NEU PRESS

FEN ve MÜHENDİSLİK BİLİMLERİ DERGİSİ

Cilt/Vol: 6 Sayı/No: 2 Yıl/Year: 2024 Araştırma Makalesi/Research Article e-ISSN: 2667-7989

https://doi.org/10.47112/neufmbd.2024.49

Güneş Enerjisi Santrallerinde Derin Öğrenme Kullanılarak Elektrik Üretiminin Değerlendirilmesi

Yunus Emre KIYMAZ[1]* Hidayet OĞUZ[2]

[1] Necmettin Erbakan University, Institute of Science, Department of Energy Systems Engineering, Konya, Türkiye

[2] Necmettin Erbakan University, Faculty of Engineering, Department of Energy Systems Engineering, Konya, Türkiye

Makale Bilgisi

Geliş Tarihi: 27.02.2024
Kabul Tarihi: 04.05.2024
Yayın Tarihi: 31.08.2024

Anahtar Kelimeler:
Derin Öğrenme,
Güneş Enerjisi Santrali,
Mevsimsel Otoregresif Bütünleşik Hareketli Ortalama,
Uzun Kısa Süreli Hafıza,
Tek Değişkenli Zaman Serileri.

ÖZET

Güneş paneli teknolojisi ortalama 25 yıl ömrü olan bu tür sistemlerin kurulumu pahalıdır. Bu sistemlerden en iyi şekilde yararlanmak için geleceğe yönelik üretim tahminleri yapmak çoğu zaman önemlidir. Bu çalışmada, Konya merkezli yıllık 1MW kapasiteye sahip güneş enerjisi santrallerine (tek değişkenli zaman serisi) ait iki yıllık üç günlük frekans veri seti ve bir yıllık saatlik frekans veri seti değerlendirilmektedir. Elektrik üretim analizi, derin öğrenme kullanılarak güneş enerjisi santrallerinden elde edilen verilere dayanılarak yapılmaktadır. Tercih edilen yöntem uzun kısa süreli hafıza (LSTM) olup, zaman serisi analizinde kullanılan diğer bir istatistiksel yöntem olan mevsimsel otoregresif bütünleşik hareketli ortalama (SARIMA) ile kıyaslanmıştır. Her bir veri seti ile elde edilmiş sonuçlar beş farklı performans ölçüm mekanizmasına (MSE, RMSE, NMSE, MAE, MAPE ve R^2) tabi tutulmuş ve LSTM modelinin genellikle SARIMA modeline göre daha gerçek verilere yakın sonuçlar verdiği tespit edilmiştir. RMSE skoruna göre dört santralin ortalama değeri LSTM'de 973, SARIMA'da 1361 olup, bu durumda LSTM, SARIMA'ya göre başarılı bir sonuç vermiştir. Güneş enerjisi santrali kurmadan önce fizibilite çalışmasının yapılması karlılığı artırıcı bir role sahiptir.

Assessment of Electricity Generation Using Deep Learning on Solar Power Plants

Article Info

Received: 27.02.2024
Accepted: 04.05.2024
Published: 31.08.2024

Keywords:
Deep Learning,
Long Short-Term Memory,
Seasonal Autoregressive Integrated Moving Average,
Solar Power Plant,
Univariate Time Series.

ABSTRACT

Solar panel technology is expensive to install such systems, which have a lifespan of about 25 years on average. It is often important to make production estimates for the future to make optimal use of these systems. This study assesses three two-year daily frequency data sets and a one-year hourly frequency data set from the solar power plants (univariate time series) based in Konya, which have a 1MW capacity per annum. Electricity production analysis is conducted based on the data from the solar power plants using deep learning. The preferred method is determined to be Long Short-Term Memory (LSTM), and it has been compared with another statistical method used in time series analysis, Seasonal Autoregressive Integrated Moving Average (SARIMA). The results obtained with each dataset have been subjected to five different performance measurement mechanisms (MSE, RMSE, NMSE, MAE, MAPE and R^2). It has been observed that the LSTM model generally provides results closer to real data compared to the SARIMA model. According to the RMSE score, the average value of four power plants is 973 in LSTM and 1361 in SARIMA, in this case LSTM gave a successful result compared to SARIMA. Before establishing a solar power plant, carrying out a feasibility study has a profitability-enhancing role.

To cite this article:
Kıymaz, Y. E. and Oğuz, H. (2024). Assessment of electricity generation using deep learning on solar power plants. *Necmettin Erbakan University Journal of Science and Engineering*, *6*(2), 289-311. https://doi.org/10.47112/neufmbd.2024.49

***Sorumlu Yazar:** Yunus Emre Kıymaz, *yunusemrekiymaz89@hotmail.com*

INTRODUCTION

Renewable energy is a term used for power sources with a constant energy flow such as solar, wind, biomass, geothermal, hydroelectric, and wave power. It is arguable that renewable energy systems (RES) are the set of systems using renewable energy sources to meet the need for electricity due to the decreasing availability of energy sources commonly used around the world such as oil, natural gas, and coal. Fossil fuels and their derivatives, which are most frequently used in electricity production around the world, not only pollute the natural environment but at the same deliver negative effects on human, animal, and plant health. These and similar reasons render renewable energy use crucial. The demand for the existing types of renewable energy sources is increasing by each day; nevertheless, not an adequate body of work is available in practice for energy efficiency. This constitutes a significant obstacle vis-a-vis fossil fuel. For instance, if we consider that solar power plants (SPP), which require high installation and maintenance costs, have a lifespan of 25 years on average, the issue of efficiency appears critical for SPPs.

In this study, we first review renewable energy sources, artificial intelligence and deep learning, time series definition, and analysis methods and then use the LSTM model (Long Short-Term Memory), a deep learning method, to make electricity production estimation for the future. For illustration, three SPPs selected from three different areas, which each have a 1 MW production capacity annually, are analyzed. This method evaluates the results by comparing with SARIMA (Seasonal ARIMA), a variation of ARIMA (Autoregressive Integrated Moving Average), which is another statistical method used in time series analysis.

Alan Turing, who proposed the question “Can machines think?” in the 1950s, to be a subbranch of artificial intelligence. Computer scientist Arthur Samuel is believed to have coined the term “machine learning” in 1959 [1].

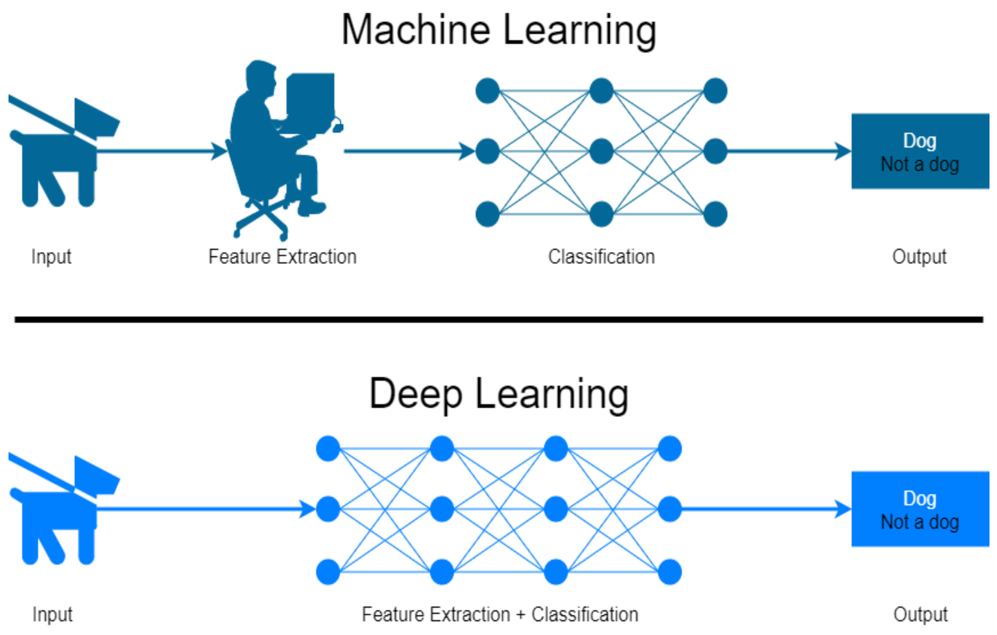

Figure 1
Structural Differences in Machine Learning and Deep Learning

Deep learning is an approach which essentially makes part of the family of machine learning and relies on artificial neural networks. Learning may be controlled, semi-controlled or uncontrolled [2]. Its fundamental difference from machine learning is that it can carry out the task of feature (or attribute) extraction (formation of new derived values through measured values introduced as input) within its own neural network without any support as Figure 1 demonstrates.

Artificial intelligence can be defined to include machine learning and deep learning respectively (Figure 2).

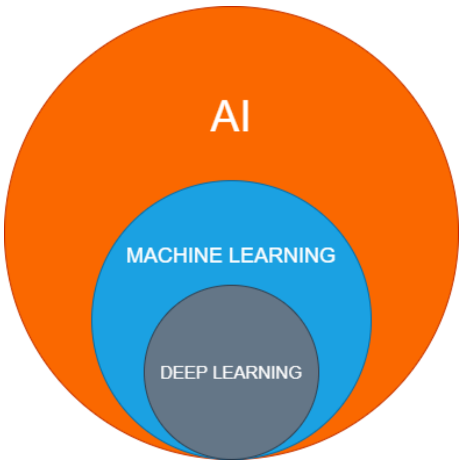

Figure 2
The Relationship of the Deep Learning Approach with Machine Learning and Artificial Intelligence $DL \subset ML \subset AI$ *[3]*

Time series refers to the frequency of data points in areas such as statistics, econometrics, signal processing, seismology, meteorology, and mathematical finance and is measured in terms of regular time periods and consecutive time slots. The daily closing value of the BIST (Borsa Istanbul) Index or the daily passenger capacity of Turkish Airlines can be offered as an example. Time series are divided into two types: univariate and multivariate. While univariate series include a single feature (for instance, only radiation data in meteorology), multivariate series may allow for several features (for instance, temperature, pressure, radiation, wind speed, or radiation time in meteorology).

Time series estimation is a conceptual model for predicting future events based on certain pre-given events. An example of time series estimation in econometrics is prediction of the opening price of a stock share based on its previous performance [4].

A few academic studies that have been conducted concerning solar power, wind power, weather forecasts, electricity demands, and other topics using deep learning and artificial intelligence techniques are briefly touched upon.

Cano et al. [5] utilized the HELIOSAT Method, which is one of the pioneering studies in solar radiation forecasting using prediction pixels derived from meteorological satellite images (cloud index (n)). Ruşen [6] aimed to understand the components of daily solar radiation on a horizontal surface in the selected region (Konya-Karaman) in the Central Anatolia Region using the HELIOSAT Method. Seven years of solar data were employed, and it was determined that the accuracy of daily global and diffuse solar radiation predictions was found to be acceptable. As a result, it was indicated that these predictions are important for calculating the performance and energy costs of solar power plants. Ruşen and Konuralp [7] conducted a study aiming to compare and evaluate the validity of global and diffuse radiation estimation methods for nine locations in Turkey to accurately determine solar radiation components for solar system analysis. They employed the HELIOSAT, Meteonom, and PVGIS satellite-based methods. It was noted that the HELIOSAT Method (with measurement metrics of relative mean bias error and relative root mean square error) is considered a reliable method in situations where global and diffuse radiation data are not available compared to the others.

Abdel-Nasser and Mahmoud [8] used the (LSTM-RNN) method to accurately estimate the output power of PV systems in their study. In their proposed method (LSTM with time steps), they used annual data sets obtained from different sites. When they compared with three different PV estimation methods (multiple linear regression (MLR), bagged regression trees (BRT) and neural networks), they stated that the prediction error in LSTM was lower.

Agrawal et al. [9] proposed a new method for long-term load estimation at hourly resolution. They focused the model on a recurrent neural network of LSTM-RNN cells. They have considered the long-term relationships in the time series data of the electrical load demand by using LSTM-RNN and have obtained results with correct predictions. They adapted this model to real-time data of the ISO New England electricity market. They collected twelve years of publicly available data from 2004 to 2015 to train and validate this model and made forecasts of electricity demand on a rounding basis over a five-year period from 2011 to 2015. They stated that they predicted correctly with a 6.54 Mean Absolute Percent Error (MAPE) in the 2.25% confidence interval.

Balluff et al. [10] using a RNN to estimate the wind speed and pressure in northern Europe (Great Britain, Ireland, France, Germany, Denmark) between 2001 and 2015 installed wind power plants, wind speed, wind direction, temperature and surface pressure data were used.

Gensler et al. [11] used combinations of the relevant algorithm to demonstrate their prediction power compared to a standard MLP and a physical prediction model in estimating the energy output of 21 GES. It has been stated that the results using Deep Learning algorithms show superior prediction performance compared to other reference models such as Artificial Neural Networks and physical models.

Sharadga et al. [12] introduced several time series estimation methods, including those based on statistical methods and artificial intelligence, and rigorously compared PV power output estimation. In addition, they investigated the effect of prediction time horizon variation for all algorithms. The data used in the current study includes 3640 hours of operation from a 20 MW grid-connected PV station in China.

Şencan [13] made a short-term electricity price estimation using LSTM in the study. In the model created, historical electricity price values are used as input. The data used are hourly electricity prices in Turkey for the years 2015, 2016 and 2017. The data used in the study are divided into two as training and testing for winter and summer seasons. The method used; Compared with the performance of RNN and Exponential Correction methods. MAPE values obtained using the LSTM method; 5.91% for winter and 5.77% for summer.

LSTM model was created by Özen et al. [14] using temperature and humidity data received from Tekirdağ Provincial Directorate of Meteorology for temperature and humidity prediction with deep learning, and the success criteria were calculated as RMSE 1.895, MSE 3.547, R-square score value 0.952 and MAE 1.614.

In their study, Hacıbeyoğlu et al. [15] used the K Nearest Neighbor (KNN) algorithm to estimate energy efficiency in buildings. In their experimental results, KNN was more successful than linear regression and produced predictions at the level of 96%.

Alparslan et al. [16] conducted a study on solar radiation prediction using Artificial Neural Network (ANN) and Adaptive Neuro-Fuzzy Inference System (ANFIS) methods. In this study, which considered the climatic conditions of the Karaman region, it was observed that the ANFIS model provided more accurate results in the prediction of monthly data compared to the ANN model.

MATERIALS AND METHODS

The study was conducted using several software tools on a computer. The general features of the computer used are indicated in Table 1.

Table 1
General Characteristics of the Computer Used

Hardware	**Hardware Features**
CPU	AMD Ryzen 1500X 3.5 GHz
GPU	NVIDIA GeForce GTX 1050Ti 4GB GDDR5
Memory (RAM)	16 GB (8+8) 3000 MHz

Software Tools

This section includes the applications used in this study and the materials needed for deep learning and statistical analysis in the software libraries. These materials were preferred since they are frequently used in statistical studies, a good number of sample works involving them are available, and they are easy to access.

Python is an object-oriented, modular, and interactive high-level programming language. Its modular structure supports any kind of data area entry. It can operate on almost any platform [17]. It is most often used in scientific studies and statistical science (along with the statistical language R).

Anaconda is a Python-supported integrated Python distribution for applications in data science and other fields. In addition to the libraries typically used in the fields of artificial intelligence and data science, it also involves devices such as Jupyter Notebook and Spyder in its system [18].

Anaconda Navigator is a desktop graphical user interface included in the Anaconda distribution that allows developers to launch applications and manage conda (package manager used in Python) packages, environments, and channels without using command-lines [19].

Numpy (Numerical Python) is an open-source library frequently used by researchers and scientists that supports Python-based scientific calculations and the formation of multidimensional arrays and matrices through code writing on fewer lines to help form high-level mathematical functions [20].

Matplotlib is an open-source Python library used for data visualization. It is useful for two- or three-dimensional graph drawing [21]. It works compatibly with Numpy. It supports many types of graphs (lines, columns, circles, image processing etc.).

Tensorflow is a free and open-source library developed by the firm Google that is used for neural networks and machine learning [22]. It operates compatibly with libraries such as Keras, used in neural networks, and scikit-learn, used for machine learning. Google integrated Keras into the system in the second version of Tensorflow. It has also recently presented the Tensorflow 3D library, which facilitates deep learning in three-dimensional environments, for use by scientific researchers [23].

Pandas is an open-source library written using the Python language that is used for data analysis and manipulation. It is especially useful for importing data into a project and supports formats such as csv, txt, and xls. It operates in the types “Dataframe” and “Series.” It usually uses “dataframes” for machine learning. We can consider “dataframes” to be like tables in relational database systems. Pandas uses operations such as adding or removing columns or lines and merging or separating tables for “dataframes” like data tasks in a table [24].

Keras is an open-source neural network library written using Python and is used for deep learning. It can operate with Tensorflow. It was developed by Google engineer François Chollet. It is used in designing deep neural networks such as CNN and RNN [25].

Statsmodels is a Python library offering classes and functions for the computation of many different statistical models, statistical tests, and statistical data discovery. It provides a comprehensive list of statistical conclusions for each estimate. It is checked against available statistical packages to verify the accuracy of its conclusions [26].

Datasets

The study analyzes four data sets from SPPs based in different locations in the province of Konya, each of which has a 1MW installed power. These power plants are in the districts of Çumra, Tuzlukçu, and Yunak in Konya. While three of these four data sets are daily frequency and span two years of electricity production data (kWh), the other is hourly frequency and spans about one year. These data consist of univariate time series. Figure 3 shows the locations of the SPPs.

In the rest of the study, the Çumra power plant will be referred to as Plant_A (daily frequency), the Tuzlukçu power plant as Plant_B (daily frequency), the Yunak power plant as Plant_C (daily frequency), and the same Çumra power plant as Plant_D for hourly frequency.

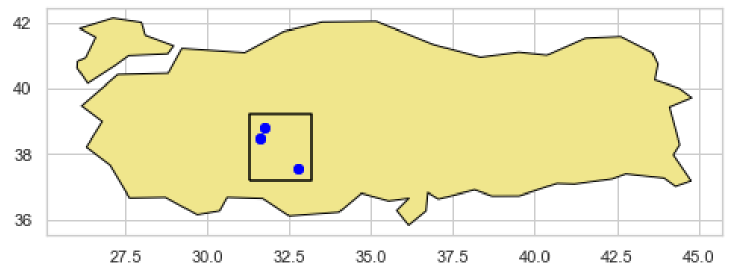

Figure 3
The Locations of SPPs Used in the Study

Figure 4 indicates the electricity production data graph for Plant_A. It includes 800 days of data. Figure 5 indicates the electricity production data graph for Plant_B. It includes 730 days of data. Figure 6 indicates the electricity production data graph for Plant_C. It includes 730 days of data. Finally, Figure 7 indicates the electricity production data graph for Plant_D. It includes 8777 hours of data.

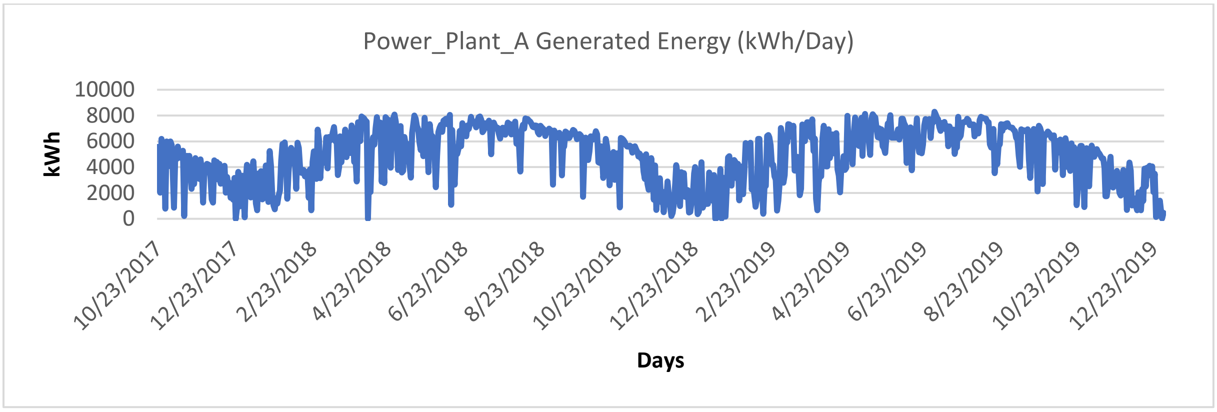

Figure 4
Electricity Production Chart of Plant_A (with daily frequency).

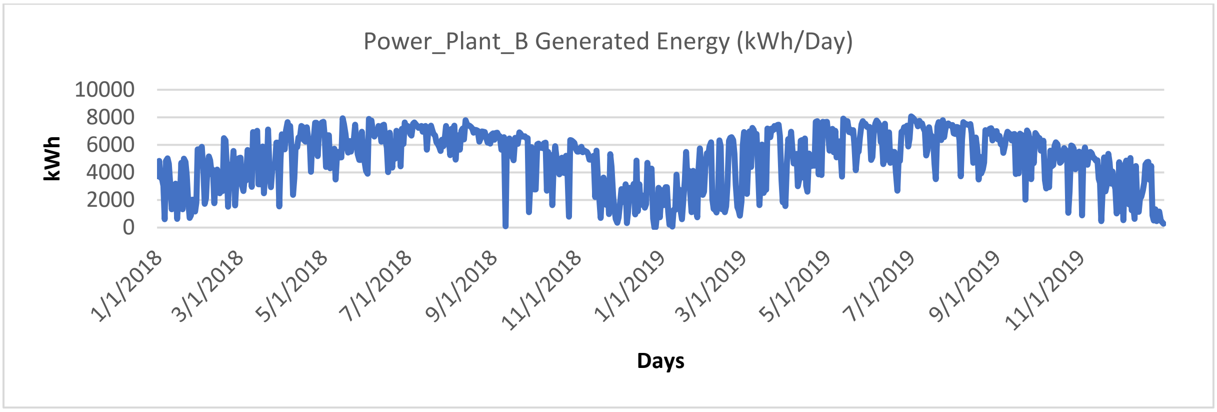

Figure 5
Electricity Production Chart of Plant_B (with daily frequency).

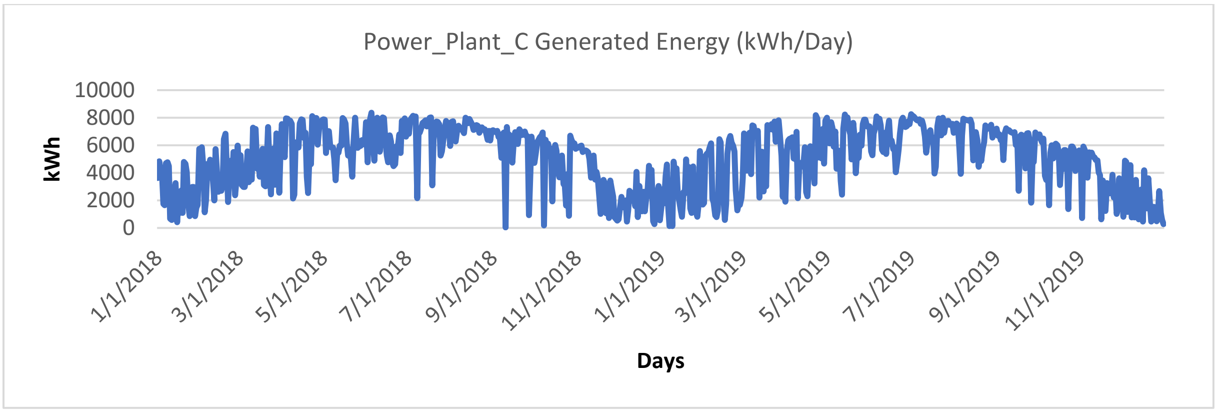

Figure 6
Electricity Production Chart of Plant_C (with daily frequency).

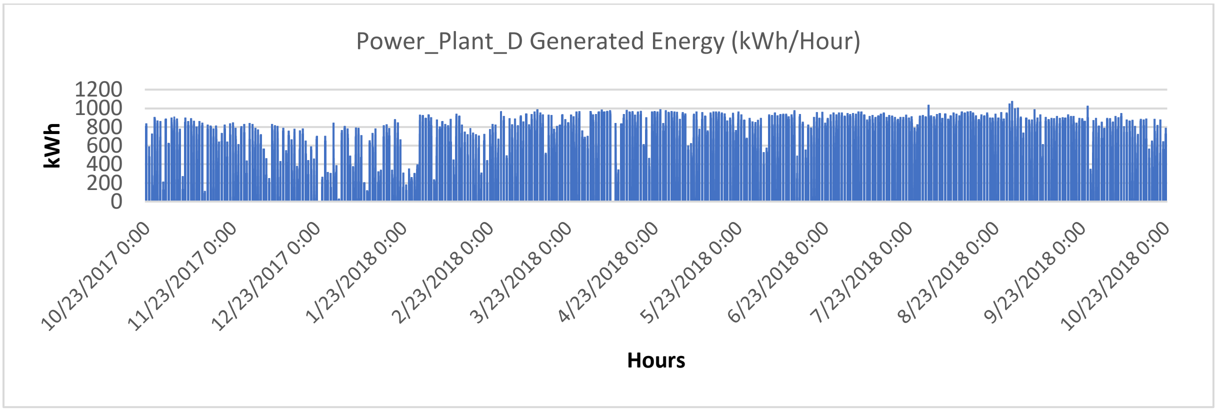

Figure 7
Electricity Production Chart of Plant_D (with hourly frequency).

Plant_D is in fact the data set derived from the Çumra power plant based on hourly frequency. It is labelled Plant_D to prevent confusion. While three data sets are offered in daily frequency, the fourth data set is offered in hourly frequency since the latter involves higher data density and we would like to be able to analyze how the system, especially the LSTM, will respond. That data density is high in deep learning is an important consideration in system training and in the forecasting mechanism. Data from Konya is used because this province is large in surface area and the region exhibits significant levels of solar heat and radiation annually [27].

The LSTM Method in Neural Networks and its Structural Properties

LSTM networks are referred to as gate cells where information can be stored. A cell determines which information to store, how to use it or whether to forget any information during the task of forecasting. Input and output gates allow for the passage or blocking of information based on trained weights. This architecture can merge flow entry, previous status and the cell memory and inform long-term dependencies. It was derived and formed out of the RNN (Recurrent Neural Network) structure [28].

All RNNs are in the form of recurrent neural module chains. In classic RNNs, these modules have a basic structure such as a single *tanh* layer. Figure 8 demonstrates the scheme of this chain.

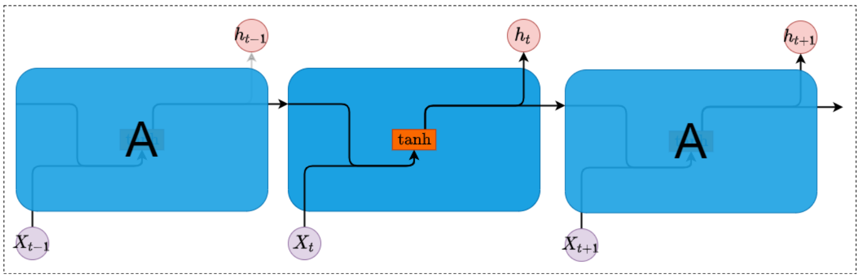

Figure 8
Repeating Module in Standard RNN Contains a Single Layer [29]

LSTMs, like RNNs, have a consecutive structure where they follow one another. The crucial difference here is that parts following each other do not have a single neural layer but a four-layered special-interaction structure. Figure 9 shows the LSTM chain.

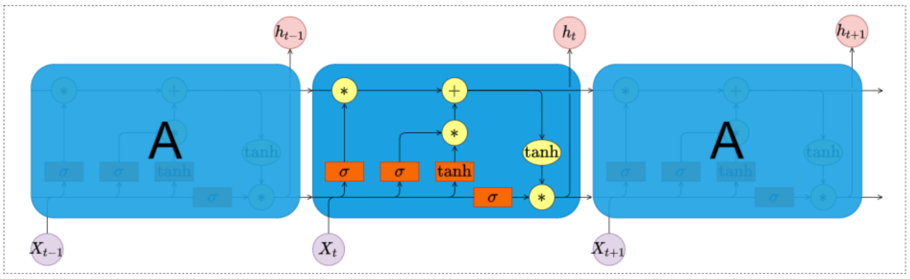

Figure 9
The Repeating Module in LSTM Includes Four Interactive Layers [29]

In the diagram in Figure 9, the entire vector is transmitted from the output of the node for each line to the input of the other modules. While neural network layers are learned in the orange boxes, the yellow circles perform pointwise operations such as vector addition. Merged lines stand for the merging line and the line forking operation suggests that the copied content will move to different locations [29].

The LSTM architecture could add or remove information organized by structures called "gates". Gates are a way to allow the passage of information on demand. They consist of a sigmoid neural network layer and a point multiplication process (Figure 10).

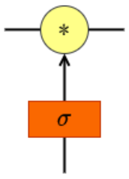

Figure 10
Sigmoid Layer with Pointwise Multiplication [29]

The sigmoid layer returns values between 0 and 1 that describe how much of each component it should allow to pass through. The value "0" means "closed to data transfers", while the value "1" means "available for data transfers". The standard LSTM has three of these gates for maintaining and controlling the cell state [29].

Steps of an LSTM Cell

The first step in the model is to determine which information to throw away from the cell state. This task is carried out by a sigmoid layer called "the forget gate layer." It looks at the points X_t and h_{t-1} and outputs a number between 0 and 1 for each number in the cell state C_{t-1} (Figure 11). An exit value of 1 represents "keep this information" while a value of 0 represents "get rid of this information". The equation as follows:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, X_t] + b_f\right) \qquad (1)$$

The next phase is the decision phase, determining what new information to keep based on the cell state. This takes place in two parts. First, the "gateway layer", which is the sigmoid layer, determines which values will be updated. Next, a $tanh$ layer creates a vector of new candidate values, $\tilde{C}_t$, that is added to the new state (Figure 12). In the next step, these two layers are combined to update the state.

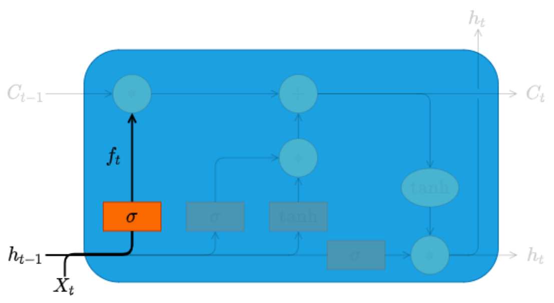

Figure 11
Sigmoid Layer with Pointwise Multiplication [29]

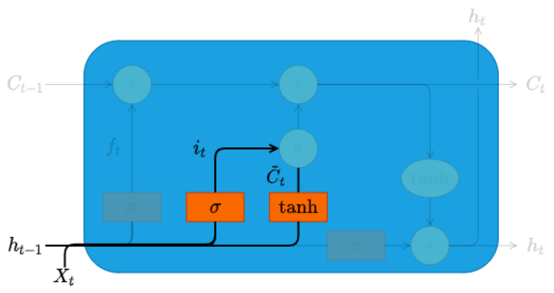

Figure 12
"Input gate" Layer in LSTM Cell [29]

The formula of the input layer is given as follows:

$$i_t = \sigma(W_i \cdot [h_{t-1}, X_t] + b_i) \quad (2)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, X_t] + b_C) \quad (3)$$

This makes it possible to update the old C_{t-1} cell state to the new C_t cell state. It had already been decided in the previous steps what to do. In this step, implementation begins.

In this step, the old situation is multiplied by f_t, forgetting the information previously decided to forget. Then add $i_t * \tilde{C}_t$ (as shown in the equation below). These values are new candidate values at scale where it is decided how much of each state value to update (Figure 13).

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (4)$$

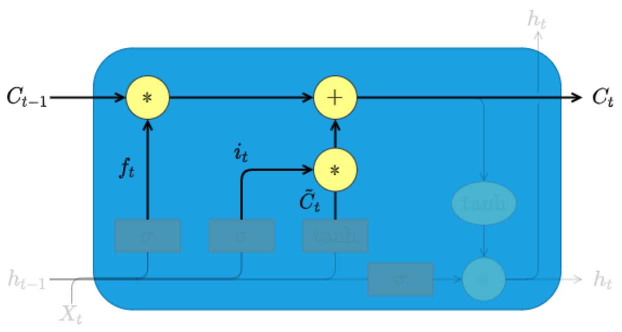

Figure 13
Merge Layers After Previous Operations in LSTM Cell [29]

In the last step, it is decided what to send as output. This output depends on the cell state but can also be a filtered version. In the first step, a sigmoid layer is run where it is decided which parts of the cell state want to be output (Figure 14). The cell state is then passed through a $tanh$ function (to keep the values between -1 and +1) and multiplied by the output of the sigmoid gate to output only the parts it was decided to include.

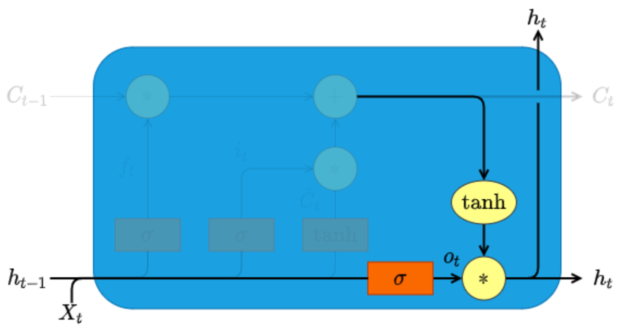

Figure 14
The h_t Gate is Created in the Last Step in the LSTM Cell [29].

The equations of the last step are shown as follows:

$$o_t = \sigma(W_o \cdot [h_{t-1}, X_t] + b_o) \tag{5}$$

$$h_t = o_t * \tanh(C_t) \tag{6}$$

ARIMA and SARIMA Methods in Statistics

The autoregressive integrated moving average model (ARIMA) is used in non-stationary time series. It uses the Box-Jenkins method [30]. While it represents local trends or differences in level, different parts of non-stationary processes display a certain level of similarity. A stable $ARIMA(m, d, n)$ model that is the difference d of the time series is shown as follows:

$$\tilde{x}_t = \sum_{i=1}^{m} \Phi_i S^d x_{t-i} + \sum_{j=0}^{n} \theta_j \omega_{t-j} \tag{7}$$

Here, $S = 1 - q^{(-1)}$ and $\Phi_m(q)$ are fixed and reversible $AR(m)$ operators. x_t, ω_t, Φ_t and θ_j are respectively the observed time series values, the error, the AR, and the MA values. ‘d’ is the number of differences (non-seasonal), ‘m’ is the number of autoregressive terms, and ‘n’ is the number of delayed forecast errors [31].

In the ARIMA process, stationarity in time series is identified and fixed and then the operation is made more forecastable using the $ARIMA$ model.

If time series involve a seasonal impact, the Seasonal ARIMA (SARIMA) method is used. This method goes through operations like those in $ARIMA$ and incorporates the parameter s, which stands for seasonality and is expressed as $SARIMA\ (m, d, n)x(M, D, N, s)$ statistically. The capitalized letters (M, D, and N) are the parameters used for the model involving seasonality.

For time series to be forecasted in the ideal fashion and in statistical terms, they are expected to be static. It is possible to use methods such as a correlogram and the Dickey-Fuller test to determine whether a time series is static. Considering such a test, it is possible to make a time series static and engage in forecasting.

Performance Measurements

The performance of forecasting methods is measured using various metrics related to forecast error. Higher error values indicate lower prediction accuracy. This section will discuss various performance metrics frequently used to calculate prediction error. The point that needs attention here is that x represents the observed value, $\tilde{x}$ stands for the estimated value, and n corresponds to the total number of cases [31]. This section will cover the performance metrics evaluated in the study.

MSE (Mean Squared Error)

The mean squared error refers to the distance between a regression curve and a series of points. The MSE measures the performance of the forecasting mechanism of a machine learning model. It always has a positive value, and it is possible to put forward that the closer the MSE result is to zero, the better performance the forecasting mechanism exhibits. The equation is shown as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\tilde{x}_i - x_i)^2 \tag{8}$$

NMSE (Normalized Mean Squared Error)

This measurement is the normalized version of the MSE as follows:

$$NMSE = \frac{n\sum_{i=1}^{n}(\tilde{x}_i - x_i)^2}{\sum_{i=1}^{n}x_i\sum_{i=1}^{n}\tilde{x}_i} \tag{9}$$

It is accepted that the model is successful in cases in which the NMSE value measured for the best model is closest to zero. The NMSE, normalized by the multiplication term of the means found in the denominator, does not show bias due to under-forecasting or over-forecasting by the model [32].

RMSE (Root Mean Squared Error)

The root mean squared error is obtained by extracting the square root of the MSE. It is a quadratic measurement used to find the distance between the value predicted by the forecasting mechanism of a machine learning model and the actual value. The RMSE accounts for the standard deviation in forecast errors. In other words, residuals are a measure of how far the regression line is from the actual data points. The RMSE does not allow for the use of absolute values, which are not desired in many mathematical computations [33]. It is considered that, by definition, the closer the RMSE is to zero, the better alignment it shows with data. However, it never takes the value of 0 in practice. In general, a lower RMSE is better than a higher one. Since the RMSE depends on the scale of numbers used for measurement, comparisons among different types of data will not have any validity. The model is frequently used in comparisons. The equation is shown below:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\tilde{x}_i - x_i)^2} \tag{10}$$

MAE (Mean Absolute Error)

The mean absolute error is the measure of the difference between two constant variables (the following equation). The MAE is both the mean vertical and the mean horizontal distance between each actual value and the line best aligning with the data. The MAE result is easy to interpret and is frequently used in regression and time series problems [33].

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\tilde{x}_i - x_i| \tag{11}$$

MAPE (Mean Absolute Percentage Error)

The mean absolute percentage error is shown below:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{\tilde{x}_i - x_i}{x_i}\right| * 100\% \tag{12}$$

The MAPE is frequently used in regression and time series models to determine the accuracy of forecasts. If the actual values include zero, the MAPE cannot be calculated since it needs to be divided by zero (it will yield a false result). For very low forecast values, the percentage error cannot go above 100%. Nevertheless, when very high forecast values are generated, there is no upper limit for the percentage error. When the MAPE is used to compare the accuracy of models used for forecasting, it is biased since it systematically selects a model with low forecasts [33].

R^2 (Coefficient of determination)

Coefficient of determination is shown below:

$$R^2 = 1 - \frac{\frac{1}{n}\sum_{i=1}^{n}(\tilde{x}_i - x_i)^2}{\frac{1}{n}\sum_{i=1}^{n}(\bar{x} - x_i)^2} \tag{13}$$

The coefficient of determination (R-squared) [34] is expressed as the ratio of the variance in the dependent variable that can be predicted by the independent variables. For example, an R-squared of 80% means that the independent variables can explain 80% of the variance in the dependent variable. R-squared generally corresponds to a value between 0 and 1.

Hyperparameter Tuning

Since the use of graphical cards allows for speed in parallel operations, researchers are now able to model complex structures with multiple layers and solving problems through deep learning has become equivalent to optimizing a multiple-layered neural structure. At this point, hyper-parameters often present themselves as tools used by researchers along with their insight [35].

In designing an ML model learning from data, designers need to use certain parameters to decide what algorithms or techniques to feature in the model. To illustrate, it is the designer who determines what value the value k will receive in the KNN classification algorithm. Similarly, it is the designer who decides which kernel function to use in the SVM algorithm. In deep neural models, the designer sets the dropout value and the number of neurons. It is initially not clear which criteria to use in the selection of such parameters. It depends on factors such as the data set and the type of problem. Thus, it is up to the designer what they will be. These parameters that vary by the characteristics of the data set are called hyperparameters. Figure 15 shows hyper-parameters and the state of model parameters [35]. In statistical science, the parameters referred to as “orders” in ARIMA models used for time series estimation are also called hyperparameters since they are adjusted according to the makeup of the series.

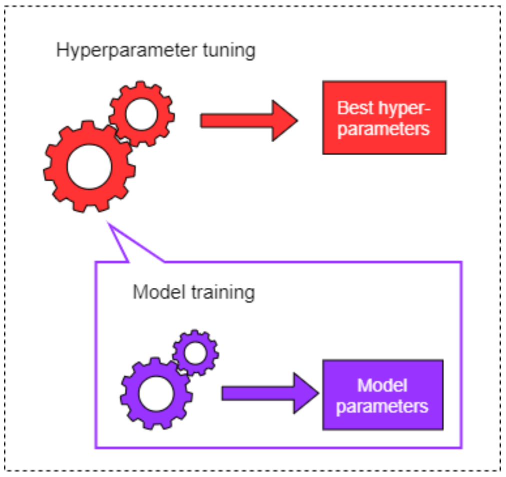

Figure 15
Hyperparameters and Relationship of Model Parameters [36].

RESULTS AND DISCUSSIONS

In the study, LSTM and Seasonal ARIMA (SARIMA) models were used for each data set (these data are univariate time series). The forecast graphs for each model used for each data set are given along with the actual values. To compare the forecast accuracy of these models, MSE, RMSE, NMSE, MAE, and MAPE error measurements were used.

The LSTM Model

The basic sequential model available in the Keras library was preferred, and among the layers of this model, primarily the LSTM model was used. The general structure of the model is shown in Table 2.

Table 2
General Structure of The LSTM Model Used

MODEL TYPE: SEQUENTIAL		
LAYER	**OUTPUT SIZE***	**NUMBER OF PARAMETERS**
Lstm	32	4864
Dropout	32	0
Lstm_1	25	5800
Dense	1	26

Total number of parameters: 10.690, Number of trainable parameters: 10.690
* Dimensions and other values are hyperparameters.

The model has a structure consisting of four layers. The input number for the LSTM in the first layer was set as 32 (since it is a hyperparameter, it is possible to change it if necessary). As the forgetting of weak information returns a positive result for the forecasting mechanism, a dropout layer was added and was set at the value 0.5. (In general, you start out with this value, but it is subject to change and is up to the designer.) Following the thinning operation, a new LSTM layer was formed at the third layer and the intermediate input number was set at 25. Finally, the Dense (conventional YSA model) layer was added, and its output dimension was given as 1 and ReLU (hyperparameter) was used as the activation function. For optimization, the RMSProp algorithm was added. The Batch dimension (hyperparameter) was given as 16. Every step is 200 epochs (hyperparameter). The standard LSTM model was used. In model comparison, the LSTM will be expressed parametrically as (32, 25). The parameters were respectively set as the first and second LSTM layer (output dimension).

The SARIMA Model

The Python-supported Statsmodels library was used in the making of this model. In addition, for the SARIMA parameters to be available in the best possible way in the SPPs involving daily data, the ‘auto_arima’ method in the pmdarima [37] library was used for control purposes only. The most appropriate model (the one with the lowest AIC value) was selected among different models, using the AIC method [38]. It is a process that requires attention statistically to determine the parameters in the operation process of SARIMA (in determining the parameters following the analysis of autocorrelation and partial autocorrelation functions etc.). For this end, in addition to AIC, the Dickey-Fuller unit root test, which we mentioned in the third section, was applied to confirm the stationary of the time series. This test is a method developed to identify whether a time series is stationary or non-stationary. It can determine whether a time series includes a unit root [39]. It is recommended that the forecasting mechanism be run in stationary time series in SARIMA. In non-stationary time series, on the other hand, it is necessary to first make them static for to determine the parameters (extracting a difference, logarithmic calculations etc.).

Since the data sets involve a seasonal impact, assessment was made using the Seasonal ARIMA (SARIMA) model. Differently from the Standard ARIMA ($ARIMA\,(m, d, n)$), the SARIMA is expressed as $(m, d, n)x\,(M, D, N, s)$. The value ‘s’ here corresponds to an integer that stands for seasonality. M, D, and N are the parameters used for seasonal ARIMA. This value is a hyperparameter, like other ARIMA expressions.

The basic flow diagram for the LSTM and SARIMA models is indicated in Figure 16.

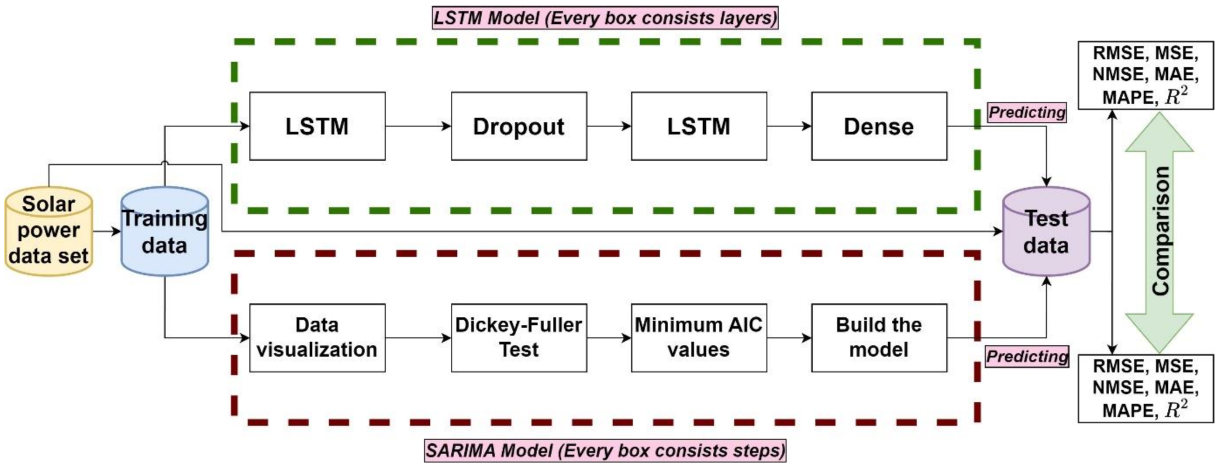

Figure 16
LSTM and SARIMA Basic Flow Chart of the Study.

The LSTM model was evaluated using the same parameters for each data set. This is because the LSTM model is a trainable one and can generate results that are based on relevant data. Training for the SARIMA is informed by the characteristics of the data set (trends, stationarity criteria), it is required to run pre-operations for data, and the parameters are unique for each data set. They apply separately for each data.

Result for the Datasets

In this section, the SARIMA and LSTM models were applied in each data set and the results of the models were applied individually for each data set. The LSTM model was formed for once only (Table 2) and the same model was applied for each data set. 70% of each data set was reserved for training, and 30% for testing. As indicated in Figure 16, for each data set, the ‘auto arima’ method, the Dickey-Fuller test, and then the AIC test were run, and the parameters were set accordingly.

Plant_A (Figure 4) includes 800 days of data. The model obtained for Plant_A is $SARIMA\,(1, 1, 1)x\,(0, 1, 1, 365)$. Since it involves yearly data, the final parameter (seasonal) was set at 365. The forecasting results for the dates between 12.5.2019 and 31.12.2019 following these operations are shown in Figure 17.

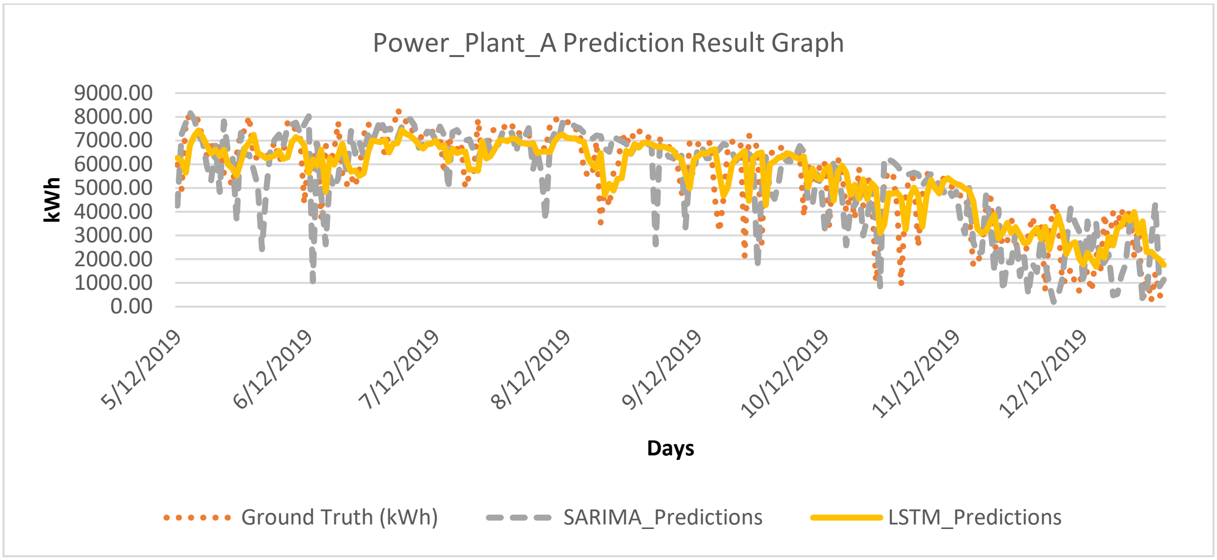

Figure 17
Estimation Results for Plant_A

The LSTM model which we formed after reviewing the performance results (error measurements) for Plant_A, which is in Çumra and has a 1MW installed power, exhibited a successful performance in comparison to the SARIMA (except for the MAPE) (Table 3).

Table 3
Performance Results of Plant_A

MODEL	RMSE	MSE	NMSE	MAE	MAPE	R^2
SARIMA (1,1,1) x (0,1,1,365)	1525.37 kWh	2326754	0.08	1104.16	61%	0,44
LSTM (32,25)	1163.62 kWh	1354012	0.05	870.46	78%	0,67

Plant_B (Figure 5) includes 730 days of data. The operations run for Plant_A were also run for Plant_B. The model thus obtained $is\ SARIMA\ (1,1,1)\ x\ (0,1,1,365)$. After the parameters for this model were set, the SARIMA and LSTM models were applied to the data set and the results in Figure 18 emerged for the dates between 1.6.2019 and 31.12.2019.

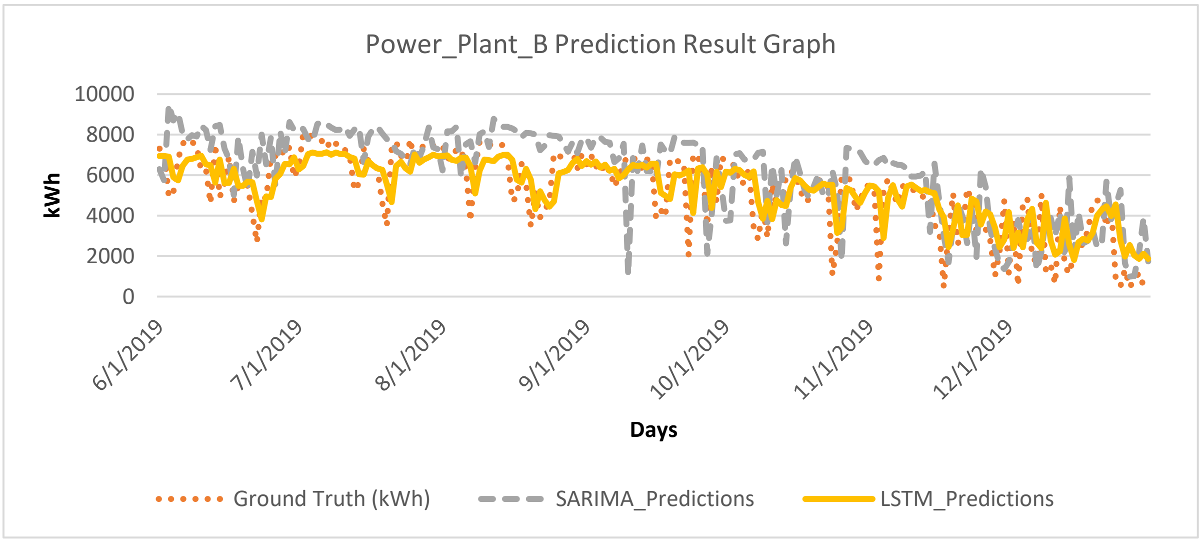

Figure 18
Estimation Results for Plant_B

The performance results for Plant_B, located in Tuzlukçu and with a 1 MW installed power, were examined. The LSTM model we developed exhibited results relatively close to actual data compared to the SARIMA model, as was the case with Plant_A (Table 4).

Table 4
Performance Results of Plant_B

MODEL	RMSE	MSE	NMSE	MAE	MAPE	R^2
SARIMA (1,1,1) x (0,1,1,365)	1994.28 kWh	3977153	0.12	1556.63	58%	-0,004
LSTM (32,25)	1341.93 kWh	1800776	0.06	1005.14	44%	0,55

Plant_C (Figure 6), like Plant_B, includes 730 days of data. The model derived is $SARIMA\ (1, 1, 1)x\ (0, 1, 1, 365)$. After the parameters for the model were set, the SARIMA and LSTM models were applied to the data set and the results shown in Figure 19 were obtained for the dates between 1.6.2019 and 31.12.2019.

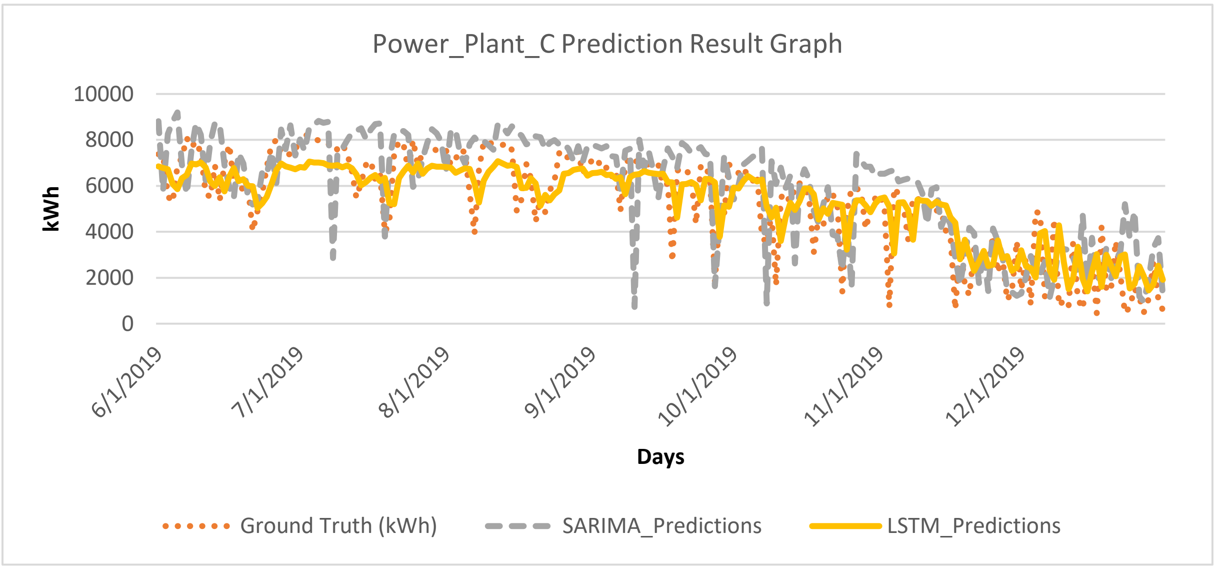

Figure 19
Estimation Results for Plant_C

If we analyze the performance results for the SPP located in Yunak, which has a 1MW installed power, the LSTM model yielded more satisfactory results than the SARIMA model (Table 5).

Table 5
Performance Results of Plant_C

MODEL	RMSE	MSE	NMSE	MAE	MAPE	R^2
SARIMA (1,1,1) x (0,1,1,365)	1756.39 kWh	3084906	0.1	1324.56	52%	0,37
LSTM (32,25)	1301.53 kWh	1693980	0.06	1004.43	44%	0,65

Plant_D (Figure 7) includes 8777 hours of data differently from the other data sets. It was formed through the conversion of the original data for Plant_A into hourly frequency (under normal circumstances, data is recorded in the system automatically in fifteen-minute periods).

The data pattern for Plant_D displays a strikingly different behavior from the data for the other three SPPs. Thus, different factors played a role in determining the parameters. Only the AIC test was applied in Plant_D for the task of selecting parameters.

The model obtained because of the test is $SARIMA\ (1,0,1)x\ (1,1,1,24)$. The value ‘s’ was set at 24 since the SPP is operational 24 hours a day (a hyper-parameter). The forecasting results of the LSTM and SARIMA models derived for the dates between 6.7.2018 and 23.10.2018 following this procedure can be viewed in Figure 20.

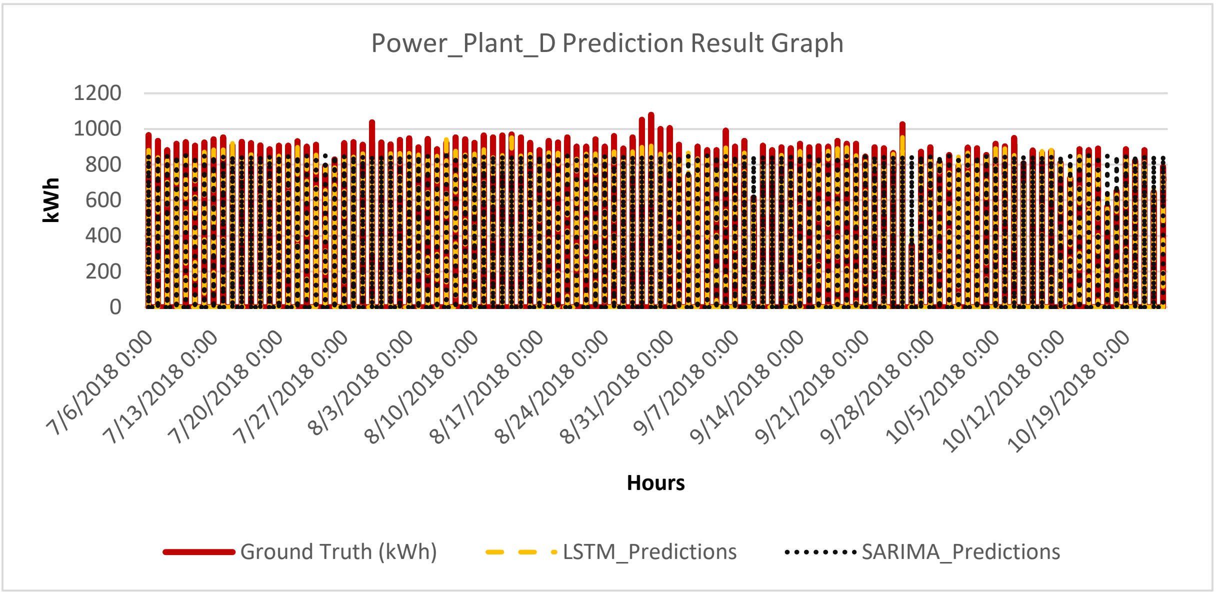

Figure 20
Estimation Results for Plant_D

If we take a glance at the performance errors for Plant_D (Table 6), the error data is lower in the LSTM model than in the SARIMA model and the former model yields more successful results. The MAPE value was not added because the actual data included zero (0).

Table 6
Performance Results of Plant_D

MODEL	RMSE	MSE	NMSE	MAE	R^2
SARIMA (1,0,1) x (1,1,1,24)	167.93 kWh	28200.48	0.38	102.86	0,75
LSTM (32,25)	86.26 kWh	7440.79	0.11	38.58	0,94

While machine learning and deep learning are not new technologies, they continue to develop and receive interest constantly. This study made use of these topics and aimed to make contributions to research focused on improving the efficiency of such systems since renewable energy systems are in high demand due to the limited availability of underground sources in the 21st century.

This study can be a precursor for work on forecasting future production values and engaging in plant maintenance in due time and with low cost to slow down the decrease in the efficiency of solar power plants over time, given that it is constantly expanded and updated, with its models improving each time. Its forecasting infrastructure can be reduced to monthly periods, and it can be used in identifying the problems encountered over a given month or time of the year in the past as soon as they recur in the future.

Thanks to its monthly production estimates, this system has the potential to offer individuals (real or legal) the chance to take measures so that they can minimize any penalties they may be required to pay due to electricity production that is above or below the level specified in agreements by electricity distribution companies which are based on monthly production in SPPs. Similarly, this study can not only be turned into a set of package applications and provide data for the SCADA systems used by electricity distribution companies but at the same time include different data from the SCADA systems and help make further inferences.

In addition, it is important to constantly improve and optimize the deep learning model. This study can be trained constantly using instant data obtained through the systems at the basic (elementary) level. The learning ability of the model can be advanced through training. If we consider that the standard lifespan of an SPP is 25 years and three years of past data are sufficient for conducting a study at the optimal level, we can argue that such work is crucial for SPPs.

CONCLUSIONS

In this study, which engages in electricity production estimation based on solar power plants, methods (LSTM and SARIMA) available in the fields of deep learning, statistics, and econometrics were used to make time series analysis. These methods were implemented on four different SPP data sets and the results were compared using the RMSE, MSE, NMSE, MAE, MAPE and R^2, which are frequently used in performance measurements.

The tests for Plant_A, Plant_B, Plant_C, and Plant_D, which are elaborated on in the following chapters, were assessed, and while results close to actual data were obtained in both methods (based on the NMSE measure, all models returned results close to zero), the LSTM results still yielded more successful values than the SARIMA. Here, it should be noted that the hyper-parameters have an impact because hyper-parameters play a crucial role in determining the direction of forecasting. Second, the computational power of the GPU used in this study (CPUs work with less speed than GPUs in terms of computational power today) made positive contributions to obtaining the desired values both in such a short time and in a consistent way. And at last, LSTM architecture provides more parameter learning makes powerful to do forecast. Particularly when a data has long-term trend.

In terms of the coefficient of determination (R^2), when examining Power Plants A, B, and C transmitting data at daily frequency, it was found that the R^2 values of LSTM models explain over 55% of their variances. Although they do not approach a value of 1, LSTM models outperform SARIMA models in all tested power plants. For instance, when examining the SARIMA forecast for Power Plant B, the R^2 value was -0.004, indicating insufficient predictive capability of the model for Power Plant B. To rectify this, it is important for the model to undergo appropriate preprocessing steps on the dataset (such as reconfiguring the Dickey-Fuller test, redefining the minimum AIC value, or re-determining SARIMA parameters using autocorrelation and partial autocorrelation functions). Surprisingly, when examining Power Plant D transmitting data at hourly frequency, both models' R^2 values are high, and the LSTM model even approaches a value of 1.

While the LSTM library used in the study made use of GPU support, the SARIMA library made use of CPU only (the Statsmodels library does not support GPU). The obstacle encountered in deep learning design makes it necessary to have expensive equipment (GPU or TPU requiring high computational power) to run fast and efficient operations.

Another issue is correction of data pollution (removal of insignificant data and value assignment based on the series trend etc.). Optimization and rearrangement of data pollution is a factor facilitating the forecasting function. Such procedures are frequently run-in analyzing data.

Ethical Declaration

This study was prepared from the master's thesis titled "Estimation electricity generation using deep learning on solar power plants", presented on 17.12.2020, under the supervision of Professor Hidayet OĞUZ.

Acknowledgements

We would like to thank engineers Kenan Kahya and Ercenk Yavuz, who provided the solar power plant data required for our study.

Funding

This research was not supported by any public, commercial, or non-profit organization.

Authorship Contribution

Research Design, Conceptualization (CRediT 1) Yunus Emre Kıymaz (%60) – Hidayet Oğuz (%40)
Data Collecting (CRediT 2) Yunus Emre Kıymaz (%70) – Hidayet Oğuz (%30)
Research- Data Analysis - Validation (CRediT 3-4-6-11) Yunus Emre Kıymaz (%60) – Hidayet Oğuz (%40)
Writing (CRediT 12-13) Yunus Emre Kıymaz (%60) – Hidayet Oğuz (%40)
Review & Editing (CRediT 14) Yunus Emre Kıymaz (%50) – Hidayet Oğuz (%50)

Conflict of Interest

The authors have no conflict of interest to disclose for this study.

Sustainable Development Goals (SDG)

Sustainable Development Goals: 7 Affordable and Clean Energy

REFERENCES

[1] A.L. Samuel, Some studies in machine learning using the game of checkers, *IBM Journal of Research and Development*. 3 (1959), 210-229. doi:10.1147/rd.33.0210.

[2] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, *Nature*. 521 (2015), 436-444. doi:10.1038/nature14539.

[3] R. Roy, AI, ML, and DL: How not to get them mixed!, (2019). https://towardsdatascience.com/understanding-the-difference-between-ai-ml-and-dl-cceb63252a6c (access date 07 January 2021).

[4] H. Lütkepohl, M. Krätzig, P.C.B. Phillips, Applied time series econometrics, *Cambridge University Press*, 2004.

[5] D. Cano, J.M. Monget, M. Albuisson, H. Guillard, N. Regas, L. Wald, A method for the determination of the global solar radiation from meteorological satellite data, *Solar Energy*. 37 (1986), 31-39. doi:10.1016/0038-092X(86)90104-0.

[6] S.E. Rusen, Modeling and analysis of global and diffuse solar irradiation components using the satellite estimation method of HELIOSAT, *Computer Modeling in Engineering & Sciences*. 115 (2018), 327-343.

[7] S. Ener Rusen, A. Konuralp, Quality control of diffuse solar radiation component with satellite-based estimation methods, *Renewable Energy*. 145 (2020), 1772-1779. doi:10.1016/j.renene.2019.07.085.

[8] M. Abdel-Nasser, K. Mahmoud, Accurate photovoltaic power forecasting models using deep LSTM-RNN, *Neural Computing and Applications*. 31 (2019), 2727-2740. doi:10.1007/s00521-017-3225-z.

[9] R.K. Agrawal, F. Muchahary, M.M. Tripathi, Long term load forecasting with hourly predictions based on long-short-term-memory networks. In *2018 IEEE Texas Power and Energy Conference (TPEC), IEEE*, 2018: ss. 1-6. doi:10.1109/TPEC.2018.8312088.

[10] S. Balluff, J. Bendfeld, S. Krauter, Short term wind and energy prediction for offshore wind farms using neural networks. In *2015 International Conference on Renewable Energy Research and Applications (ICRERA), IEEE*, 2015: ss. 379-382. doi:10.1109/ICRERA.2015.7418440.

[11] A. Gensler, J. Henze, B. Sick, N. Raabe, Deep Learning for solar power forecasting — An approach using AutoEncoder and LSTM Neural Networks. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), IEEE*, 2016: ss. 002858-002865. doi:10.1109/SMC.2016.7844673.

[12] H. Sharadga, S. Hajimirza, R.S. Balog, Time series forecasting of solar power generation for large-scale photovoltaic plants, *Renewable Energy*. 150 (2020), 797-807. doi:10.1016/j.renene.2019.12.131.

[13] U. Şencan, Short term electricity price forecasting using Long Short-Term Memory, Thesis, *Bahçeşehir University*, 2018.

[14] F. Özen, R. Ortaç Kabaoğlu, T.V. Mumcu, Deep learning based temperature and humidity prediction, *Necmettin Erbakan University Journal of Science and Engineering*. (2023). doi:10.47112/neufmbd.2023.20.

[15] M. Hacibeyoglu, M. Çelik, Ö. Erdaş Çiçek, Energy efficiency estimation in buildings with K nearest neighbor algorithm, *Necmettin Erbakan University Journal of Science and Engineering*. 5 (2) (2023), 65-74. doi:10.47112/neufmbd.2023.10.

[16] N.C. Alparslan, A. Kayabasi, S.E. Rusen, Estimation of global solar radiation by using ANN and ANFIS. In *2019 Innovations in Intelligent Systems and Applications Conference (ASYU), IEEE*, 2019: ss. 1-6. doi:10.1109/ASYU48272.2019.8946448.

[17] W. Donat, What is Python: An Intro to a Cross-Platform Programming Language, (2015). https://www.atlantic.net/vps-hosting/what-is-python-intro-cross-platform-programming-language/ (access date 01 June 2021).

[18] Anaconda Software Distribution, *Anaconda Documentation*. (2020). https://docs.anaconda.com/ (access date 01 October 2021).

[19] Anaconda Navigator, (2020). https://docs.anaconda.com/anaconda/navigator/ (access date 07 January 2021).

[20] C.R. Harris, K.J. Millman, S.J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N.J. Smith, R. Kern, M. Picus, S. Hoyer, M.H. van Kerkwijk, M. Brett, A. Haldane, J.F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, T.E. Oliphant, Array programming with NumPy, *Nature*. 585 (2020), 357-362. doi:10.1038/s41586-020-2649-2.

[21] J.D. Hunter, Matplotlib: A 2D Graphics Environment, *Computing in Science & Engineering*. 9 (2007), 90-95. doi:10.1109/MCSE.2007.55.

[22] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems, (2016). https://arxiv.org/abs/1603.04467.

[23] M. Najibi, G. Lai, A. Kundu, Z. Lu, V. Rathod, T. Funkhouser, C. Pantofaru, D. Ross, L.S. Davis, A. Fathi, DOPS: Learning to Detect 3D Objects and Predict their 3D Shapes, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. (2020), 11913-11922. http://arxiv.org/abs/2004.01170.

[24] W. McKinney, Data Structures for Statistical Computing in Python. In 2010: ss. 56-61. doi:10.25080/Majora-92bf1922-00a.

[25] F. Chollet, Keras: Deep learning for humans, (2015).

[26] S. Seabold, J. Perktold, Statsmodels: Econometric and Statistical Modeling with Python. In 2010: ss. 92-96. doi:10.25080/Majora-92bf1922-011.

[27] Turkish State Meteorological Service, Türkiye Global Güneş Radyasyonu Uzun Yıllar Ortalaması (2004-2018), (2018). https://www.mgm.gov.tr/kurumici/radyasyon_iller.aspx (access date 07 January 2021).

[28] S. Hochreiter, J. Schmidhuber, Long Short-Term Memory, *Neural Computation*. 9 (1997), 1735-1780. doi:10.1162/neco.1997.9.8.1735.

[29] C. Olah, Understanding LSTM Networks, (2015). http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (access date 07 January 2021).

[30] G.E.P. Box, G.M. Jenkins, Time series analysis: forecasting and control, *Holden-Day*, 1970. https://books.google.com.tr/books?id=5BVfnXaq03oC.

[31] M. Ghofrani, M. Alolayan, Time Series and Renewable Energy Forecasting. In *Time Series Analysis and Applications*, *InTech*, 2018: ss. 77-92. doi:10.5772/intechopen.70845.

[32] Ö. Zeydan, Zonguldak bölgesi pm10 konsantrasyonu dağılımının modellenmesi, Thesis, *Kocaeli Üniversitesi*, 2014. http://dspace.kocaeli.edu.tr:8080/xmlui/handle/11493/856.

[33] Anonymous, MSE, RMSE, MAE, MAPE ve Diğer Metrikler, (2017). https://veribilimcisi.com/2017/07/14/mse-rmse-mae-mape-metrikleri-nedir/ (access date 07 January 2021).

[34] S. Wright, Correlation and causation, *Journal of agricultural research*. 20 (1921), 557.

[35] N. Çarkacı, Derin Öğrenme Uygulamalarında En Sık kullanılan Hiper-parametreler, (2018). https://medium.com/deep-learning-turkiye/derin-ogrenme-uygulamalarinda-en-sik-kullanilan-hiper-parametreler-ece8e9125c4 (access date 07 January 2021).

[36] T. Okonkwo, Hyperparameter Optimization-Building an Optimal Model, (2019). https://medium.com/@THOR_mas/hyperparameter-optimization-building-an-optimal-model-b11677bf3dfc (access date 07 January 2021).

[37] T.G. Smith, others, pmdarima: ARIMA estimators for Python, (n.d.). http://www.alkaline-ml.com/pmdarima (access date 07 January 2021).

[38] H. Akaike, A new look at the statistical model identification, *IEEE Transactions on Automatic Control*. 19 (1974), 716-723. doi:10.1109/TAC.1974.1100705.

[39] D.A. Dickey, W.A. Fuller, Distribution of the Estimators for Autoregressive Time Series with a Unit Root, *Journal of the American Statistical Association*. 74 (1979), 427-431. doi:10.1080/01621459.1979.10482531.