

PAPER • OPEN ACCESS

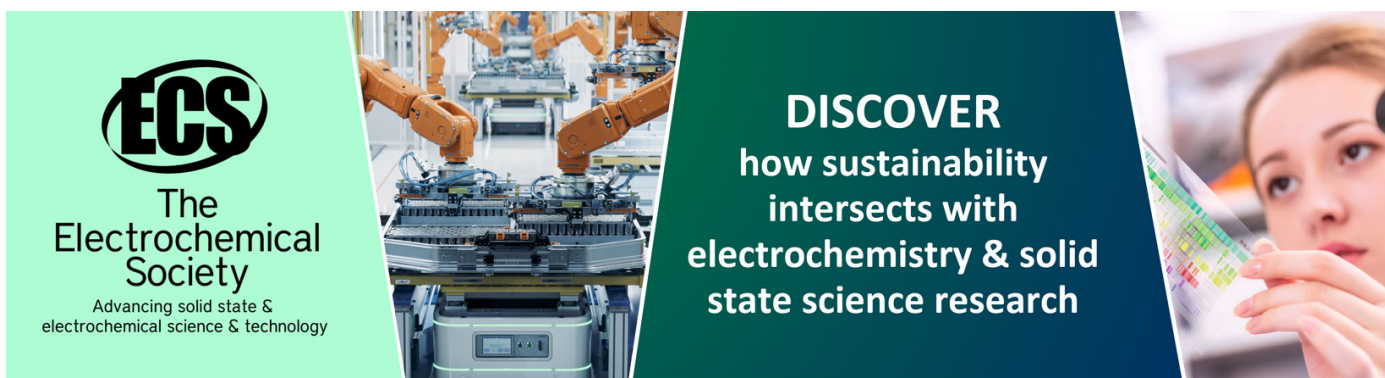
Forecasting solar radiation using a deep *long short-term memory* artificial neural network

To cite this article: Mihail-Bogdan Carutasu *et al* 2021 *IOP Conf. Ser.: Earth Environ. Sci.* **664** 012043

View the [article online](#) for updates and enhancements.

You may also like

- [Cloud Server Oriented FPGA Accelerator for Long Short-Term Memory Recurrent Neural Networks](#)
Jiasheng Wang, Yu Zhou, Yuyang Sun et al.
- [A sequence-to-sequence remaining useful life prediction method combining unsupervised LSTM encoding-decoding and temporal convolutional network](#)
Jialin Li, Renxiang Chen and Xianzhen Huang
- [Research on remaining useful life of rolling bearings using EWT-DI-ALSTM](#)
Runxia Guo and Bo Gong



ECS
The
Electrochemical
Society
Advancing solid state &
electrochemical science & technology

DISCOVER
how sustainability
intersects with
electrochemistry & solid
state science research

Forecasting solar radiation using a deep *long short-term memory* artificial neural network

Mihail-Bogdan Carutasu^{1,*}, Alin Ionescu¹, Constantin Ionescu¹, Horia Necula¹

¹Power Engineering Faculty, University POLITEHNICA of Bucharest, Romania

Corresponding author: carutasu@gmail.com

Abstract. Solar systems are widely used to mitigate the environmental impact of the energy sector and their importance has constantly increased due to the recent EU's strategy to lower the CO₂ emissions. Moreover, the newest Energy of Buildings Directive empathises the importance of producing energy from renewable sources to decrease the overall impact of buildings over the total end-use energy consumption. Generally, the systems' performances are highly correlated with the incident solar radiation and outdoor air temperature. Thus, being able to accurately forecast these two parameters represents a vital step in dimensioning and maximizing the overall energy production. This Paper presents the results obtained by implementing a deep recurrent artificial neural network (ANN) trained with on year solar radiation data harvested from the UPB campus. The time series data was modelled using a special ANN architecture – the LSTM (Long Short-Term Memory) – due to its special designed internal 'memory' which increases its capabilities of predicting temporal sequence data. The model uses sequences of 24 hours and the resulted mean squared error (mse) for both training and validation data is under 30%.

1. Introduction

The importance of increasing the portion of renewable energy usage to mitigate the energy intensity of buildings is highlighted in the newest EU's energy strategies (Energy Performances of Buildings Directive – Directive 2018/844 and Renewable Energy Directive – Directive 2018/2001/EU) [1, 2]. As generally defined in [3] achieving nZEB (nearly Zero Energy Building) standards cannot be done without solar systems, as they have great building integrability and the solar industry is mature enough to provide efficient systems [2].

The solar systems' performances are highly influenced by the incident solar radiation characterised by great volatility due to weather conditions' randomness. Even though local or national meteorological organization provide estimations of some weather data, mostly in form of next day forecasts of outside's air temperature and humidity, sky's clearness, wind speed, etc, they provide little or no suggestive information of solar irradiance. Being able to accurately predict and forecast the solar radiation leads to properly designed solar (PV and thermal) systems to assure quality power to the electrical grid. Moreover, this has a direct impact over smart-grids' efficiency as it allows the stakeholders to efficient develop demand-supply energy strategies. Moreover, an accurate forecast model is a prerequisite for optimizing the functioning of solar panels and design control tools.

Due to these implications, models for estimating the solar radiation have been widely developed in the literature and can be classified in pure physical, statistical, and artificial intelligence methods [4]. The latter category implies machine learning or, even better, deep learning techniques which requires labelled data from which the algorithm can 'learn' and develop generalization capabilities. Deep learning techniques comprise Artificial Neural Networks (ANN) among other techniques and are widely used in classification, objects and face recognition and general regression problems [5] When the predicted feature is a real value, as in the case of solar radiation, the problem becomes a regression problem, and can be tackled using various ANN's architectures. For this paper, a relatively new recurrent ANN (LSTM – Long Short-Term Memory) was used, mainly due to its internal 'memory' which makes it appropriate for tackling long time dependency timeseries data. The LSTM architecture is described in Chapter 3.



2 Data description and preparation

For this paper data collected by a pyranometer mounted on the roof of UPB's passive house was used. Details about the construction, data acquisition system and results can be found in [6, 7]. The data was harvested for one year, with hourly timestep. After analysing the data, it could be found that the pyranometer had a small drawback and during the night it transmitted a small signal which translates into negative solar radiation values (Figure 1). Thus, the negative values recorded during those periods were shifted to 0 W/m².

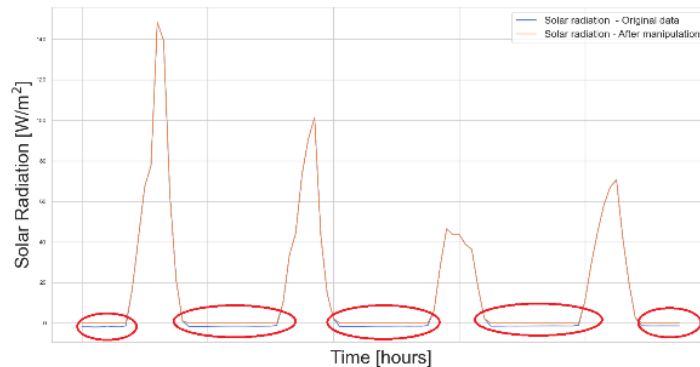


Figure 1. Solar Radiation for the first 96 hours before and after manipulation

The device was not operational for several hours; thus, no data was recorded, resulting in just 8524 datapoints instead of 8760. A brief statistical description of the used data can be found in Table 1, where: *count* – total non-null values, *mean* – mean value, *std* – standard deviation, *min/max* – minimum and maximum values, 25%, 50% and 75% - percentiles (a theoretical raw score which corresponds to a given percentile rank in a specified distribution [8]).

Table 1. Solar radiation descriptive statistics

count	8524.000
mean	151.838
std	255.350
min	0.000
25%	0.000
50%	2.608
75%	186.251
max	1086.397

Descriptive statistics include those that summarize the central tendency, dispersion, and shape of a dataset's distribution. As can be observed, the value presents high variation (from 0 to 1086 W/m²) implying high dispersion. To use them in deep learning algorithms and forecasting problems, the data must be rescaled, usually normalized or standardised. This is very useful when computing the gradient descent algorithm, which, in fact, is used as optimizer in machine learning field and minimizes the error between model's output and real data; thus, the forecasting capabilities of the neural net is, first, impacted by the scale of the data.

Given the fact that there should be no negative values, the data was scaled using the min-max scaler algorithm [9, 10] and the new values were computed as follows:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \cdot (\max - \min) + \min \quad (1)$$

where, in this case, min = 0 and max = 1 and the fraction is the standard deviation of the data. Scaled data variation is represented in Figure 2:

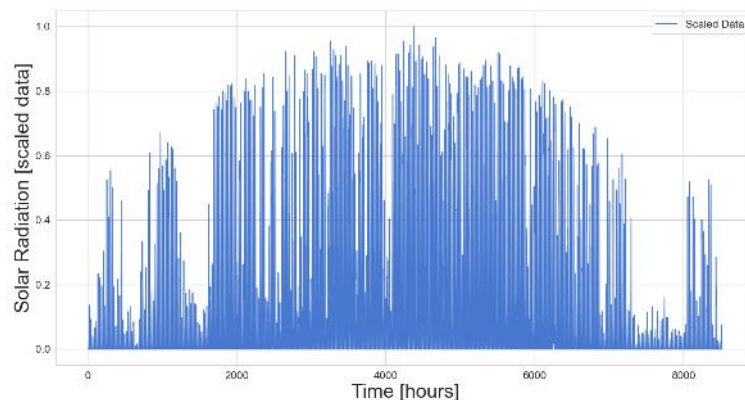


Figure 2. Scaled data

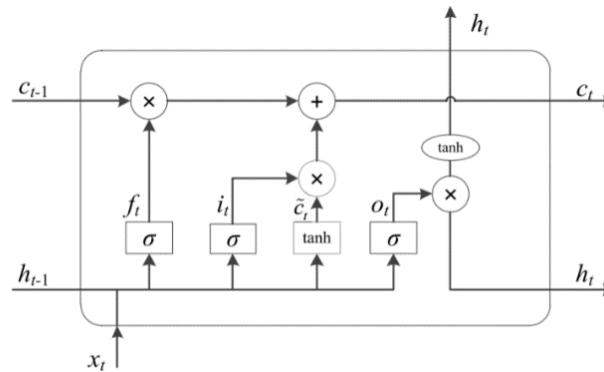
After scaling, we split the data in two: a part to be fed to the ANN in the training period and a part to be used to test the model's accuracy. Given the fact that the time dependency is crucial in this type of problem, the data has not been shuffled. Therefore, the model can learn the hourly, daily and seasons variations. Thus, data was divided as follows: **training data** – 80% (from which 10% represents the **testing data**) and **validation data** – 20%.

3. Model development

Any data observed sequentially over time is called time series data [11]. Moreover, if only one value is used to forecast itself, the series is called univariate time. To model and forecast this type of data, numerous algorithms are presented in the literature, ranging from simple (average, naïve, seasonal naïve, drift) to complex (linear regression, smoothing, ARIMA, vector autoregression – VAR), and extremely complex models based on artificial intelligence (through deep neural networks) [11].

Recurrent neural networks (RNNs) are designed with a memory cell which allows them to keep track of short time memory by updating this cell at each considered time step [12]. These features allow RNNs to model sequential data by predicting the next real value or to classifying categorical data. However, these architectures of ANNs have a problem in 'remembering' long time dependencies in sequential data due to decaying error gradients (gradient vanishing issue); these usually implies getting stuck around local minima instead of global minimal error, decreasing the model's efficiency [13].

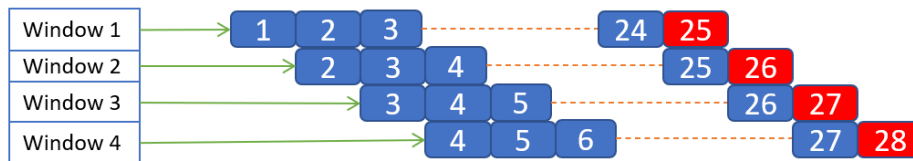
These drawbacks were limited once a new architecture of RNN – long short-term memory (LSTM) – was presented in 1997 by S. Hochreiter and J. Schmidhuber [14]. The LSTM structure presents an extra cell, called 'memory cell' or 'cell state' which adds an extra weight to the passing-through data, giving the model the capability to 'forget' irrelevant data. The functionality of the LSTM neuron is assured by its multistep process: first, the data flow through the forget gate, where a logistic sigmoid activation function decides what information is kept; secondly, the input gate's tanh activation function assures that relevant data is stored in the memory cell; last, the output is produced by passing the pre-processed information through a last activation function and the output is produced [15]. An example on an LSTM neuron is present in Figure 3.

**Figure 3.** LSTM internal architecture

[from https://commons.wikimedia.org/wiki/File:Long_Short_Term_Memory.png]

These complex continuous computations result in very high dimension matrices (or tensors) which can be manipulated using the TensorFlow library [16] developed by Google and used for creating deep ANNs. Even more, an extreme useful API was developed by François Chollet [5] and named *Keras* (*κέρας*), after the Greek horn of abundance [17]. These tools were used in conjunction with Python to develop the Sequential model used to forecast the solar radiation data.

When implementing the LSTM architecture in TensorFlow and Keras, the data used as inputs must be reshaped as an 3D tensor in the following shape: batch x timesteps x features [17]. The batch size is treated automatically by Keras in the training period and is not a required hyperparameter when first creating the model. However, the batch size was considered equal to 64 hours; the timestep of 24 hours (as the data has a 24-hour period; there is only one feature (the solar radiation value at a given timestep). Moreover, we intent to forecast the value of the 25th hour by using the previous 24 hours and so one, generating a dynamic sliding window of values. The first 4 sliding windows are depicted in Figure 4, where the red boxes represent the forecasts; the forecasted value from first window represents the 24th known value for the second window and so on.

**Figure 4.** Sequential data forecast for 4 windows

Considering all these facts, we built a deep network with the following configuration:

Table 2. LSTM network's configuration

Type of layer	Number of neurons	Activation function
LSTM	132	'relu'
Dropout	0.2	-
LSTM	64	'relu'
Dropout	0.2	-
LSTM	13	'relu'
Dense	1	'relu'

The input data flows through the first LSTM layer consisting in 132 neurons and after being processed here, it serves as input for the second layer: Dropout layer; this is a very useful tool when

training a neural network with large datasets because it prevents the network to overfit. Dropping out a randomly number of a layer's output features in the training process is considered the most effective regularization techniques [5]. The data flow is then processed through two LSTM layers (64 and 13 neurons), two 20% Dropout layer) and a Dense layer which provides the output.

After establishing the configuration, the model is compiled and trained. Compiling the model means choosing the right optimization technique; it used to minimize the error between the model's output and the real data. For this type of model, Adam optimizer with 0.0013 learn was chosen to find the global mean squared error (mse).

The training process uses the 80% from the data, 20 epochs (an epoch means that all the data is process by the network) and a batch size of 168 samples to find the best weights that satisfies the global minimal error. Tuning all the hyperparameters described in this section was made through several tries-and-error analysis.

4. Results

The first thig plotted after the optimization and training processes was the loss values for both training and testing data. The model is trained on 6881samples and validated on 765 samples; the loss values were computed after each epoch and are presented in Figure 5.

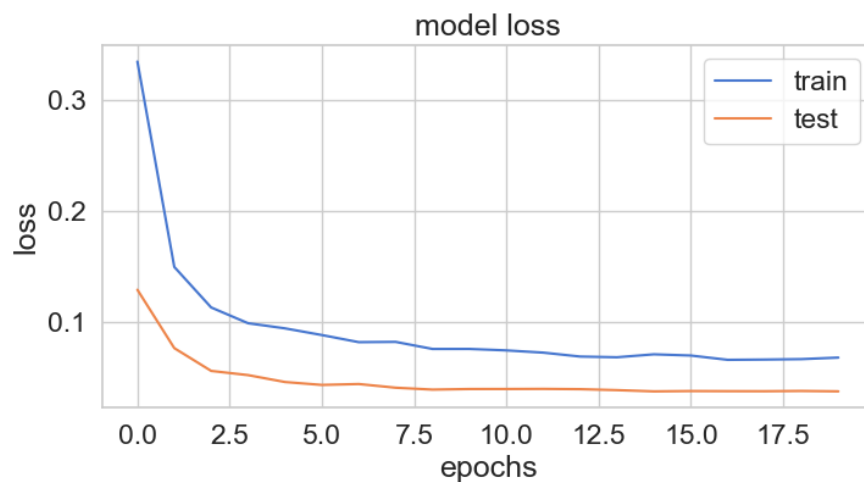


Figure 5. Loss values for both training and testing data for each of the 20 epochs considered

As observed, both losses have a descendent trend and over or underfitting did not occurred. As expected, the highest loss value is at the first epoch, when the computed value was 0.335; the lowest loss value occurred for the testing dataset at the last epoch (0.038). As the graph suggests, after the tenth epoch, there are no remarkable decrease in the loss value, therefore the model would have the same accuracy even if trained less. Matter of fact, the optimum number of epochs is 13. It is worth mentioning that the simulation time improvement is out of the scope of this Paper.

After analysing the training and testing errors, we plotted the next hour forecast (hour number 6882); this value was not seen by the ANN, because it is the first value from the validation dataset. We also compared it with the real value and the difference is presented in Figure 6. The absolute error between the forecasted value and the real value is 13 W/m², equivalent to 32% relative error.

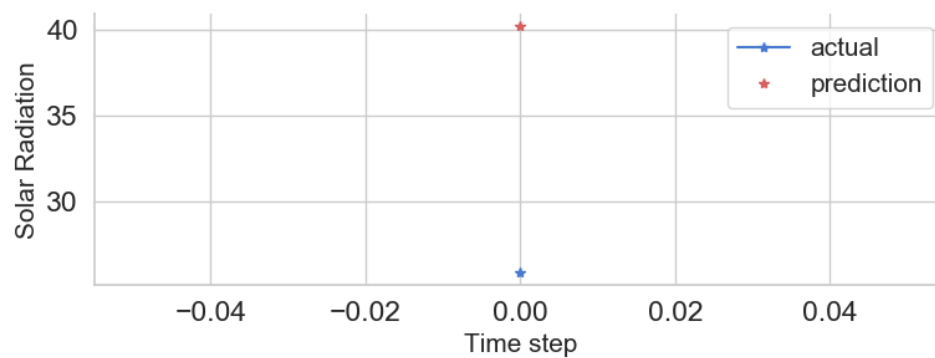


Figure 6. Forecasted vs. real value for the 6882nd hour

The forecast for 72 hours period is depicted in Figure 7, while the values for a 144-hour period are presented in Figure 8.

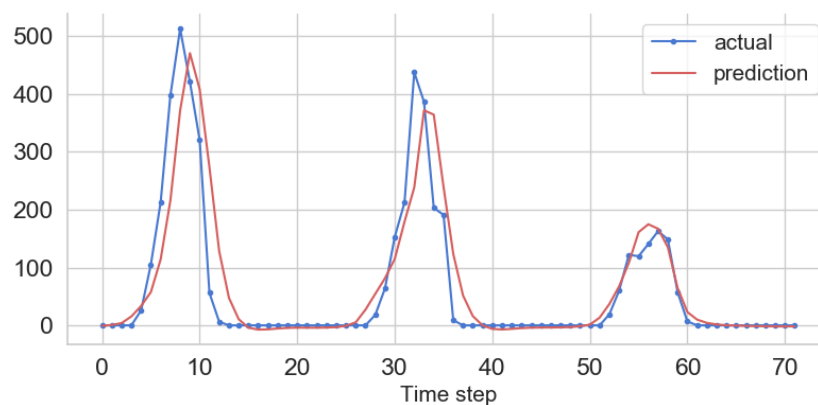


Figure 7. Comparison between forecasted and real data for the first 72 hours form the validation dataset

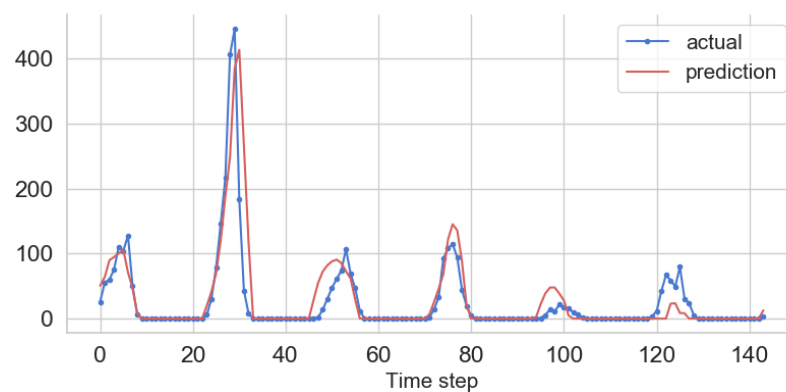


Figure 8. Comparison between forecasted and real data for the next 1442 hours form the validation dataset

In figure 9 it is presented the last values of the training set, considered to be the history data and all the forecasted values – on unseen data; in the lower part is presented the comparison between real and forecasted data. As can be seen, the model proposed algorithm shows very good accuracy and captures well daily and seasonal solar radiation.

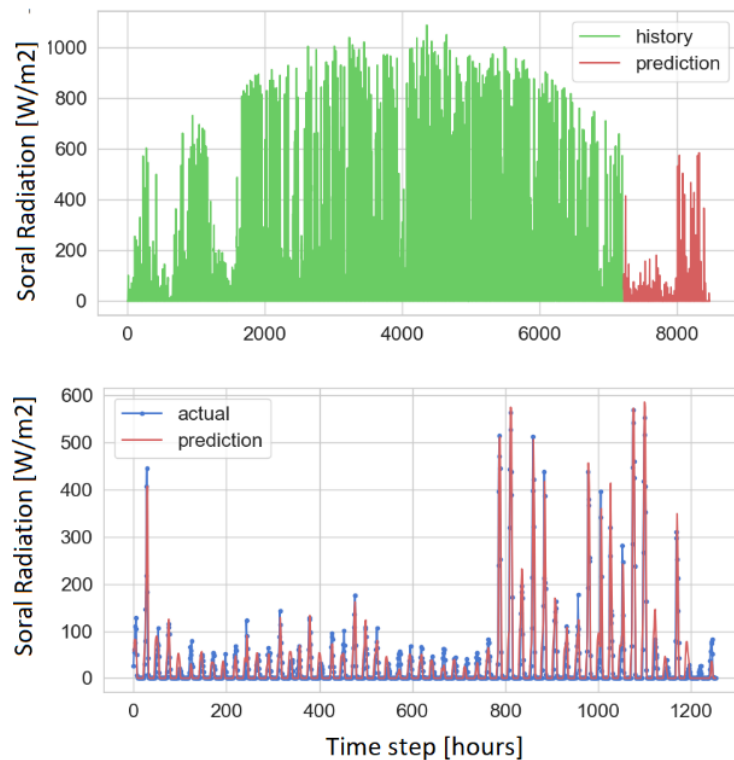


Figure 9. Comparison between the validation dataset and forecasted solar radiation values.

For the entire data set, the train and validation errors ware:

- **Train** Mean Absolute Error: 40.70 %
- **Train** Root Mean Squared Error: 79.34 %
- **Test** Mean Absolute Error: 16.40 %
- **Test** Root Mean Squared Error: 37.89 %

5 Conclusions and future work

This paper presents an end-to-end project on timeseries forecast using a special designed type of artificial neuronal network – long short-term memory (LSTM). Even if there are several related Papers in the literature, no one presents the full algorithm on how to prepare the data and how to set the network's hyperparameters; these are the most important factors in the research's reproducibility.

The LSTM network was used to forecast the solar radiation using a data harvested on the University POLITEHNICA of Bucharest Campus. The proposed model showed very good accuracy if data was pre-processed by eliminating the nonvalues and scaling it to values between 0 and 1; this was achieved using a min-max scaler algorithm. Using Python programming language and TensorFlow and Keras APIs, we built a deep neural network with one input layer, three LSTM hidden layers (each one with a Dropout layer) and one output layer. This model forecasted every 25th value from a 24-value sliding window and the loss accuracy for both training and testing data was under 0.3; in addition, the loss for the test dataset was lower than the training dataset, meaning that the model accurately learnt the daily solar radiation path. The plots of validation data set vs. forecasts highlights the model's efficiency.

Next, we will extend the model to forecast the outdoor air temperature and the soil temperature using data collected from the same site. This can improve even mode the forecast of the solar radiation, as there is a tight connection between these tree weather parameters. Moreover, if the future model will provide good accuracy for the soil temperature, it will be used in analysing the energy production of different heat pumps or air-to-ground heat exchangers (for example Canadian wells).

Acknowledgement

This work was supported by the Operational Programme Human Capital of the Ministry of European Funds through the Financial Agreement “Developing the entrepreneurial skills of the PhD students and postdoctoral students - key to career success (A-Succes)” contract number 51675/09.07.2019 POCU/380/6/13, SMIS code 125125.

References

- [1] Directie (EU) 2018/844 of the European Parliament and of the Council of 30 May 2018 amending Directive 2010/31/EU on the energy performance of buildings and Directive 2012/27/EU on energy efficiency.
- [2] Directive (EU) 2018/2001 of the European Parliament and of the Council of 11 December 2018 on the promotion of the use of energy from renewable sources
- [3] Directive 2010/31/EU of the European Parliament and of the Council 19 May on the Energy Performances of Buildings (Recast) (2010)
- [4] Aggarwal S K and Saini L M 2014 Solar energy prediction using linear and non-linear regularization models: a study on AMS (American Meteorological Society) 2013-14 Solar energy prediction contest *Energy* **78** 247-256
- [5] F Chollet 2018 Deep Learning with Python
- [6] Carutasu M B, Tanasiev V, Ionescu C, Danu A, Necula H and Badea A 2015 Reducing energy consumption in low energy buildings through implementation of a policy system used in automated heating systems *Energy Build.* **94** 227-239
- [7] Carutasu M B, Ionescu C and Necula H 2016 The influence of genetic algorithm parameters over the efficiency of the energy consumption estimation in a low-energy building *Energy Procedia* **85** 99-108
- [8] Weinberg S L and Abramowitz S. K 2008 Statistics using SPSS: an integrative approach
- [9] Khare K, Darekar O, Gupta P and Attar V Z 2017 Short term stock price prediction using deep learning *2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)* 482-486
- [10] scikit-learn preprocessing algorithm. Available at: <https://scikit-learn.org/>
- [11] Hyndman R J and Athanasopoulos G Forecasting: Principles and Practice 2nd edition
- [12] Kong W, Dong Z Y, Hill D J, Luo F and Xu Y 2018 Short-Term Residential Load Forecasting Based on Resident Behaviour Learning *IEEE Transactions on Power Systems* **33** 1087-1088
- [13] Rahman A, Srikumar V and Smith A D 2018 Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks *Appl. Energy* **211** pp. 372-385
- [14] Hochreiter S and Schmidhuber J 1997 Long Short-Term Memory *Neural Computation* **9**(8) 1735-1780
- [15] Khare K, Gupta O D P, Attar V Z 2017 Short Term Stock Price Prediction Using Deep Learning *2nd IEEE International Conference on Recent Trends in Electronics Information & Communication Technology (RTEICT)*
- [16] TensorFlow: <https://www.tensorflow.org/>
- [17] Keras: <https://keras.io/>