83	0.81	0.82	7.41%	7.41%	3.61%	8.64%	7.32%

6. Conclusions

In this work, we evaluated the effectiveness of using exogenous inputs for short-term solar radiation forecasting. To this aim, we identified a subset of relevant input variables for predicting solar radiation by applying different feature selection techniques to a larger set of variables. The results of feature selection revealed that the most significant input variables for predicting solar radiation are UV index, cloud cover, air temperature, relative humidity, dew point, wind bearing, sunshine duration and hour of the day. On the other hand, features like precipitation intensity, precipitation probability, wind speed, sea-level air pressure, day of the year and minute of the hour were discarded. To assess the usefulness of the selected features, we evaluated and compared the prediction performance of five different machine learning models, namely a Feedforward Neural Network (FNN), an Echo State Network (ESN), a 1D Convolutional Neural Network (1D-CNN), a Long Short-Term Memory (LSTM) neural network and a Random Forest (RF). Overall, the LSTM demonstrated the best prediction performance among the five models, producing acceptable forecasting errors up to 4 h ahead. The 1D-CNN exhibited forecasting performance comparable to those of the LSTM for prediction horizons shorter than 2 h. The FNN demonstrated also very good

prediction performance, but definitely lower than the ones achieved by the LSTM and the 1D-CNN. The ESN presented the best performance for very short-term predictions (15 min), but presented the highest forecasting errors for larger forecasting horizons, revealing poor prediction performance in modelling multivariate time series. The RF performed slightly better than the ESN, showing promising results. Finally, in order to demonstrate the effectiveness of using exogenous inputs for short-term solar radiation forecasting, we compared the multivariate models with their univariate counterparts. The results showed that the adoption of exogenous inputs can significantly improve the forecasting performance for prediction horizons greater than 15 min, while for shorter prediction horizons the performance improvement due to exogenous inputs can be considered as negligible. Overall, the results demonstrated the effectiveness of using exogenous inputs for short-term solar radiation forecasting.

In this study, we implemented different feature selection techniques able to identify the most relevant input variables for short-term solar radiation forecasting. However, we considered a limited number of variables, due to the lack of availability of high quality meteorological data with minutely frequency. We believe that the model can be further improved by introducing additional features such as real-time weather forecast and other atmospheric components. Furthermore, we plan to improve our methodology by exploiting also spatial information and considering meteorological observations from neighboring locations.

List of acronyms

1D-CNN	1 Dimensional Convolutional Neural Network
ANN	Artificial Neural Network
ARIMA	Autoregressive Integrated Moving Average
ESN	Echo State Network
ESP	Echo State Property
FNN	Feedforward Neural Network
GHI	Global Horizontal Irradiance
IEA	International Energy Agency

LASSO	Least Absolute Shrinkage and Selection Operator Regression
LSTM	Long Short-Term Memory
MAD	Mean Absolute Deviation
MI	Mutual Information
MIMO	Multiple-Input Multiple-Output
MSE	Mean Square Error
nRMSE	Normalized Root Mean Square Error
PV	Photovoltaic
R^2	Coefficient of Determination
RF	Random Forest
RMSD	Root Mean Square Deviation
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
SBS	Sequential Backward Selection
SFS	Sequential Forward Selection
SVR	Support Vector Machine
UV	Ultraviolet
VRE	Variable Renewable Energy

References

Albadi, M. (2016). Solar pv power intermittency and its impacts on power systems–an overview. *Sola*, *2012*, 2018.

Aliberti, A., Bottaccioli, L., Cirrincione, G., Macii, E., Acquaviva, A., & Patti, E. (2018a). Forecasting short-term solar radiation for photovoltaic energy predictions. In *SMARTGREENS* (pp. 44–53).

Aliberti, A., Bottaccioli, L., Cirrincione, G., Macii, E., Acquaviva, A., & Patti, E. (2018b). Non-linear autoregressive neural networks to forecast short-term solar radiation for photovoltaic energy predictions. In *Smart Cities, Green Technologies and Intelligent Transport Systems* (pp. 3–22). Springer.

Ben Taieb, S., Sorjamaa, A., & Bontempi, G. (2010). Multiple-output modeling for multi-step-ahead time series forecasting. *Neurocomput.*, *73*, 1950–1957. URL: http://dx.doi.org/10.1016/j.neucom.2009.11.030. doi:10.1016/j.neucom.2009.11.030.

Benali, L., Notton, G., Fouilloy, A., Voyant, C., & Dizene, R. (2019). Solar radiation forecasting using artificial neural network and random forest methods: Application to normal beam, horizontal diffuse and global components. *Renewable Energy*, *132*, 871 – 884. URL: http://www.sciencedirect.com/science/article/pii/S0960148118309947. doi:https://doi.org/10.1016/j.renene.2018.08.044.

Berger, L. T., & Iniewski, K. (2012). *Smart Grid Applications, Communications, and Security*. John Wiley and Sons.

Bontempi, G., Taieb, S. B., & Le Borgne, Y.-A. (2012). Machine learning strategies for time series forecasting. In *European business intelligence summer school* (pp. 62–77). Springer.

Bottaccioli, L., Patti, E., Macii, E., & Acquaviva, A. (2017). Gis-based software infrastructure to model pv generation in fine-grained spatio-temporal domain. *IEEE Systems Journal*, *PP*, 1–10. doi:10.1109/JSYST.2017.2726350.

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5–32.

Chollet, F. et al. (2015). Keras. URL: https://keras.io (Accessed on: 2019-06-01).

Diagne, H. M., David, M., Lauret, P., Boland, J., & Schmutz, N. (2013). Review of solar irradiance forecasting methods and a proposition for small-scale insular grids. *Renewable and Sustainable Energy Reviews*, *27*, 65 – 76. URL: https://hal.archives-ouvertes.fr/hal-01090087. doi:10.1016/j.rser.2013.06.042.

Esfe, M. H., Razi, P., Hajmohammad, M. H., Rostamian, S. H., Sarsam, W. S., Arani, A. A. A., & Dahari, M. (2017). Optimization, modeling and accurate prediction of thermal conductivity and dynamic viscosity of stabilized ethylene glycol and water mixture al2o3 nanofluids by nsga-ii using ann. *International Communications in Heat and Mass Transfer*, *82*, 154–160.

Grossman, A., & Turner, J. (2019). Dark sky. URL: `https://darksky.net` (Accessed on: 2019-06-01).

Gueymard, C. (2014). A review of validation methodologies and statistical performance indicators for modeled solar radiation data: Towards a better bankability of solar projects. *Renewable and Sustainable Energy Reviews*, *39*, 1024–1034. doi:`10.1016/j.rser.2014.07.117`.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *J. Mach. Learn. Res.*, *3*, 1157–1182. URL: `http://dl.acm.org/citation.cfm?id=944919.944968`.

Hassan, M. A., Khalil, A., Kaseb, S., & Kassem, M. (2017). Exploring the potential of tree-based ensemble methods in solar radiation modeling. *Applied Energy*, *203*, 897 – 916. URL: `http://www.sciencedirect.com/science/article/pii/S0306261917308437`. doi:`https://doi.org/10.1016/j.apenergy.2017.06.104`.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*, 1735–1780. URL: `https://doi.org/10.1162/neco.1997.9.8.1735`. doi:`10.1162/neco.1997.9.8.1735`. `arXiv:https://doi.org/10.1162/neco.1997.9.8.1735`.

IEA (2019). Renewables 2019. URL: `https://www.iea.org/reports/renewables-2019`.

Ineichen, P., & Perez, R. (2002). A new airmass independent formulation for the linke turbidity coefficient. *Solar Energy*, *73*, 151–157. doi:`10.1016/S0038-092X(02)00045-2`.

Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D. J. (2019). 1D Convolutional Neural Networks and Applications: A Survey. *arXiv e-prints*, (p. arXiv:1905.03554). `arXiv:1905.03554`.

Lauret, P., Voyant, C., Soubdhan, T., David, M., & Poggi, P. (2015a). A benchmarking of machine learning techniques for solar radiation forecasting in an insular context. *Solar Energy*, *112*, 446–457.

Lauret, P., Voyant, C., Soubdhan, T., David, M., & Poggi, P. (2015b). A benchmarking of machine learning techniques for solar radiation forecasting in an insular context. *Solar Energy*, *112*, 446 – 457. URL: http://www.sciencedirect.com/science/article/pii/S0038092X14006057. doi:https://doi.org/10.1016/j.solener.2014.12.014.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, *50*, 1–45.

McCandless, T., Haupt, S., & Young, G. (2016). A regime-dependent artificial neural network technique for short-range solar irradiance forecasting. *Renewable Energy*, *89*, 351 – 359. URL: http://www.sciencedirect.com/science/article/pii/S0960148115305346. doi:https://doi.org/10.1016/j.renene.2015.12.030.

Mills, A., Ahlstrom, M., Brower, M., Ellis, A., George, R., Hoff, T., Kroposki, B., Lenox, C., Miller, N., Stein, J., & Wan, Y.-H. (2009). Understanding variability and uncertainty of photovoltaics for integration with the electric power system.

Monjoly, S., André, M., Calif, R., & Soubdhan, T. (2017). Hourly forecasting of global solar radiation based on multiscale decomposition methods: A hybrid approach. *Energy*, *119*, 288 – 298. URL: http://www.sciencedirect.com/science/article/pii/S0360544216316668. doi:https://doi.org/10.1016/j.energy.2016.11.061.

Palage, K., Lundmark, R., & Söderholm, P. (2019). The innovation effects of renewable energy policies and their interaction: the case of solar photovoltaics. *Environmental Economics and Policy Studies*, *21*, 217–254.

Paoli, C., Voyant, C., Muselli, M., & Nivet, M.-L. (2010). Forecasting of preprocessed daily solar radiation time series using neural networks. *Solar Energy*, *84*, 2146 – 2160. URL: http://www.sciencedirect.com/science/article/pii/S0038092X10002793. doi:https://doi.org/10.1016/j.solener.2010.08.011.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python . *Journal of Machine Learning Research*, *12*, 2825–2830.

Pedro, H. T., & Coimbra, C. F. (2012). Assessment of forecasting techniques for solar power production with no exogenous inputs. *Solar Energy*, *86*, 2017 – 2028. URL: http://www.sciencedirect.com/science/article/pii/S0038092X12001429. doi:https://doi.org/10.1016/j.solener.2012.04.004.

Pudil, P., Novovičová, J., & Kittler, J. (1994). Floating search methods in feature selection. *Pattern recognition letters*, *15*, 1119–1125.

Qing, X., & Niu, Y. (2018). Hourly day-ahead solar irradiance prediction using weather forecasts by lstm. *Energy*, *148*, 461 – 468. URL: http://www.sciencedirect.com/science/article/pii/S0360544218302056. doi:https://doi.org/10.1016/j.energy.2018.01.177.

Rana, M., Koprinska, I., & Agelidis, V. (2016). Univariate and multivariate methods for very short-term solar photovoltaic power forecasting. *Energy Conversion and Management*, *121*, 380–390. doi:10.1016/j.enconman.2016.05.025.

Reikard, G. (2009). Predicting solar radiation at high resolutions: A comparison of time series forecasts. *Solar Energy*, *83*, 342 – 349. URL: http://www.sciencedirect.com/science/article/pii/S0038092X08002107. doi:https://doi.org/10.1016/j.solener.2008.08.007.

Reno, M., Hansen, C., & Stein, J. (2014). Global horizontal irradiance clear sky models : implementation and analysis. *IEEE Systems Journal*, . doi:10.2172/1039404.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, *27*, 379–423.

Shumway, R. H., & Stoffer, D. S. (2017). *Time Series Analysis and Its Applications*. Springer International Publishing.

Siano, P. (2014). Demand response and smart grids—a survey. *Renewable and Sustainable Energy Reviews*, *30*, 461 – 478. URL: http://www.sciencedirect.com/science/article/pii/S1364032113007211. doi:https://doi.org/10.1016/j.rser.2013.10.022.

Sugihara, H., Funaki, T., & Yamaguchi, N. (2017). Evaluation method for real-time dynamic line ratings based on line current variation model for representing forecast error of intermittent renewable generation. *Energies*, *10*, 503.

Tibshirani, R. (1996). Regression selection and shrinkage via the lasso. *Journal of the Royal Statistical Society Series B*, *58*, 267–288.

Vindel, J., & Polo, J. (2014). Intermittency and variability of daily solar irradiation. *Atmospheric Research*, *143*, 313 – 327. URL: http://www.sciencedirect.com/science/article/pii/S0169809514001252. doi:https://doi.org/10.1016/j.atmosres.2014.03.001.

Voyant, C., Muselli, M., Paoli, C., & Nivet, M.-L. (2011). Optimization of an artificial neural network dedicated to the multivariate forecasting of daily global radiation. *Energy*, *36*, 348 – 359. URL: http://www.sciencedirect.com/science/article/pii/S0360544210005955. doi:https://doi.org/10.1016/j.energy.2010.10.032.

Voyant, C., Notton, G., Kalogirou, S., Nivet, M.-L., Paoli, C., Motte, F., & Fouilloy, A. (2017). Machine learning methods for solar radiation forecasting: A review. *Renewable Energy*, *105*, 569 – 582. URL: http://www.sciencedirect.com/science/article/pii/S0960148116311648. doi:https://doi.org/10.1016/j.renene.2016.12.095.

Zimmermann, R. (2017). easyesn. URL: https://github.com/kalekiu/easyesn (Accessed on: 2019-06-01).