An Automatic Supervision Method for Photovoltaic-Arrays Systems

Virgilio López-Morales, *Member, IAENG,* Iván S. Razo-Zapata,
Joel Suarez-Cansino and Manuel A. Ojeda-Misses

***Abstract*—Machine Learning (ML) techniques provide powerful tools to create intelligent supervisory systems. This study demonstrates their effectiveness in the context of distributed photovoltaic systems (DPVs) by introducing an automated supervision method. The method has been designed based on various ML models that were trained on historical data encompassing environmental conditions (e.g., temperature, humidity) and real-time measurements from DPVs' inverters. The method can automatically forecast DPVs' energy generation for different time periods. The final supervisory system, implemented using the best-performing Deep Learning model obtained, is validated through comprehensive metrics, demonstrating its accuracy and utility in monitoring DPV's performance based on physical and internal variables.**

***Index Terms*—Supervisory System, Discrete Integral, Deep Learning Modeling, LSTM, Machine Learning.**

I. Introduction

RENEWABLE energy systems are growing exponentially to meet worldwide electricity demand and reduce greenhouse gas emissions. Government policies and cost reductions in photovoltaic systems (PV) have led to their incredible growth in recent years. As stated in [1], for decades PV systems have shown their scalability and reliability in a wide variety of applications such as power for remote weather stations and telecommunications towers, residential installations, community micro-grids, and grid-scale power plants.

According to the International Renewable Energy Agency (IRENA), in 2020 the world installed a historic 260 gigawatts (GW) of new renewable power generation capacity, exceeding 2019 by almost 50% and constituting more than 80% of all new electricity capacity added that year. The majority of these new renewable additions (91%) were only on Solar and wind power [2]. This growth in PV installed capacity has resulted in an unconventional contribution to global electricity generation by more than 2% and an extrapolation demonstrates a future planet-wide capacity of 1700 GW by 2030 [3]. Furthermore, the above mentioned report is supported by the latest communication from several environmental and energy agencies that point out the increasing demand for solar energy and especially for photovoltaic systems, which promotes the need for accurate and reliable forecasting of photovoltaic energy [4], [5].

This exponential growth is undeniable, despite the intermittent and variable nature of PV power generation due to different reasons as meteorological events as for instance humidity, cloud coverage, temperature, solar irradiation, and dust accumulation, which can alter in a negative manner PV panel performance and production. This variability can affect the operation of the electricity system and brings several challenges related to the integration of PV systems within grid power systems; i.e., changes in meteorological conditions lead naturally to variations in PV power, voltage, and frequency as well as reverse power flows among other problems. These power and frequency varitions influence the reliability, stability, and scheduling of the entire electrical system. Furthermore, the intrinsic uncertainty of the PV system' output power poses economic complications, underscoring the importance of accurate prediction models that support efficient economic models for a proper PV system operation [6], [7], [8].

Therefore, having an effective PV power forecasting model is essential for developing a cost-effective solution to address the aforementioned uncertainties, improve overall system stability, and thus achieve improved economic performance.

Using such a reliable and effective PV power prediction model, an application can be built to classify the operation of the entire system or to detect those modules of the system that are presenting some type of failure or recurring errors due to hardware conditions (interruption of communication between modules, shutdown of a measurement module, blown fuse, etc.), meteorological, environmental (seasonal) or human error.

Real-world data presents highly nonlinear dynamics and in addition irregularity and high complexity, thus, it is very difficult for classical methods or traditional statistical models, to achieve high-precision predictions. Such methods as Autoregressive Moving Average (ARMA) or Autoregressive Integrated Moving Average (ARIMA) frequently, they only perform well with short-term forecasts. Furthermore, their performance deteriorates significantly when parameters are not properly selected; for example, the ARIMA-based model is not suitable for time series data with weak periodic characteristics (Cf. [9], [10]).

Machine learning (ML) techniques leverage training and learning iterations to obtain more accurate prediction results and, consequently, a better synthetic model of the real system. For example, support vector regression or classification-based methods [11], kernel-based methods and artificial neural networks [12], and the tree-based ensemble learning

Manuscript received September 30, 2024; revised October 15, 2025.

This work was supported in part by the CONAHCYT-PRODEP under Grant 15102024.

Virgilio López-Morales is a professor and academic group head at the Information Technologies and Systems Research Center, ICBI-UAEH, C.P. 42184, Hgo., Mexico (correspondant e-mail: virgilio@uaeh.edu.mx).

Iván S. Razo-Zapata is the Technology Director of COCOA B.V., The Netherlands (e-mail: ivan@cocoa-ci.org).

Joel Suarez-Cansino is the Technology Director of QueryThem Co., Mexico (email: querythemx@gmail.com).

Manuel A. Ojeda-Misses is a professor at the Information Technologies and Systems Research Center, ICBI-UAEH, C.P. 42184, Hgo., Mexico (e-mail: manuel_ojeda@uaeh.edu.mx).

method (gradient boosting regression or decision tree [13]) have also been used due to their strong ability to approximate nonlinear functions. However, they have some shortcomings when addressing the sequential dependence between input variables in the time series forecasting problem.

Recurrent neural networks use internal memory cells to manage temporal data and are used to model historical and future states ([14], [15], [16], [17]). However, the problem of vanishing gradients in these networks limits their performance. To overcome this problem, Long-Short-Term Memory (LSTM) has introduced a *gate* that determines what information to remember and forget to obtain long- and short-term memory for historical data. For example, [18] combines an LSTM and bidirectional LSTM networks to perform transportation predictions. In [19], a unified decomposition-based network architecture (LSTM-MSNet) is proposed to predict multiple seasonal time series [16], [17].

Transformer-based (TB) time series modeling is a very popular research axe, with modeling capabilities similar to those of traditional neural networks. Their inherent advantages in processing and predicting long sequence data allow TB models to achieve good performance on most temporal tasks [20], [21]. However, their particular complexity leads to a considerable increase in memory usage when running on long sequence data, as in this case study. Therefore, current research mainly focuses on optimizing efficiency and reducing algorithm complexity [22], [23], [24]. For a more detailed review, cf. [25], which provides a comprehensive review of various models, from regression to deep learning algorithms.

Efficient time series forecasting allows for the optimization of existing information for analysis and decision-making. Its wide range of applications also includes clinical medicine [26], financial forecasting [27], traffic flow prediction [28], and human action prediction, among other fields.

Unlike other predictive modeling tasks, time series add the complexity of sequence dependencies between input variables.

Unlike other predictive modeling tasks, an additional complexity is added by time series such as sequence dependencies between input variables. Building a predictive model tailored to real-world data is crucial to take full advantage of complex sequence dependencies. We conducted a comprehensive evaluation of the leading ML techniques using two metrics: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). The three best ML techniques were selected based on the aforementioned metrics, which measure their performance: Random Forest (RF), Support Vector Machines (SVM), and Long Short-Term Memory (LTM) techniques. The LSTM architecture performed best and was therefore selected. Furthermore, it can learn long-term dependencies using gates, enabling information retention and robust results in long-term prediction.

In this paper, we propose a methodology for building a reliable PV power prediction model and a classifier to systematize the operational categories of a photovoltaic power system's monitoring operation. This method relies on a Long Short-Term Memory (LSTM) architecture and a discrete integration model-based function to display the different operating categories of the PV system. The choice of this technique is based on a series of numerical simulations and a comparison with the two best-performing LSTM models (based on the aforementioned metrics).

The selection of the LSTM is based on a series of numerical simulations that evaluate the best possible performance of the analyzed models. After several tests, we chose the LSTM architecture due to its excellent performance (based on RMSE and MAE) and its greater reliability for building our monitoring system.

The paper organization is as follows: Section II introduces the data analysis methodology to provide a comprehensive framework for data mining steps and a brief recall of SVM, RF and the architecture of an LSTM. Section III presents the PV power forecasting algorithm that is coupled with the classifier. Section IV shows some results based on the data collected through PV power panels system located in Mexico. Finally, Section V highlights conclusions and future work.

II. Data Analysis Methodology and three main ML techniques

In our case the time period for output forecasting or the time duration between actual and effective time of prediction, defined as the forecast horizon, is a short-term forecasting (one-day or intraday market forecasting) after the categories stated in [29]. In these categories, the temporal horizon for the short-term forecasting can range from 30 to 60 minutes, up to a day. However, some authors consider one to several hours, one day, or up to seven days as a short-term forecast horizon (Cf. [30]). To have a sound management of the power system operation short-term forecasting is here selected.

Short-term forecasting supports the day-ahead market since it can offer sellers-buyers to trade closer to the moment of physical delivery. However, this transaction is typically available one day before delivery and offers the opportunity to place bids and orders to buy and sell energy from 0 to 24 hours before actual delivery [31].

Due to the nature of the present project, the main elements to allow for implementing the application, are a dataset containing the meteorological physical variables of the precise location where the PV system is located and the physical variables from a historical dataset to provide the forecast outcomes (PV power generation, performance status and eventually predict the origin of poor performances).

In this paper, we have collected the electricity power generated, directly from the smart inverters connected to a PV system located in Queretaro, Mexico.

The historic meteorological physical variables were collected from the database of a Government Agency (Nasa Power: Prediction Of Worldwide Energy Resources) that provides solar and meteorological datasets from NASA research for support of renewable energy, building energy efficiency, and agricultural needs [32].

To measure and evaluate the performance of the different solar PV power forecasting models, several metrics are used depending on user requirements, application type, and its corresponding forecast horizon. Thus, actually there is no a common evaluation method. Broadly speaking, error metrics can be subdivided into four groups: ramp rate, statistical, uncertainty, and economic metrics [33].

The most commonly used error metrics in solar forecasting are the statistical error metrics, which provide a quantitative

description of forecast accuracy. Since our goal is to achieve good performance in solar energy forecasting, to evaluate the deviation between model predictions and actual measured values, we use mean absolute error (MAE) and root mean square error (RMSE), two well-known statistical error metrics.

The MAE metric is defined as:

$$MAE = \frac{1}{T}\sum_{i=1}^{T}|y_i - \hat{y}_i|, \tag{1}$$

where the horizon can be based on an evaluation of the classical persistence model as follows:

$$\hat{y}_{(i+k|i)} = y_i. \tag{2}$$

In contrast, the RMSE metric is quite sensitive to large forecasting errors and is therefore suitable for applications where small errors are more tolerable and larger errors cause disproportionately high costs. The RMSE is defined as follows:

$$RMSE = \sqrt{\frac{1}{T}\sum_{i=1}^{T}(y_i - \hat{y}_i)^2}, \tag{3}$$

There is another model known as the clear-sky persistence model to account for daily and seasonal variations, as well as the energy production of the PV system under clear-sky conditions, which is defined as: [34]:

$$\hat{y}_{(i+k|i)} = y_i \frac{k^m_{i+k}}{k^m_i}, \tag{4}$$

where k^m is the power production of the PV system under clear-sky conditions and can be calculated by using a clear-sky model as input to the physical model. The term $\frac{y_i}{k^m}$ is also known as the clear-sky production index. To obtain and use this former model is mandatory to have a considerable amount of data with historical measurements with the particular PV system to validate the indices under these conditions. Nevertheless, in this paper since our goal is to show that a supervision system can be implemented using a machine learning modeling method, we will utilize RMSE and MAE as our metrics.

A. SEMMA Data Mining Methodology

There are many different methods to extract knowledge from data. However, the data science lifecycle generally describes the step-by-step approach to take to deliver a project. Therefore, it is important to follow a method to ensure a certain level of quality of the information obtained from the data throughout the process, as data science teams are increasingly expanding into others areas of work, such as legal or commercial.

The SEMMA methodology was developed by the SAS (Statistical Analysis Systems) Institute, founded in 1976 in part by Jim Goodnight. Its acronym refers to its five-phase process: Sampling, Exploration, Modification, Modeling, and Evaluation.

SEMMA offers a comprehensive framework for data mining projects that prioritizes flexibility and helps guide practitioners through the steps required to extract valuable insights from large amounts of data [35].

A. Sample: The step aims to select the representative data which must be of high quality and relevance. Some key sub-steps are: a1) Collection of large datasets from all possible sources. a2) Selection of the representative subsets to have an efficient numerical treatment. a3) Consider in detail the complexities, different dynamics and patterns of the entire dataset framework, to have a complete dataset in the sense of variability.

B. Explore: The step aims to uncover initial patterns, dynamics, outliers, or factors influencing subsequent phases. The sub-steps are usually: b1) Apply a descriptive statistics. b2) Visualize data to find some repetitive patterns or trends. b3) Identify outliers, their location and their relationships with the whole dataset.

C. Modify: This step aims to transform the initial dataset into a cleaner and more refined one, suitable for modeling. The common sub-steps are: c1) Clean the data by handling the missing values, anomalies or outliers. c2) Create, select and transform variables. c3) Provide a correct format or structure to the dataset according to the modeling method.

D. Model: The step aims to compare several modeling methodologies to seek for hidden patterns, singularities, useless data and tuning the parameters' methodology to discover better performances. Some considered sub-steps are: d1) Choose the appropriate data mining techniques as classification, regression, decision trees, deep learning techniques, etc). d2) Build several models by using a part of the complete dataset. d3) Fine-tuning the parameters or variables used in the method to find the best performance.

E. Assess: This step aims to evaluates the model's accuracy, reliability, and overall utility. Usual sub-steps are: e1) Evaluate model performance on validation datasets (data that was not used in the modeling/training process). e2) Compare obtained results and performance against metrics and other methodologies, as well as against the project objectives. e3) Fine-tuning in an iterative manner to find an optimal solution.

Given its completness, the SEMMA methodology is utilized in this paper.

B. Random Forest modeling technique

A Random Forest is an ensemble learning method that builds numerous decision trees on different random subsets of the training data to make predictions. By combining the results of these individual "weak" trees through techniques like majority voting (for classification) or averaging (for regression), the Random Forest model achieves higher accuracy, reduced risk of overfitting, and better stability than a single decision tree. Its key advantage lies in its robustness and its ability to naturally handle both classification and regression tasks while also providing insights into feature importance.

In the strong law of large number, it can be formally stated as the following. Let X_1, X_2, X_3, $\cdots$, X_n be identically distributed random variables with a finite expected value $E(X_i) = \mu < \infty$. Let also $M_n = (X_1 + X_2 + ... + X_n)/n$. Then, $M_n \longrightarrow \mu$, is almost certain.

Therefore, RF has some advantages over other machine learning methods due to the randomness in data and feature selection, helps prevent the model from memorizing the training data, improving its ability to generalize to new, unseen data. [36].

Thus, a random forest is a meta estimator that fits a series of decision tree regressors on various sub-samples of the dataset. It uses averaging to improve the predictive accuracy and control the over-fitting. The algorithm can quantify the importance of each feature in making predictions, helping with feature selection and understanding the data [37].

The random forest technique has a set of main parameters as:

- Random state: Controls both the randomness of the bootstrapping of the samples used when building trees (if bootstrap=True) and the sampling of the features to consider when looking for the best split at each node (if max-features $<$ n-features).
- Number of estimators (N): The number of trees in the forest.
- Max features: The number of features to consider when looking for the best split. See also [38], for a deeper understanding on theory and implementation details.

C. Support Vector Machine modeling technique

A Support Vector Machine (SVM) is a supervised learning algorithm used for classification and regression. It is generally designed to find the optimal hyperplane that maximally separates different classes (within the data). When data has non-linearly separable classes, it utilizes kernel functions to unconditionally map the data into a higher dimensional space where it evolves into a linearly separable one. It is known as the *kernel trick*, and the choice of kernel function, such as linear kernels polynomial kernels, radial basis function (RBF) kernels, or sigmoid kernels, depends on the characteristics of the data and the specific use case.

Support vectors are paramounts, since they are the data points closest to the optimal hyperplane that define its position and ultimately determine the maximal margin [39], [40].

SVMs are commonly used within classification (linear and nonlinear) problems where the number of features in the input data determine if the hyperplane is a line in a 2-D space or a hyperplane in a n-dimensional space. Since multiple hyperplanes can be found to differentiate classes, maximizing the margin between points allows the algorithm to find the best decision boundary between classes. This, in turn, allows it to generalize correctly to new data and make accurate classification predictions. For example, [41] proposes forecasting a building's energy consumption to facilitate energy management by combining this technique with an optimization algorithm. See also [42], for a deeper understanding on theory and implementation details.

D. An LSTM architecture modeling technique

Forecasting, in general, is a complex task, and the data needed to accurately model a complex, real-world system is not always easy to select and validate. Using a Long Short-Term Memory (LSTM) model, persistent prior information, both short-and long-term, is incorporated into a neural memory network's memory (Cf. [43]).

There are several types of LSTM architectures in the literature, then to give a concise explanation of how an LSTM cell works, let us take the standard architecture shown in Fig. 1. An LSTM cell has a persistent characteristic and the original input signals is completed with its own states and cell activations, i.e., $\vec{x}(t)$, $\vec{s}(t-1)$ and $\vec{h}(t-1)$ are the input signals (at time t and $t-1$, respectively). It consists of three main gates: the forget gate, the input gate, and the output gate. Each gate uses a sigmoid activation function (σ) to produce values between 0 and 1, controlling the flow of information.

Note that the vector state $\vec{s}$ and the vector activation $\vec{h}$ have the same dimension (m inputs each). Naturally, m is often different from the dimension of the input vector X ($dim = n$). An augmented vector ($\left(\vec{h}(t-1), \vec{x}(t)\right)$) is then obtained at the output of the mux operator.

There are a set of nodes in the gates ("hidden units") where for every gate, there are many hidden units as the dimension of the activation vector $\vec{h}$ or the cell state vector $\vec{s}$ ($dim = m$).

First gate is called "forget gate"(f), who is in charge of determining which information from the previous cell state ($\vec{s}(t-1)$) should be discarded. It takes the previous hidden state ($\vec{h}(t-1)$) and current input ($\vec{x}(t)$) as inputs.

$$\vec{\sigma_f} = \vec{\sigma}\left(\Omega_f \left[\vec{h}(t-1), \vec{x}(t)\right]^T\right), \tag{5}$$

where if $\vec{\sigma_f}$ is close to 0, information is forgotten while if it is close to 1, it is retained.

To achieve that, the rows of the forget gate's weight matrix, multiply each component of the augmented input vector ($\vec{h}(t-1), \vec{x}(t)$). Therefore, the first row will correspond to the first hidden neuron in the block, from left to right, and so on. A similar thing happens to the gates i, g, and o with the weight matrices Ω_i, Ω_g, and Ω_o, respectively.

The Input Gate ($\vec{\sigma_i}$) and a Candidate State ($\vec{\eta_g}$) control which new information will be added to the cell state.

$$\vec{\sigma_i} = \vec{\sigma}\left(\Omega_i \left[\vec{h}(t-1), \vec{x}(t)\right]^T\right), \tag{6}$$

$$\vec{\eta_g} = \vec{\eta}\left(\Omega_g \left[\vec{h}(t-1), \vec{x}(t)\right]^T\right), \tag{7}$$

where the Candidate State constains new content (information) that could be stored, and $\vec{\sigma}_i$ *decides* (weighting) how much of it to keep.

A fifth component, s at the top of Fig. 1 is to update the cell state by combining the old and the new information

$$\vec{s}_t = \vec{\sigma_f} * \vec{s}(t-1) + \vec{\sigma_i} * \vec{\eta}_g. \tag{8}$$

The weight matrix Ω_s is the identity matrix with order m. Note that sigmoid or logsig $\sigma(x)$, and tansig $\eta(x)$ functions, help perform the former operations.

Finally, the Output Gate ($\vec{\sigma_o}$) determines what the next hidden state ($\vec{h}(t)$) will be, which also serves as output:

$$\vec{\sigma_o} = \vec{\sigma}\left(\Omega_o \left[\vec{h}(t-1), \vec{x}(t)\right]^T\right), \tag{9}$$

$$\vec{h}(t) = \vec{\sigma_o} * \tanh \vec{s}_t. \tag{10}$$

This architecture allows LSTMs to learn context over long time spans, preserving information relevant to future predictions while discarding noise. Consequently, LSTMs are widely used where understanding temporal patterns is crucial (v.gr. speech recognition, time-series forecasting, natural language processing, among other sequential data tasks).

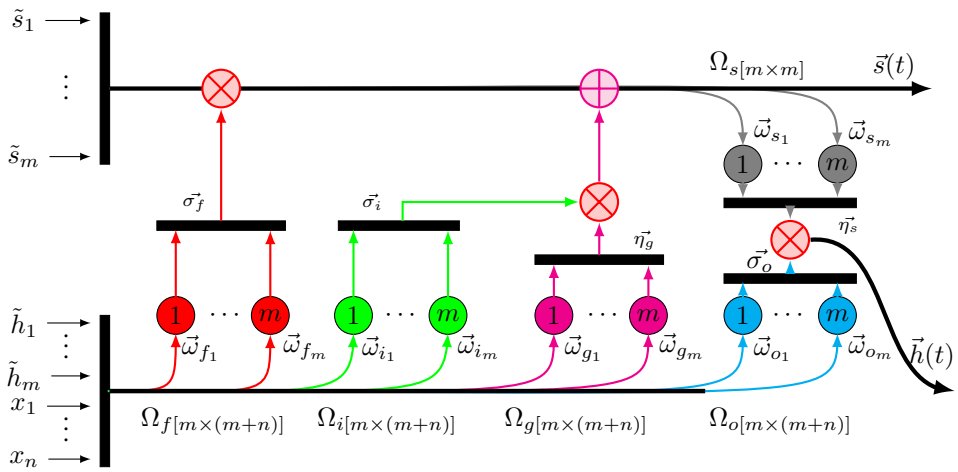

Fig. 1. Internal structure of an LSTM where $\tilde{s}_i, \tilde{h}_j;\ \ i,j = 1,2,\cdots,m$ stand for $s_i(t-1)$ and $h_j(t-1)$ respectively,

$$\Omega_s = \begin{bmatrix} \vec{\omega}_{s_1} \\ \vdots \\ \vec{\omega}_{s_m} \end{bmatrix}; \Omega_f = \begin{bmatrix} \vec{\omega}_{f_1} \\ \vdots \\ \vec{\omega}_{f_m} \end{bmatrix}; \Omega_i = \begin{bmatrix} \vec{\omega}_{i_1} \\ \vdots \\ \vec{\omega}_{i_m} \end{bmatrix}; \Omega_g = \begin{bmatrix} \vec{\omega}_{g_1} \\ \vdots \\ \vec{\omega}_{g_m} \end{bmatrix}; \Omega_o = \begin{bmatrix} \vec{\omega}_{o_1} \\ \vdots \\ \vec{\omega}_{o_m} \end{bmatrix};$$

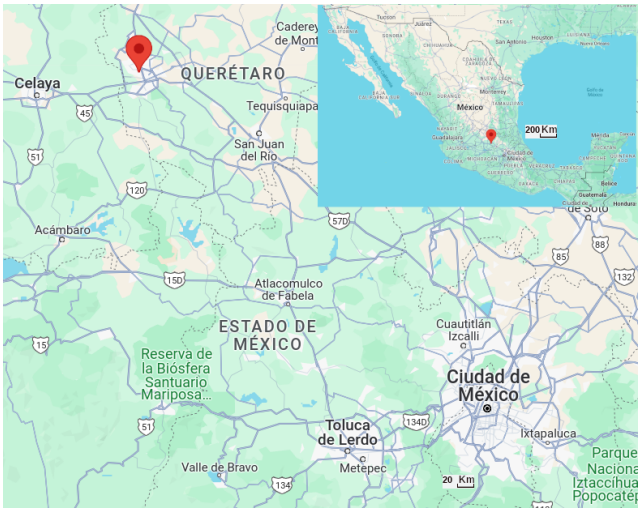

Fig. 2. PV arrays System Location in Queretaro, Mexico.

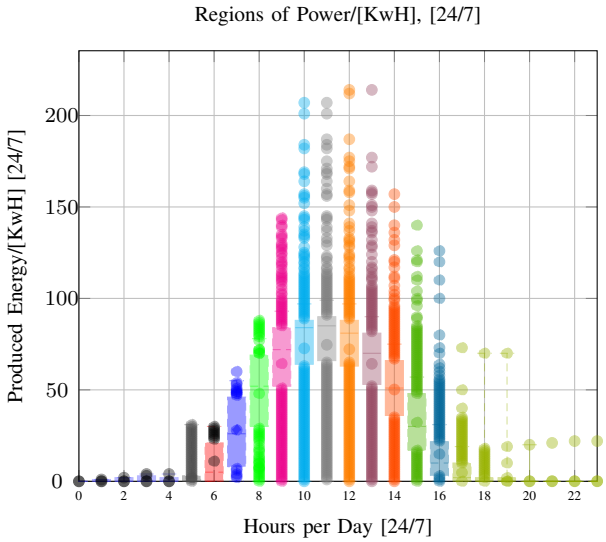

Fig. 3. Temperatures' Accumulation before removing outliers

III. PV Power System Modeling.

To provide a contextual analysis of the current project's results, the analyzed dataset includes features such as power measurements from rooftop PV systems, all composed of mono-Si panels.

As mentioned above, historical meteorological variables were collected from the NASA database. PV array system power output was collected at one-minute resolution between January and March 2022 and averaged to obtain mean hourly production values. Since the NASA database has hourly resolution, we modified the PV system power output accordingly.

Cleaning the database:

System behavior can be identified across the entire database by aggregating all data into a single 24-hour format. Classification can be performed regardless of whether the measurements correspond to different days, months, or years.

After processing the raw data and performing normality tests, outliers can be detected and removed. Finally, the filtered data can be renormalized to validate its normal distribution, and a confidence interval can be calculated and displayed. This process is shown in Fig. 3. This figure shows that valuable data is found between 5:00 a.m. and 5:00 p.m., and then outliers outside this interval can be removed. It is worth noting that, despite the database processing, outliers (in the 5:00 a.m. to 5:00 p.m. range) still exist, which could be due to the low fidelity of the sensor measurements.

A. Numerical weather variables

Approach: After a thorough evaluation of the aforementioned three leading machine learning (ML) techniques, this paper selects an LSTM model for use in the PV Power Forecasting System. The main variables considered to determine the best model are the following (according to the acronyms assigned by the NASA database):

- CLRSKY_SFC_SW_DWN: Clear Sky Surface Shortwave Downward Irradiance (Wh/m^2),
- CLRSKY_SFC_PAR_TOT: Clear Sky Surface PAR Total (W/m^2),

- T2M: Temperature at 2 Meters (C),
- RH2M: Relative Humidity at 2 Meters (%),
- PRECTOTCORR: Precipitation Corrected (mm/hour),
- WS10M: Wind Speed at 10 Meters (m/s),
- ALLSKY_KT: All Sky Insolation Clearness Index (dimensionless),
- T2MWET: Wet Bulb Temperature at 2 Meters (C),

B. Historical Dataset

The numerical weather variables were collected from NASA weather site [32]. The resolution chosen was hourly, while our generated energy measurements were obtained directly from the photovoltaic systems using inverters. Like any raw dataset, it contains errors and missing numerical values. Therefore, in a first step, the dataset is preprocessed.

The dataset was divided into three parts to ensure that the resulting model worked with both the internal data of the values observed by the training method and those used for pure prediction. Thus, the training variables were defined as if they had been measured at a conventional weather station. Furthermore, in an initial test of the modeling methodologies, these variables showed greater influence in capturing the dynamics of the target variable (the electrical power of the PV systems).

Then, the final variables retained for the modeling phase, are:

Inputs:

- NumData: Measurement's position within the DB,
- T2M: Temperature at 2 Meters (^{o} C),
- RH2M: Relative Humidity at 2 Meters (%),
- WS10M: Wind Speed at 10 Meters (m/s),
- CLRSKY: Clear Sky Surface PAR Total (W/m^2),
- Rain: Precipitation Corrected (mm/hour),

Output:

- WH: Power measurement of a representative subset of the entire PV array system measured directly from the inverters ([0 5500] Watts/hr.)

C. Pre-processing

The variables are collected in a matrix $X_{m\times n}$ where m indicates the number of observations (hours) downloaded, the first $n-1$ variables indicate the number of predictor variables and, in the $n-th$ column the generated power is concatenated.

During the night, WH was set to zero to remove outliers. If negative values, blanks, or other errors are present, these cells are labeled as NaN. Previous NaN values were replaced by the results of a linear interpolation calculated with the values at the corresponding time steps. The training data were normalized, as machine learning (ML) methodologies are sensitive to high values in the dataset. In a first round of testing, several ML techniques were evaluated, and the three best modeling methods were ultimately selected: Support Vector Machines (SVM), Random Forest (RF), and Long Short-Term Memory (LTSM).

D. Modeling and Classification.

To make a fair comparison between the three methods, we performed several rounds of training and evaluation of various sets of parameters to obtain the best candidate for each method.

TABLE I
BEST MACHINE LEARNING METRICS OBTAINED.

Training & Validation Metrics for the Best Three ML Model.		
Best result:	**RMSE**	**MAE**
RF	733.1574	433.1217
SVM	1019.3498	637.2480
LSTM	541.2866	300.8616

The best machine learning model was selected based on the MAE and RMSE metrics (Cf. Eq. 3). The MAE and RMSE metrics (training and validation) for these three ML modeling techniques are shown in Table I. The final configuration of the three models is as follows:

- RF: No. Estimators = 73; Max features = n; Random State = 16;
- SVM: kernel = linear; rndm state = 1; degree = 3;
- LSTM: No. Hidden Units = 128; Optimizer = adam; DropOut Prob = 0.2.

In the end, the best model was the LSTM neural network.

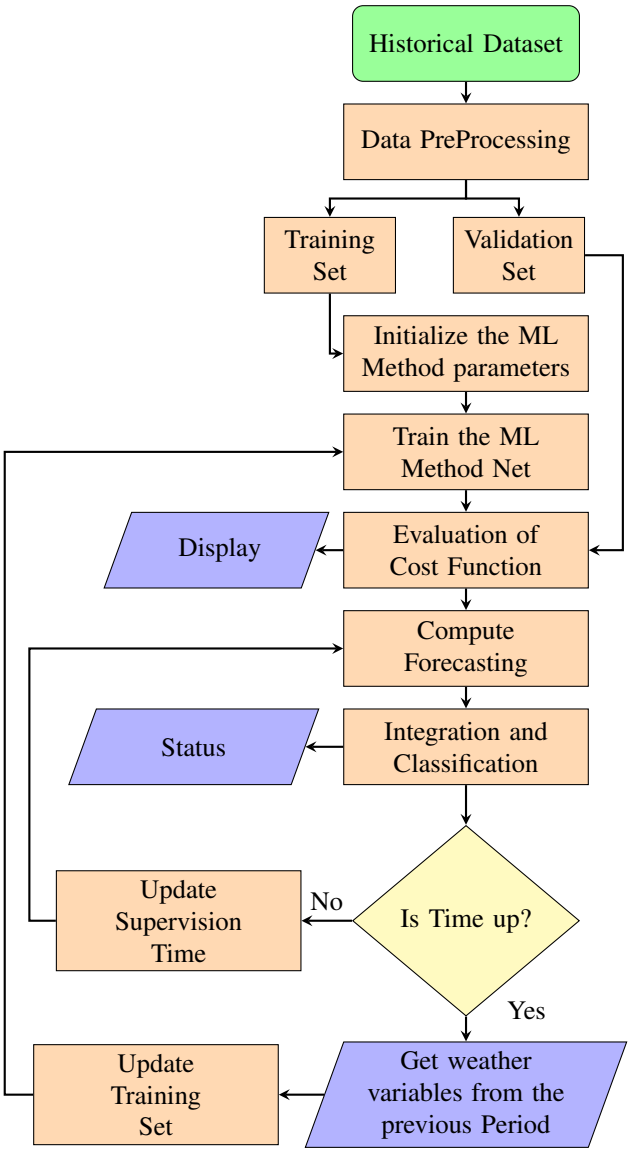

Fig. 4. Process Flowchart for the Supervisory System.

E. Flowchart of the implemented LSTM and the Decision Making Process.

Figure 4, shows the process flowchart, which reveals the steps followed for the implementation of the supervisory system and the decision-making process.

As can be seen in flowchart 4, the dataset is divided into a training set ($X_{training}$) and a test set (X_{test}). The training set represents 75% of the total dataset. The validation set (the remaining 25%) was selected from the end of the dataset (March 2022).

Metric functions are then sent to the user to assess whether the hyperparameters of a new LSTM model need to be refactored. A first forecast (accurate to within one hour) is then calculated. Discrete integration of PV power generation provides the sum up to the nearest elapsed hour (due to the granularity of the meteorological dataset) to compare the expected PV power generation (from the LSTM model) with the currently generated PV power. Based on this information, a discrete integral classifier is used to calculate the current operating state of the PV system, which will be displayed to the user (Degraded:Red, Caution:Orange, and Normal:Green).

The pseudocode used for this classifier is presented below:

```
import discreteintegral as integrate

def DIntegral(Yt,Yp,Yr,Ip,Fp):
#Yt:Output used for training data
#Yp:Output predicted through the Model
#Yr:Output real
#Ip:Initial point to integrate & compare
#Fp:Final point to integrate & compare

def definebounds-grid-labels-axis
def defineFpred-as-a-function-of-t
def defineFreal-as-a-function-of-t

FpredI = integrate(Fpred(FromIp-toFp))
FrealI = integrate(Freal(FromIp-toFp))

outputPerCent = FrealI/FpredI*100

if outputPerCent >= 85 &&
   outputPerCent <=115
% if PV_output =[85% 115%] then
            production High=greenLight
Light = 'Green'

elseif outputPerCent >= 65 &&
       outputPerCent <85
% if PV_output =[65% 85%] then
    production average=orangeLight
Light = 'Orange'

else
Light = 'Red'  % redLight Otherwise

return Light
```

The system is programmed so that if the difference between the actual and predicted power value is within a range of 15%, or another provided by the end user, the system is considered to be operating normally. If the difference is between 15% and 30%, the system is operating with slight degradation. If the difference is greater, the DPV system should be inspected, as a problem is likely present.

After the time period set by the user or the program has elapsed (by default, two hours before sunrise and two hours after sunset), the LSTM model can be updated with the numerical meteorological variables measured and stored by the local weather station. A new model is then trained and updated to reflect changes in the current weather system's trends and is ready to forecast the next day. In this case, a portion of the actual input variables (future days) is reserved to simulate this last scenario, as a local weather station is not yet available.

IV. Results

The resulting model is then fed with input variables from the near future (an independent data set), and the results are compared to the actual PV power obtained during these time intervals. The time values (Time/[Hour]) are based on sequential measurements collected in the data set.

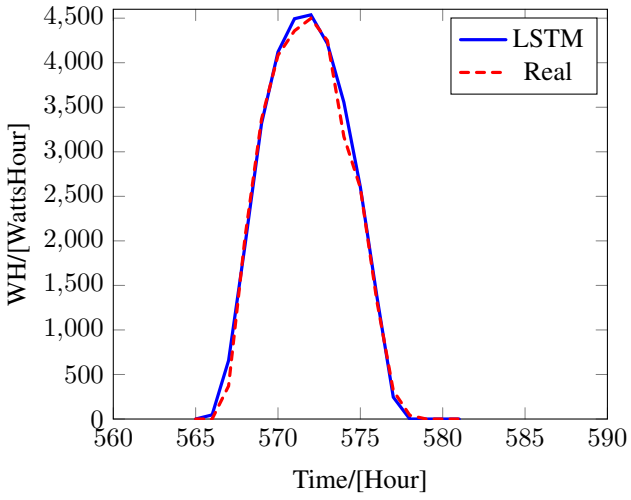

Fig. 5. Real generated power and forecast for a regular day.

The forecast of the generated PV energy (cf. Fig. 5) provides a good approximation of the photovoltaic energy throughout the day, since it closely follows the actual photovoltaic energy generation.

To push the prediction model to its limits, the results obtained on a very rainy day (atypical for the area) were evaluated, obtaining the results shown in Fig. 6.

In Fig. 7, a 10-day period during a rainy and cloudy period was selected to observe the model's behavior. It is worth noting that the model was built to forecast the next day's generated power; however, its 10-day forecast works correctly, even when the days presented do not correspond to the interval that the model has learned.

The three ML models (RF, SVM, and LSTM) were programmed in Python, and the dataset was stored in a CSV file. To compare the results of the LSTM modeling method, it, like the classifier, was also programmed in MatLab©. An excerpt of the data and code will be published in a repository for further research. Additionally, the code for evaluating predicted and actual PV power generation using the discrete integral-based classifier can also be retrieved there [44].

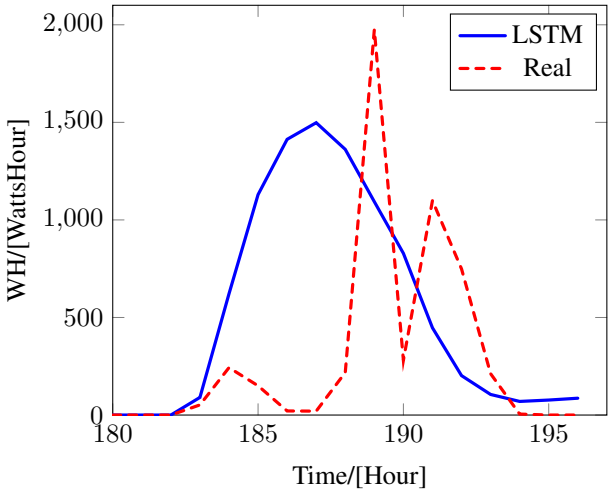

Fig. 6. Real generated power and forecast in a heavy rainy day.

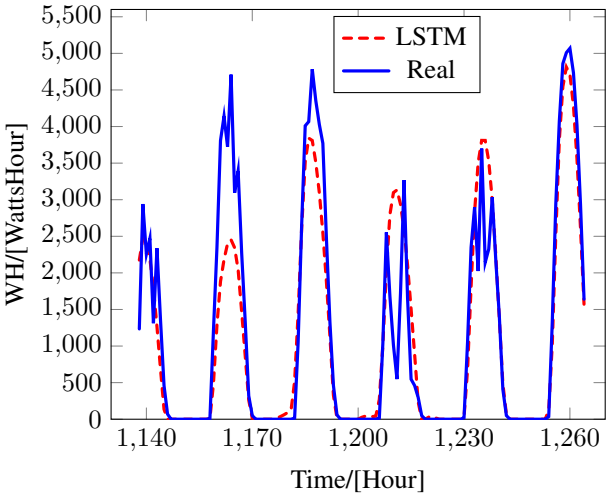

Fig. 7. Excerpt of actual power generated vs. forecast of 10 days.

V. Conclusions and Future Works

When addressing a forecasting problem based on physical, human, and meteorological effects, it is essential to evaluate various modeling techniques and different training data presentations. Therefore, the best modeling technique should be chosen based on its ease of implementation, reliability, and lowest computational cost.

We conducted a series of evaluations to determine, for the specific problem of PV power generation prediction, the three best modeling techniques based on two metrics (MAE and RMSE). The best-performing method (LSTM) was applied to a supervisory system and its performance was evaluated, obtaining good short-term results. Over the long term, its performance declines, especially when the input variables are infrequent (as in this case, with rainy days in a desert environment).

An LSTM modeling method is an excellent option and has demonstrated good performance in predicting photovoltaic energy production, as demonstrated in the former experiments. Its performance has been tested using two metrics.

An opportunity to further develop this work is to add a real-time fault prediction and diagnosis module based on common photovoltaic panel failures.

References

[1] USAID, 2024, photovoltaic (PV) Systems, In Website at https://www.usaid.gov/energy/powering-health/system-components/photovoltaic-systems, accessed in June, 2024.

[2] I. R. E. Agency, "Publications," in *https://www.irena.org/Publications/*, last visit september 2025.

[3] M. van der Hoeven, "Technology roadmap: solar photovoltaic energy." *Int. Energy Agency, France*, jan 2014, https://iea.blob.core.windows.net/assets/3a99654f-ffff-469f-b83c-bf0386ed8537/pv_roadmap.pdf.

[4] A. IEA, IRENA, UNSD, and WHO, 2024, tracking SDG7: The Energy Progress Report 2024, accessed in June, 2024, SDG7-Report2024-0611-V9-highresforweb.pdf at https://www.iea.org/reports/tracking-sdg7-the-energy-progress-report-2024.

[5] G. Eduful, K. Atanga, and D. Pulfrey, "Simulation study of stand-alone and grid-tied photovoltaic systems in ghana," in *Lecture Notes in Engineering and Computer Science: World Congress on Engineering and Computer Science 2021*, 2021, pp. 184–190.

[6] N. Rathore and N. Panwar, "Outline of solar energy in india: advancements, policies, barriers, socio-economic aspects and impacts of covid on solar industries," *Int. J. Ambient Energy*, vol. 1, no. 43, pp. 7630–7642, 2022, doi: https://doi.org/10.1080/01430750.2022.2075925.

[7] M. Talaat, B. E. Sedhom, and A. Hatata., "A new approach for integrating wave energy to the grid by an efficient control system for maximum power based on different optimization techniques," *International Journal of Electrical Power & Energy Systems*, vol. 128, no. 1, 2021, doi: https://doi.org/10.1016/j.ijepes.2021.106800.

[8] L. Hou, H. J. Lin, X. Yang, T. K. Yang, F. Qu, and D. Shao, "Optimal scheduling of integrated energy systems for high-speed railway stations considering integrated demand response under the carbon market," *IAENG International Journal of Applied Mathematics*, vol. 53, no. 2, pp. 497–506, 2023, https://www.iaeng.org/IJAM/issues_v53/issue_2/IJAM_53_2_06.pdf.

[9] R. R. Sharma, M. Kumar, S. Maheshwari, and K. P. Ray, "EVDHM-ARIMA-based time series forecasting model and its application for COVID-19 cases," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 01–10, 2021, doi.org/10.1109/TIM.2020.3041833.

[10] L. R. de Araújo Morais and G. S. da Silva Gomes, "Forecasting daily covid-19 cases in the world with a hybrid arima and neural network model," *IEEE Trans. Instrum. Meas.*, vol. 126, p. 109315, 2022, doi.org/10.1016/j.asoc.2022.109315.

[11] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik, *Support Vector Machines in Advances in Neural Information Processing Systems*, 1st ed. Eds. M.C. Mozer and M. Jordan and T. Petsche, MIT Press, 1996.

[12] K. Davoian and W.-M. Lippe, "Time series prediction with parallel evolutionary artificial neural networks," in *Proc. of Int. Conf. on Data Mining (DMIN) June, Las Vegas, Nevada, USA, Eds. Robert Stahlbock and Sven F. Crone and Stefan Lessmann, CSREA Press*, 2007, pp. 10–15.

[13] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree, vol. 30," in *Proc. of Eds. I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett, Curran Associates, Inc., In Proc. Advances in Neural Inf. Processing Syst.*, 2017, pp. 1–15.

[14] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990, doi.org/10.1016/0364-0213(90)90002-E.

[15] M. Canizo, I. Triguero, A. Conde, and E. Onieva, "Multi-head CNN–RNN for multi-time series anomaly detection: An industrial case study," *Neurocomputing*, vol. 363, no. 1, pp. 246–260, 2019, doi.org/10.1016/j.neucom.2019.07.034.

[16] Q. Ni and X. Cao, "MBGAN: An improved generative adversarial network with multi-head self-attention and bidirectional RNN for time series imputation," *Engineering Applications of Artificial Intelligence*, vol. 115, no. 1, p. 105232, 2022, doi.org//10.1016/j.engappai.2022.105232.

[17] M. Hu, K. Jiang, Z. Nie, and Z. Wang, "You only align once: Bidirectional interaction for spatial-temporal video super-resolution," in *Proc. of 30th ACM Int. Conf. on Multimedia, ACM, New York, USA*, 2022, pp. 847–855, doi.org/10.1145/3503161.3547874.

[18] C. Ma, G. Dai, and J. Zhou, "Short-term traffic flow prediction for urban road sections based on time series analysis and LSTM_BILSTM method," *IEEE Trans. on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 5615–5624, 2022, doi.org/10.1109/TITS.2021.3055258.

[19] K. Bandara, C. Bergmeir, and H. Hewamalage, "LSTM-MSNet: Leveraging forecasts on sets of related time series with multiple seasonal patterns," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 4, pp. 1586–1599, 2021, doi.or/10.1109/tnnls.2020.2985720.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Inf. Processing Syst., 30, Eds. I. Guyon and U. Von Luxburg and S. Bengio and H. Wallach and R. Fergus and S. Vishwanathan and R. Garnett*, 2017.

[21] Y. Xiao, Q. Yuan, J. He, Q. Zhang, J. Sun, X. Su, J. Wu, and L. Zhang, "Space-time super-resolution for satellite video: A joint framework based on multi-scale spatial-temporal transformer," *Int. Jrnl. Applied Earth Observation and Geoinformation*, vol. 108, p. 102731, 2022, doi.org/10.1016/j.jag.2022.102731.

[22] N. Kitaev, Łukasz Kaiser, and A. Levskaya, "Reformer: The efficient transformer, 2020," in *In arXiv.org: 2001.04451*, 2020.

[23] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and X. Yan, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," in *In arXiv.org: 1907.00235*, 2020.

[24] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting, 2021," in *In arXiv.org: 2012.07436*, 2020.

[25] L. Visser, T. AlSkaif, and W. van Sark, "Operational day-ahead solar power forecasting for aggregated PV systems with a varying spatial distribution," *Renewable Energy*, vol. 183, pp. 267–282, 2022, doi.org/10.1016/j.renene.2021.10.102.

[26] X. J, W. Z, Y. Z, D. Y, and G. B., "Prototype learning for medical time series classification via human-machine collaboration," *Sensors*, vol. 24, no. 8, pp. 1–15, 2024, https://doi.org/10.3390/s24082655.

[27] A. G. Cherstvy, D. Vinod, E. Aghion, A. V. Chechkin, and R. Metzler, "Time averaging, ageing and delay analysis of financial time series," *New Journal of Physics*, vol. 19, no. 6, pp. 1–11, 2017, doi:10.1088/1367-2630/aa7199.

[28] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," in *Proc. from 34th Conference on Neural Information Processing Systems, pp. 17804-17815, Eds.: H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan and H. Lin*, 2020.

[29] R. A. de Marcos, A. Bello, and J. Reneses, "Electricity price forecasting in the short term hybridising fundamental and econometric modelling," *Electric Power Systems Research*, vol. 167, pp. 240–251, 2019, https://doi.org/10.1016/j.epsr.2018.10.034.

[30] U. K. Das, K. S. Tey, M. Seyedmahmoudian, S. Mekhilef, M. Y. I. Idris, W. Van Deventer, B. Horan, and A. Stojcevski, "Forecasting of photovoltaic power generation and model optimization: A review," *Renewable and Sustainable Energy Reviews*, vol. 81, no. 1, pp. 912–928, 2018, https://doi.org/10.1016/j.rser.2017.08.017.

[31] L. Silva-Rodriguez, A. Sanjab, E. Fumagalli, A. Virag, and M. Gibescu, "Short term wholesale electricity market designs: A review of identified challenges and promising solutions," *Renewable and Sustainable Energy Reviews*, vol. 160, no. 1, pp. 112–128, 2018.

[32] N. Powers, "Prediction of worldwide energy resources," in *Site accessed in June, 2024, In https://power.larc.nasa.gov*, 2024.

[33] J. Zhang, A. Florita, B.-M. Hodge, S. Lu, H. F. Hamann, V. Banunarayanan, and A. M. Brockway, "A suite of metrics for assessing the performance of solar power forecasting," *Solar Energy*, vol. 111, pp. 157–175, 2015, 10.1016/j.solener.2014.10.016.

[34] F. Antonanzas-Torres, R. Urraca, J. Polo, O. Perpiñán-Lamigueiro, and R. Escobar, "Clear sky solar irradiance models: A review of seventy models," *Renewable and Sustainable Energy Reviews*, vol. 107, pp. 374–387, 2015, doi.org/10.1016/j.rser.2019.02.032.

[35] R. Matignon, *Data Mining Using SAS Enterprise Miner*, 1st ed. Wiley, 2007, https://doi.org/10.1016/j.neucom.2015.12.030.

[36] L. Breiman, "Random forests," *Machine Learning*, pp. 5–32, 2001.

[37] S.-K. Learn, "Random forests regressor," in *Site accessed in June, 2024 In https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html*, 2024.

[38] G. Louppe, "Understanding random forests: From theory to practice," in *https://arxiv.org/abs/1407.7502*, 2015.

[39] V. Vapnik, S. E. Golowich, and A. Smola, "Support vector method for function approximation, regression, estimation and signal processing," *NIPS'96: Proceedings of the 9th International Conference on Neural Information Processing Systems*, pp. 281–287, 1996.

[40] I. Solutions, "What are support vector machines," in *Site accessed in June, 2024, In https://https://www.ibm.com/topics/support-vector-machine*, 2015.

[41] A. Wahba, R. El-khoribi, and S. Taie, "A new hybrid model for energy consumption prediction based on grey wolf optimization," *IAENG International Journal of Computer Science*, vol. 49, no. 2, pp. 469–481, 2022, https://www.iaeng.org/IJCS/issues_v49/issue_2/IJCS_49_2_21.pdf.

[42] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing*, vol. 408, pp. 189–215, 2020, https://doi.org/10.1016/j.neucom.2019.10.118.

[43] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997, 10.1162/neco.1997.9.8.1735.

[44] P. V. System, "Supervisory PV array system," in *https://drive.google.com/drive/folders/1Dxgzm5dReh0hSHjVLrPCCMwiKegJNQBX?usp=sharing*, 2025.