energies

MDPI

Article

Forecasting Solar Energy Generation and Household Energy Usage for Efficient Utilisation

Aistis Raudys * and Julius Gaidukevičius

Institute of Computer Science, Vilnius University, Didlaukio g. 47, LT-08303 Vilnius, Lithuania; juliuss.ga@gmail.com

* Correspondence: aistis.raudys@mif.vu.lt

Abstract: In this study, a prototype was developed for the effective utilisation of a domestic solar power plant. The basic idea is to switch on certain electrical appliances when the surplus of generated energy is predicted one hour in advance, for example, switching on a pump motor for watering a garden. This prediction is important because some devices (motors) wear out if they are switched on and off too frequently. If a solar power plant generates more energy than a household can consume, the surplus energy is fed into the main grid for storage. If a household has an energy shortage, the same energy is bought back at a higher price. In this study, data were collected from solar inverters, historical weather APIs and smart energy meters. This study describes the data preparation process and feature engineering that will later be used to create forecasting models. This study consists of two forecasting models: solar energy generation and household electricity consumption. Both types of model were tested using Facebook Prophet and different neural network architectures: feedforward, long short-term memory (LSTM) and gated recurrent unit (GRU) networks. In addition, a baseline model was developed to compare the prediction accuracy.

Keywords: neural networks; feedforward; long short-term memory;recurrent neural networks; LSTM; GRU; photovoltaic; solar

Citation: Raudys, A.; Gaidukevičius, J. Forecasting Solar Energy Generation and Household Energy Usage for Efficient Utilisation. *Energies* **2024**, *17*, 1256. https://doi.org/10.3390/en17051256

Academic Editor: Alon Kuperman

Received: 22 January 2024
Revised: 23 February 2024
Accepted: 28 February 2024
Published: 6 March 2024

1. Introduction

The proportion of electricity generated from renewable energy sources is increasing in most parts of the world. This reduces dependence on oil and gas and creates a cleaner and healthier environment. Renewable energy is generally more resilient to price changes and can often be a very good investment. One of the most popular renewable energy sources is solar energy, which is rapidly gaining popularity due to its relatively cheap installation and low maintenance costs. It was once considered unaffordable in developing countries, but today, with the European Union funding some of the investment and the price of solar panels falling, many people are looking forward to meeting their energy needs by installing small solar power plants [1].

However, households that decide to install such a system cannot rely on it completely, as solar energy is not constant, while it is possible to install different battery storage systems for solar energy, they have a dramatic impact on the price. In addition, the batteries used to store energy are harmful to the environment [2], and their capacity decreases over time [3]. Normally, the energy generated by a solar power plant that is not used for household consumption is fed into the general electricity grid. In this way, surplus energy is bought by the electricity supplier at a certain price and used to meet the energy needs of other households. The problem is that the utility buys the energy at a lower price; however, if there is a shortage of electricity in the same household, the energy is bought back at a higher price, resulting in some loss of generated energy.

The present work attempts to make efficient use of the surplus electricity in the local grid by switching on certain appliances that are not necessary at the moment but could be

useful to the consumer, such as a heat pump, a ventilation or air conditioning system, a charging unit for an electric car, a sprinkler for watering the garden, etc. The aim of this work is to develop a prototype for an efficient solar energy system that can redirect the surplus energy to a specific electrical device.

2. Collected Data

2.1. Energy Generated by the Solar Power Plant

Typical small solar power plants usually consist of two elements—solar modules and a solar inverter. The energy generated by the solar modules is converted into direct current (DC). Before the household to which a solar power system is connected can utilise this generated electricity or feed it into the external power grid, the direct current must be converted into alternating current. A solar inverter is used for this purpose. Modern solar inverters usually have the ability to monitor the electrical current or short-term parameters of a solar power plant. Unfortunately, these inverter features are insufficient for long-term monitoring and data acquisition, as different inverters store data in different time periods, formats, etc.

Most of these devices also support the Modbus serial communication protocol, which can be used to monitor and collect a variety of solar inverter parameters. Several "master"/"slave" devices are normally required to activate this protocol. The "master" device in this model normally sends a request to the "slave" device, which sends back the result of this request. The request usually consists of a function code that specifies the action to be performed (e.g., read or write), a specification of the action to be performed and the address of the parameter to be read or changed. The "slave" device that processes this request usually returns the same function code and the result of the request if the action was performed successfully. In the event of an error, the named device returns the function code together with an error and action error code [4].

In order to collect periodic data from solar inverters, it was decided to connect a microcontroller to each of them, which can communicate with the inverter via the Modbus protocol described above. In order to make the system as versatile and widely applicable as possible, it was decided to develop a microcontroller firmware that can be customised to any solar inverter that can communicate via the Modbus protocol. The firmware created was used to collect the generation data from several different solar inverters used in the present study. The solar inverters used in this work were connected to two power plants, one of which can generate a maximum power of 5.5 kilowatts and the other a power of 30 kilowatts. In the following sections, these solar power plants are referred to as "Solar 5K" and "Solar 30K", respectively.

2.2. Obtaining Energy Consumption Data

In order to forecast energy consumption, data on the electricity consumption of two households were obtained. The consumption data are regularly uploaded to the electricity supplier's database, where it can be easily downloaded. The data collected indicate the hourly electricity consumption of a particular household in kilowatt hours. The data were collected for two households, which are referred to as "Household 1" and "Household 2" in the following sections of this study.

2.3. Obtaining Historical Weather Data

The models created in this work also use historical weather data to predict the amounts of energy both produced and consumed. The amount of energy produced by a properly installed solar power plant is highly dependent on weather conditions, especially solar radiation and cloud cover, but the current air temperature and the strength of the wind can also affect efficiency, as the batteries in the solar power plant can become very hot; however, with the right wind direction and speed, the loss of efficiency can be avoided due to the cooled surface of the solar cells [5].

Although to a lesser extent than the amount of energy generated, household consumption is also dependent on weather conditions, with lights being switched on when cloud density is higher and heaters or air conditioners being switched on when temperatures are cooler or warmer. For this reason, some weather data were also used to predict energy consumption.

To obtain historical weather data, the Solcast Historical and TMY API (https://solcast.com/historical-and-tmy (accessed on 25 March 2022)) was used to access weather data for a specific time period and location. Data on weather conditions were collected for each hour, which are described in more detail below.

2.4. Data Collected

After collecting various types of data defined in the current section, the following data sets were collected:

- "Solar 5K": The amount of energy generated by a solar power plant at a given hour. The period of data collection was from 15 March 2020 18:00 to 28 March 2022 01:00.
- "Solar 30K": The amount of energy generated by a solar power plant at a given hour. The data collection period was from 27 February 2021 15:00 to 28 March 2022 01:00.
- "Household 1": The amount of energy consumed by a household at a given hour. The period of data collection was from 8 October 2020 14:00 to 28 March 2022 01:00.
- "Household 2": The amount of energy consumed by a household at a given hour. The period of data collection was from 29 June 2020 13:00 to 28 March 2022 01:00.
- Historical weather data: The period of data collection was from 31 December 2018 03:00 to 28 March 2022 01:00.

3. Data Preparation

The data must be well prepared so that they can be successfully used to train and test the models to be developed. This section of the study describes methods of data preparation and feature engineering that were later used to train and test the models created.

3.1. Creation of New Features

3.1.1. Calculation of Times for Sunset, Midday Sun and Sunrise

Using coordinates for a specific location (in this case the location of a solar power plant), it is possible to calculate timestamps for a specific date at which sunrise, noon and sunset occur. For each date in the data set, the values for sunrise, noon and sunset were calculated using the algorithm developed by the United States National Renewable Energy Laboratory [6]. Although the calculated values were not used directly in a model, they were used to derive other features that are described in the following sections of this study.

3.1.2. The Target Value of the Last Hour and the Difference between the Last Two Target Values

In order to develop two types of models that can predict electricity generation and consumption at a given hour, the values of the previous hour and the difference between the previous two hours were appropriately included in the feature set. The extraction of the new features is described below. Suppose we have a data set with m rows of data, each of which contains $n+1$ features. The feature with index $n+1$ in this data set is the target value, i.e., the feature to be predicted by the created model. We can define the data set using the following matrix X:

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} & x_{1(n+1)} \\ x_{21} & x_{22} & \cdots & x_{2n} & x_{2(n+1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} & x_{m(n+1)} \end{bmatrix} \quad (1)$$

A new data set X' with two additional features was obtained from this data set X—the target values of the previous hour and the difference between the target values of the two previous hours with the indices α and β, respectively. The characteristics α and β of this data set are calculated using the following expressions:

$$x'_{i\alpha} = x_{(i-1)(n+1)} \tag{2}$$

$$x'_{i\beta} = x_{(i-1)(n+1)} - x_{(i-2)(n+1)} \tag{3}$$

3.1.3. Solar Activity

As already mentioned, the calculated times of sunset, midday and sunrise are not suitable for use in the models, as they have no specific influence on the amount of energy generated or consumed at a given hour. However, it is clear that energy production is only possible when the sun is visible in the sky, and a household's energy consumption also depends on this. For this reason, it was decided to introduce a new feature derived from these timestamps. This property is intended to describe the part of an hour when the sun is visible in the sky. The newly created property can take values from 0 to 1, which means that the sun is not visible in the sky during the entire hour or that the sun is visible in the sky during the entire hour. These are the definitions of the following timestamps as vectors with three components that correspond to the hours, minutes and seconds of the timestamp:
Current timestamp $d = (d_h, d_m, d_s)$;
Timestamp of a sunrise $s = (s_h, s_m, s_s)$;
Timestamp of a sunset $p = (p_h, p_m, p_s)$.

The feature created can be expressed with the following function $f(d, s, p)$:

$$f(d,s,p) = \begin{cases} 1, & s_h < d_h < p_h \\ \dfrac{60 - \left(\dfrac{s_s}{60} + s_m\right)}{60}, & s_h = d_h \\ \dfrac{\dfrac{p_s}{60} + p_m}{60}, & p_h = d_h \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

3.1.4. Distance to Midday Sun

In view of the fact that electricity generation is likely to be highest at midday, it was decided to create a property whose value increases as the hour approaches midday and whose value decreases as the hour approaches sunrise. In simple terms, the purpose of this property is to define the distance to noon between sunset and sunrise, thus combining the three resulting timestamps. Further, by defining the noon timestamp as a vector $v = (v_h, v_m, v_s)$ (the same as the timestamp for the solar activity feature) and defining the function $t(x)$ that expresses the given timestamp vector as an integer, the feature created can be expressed by the following function $g(d, s, p, v)$:

$$g(d,s,p,v) = \begin{cases} \dfrac{t(d) - t(s)}{t(v) - t(s)}, & t(s) \le t(d) \le t(v) \\ 1 - \dfrac{t(v) - t(d)}{t(v) - t(p)}, & t(v) \le t(d) < t(p) \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where

$$t(x) = 60^2 x_h + 60 x_m + x_s \tag{6}$$

Current timestamp $d = (d_h, d_m, d_s)$;
Timestamp of a sunrise $s = (s_h, s_m, s_s)$;

Timestamp of a sunset $p = (p_h, p_m, p_s)$;
Timestamp of a solar noon $v = (v_h, v_m, v_s)$.

3.1.5. Distance to Summer Solstice

As already mentioned, it makes no sense to use the date and timestamps in the models, as the models cannot take them into account correctly. In order to be able to use not only the time of day but also the date in the models, a new feature, Distance to summer solstice, was created using the date. The days of the summer and winter solstices were used to create this feature. These days can be considered as events in the year when the sun can be seen the longest in the sky and the highest solar radiation can be expected (summer solstice) or when the sun can be seen the least in the sky and the lowest solar radiation can be expected (winter solstice) [7]. The solstices occur in December and June, depending on the year on 20 to 22 December; however, for the sake of simplicity, these days are referred to as 22 December and 22 June in the models created. The graphs in the Figure 1 show the average electricity generation per hour for a given month and the approximate day of the summer solstice, which is marked by a red line. The graphs show that, on average, the highest amount of energy is generated in the sixth and seventh months, when the date is closest to the summer solstice.

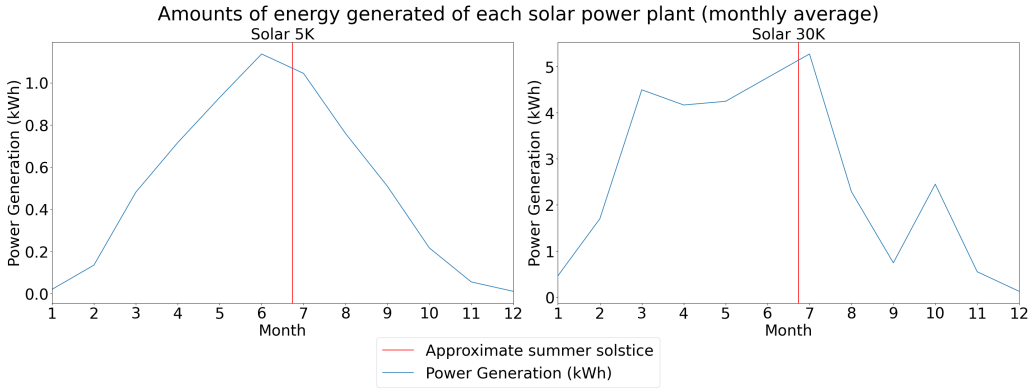

Figure 1. Average monthly energy generated by solar power plants.

Therefore, a new feature was used in the models to describe the distance to the summer solstice instead of the date. This feature was implemented as follows: The closer the date came to the summer solstice, the closer the feature was moved to 1, while the feature was moved to 0 the closer the date came to the winter solstice. By defining the function $d(x)$, which expresses the date x as an integer representing the number of days that have elapsed since 1 January 1970, the generated feature can be expressed in terms of the following function:

$$h(x, y_d, y_b, z_p, z_d) = \begin{Bmatrix} \frac{1-(d(y_d)-d(x))}{d(y_d)-d(z_p)}, & d(z_p) < d(x) < d(y_d) \\ \frac{1-(d(x)-d(y_d))}{d(z_d)-d(y_d)}, & d(y_d) < d(x) < d(z_d) \\ \frac{1-(d(y_b)-d(x))}{d(y_b)-d(z_d)}, & d(z_d) < d(x) < d(y_b) \\ 1, & x = y_d \\ 0, & \text{otherwise} \end{Bmatrix} \quad (7)$$

where
x is the current date.
y_d and y_b are dates of this year's and next year's summer solstice, respectively.
z_p and z_d are dates of last year's and this year's winter solstice, respectively.

3.2. Feature Engineering

Though some features were not encoded using additional data, they were prepared so that their values could be used efficiently in the models created. This subsection presents the ways in which certain features were prepared for their use in the models.

3.2.1. The Sun's Position in the Sky

The hourly solar azimuth and zenith angles describing the position of the sun in the sky were obtained from the above-described Solcast API. In simple terms, the azimuth angle describes the direction from which the sun is currently shining on the location, while the zenith angle describes the altitude at which the sun is currently located [8]. The below-given graphs illustrate how the amount of energy generated by "Solar 5K" (y axes) depends on these angles (x axes).

As can be seen from the graphs in the Figure 2, the amount of energy generated increases as the azimuth angle moves away from 0° and the zenith angle approaches 0°. Both the zenith and azimuth angles were coded keeping only their absolute value. This encoding can be expressed in terms of a simple modulus function $q(x) = |x|$. The graph in the Figure 3 illustrates the dependence of the amount of energy generated on the azimuth angle feature after applying the modulus operation to it. Please note that zenith angle never goes below zero in Lithuania, so the dependence graph would be the same.

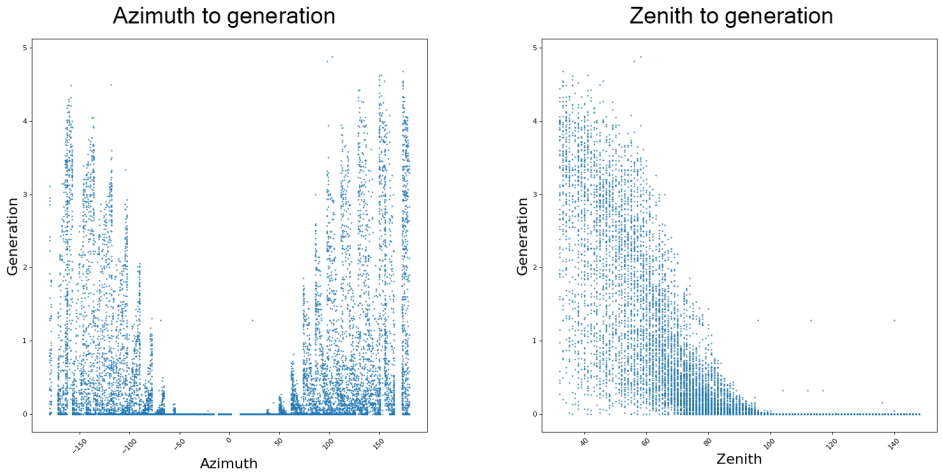

Figure 2. The correlation between the energy generated on a specific hour and the azimuth angle (**left**) and zenith angle (**right**) using the Solar 5K data set.

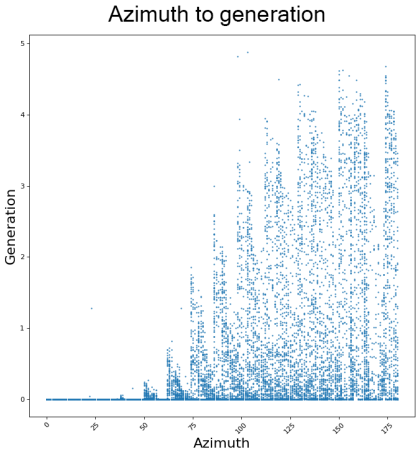

Figure 3. The correlation between the energy generated at a specific hour and the azimuth angle, with module operation applied.

3.2.2. Wind Speed and Direction

The study by Ahmad Vasel and Frantzis Iakovidis [9] found that the amount of energy generated by a solar power plant also depended on the wind direction. The study monitored solar power plants and identified 42 pairs of cases where solar radiation, temperature and wind speed were the same, and only the wind direction differed—in one case the wind blew from the south, in the other case from the north. The hypothesis that the wind from the south should result in a higher amount of energy being generated by a solar plant as compared to other wind directions was true for all 42 pairs of cases. Considering that the wind direction is given in degrees ranging from 0° to 360° in the data, the wind direction is given as 180° when the wind blows from the south. In order to use this value efficiently in the models, it was decided to encode the wind direction in terms of the absolute difference between the southerly wind direction and the present value, multiplied by the wind speed. It was decided to include the wind speed in this encoding, taking into account its ability to cool the PV modules [10]. Thus, the encoded property of the wind speed and its direction can be expressed as the following function:

$$w(x, y) = |180 - x| \cdot y \tag{8}$$

where
x—a wind direction.
y—a wind speed.

3.2.3. Time-of-Day Encoding

Household electricity consumption is much more difficult to predict than the amount of energy produced by a solar power plant, as the amount of energy produced is directly dependent on the weather conditions, while electricity consumption can be highly erratic, human-dependent and totally independent of external factors. Although, on average, energy consumption of a typical household is fairly regular, with occasional abnormalities, it is quite difficult to find factors it depends on. As mentioned above, since the time and date were not used in the models, they had to be replaced with appropriate features. The analysis of the collected energy consumption data revealed certain trends that are especially apparent on weekdays and at weekends. These trends are reflected in the graphs of the Figure 4. The red lines in the graphs show the division of the day into different parts during which similar energy consumption is expected.

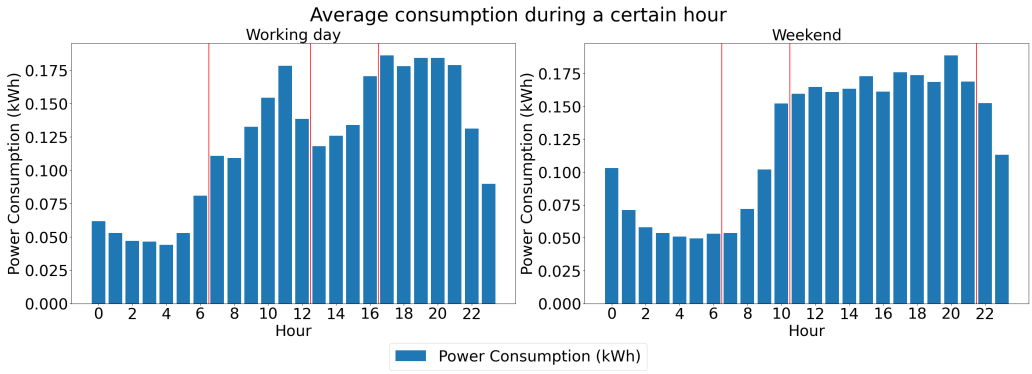

Figure 4. Average energy consumption per hour on weekdays (**left**) and at weekends (**right**).

After splitting the working days and weekends into the parts visible in the graphs, it was decided to introduce some new features to indicate which part of the resulting day the current hour falls into. In total, eight new features were obtained, which can only take values of 0 or 1. This type of encoding is usually called one-hot encoding. Each of these properties checks two conditions: whether the current day falls within the range of days belonging to the feature, and whether the current hour falls within the range of hours

belonging to the feature. If both conditions are satisfied, the feature takes a value of 1, otherwise it takes a value of 0. For the purposes of this encoding, the days of the week were assumed to be defined by numbers 1 to 7, while the hours were assumed to be defined by numbers 0 to 23. These characteristics and their ranges are defined in the Table 1.

Table 1. Feature ranges.

Feature	Range of Days	Range of Hours
Working day 1/4	[1; 5]	[0; 6]
Working day 2/4		[7; 12]
Working day 3/4		[13; 16]
Working day 4/4		[17; 23]
Weekend 1/4	[6; 7]	[0; 6]
Weekend 2/4		[7; 10]
Weekend 3/4		[11; 21]
Weekend 4/4		[22; 23]

3.2.4. Time-of-Year Encoding

As seasons change, so do electricity consumption patterns. This was confirmed by the analysis of the consumption data collected. The graphs presented in the Figure 5 show average one-hour electricity consumption by both households at a given time of the year.

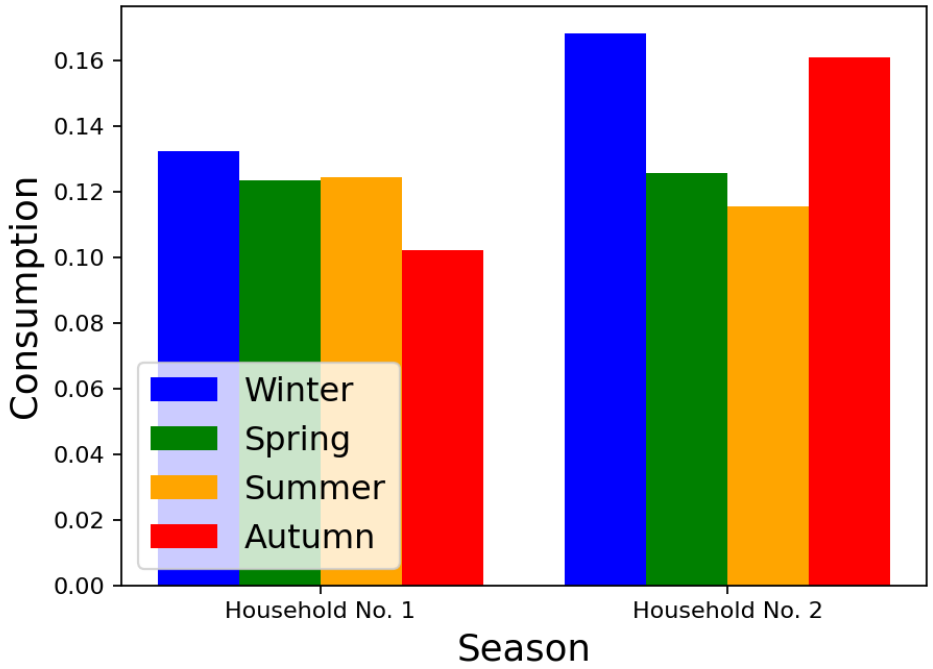

Figure 5. Average electricity consumption at a given time of the year. Consumption of the first household is shown on the left and that of the second household is shown on the right.

Although differences in seasons are not too noticeable for "Household 1", the average hourly consumption varied quite strongly for "Household 2". For this reason, as in the case of the time-of-day encoding, it was decided to introduce four additional features representing the current time of the year, which can take values of 0 or 1.

3.2.5. Solar Radiation Feature Extraction

Having collected the data, several features related to solar radiation were obtained. As all of them describe the intensity of solar radiation from different angles, it makes no

sense to use them all in the models being created. The features were selected by calculating the Pearson's correlation coefficient between the feature and the amount of electricity generated by "Solar 5K". The Pearson's correlation coefficient indicates the strength of the linear relationship between two variables and can take values between -1 (as one variable decreases, the other increases and vice versa) and 1 (as one variable either increases or decreases, the other increases or decreases accordingly). The closer the absolute value of this coefficient is to 1, the stronger the relationship between these variables is. Solar radiation features and their definitions are given below:

- Diffuse horizontal irradiance (DHI)—The horizontal component of diffuse irradiance (irradiance that is scattered by the atmosphere).
- Direct normal irradiation (DNI)—Solar irradiance arriving in a direct line from the sun as measured on a surface held perpendicular to the sun.
- Direct (beam) horizontal irradiance (EBH)—The horizontal component of direct normal irradiance.
- Global horizontal irradiance (GHI)—The total irradiance received on a horizontal surface. It is the sum of the horizontal components of direct (beam) and diffuse irradiance.
- Global tilted irradiance (GTI)—The total irradiance received on a surface with defined tilt and azimuth, fixed or tracking [11].

After calculating the correlation coefficient for all six features describing the solar radiation, the results given in the Table 2 were obtained.

Table 2. Correlation of a feature with the target value (amount of energy generated).

Marking	Feature	Pearson's Correlation Coefficient
A	DHI	0.557345
B	DNI	0.559265
C	EBH	0.730474
D	GHI	0.895749
E	GTI (fixed tilt)	0.800130
F	GTI (tracking)	0.781131

As can be seen from the above table, the strongest correlations were obtained with horizontal solar radiation and total solar radiation at a fixed position. This is to be expected as the solar power plants studied are not automatically rotated in the direction of the sun. The graphs in the Figure 6 show the dependence of the amount of energy generated (x axes) on a particular solar radiation feature (y axes), which also shows these relationships. The letters next to the graphs correspond to the feature designation in the preceding table.

3.3. Data Normalisation

After creating the data sets using the preparation methods outlined in this section, we applied the MinMax scaler to normalise all features within the data to the interval $[0, 1]$. The MinMax data scaling expression is presented below:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{9}$$

where
x is the value of the feature to be scaled.
x_{min} and x_{max} are the minimum and maximum values of the feature, respectively [12].

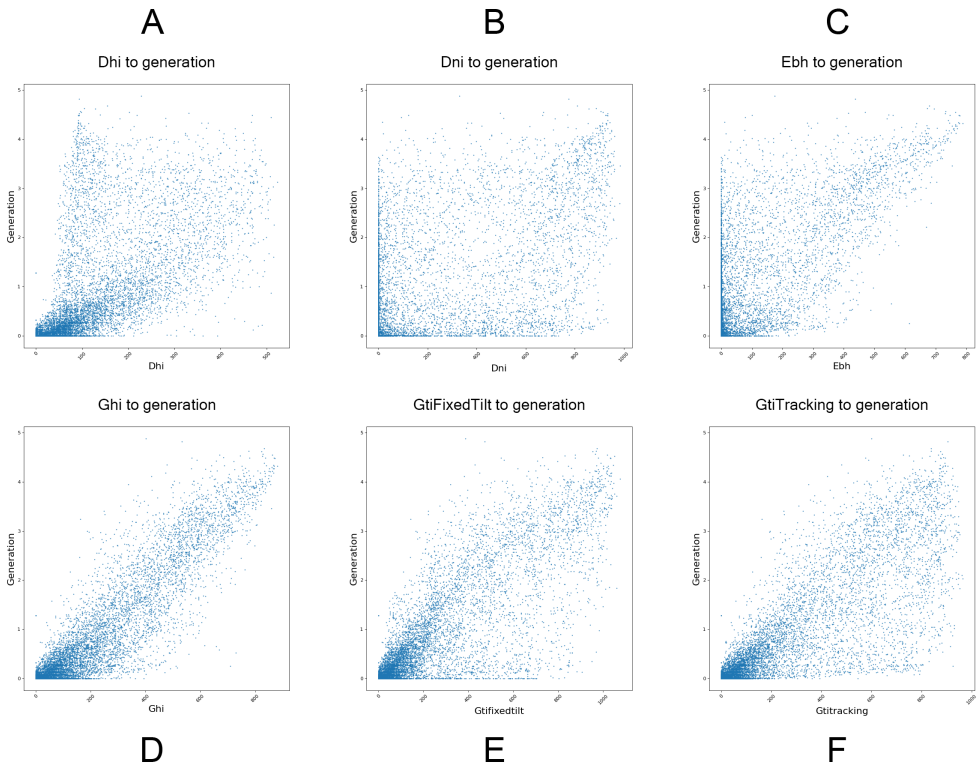

Figure 6. The correlation between the energy generated at a specific hour and different solar radiation characteristics. (**A**) DHI; (**B**) DNI; (**C**) EBH; (**D**) GHI; (**E**) GTI (fixed tilt); (**F**) GTI (tracking)

4. Created Prediction Models

4.1. Baseline Generation Model

To compare the models created during this study, a baseline model was created using only historical monthly data along with the distance to the above-defined solar noon feature. Historical data used to create this model were obtained from [13] and contained generation data for every month between the years 2014 and 2019. After collecting the data, average generation for every month was calculated. To make the baseline model more accurate, average daily generation was calculated for each day of the year. The calculation of the daily averages can be expressed using the following mathematical function $g_d(x, x_t, x_p, x_f, y_t, y_p, y_f)$:

$$g_d(...) = \begin{cases} \dfrac{y_t(d(x) - d(x_p)) + y_p(d(x_t) - d(x))}{d(x_t) - d(x_p)}, & d(x) < d(x_t) \\ \dfrac{y_f(d(x) - d(x_t)) + y_t(d(x_f) - d(x))}{d(x_f) - d(x_t)}, & d(x) > d(x_t) \\ y_t, & \text{otherwise} \end{cases} \quad (10)$$

where
$d(x)$ is a function that expresses the date x as an integer representing the number of days that have elapsed since 1 January 1970.
x_t, x_p and x_f are middle days of the current, previous and future months, respectively.
y_i, y_p and y_f are average generations of the current, previous and future months, respectively.
x is the current date.

Having average generation for every day, generation per hour on that day can be calculated using the distance to the solar noon feature. It can be expressed using the following function:

$$g_h(g_d, x, x_{sum}, r) = g_d \times r \times \frac{x}{x_{sum}} \tag{11}$$

where
g_d is daily generation.
x is the distance to the solar noon of the current hour.
x_{sum} is the sum of the distance to the solar noon feature of this day.
r is the generation power ratio. It is calculated by dividing the maximum output power by the maximum power (5.5 kW for the "Solar 5K" and 30 kW for the "Solar 30K") of the original power plant (10.7 kW).

4.2. Prophet

Prophet is open-source software released by Facebook's Core Data Science team. It uses a decomposable time series model with trend, seasonality and holidays being its main components, which are combined into the following equation:

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t \tag{12}$$

where

- $g(t)$ is the trend function that models nonperiodic changes in the value of the time series.
- $s(t)$ is a representation of periodic changes (daily, weekly and yearly seasonality).
- $h(t)$ represents the effects of holidays that occur on potentially irregular schedules over one or more days.
- ϵ_t is an error term, which represents any idiosyncratic (individual) changes that are not accommodated by the model.

The Prophet package has two different trend models: a saturating growth model and a piecewise linear model. A nonlinear saturating growth model is used for growth forecasting which has similarities with population growth in natural ecosystems where there is nonlinear growth that saturates at the carrying capacity. This growth is modelled using the logistic growth model:

$$g(t) = \frac{C}{1 + exp(-k(t - m))} \tag{13}$$

where

- C is the carrying capacity. Note that C is not a constant; it actually depends on time $C(t)$.
- k is the growth rate.
- m is an offset parameter.

Since the growth rate k is adjusted, the offset parameter must also be adjusted to connect the endpoints of segments [14]. Then $g(t)$ can be expressed this way:

$$g(t) = \frac{C(t)}{1 + exp\Big(- \big(k + a(t)^T\delta\big)\Big(t - \big(m + a(t)^T\gamma\big)\Big)\Big)} \tag{14}$$

where

- δ and γ are the adjustments of vector rates, which define the change of rate which occurs at the time s_j.
- $a(t) = \left\{ \begin{array}{ll} 1, & \text{if } t \geq s_j \\ 0, & \text{otherwise} \end{array} \right\}$

The second linear model is used for forecasting problems. It can be expressed as

$$g(t) = \big(k + a(t)^T\delta\big)t + \big(m + a(t)^T\gamma\big) \tag{15}$$

where γ_j is set to $s_j\delta_j$ [14].

The equation of seasonal effects can be represented as

$$s(t) = \sum_{n=1}^{N} \left(a_n cos\left(\frac{2\pi nt}{P}\right) + b_n sin\left(\frac{2\pi nt}{P}\right) \right) \quad (16)$$

where P is a regular period we expect a time series to have (for example, for a yearly period, $P = 365.25$; for a weekly period, $P = 7$) and $\beta = [a_1, b_1, ..., a_N, b_n]^T$ are the parameters of seasonality fit.

4.3. Neural Networks

In order to examine different architectures used in the present study, neural network models were tested using different sets of parameters. Experiments were carried out with different network architectures, LSTM and GRU containing two to four layers (excluding the output) and a feedforward network containing four to eight layers (excluding the output), to compensate for the lower number of parameters.

The reasoning for selecting these three distinct architectures was to establish one simple network as an additional baseline model, or in simpler terms, a baseline model of neural networks. It is worth noting that feedforward neural networks function in a unidirectional manner, with the outputs of each layer being passed on to the subsequent layer, thereby lacking the ability to retain past information. In an RNN, the decision made at time $t-1$ influences the decision at time t. Consequently, the network's response to new data hinges on two factors: the current input and the output from the recent past [15]. Both LSTM and GRU architectures demonstrate equal proficiency in managing long-term dependencies. They have been extensively evaluated and compared in various machine learning tasks, consistently demonstrating comparable efficiency, as stated in [16,17]. Keeping that in mind, it was expected that these network architectures would be particularly useful in forecasting the amount of energy produced by a solar power plant and the amount of energy consumed by a household due to their ability to capture temporal dependencies and patterns in sequential data, which are essential characteristics for time-series forecasting tasks.

Each layer, except for the first and the last ones, had exactly 64 cells of corresponding architecture. In the feedforward network, all layers (except for the output) used the Rectified linear unit (ReLU) activation function. In all these networks, an additional layer containing one neuron was added to the output of the network where the LeakyReLU activation function was used, with $\alpha = 0.2$ to avoid the Dying ReLU function in [18]. Furthermore, each set of parameters was tested on all four data sets, "Solar 5K" and "Solar 30K" for forecasting electricity generation and "Household 1" and "Household 2" for forecasting electricity consumption, in order to see if the data sets developed are suitable for different objects and how the accuracy of the models changes with a larger amount ("Solar 5K") or smaller amount ("Solar 30K") of data. It is also worth mentioning that with LSTM and GRU architectures, in the case of generated energy, the number of rows in the data set was slightly reduced depending on the sequence length l due to noncontiguous data—the data were prepared so that the model should use, namely, the previous $l-1$ h of the sequence rather than the previous $l-1$ rows of the sequence. The above experiments were carried out using the following variable parameters:

- Optimisation algorithms: Adam and RMSProp.
- Loss functions: mean absolute error (MAE), mean squared error (MSE) and mean absolute percentage error (MAPE).
- Learning rate: 0.001 and 0.0001.
- Sequence length (only applies to GRU and LSTM networks): 12, 24 or 36.
- Number of layers: 2 or 4 (4 and 8 for feedforward networks).
- Number of epochs: 50, 100, 200 or 400.

The models created were trained and tested using data that were not used in the training process. In order to test the performance of the models created using as large an amount of data as possible, the collected data sets were divided into the periods of 3 days exactly (72 h) from each season. It is noteworthy that these periods were specifically chosen so that each object under study, in terms of both energy production and consumption, should have 72 h in its own data set, without any missing hour points. The testing periods are listed in the Table 3.

Table 3. Testing periods.

	Winter	Spring	Summer	Autumn
Start	26 February 2022	20 March 2021	10 July 2021	28 October 2021
End	28 February 2022	21 March 2021	12 July 2021	30 October 2021

4.4. Acquired Results

After training 2688 models with different sets of parameters, models were tested using the described testing data set and compared by calculating the coefficient of determination (R^2) between the actual values and the ones predicted by the models created.

This coefficient can take values ranging from $-\infty$ (in the worst case) to 1 (when the predicted values are exactly the same as the actual ones) [19].

In the following parts of this subsection, the results are summarised by examining the models obtained using specific parameters, taking into account the average and maximum coefficient of determination obtained for the parameter in question, for a given object. The performance the acquired models and the disparities between the predicted and actual values are presented in the graphs of Appendices A–D of this article.

Taking into account optimisation algorithms and the learning rate, two algorithms tested gave similar results for all the objects tested. As can be seen from the results, algorithms with a higher learning rate often performed better compared to those with a lower learning rate. However, there were some cases, in particular with the RMSProp algorithm for "Household 2", for which a closer look at the resulting models revealed that the average coefficient of determination was heavily skewed by a few particularly bad results obtained using the feedforward network. It is likely that if these few models were discarded, the difference would not be so large. The comparison of optimization algorithms and learning rates is given in the Table 4.

When comparing the results by a loss function, the MAE and MSE functions gave quite similar results, while the MAPE loss function performed on average much worse compared to other two. It is also noteworthy that the few above-mentioned very poor models trained with the help of "Household 2" data and the RMSProp algorithm were also obtained using the MAPE as a loss function, as the loss function comparison table shows. The comparison of loss functions is given in the Table 5.

Moving on to the comparison of artificial neural network architectures, the results are quite different. Although both LSTM-type and GRU-type models performed very similarly, making use of different sequence lengths in the case of the generated energy prediction, in the case of the consumption prediction, "Household 1" models were more accurate when employing a smaller sequence length, while "Household 2" ones were more accurate using a larger one. This illustrates clearly that in principle, solar power plants perform similarly; however, different households have different energy consumption trends.

Very interesting results were obtained for all three different architectures. The results showed that the number of layers was not so important in the problems, as the results were quite similar except for the aforementioned "Household 2". However, in comparing the obtained maximum R^2 metrics with the average ones, it can be seen that in the average case, the feedforward neural networks performed even better than LSTM or GRU did; however, taking into account the best models, the advantages of the recurrent networks and the importance of selecting appropriate parameters become apparent, since the results

obtained with these networks exceeded the results of the feedforward neural networks in almost all cases. Furthermore, considering the number of epochs, models with a higher number often performed more accurately; however, there were also cases where it was possible to see signs of overfitting with a decrease in accuracy as the number of epochs increased. The comparison of architectures and sequence lengths is given in the Table 6.

Summing up the results, the differences between different households are particularly striking. The prepared data sets were more suitable for forecasting "Household 1" due to a potentially more constant and predictable consumption. Furthermore, even though data sets for the prediction of energy generated differed in size, no drastic differences were observed in the models created, although on average, the models created for a solar power plant with a bigger data set performed slightly better than those with a smaller data set.

Table 4. Comparison of optimisation algorithms and learning rates.

Object	Optimisation Algorithm	Learning Rate	Max R^2	Average R^2
"Solar 30K"	Adam	0.0001	0.964	0.525
		0.001	0.971	0.517
	RMSProp	0.0001	0.969	0.52
		0.001	0.972	0.538
"Solar 5K"	Adam	0.0001	0.975	0.575
		0.001	0.981	0.583
	RMSProp	0.0001	0.977	0.587
		0.001	0.98	0.574
"Household 1"	Adam	0.0001	0.501	−0.204
		0.001	0.52	−0.227
	RMSProp	0.0001	0.507	−0.185
		0.001	0.523	−0.217
"Household 2"	Adam	0.0001	0.273	−0.048
		0.001	0.315	−0.355
	RMSProp	0.0001	0.273	−0.026
		0.001	0.25	−13.822

Table 5. Loss functions comparison.

Object	Loss Function	Maximum R^2	Average R^2
"Solar 30K"	MAE	0.972	0.941
	MSE	0.969	0.938
	MAPE	0.911	−0.305
"Solar 5K"	MAE	0.981	0.958
	MSE	0.977	0.953
	MAPE	0.903	−0.173
"Household 1"	MAE	0.523	0.38
	MSE	0.487	0.293
	MAPE	0.068	−1.298
"Household 2"	MAE	0.247	0.092
	MSE	0.315	0.097
	MAPE	−0.037	−10.878

Table 6. Architecture and sequence length comparison.

Object	Architecture	Sequence Length	Maximum R^2	Average R^2
"Solar 30K"	LSTM	12	0.965	0.477
		24	0.956	0.467
		36	0.964	0.467
	GRU	12	0.971	0.487
		24	0.97	0.49
		36	0.972	0.489
"Solar 5K"	LSTM	12	0.976	0.549
		24	0.974	0.547
		36	0.977	0.547
	GRU	12	0.979	0.55
		24	0.979	0.551
		36	0.981	0.552
"Household 1"	LSTM	12	0.52	−0.207
		24	0.498	−0.278
		36	0.493	−0.302
	GRU	12	0.523	−0.213
		24	0.497	−0.22
		36	0.511	−0.222
"Household 2"	LSTM	12	0.235	−0.093
		24	0.273	−0.117
		36	0.251	−0.133
	GRU	12	0.246	−0.093
		24	0.25	−0.103
		36	0.267	−0.077

4.5. Comparing Different Models

The Table 7 shows a comparison between the prediction models created, baseline, Prophet and the most accurate neural network-based models for each object used in this study, using R^2 score, calculated using actual values of the testing data set and the ones predicted by a model, as a comparison metric:

Table 7. Comparison of different prediction models.

	Object			
Prediction Model	**"Solar 30K"**	**"Solar 5K"**	**"Household 1"**	**"Household 2"**
Baseline	0.555	0.614	-	-
Prophet	0.922	0.92	0.42	0.241
Neural networks	0.972	0.981	0.523	0.315

As can be seen from the above table, both Prophet and neural network-based models showed a huge improvement compared to the baseline model. Even though solar generation prediction models created using neural networks were somewhat more accurate compared to the Prophet, the difference was not great. When comparing models of energy consumption of a household, the difference between the Prophet and neural network-based models was more noticeable, especially with "Household 1".

5. Results and Conclusions

Microcontroller firmware that is versatile enough to be used with different solar PV inverters was developed and designed in the present work. Although data from solar power plants has been collected before, the methods employed were not universal, and separate microcontroller software was written for each solar inverter. The firmware developed here runs on microcontrollers connected to the solar inverters using the Modbus interface, and it is regularly upgraded.

With the help of the software designed and other additional tools, the data used in the models developed were collected and prepared accordingly. After testing the developed artificial neural network models, it was found that in the average case, neural networks designed using GRU architecture were the most accurate; however, the performance of the most accurate LSTM-type models was quite similar. The results also showed an interesting case where the feedforward neural network gave better results for a specific object compared to the other two architectures. These results can be found in the Table 8.

Table 8. Most accurate models.

	Objects			
	"Solar 30K"	**"Solar 5K"**	**"Household 1"**	**"Household 2"**
Architecture	GRU	GRU	GRU	Feedforward
Optimiser	RMSProp	Adam	RMSProp	Adam
Loss function	MAE	MAE	MAE	MSE
Learning rate	0.001	0.001	0.001	0.001
Sequence length	36	36	12	-
Number of layers	2	4	2	8
Learning R^2	0.979	0.997	0.416	0.354
Testing R^2	0.972	0.981	0.523	0.315

While comparing the models developed using neural networks with the ones developed by means of the Prophet package, the results were quite similar when predicting the generation output of a solar power plant; however, as mentioned above, that was not the case with household energy consumption prediction. Even though the Prophet requires much less computational power, as shown in this study, it may noticeably decrease the predictive accuracy. When comparing the baseline model with more advanced prediction models, it became clear that both the Prophet and neural network models could produce a huge improvement, as the baseline model simply could adjust prediction output according to the weather conditions and other important factors.

The testing of the models created also showed that generated energy prediction models performed much more accurately compared to energy consumption prediction models. This disparity can be attributed to the predictability of factors influencing solar energy generation, such as weather conditions, time of day and geographical location. Solar power output is closely tied to these factors, contributing to the higher accuracy of a model.

In contrast, forecasting household energy usage is more complex. This is because it is hard to predict how people will use energy in their homes. Factors like the variety of appliances and devices in households contribute to this unpredictability. Even small changes, such as obtaining new appliances or having more people at home, can affect energy usage patterns. Despite these difficulties, the models still managed to provide reasonably accurate estimates of household energy consumption.

This work succeeded in achieving the goal of developing a prototype for an efficient solar energy utilisation system by directing the overflow energy to a specific electrical device. To fully implement the system created, an appropriate firmware program should be developed. In predicting the amount of electricity to be generated by a solar power plant

alongside household consumption, the firmware developed could reveal the difference between these two predicted values. If the models predict the overflow of energy, some electrical devices can be turned on and vice versa.

In the future, it would be beneficial to explore additional architectures for predicting both values. For instance, transformer-based models have shown promise in surpassing traditional recurrent neural networks like LSTM and GRU [20]. Moreover, expanding the training and testing data sets to include more data from various solar power plants and households, and potentially adopting a universal approach that eliminates the need for creating separate models for each data set, could significantly enhance the prototype's applicability in real-world scenarios. Furthermore, once the previously mentioned electrical device is fully implemented, evaluating the long-term energy performance of the proposed prototype would be valuable.

Author Contributions: Writing—original draft, J.G.; Supervision, A.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article, and GitHub repository https://github.com/juliusga/Effective-Solar, accesed on 21 January 2024.

Conflicts of Interest: The authors declare no conflict of interest.

Nomenclature

GRU	Gated recurrent unit
MAE	Mean absolute error
MSE	Mean squared error
LSTM	Long short-term memory
MAPE	Mean absolute percentage error
DHI	Diffuse horizontal irradiance
DNI	Direct normal irradiation
EBH	Direct (beam) horizontal irradiance
GHI	Global horizontal irradiance
GTI	Global tilted irradiance
ReLU	Rectified linear unit

Appendix A. Comparison Plots of the Most Accurate Neural Network Models for Each Object

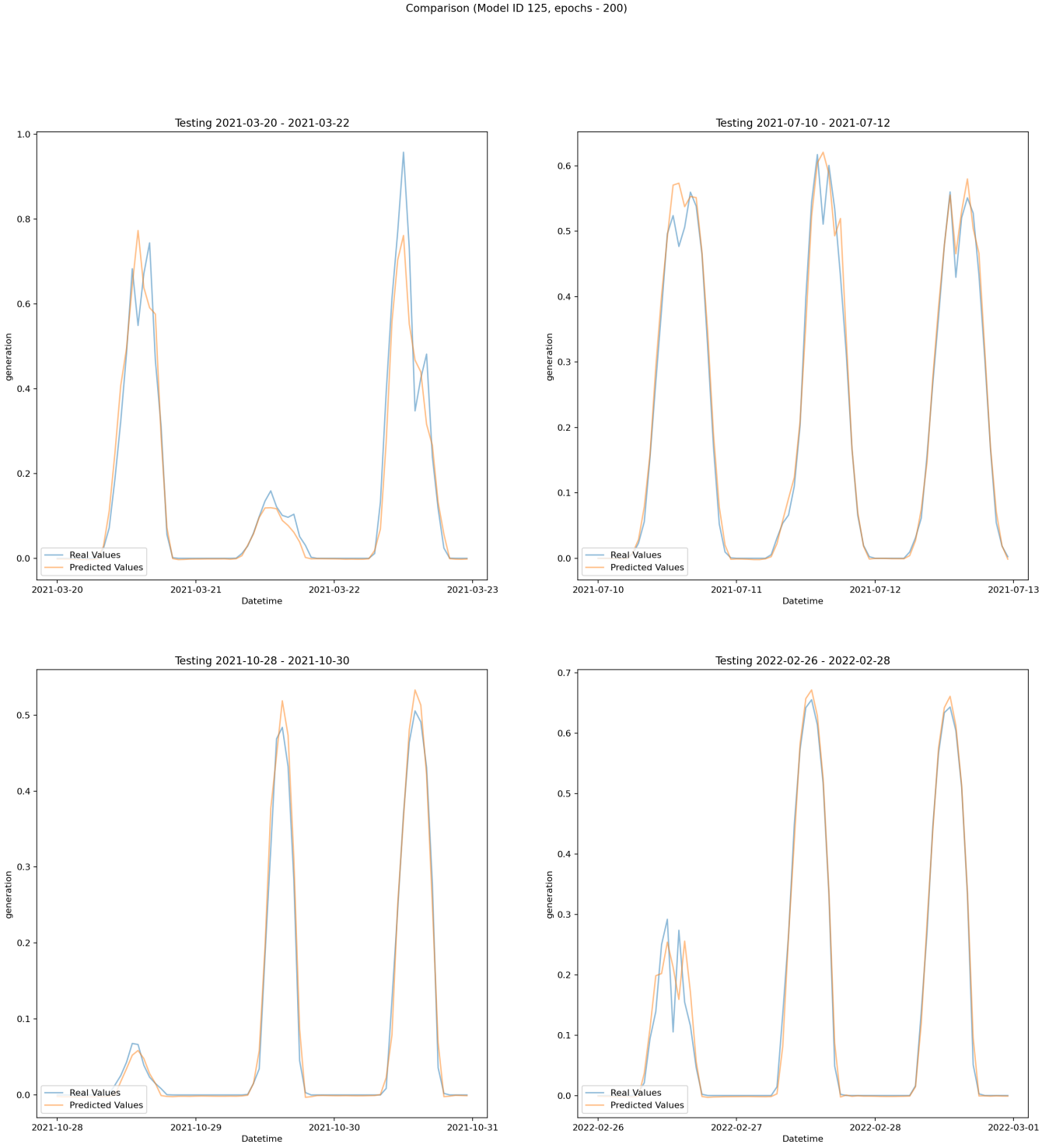

Figure A1. Most accurate "Solar 30K" model comparison plots.

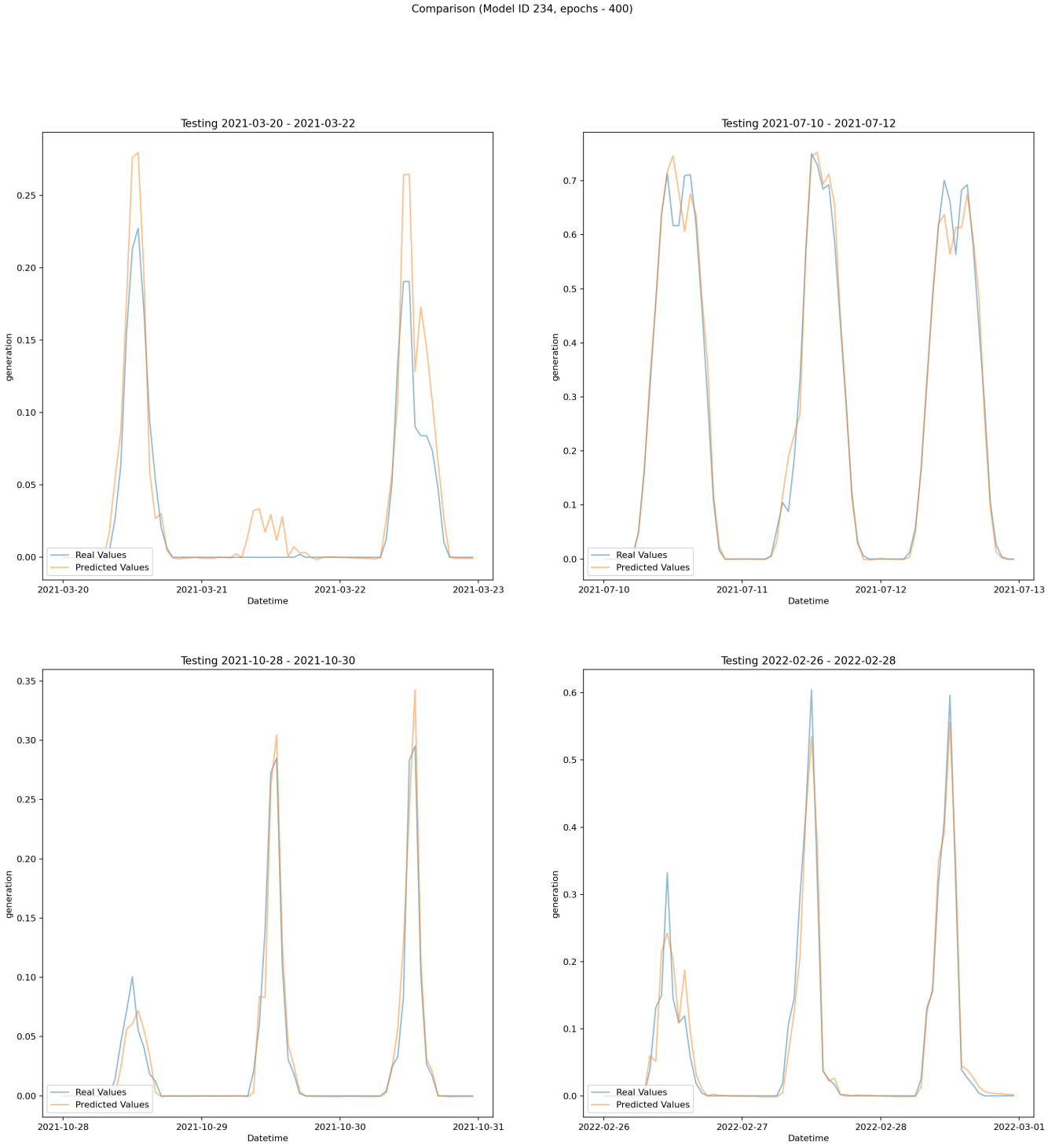

Figure A2. Most accurate "Solar 5K" model comparison plots.

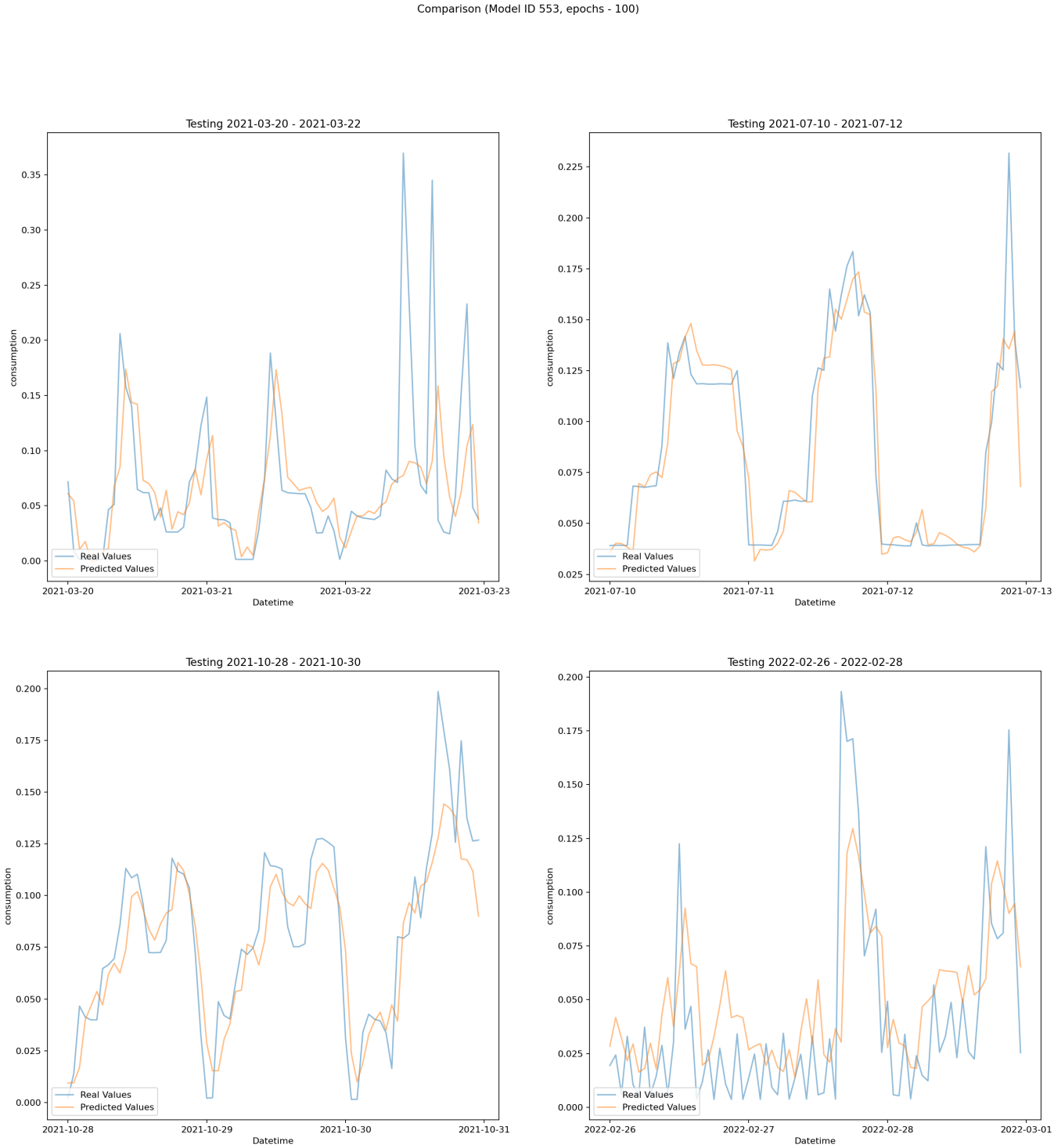

Figure A3. Most accurate "Household 1" model comparison plots.

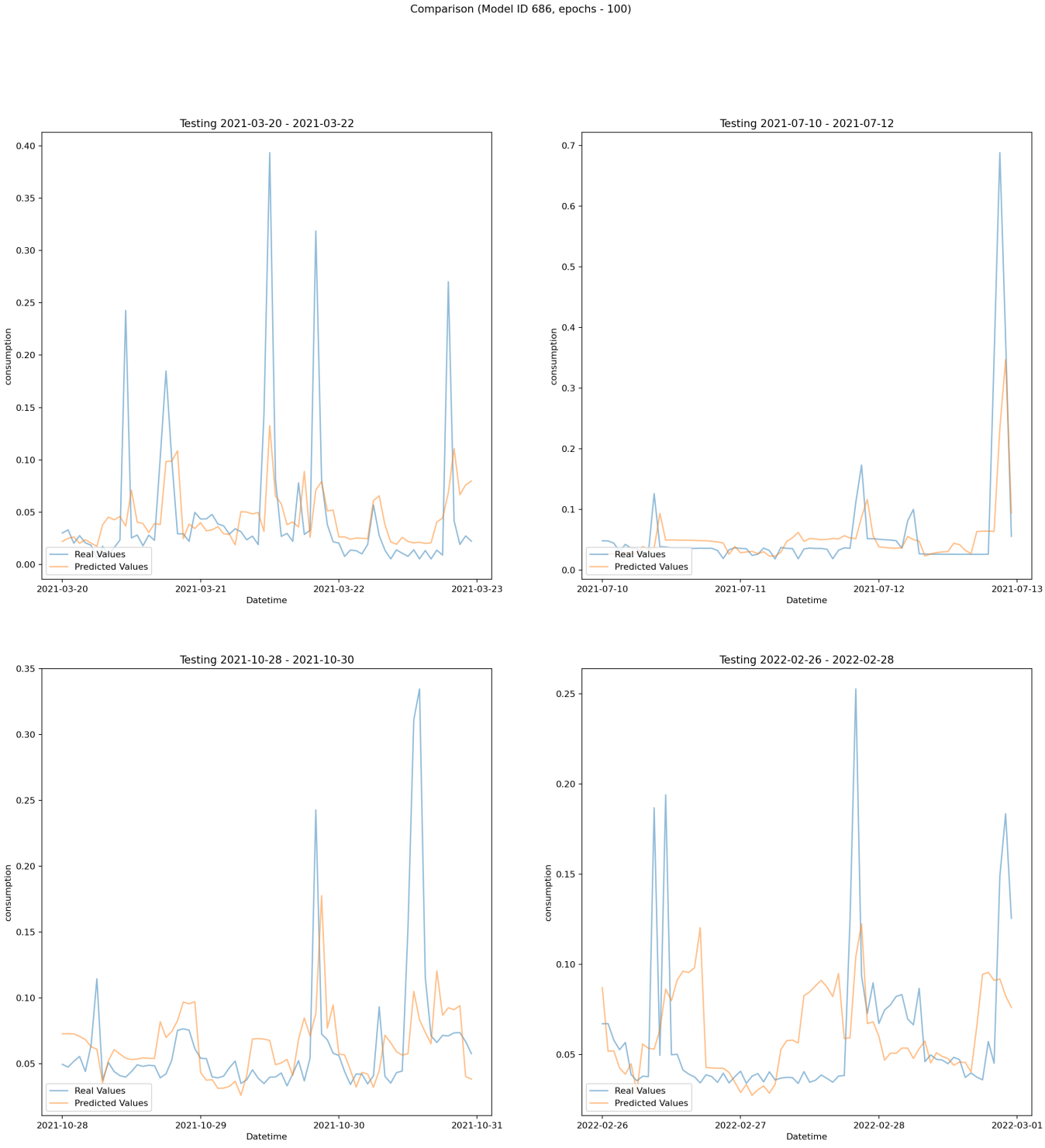

Figure A4. Most accurate "Household 2" model comparison plots.

Appendix B. Comparison Plots of the Prophet Models for Each Object

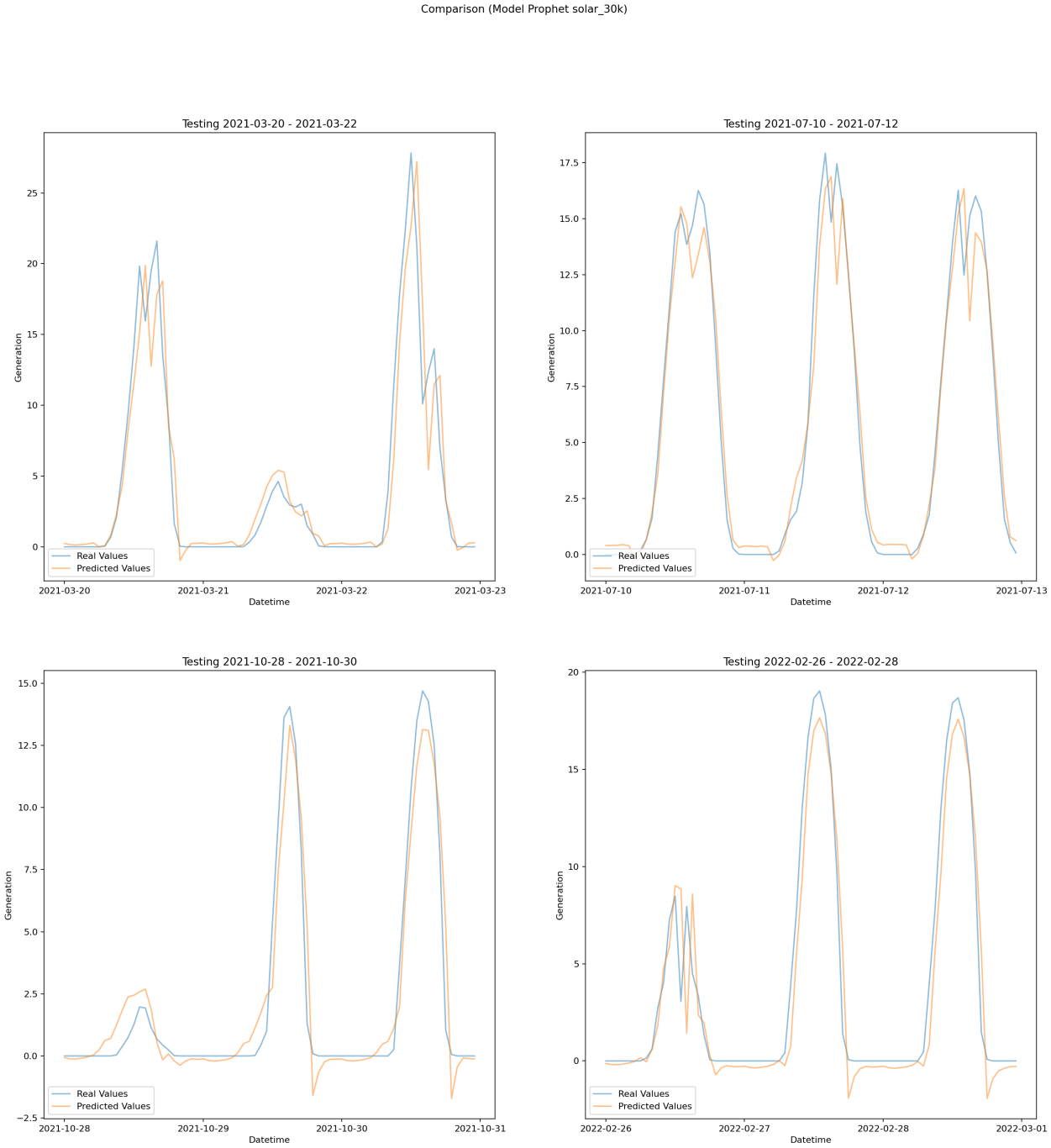

Figure A5. "Solar 30K" Prophet model comparison plots.

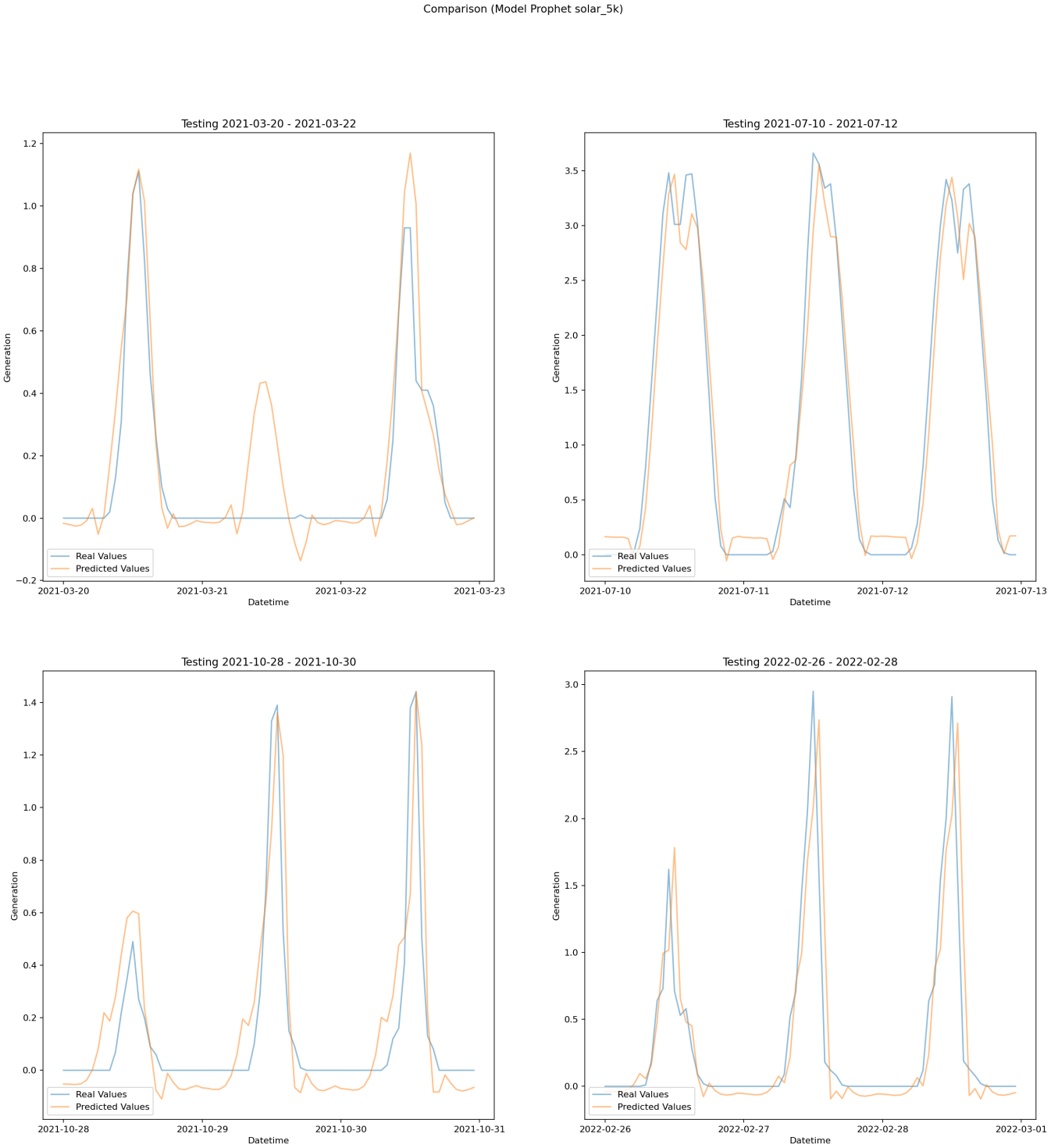

Figure A6. "Solar 5K" Prophet model comparison plots.

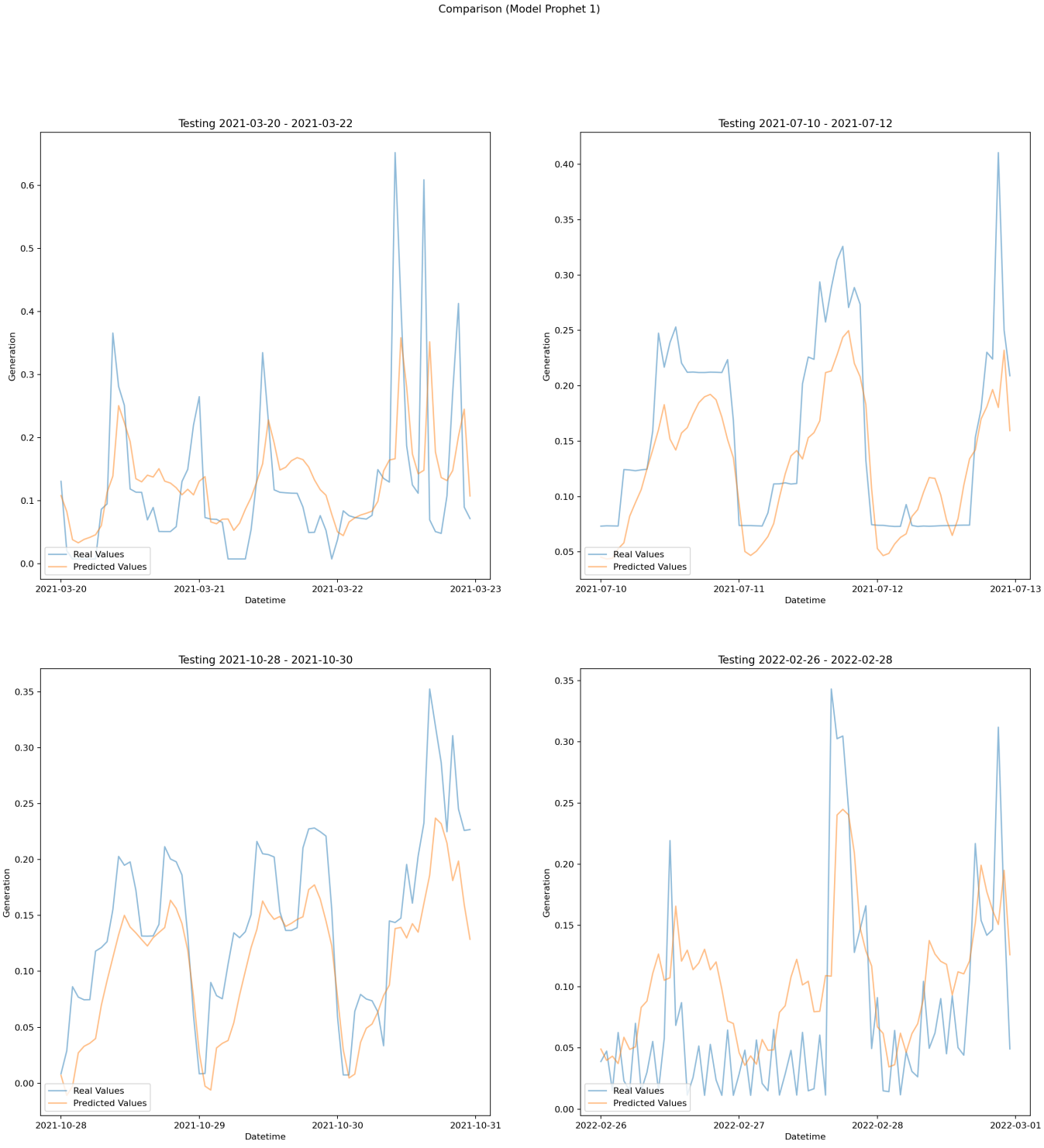

Figure A7. "Household 1" Prophet model comparison plots.

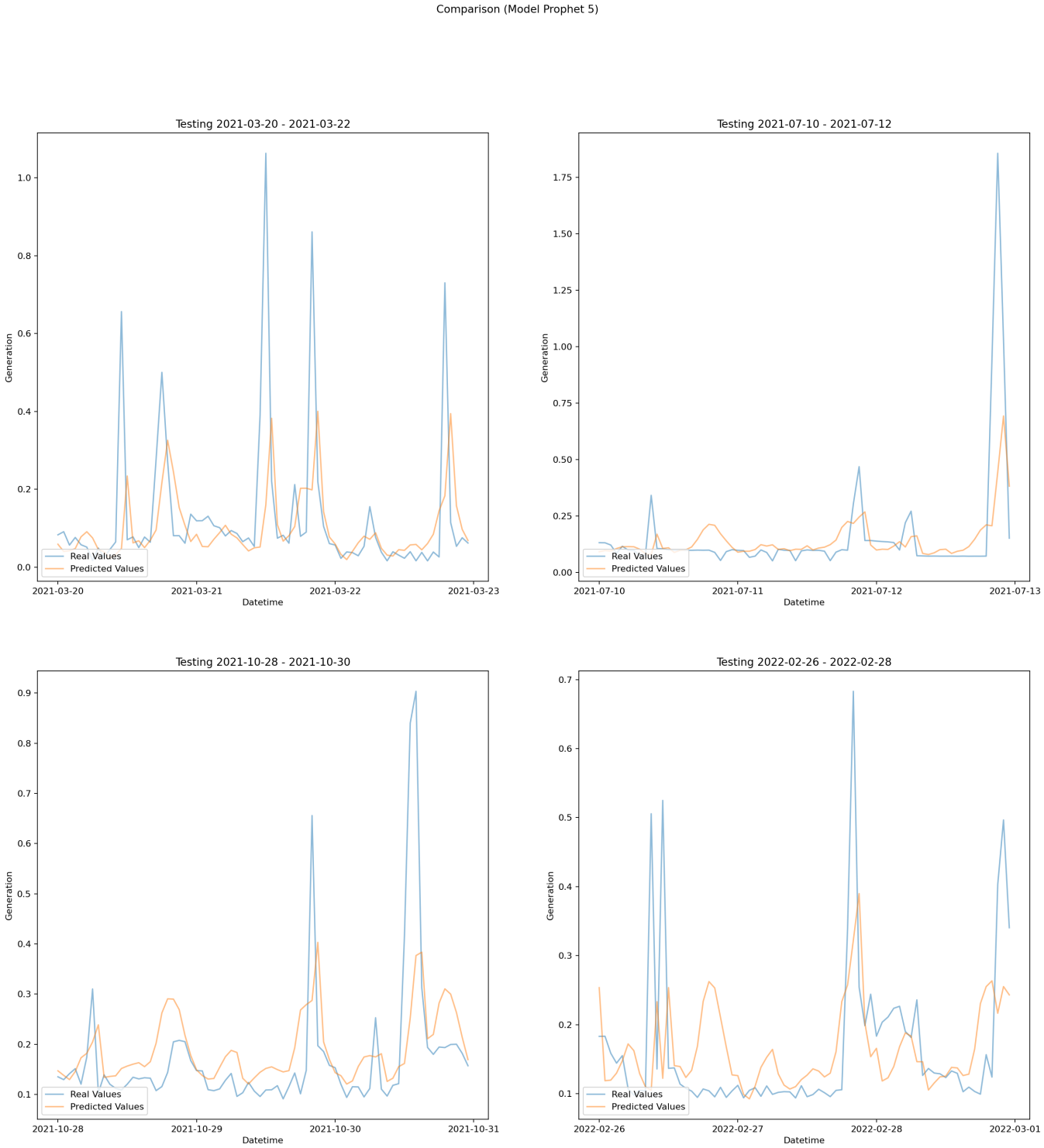

Figure A8. "Household 2" Prophet model comparison plots.

Appendix C. Comparison Plots of the SVR Models for Each Object

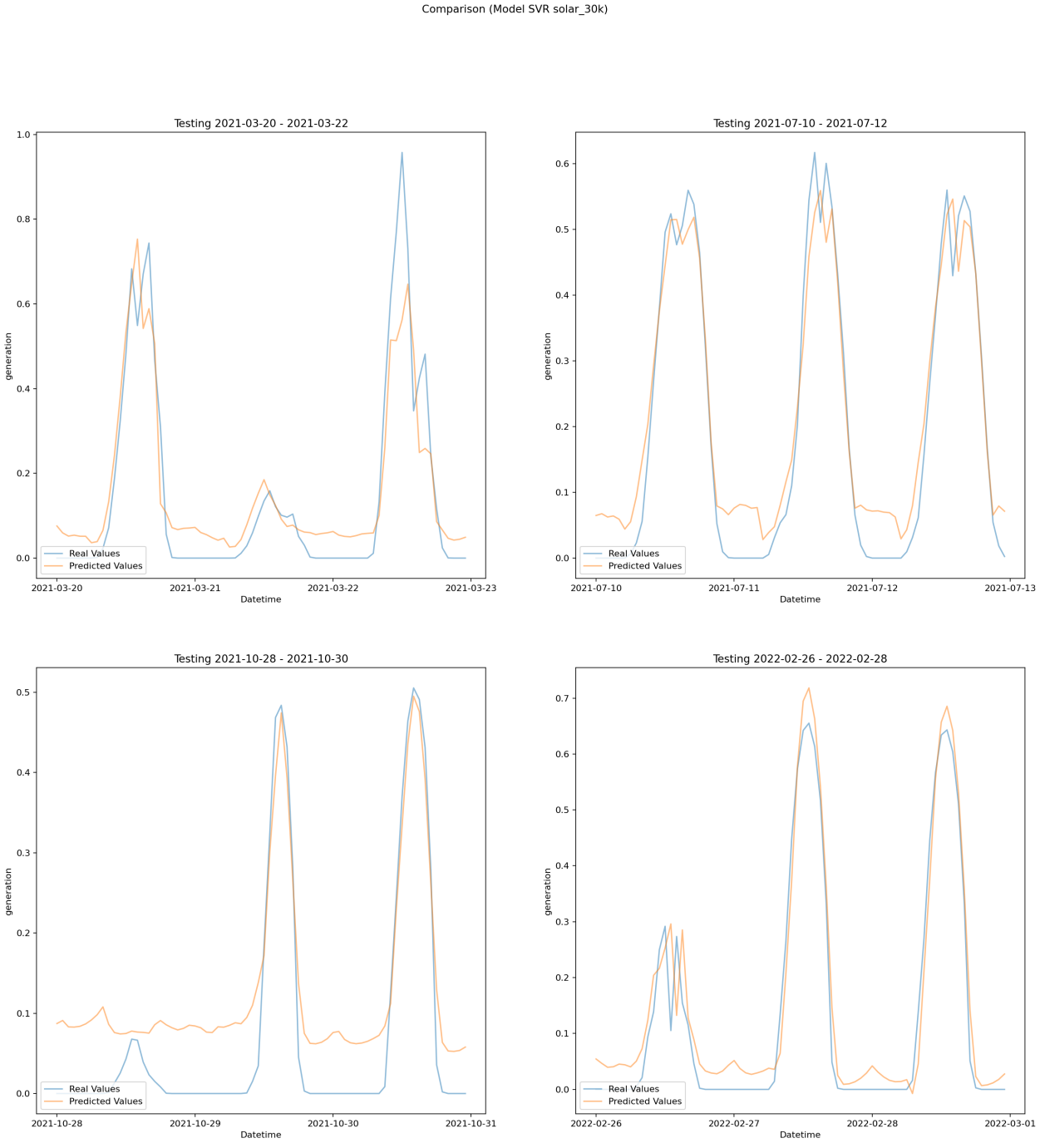

Figure A9. "Solar 30K" SVR model comparison plot

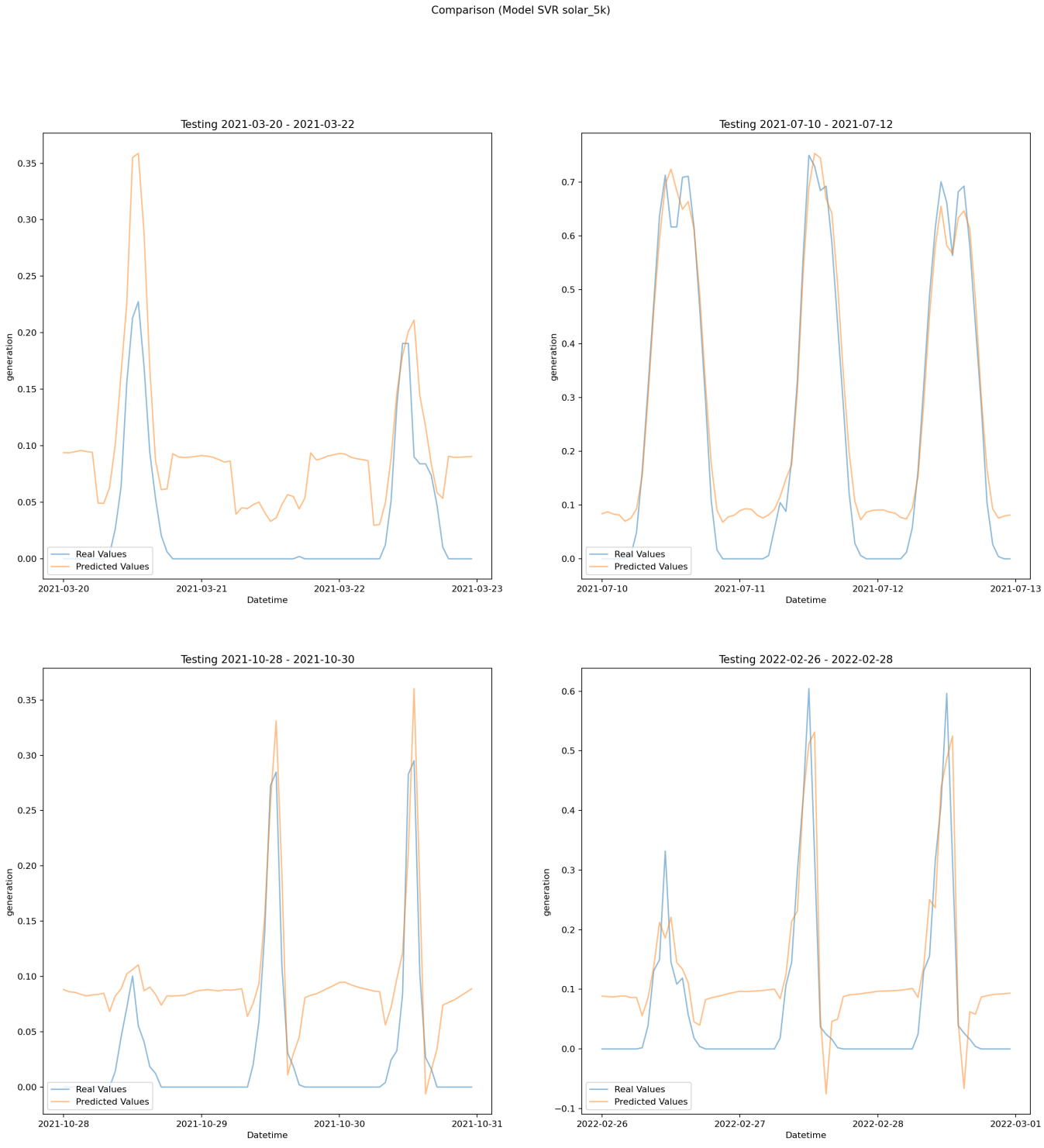

Figure A10. "Solar 5K" SVR model comparison plot

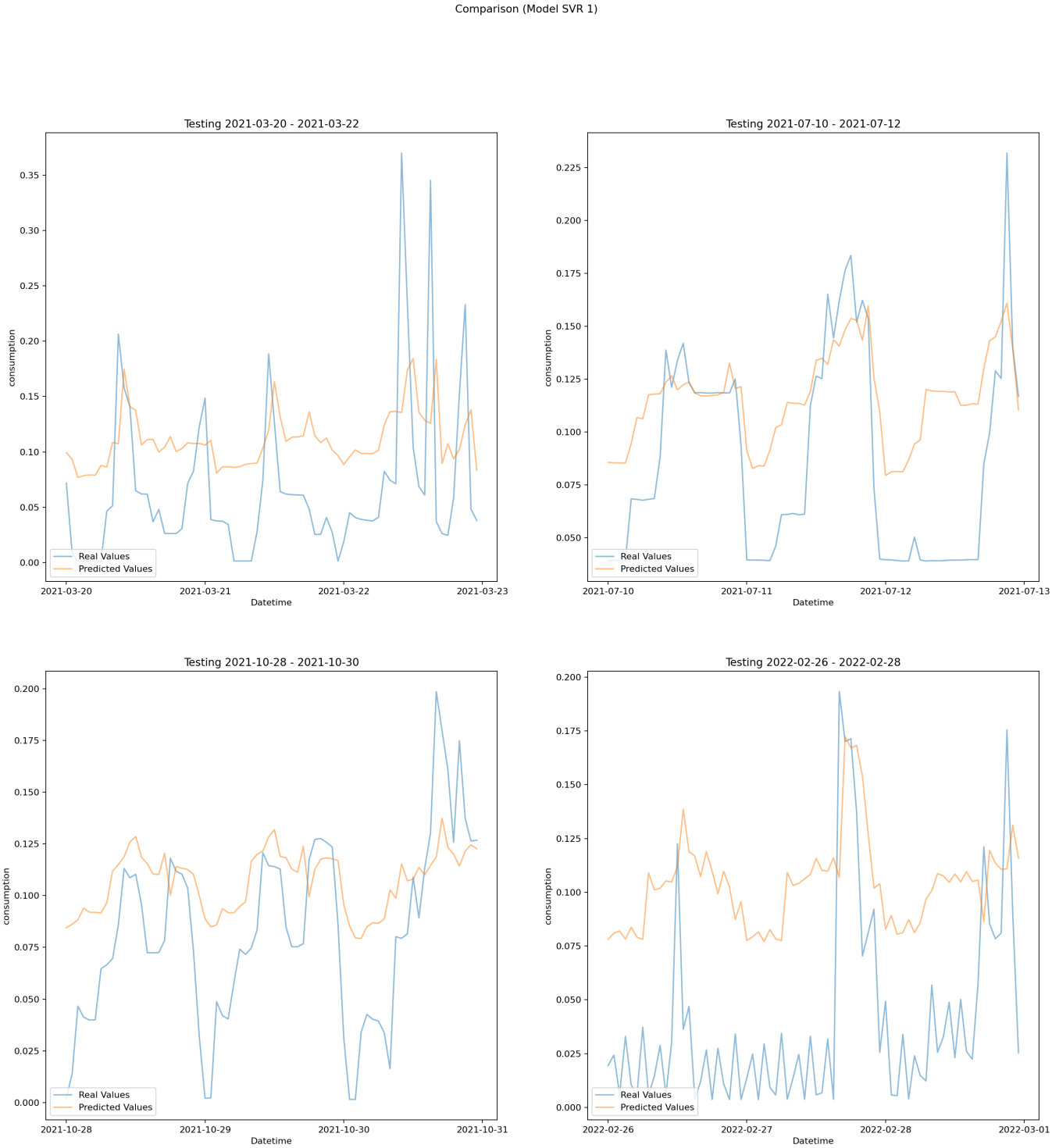

Figure A11. "Household 1" SVR model comparison plot

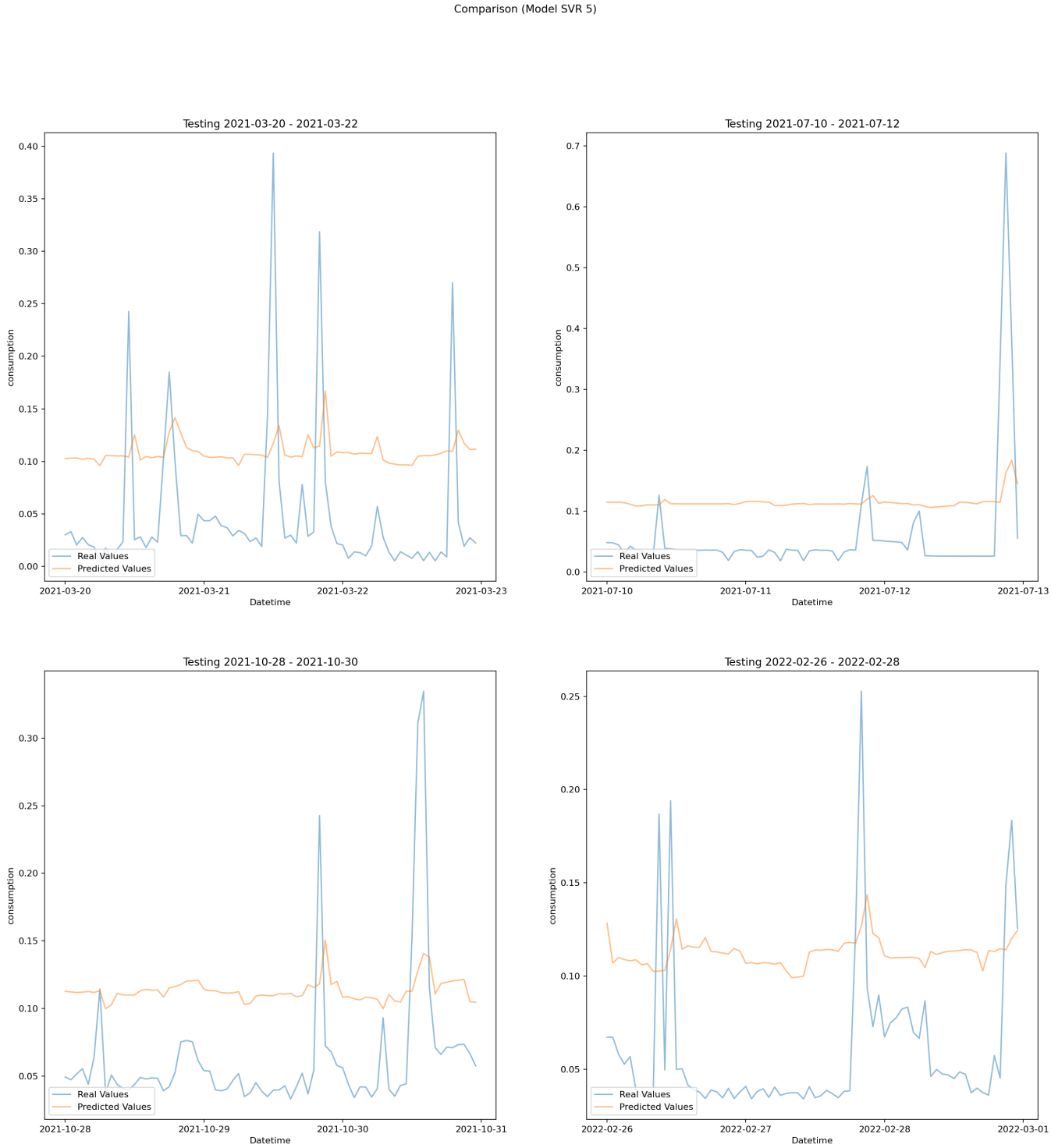

Figure A12. "Household 2" SVR model comparison plot

Appendix D. Comparison Plots of the Baseline Models for Each Generation Object

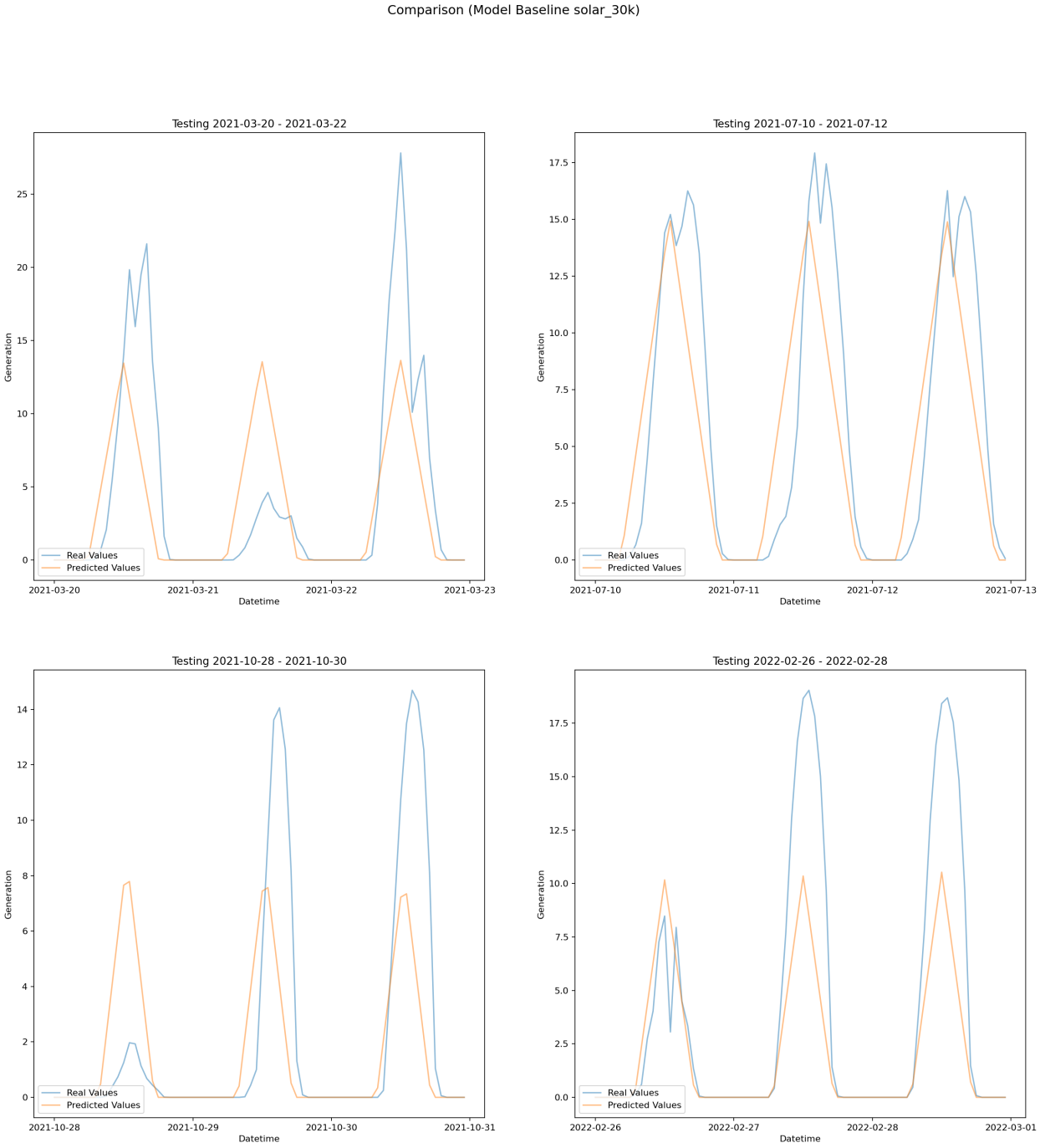

Figure A13. "Solar 30K" baseline model comparison plots.

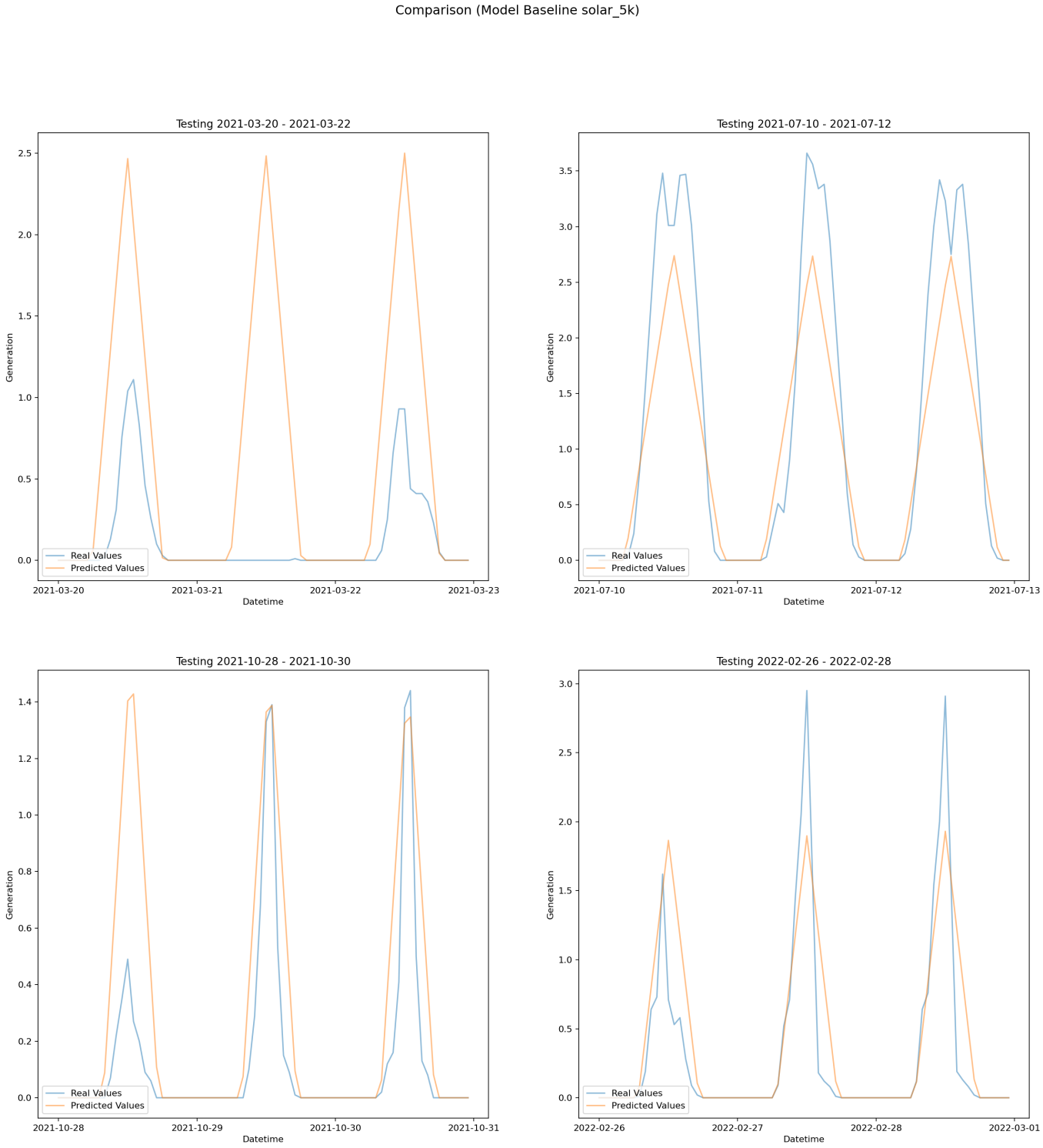

Figure A14. "Solar 5K" baseline model comparison plots.

Appendix E. Flowchart of a Proposed Work

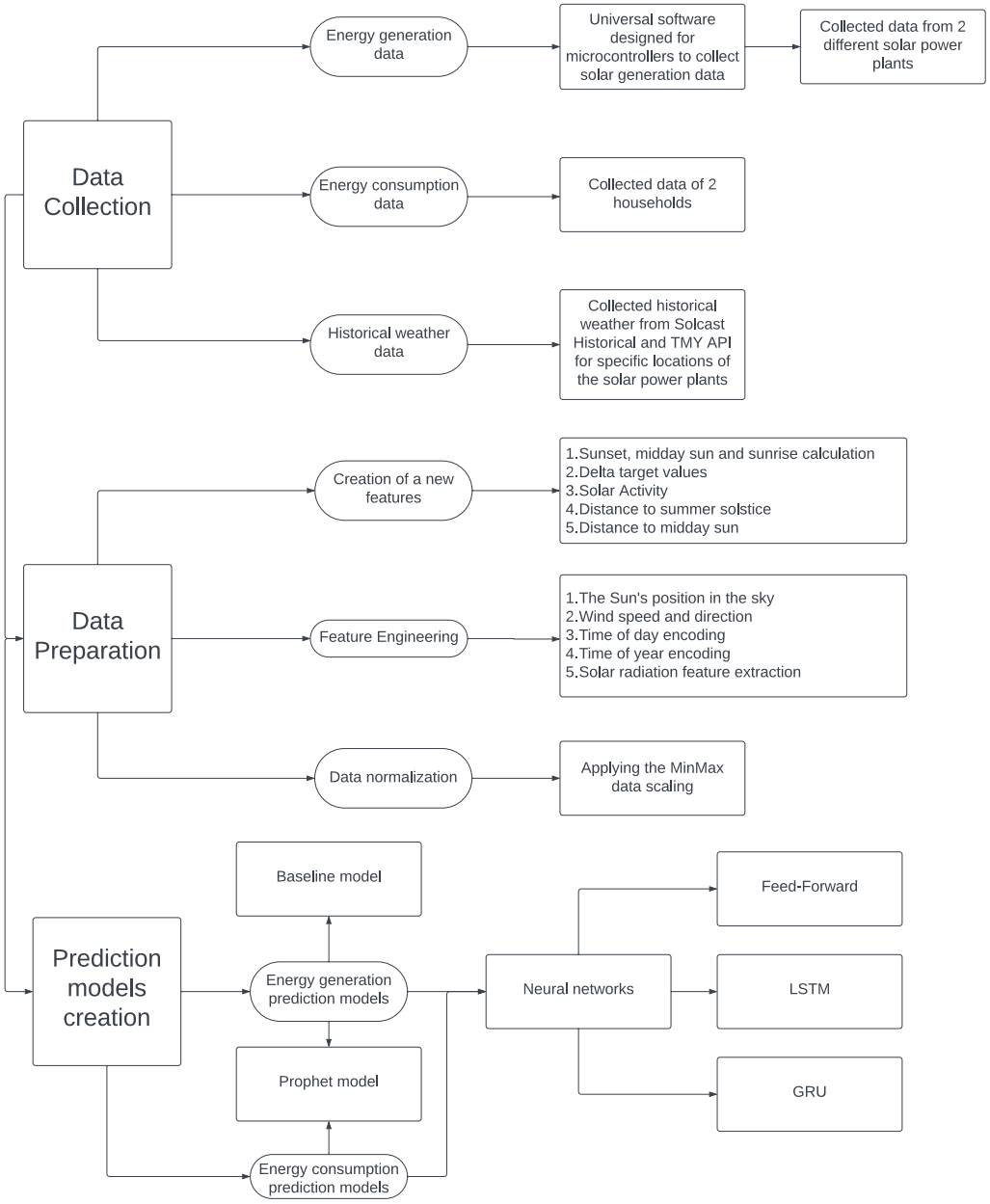

Figure A15. Flowchart of a proposed work.

References

1. Lugo-Laguna, D.; Arcos-Vargas, A.; Nuñez-Hernandez, F. A European assessment of the solar energy cost: Key factors and optimal technology. *Sustainability* **2021**, *13*, 3238. [CrossRef]
2. Kramarz, T.; Park, S.; Johnson, C. Governing the dark side of renewable energy: A typology of global displacements. *Energy Res. Soc. Sci.* **2021**, *74*, 101902. [CrossRef]
3. Basit, M.A.; Dilshad, S.; Badar, R.; Sami ur Rehman, S.M. Limitations, challenges, and solution approaches in grid-connected renewable energy systems. *Int. J. Energy Res.* **2020**, *44*, 4132–4162. [CrossRef]
4. Modbus Organization Inc. Modbus Application Protocol Specification V1.1b3. 795 KB. 2012. Available online: https://modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf (accessed on 10 March 2022).
5. Cai, W.; Chao, F.; JinLong, T.; DeXiong, L.; Sifu, H.; ZhiGang, X. The influence of environment temperatures on single crystalline and polycrystalline silicon solar cell performance. *Sci. China* **2012**, *55*, 235–241.
6. Reda, I.; Andreas, A. *Solar Position Algorithm for Solar Radiation Applications*; National Renewable Energy Laboratory: Golden, CO, USA, 2008; pp. A-1–A-4.

7. Sharmaa, K.; Kumara, V.; Bishta, D.S.; Garg, H. Comparative study of acrylic flat plate and dome shaped collector forsummer and winter solstice conditions. *Mater. Today Proc.* **2021**, *45*, 5489–5493. [CrossRef]
8. Jacobson, M.Z. *Fundamentals of Atmospheric Modeling*; Cambridge University Press: New York, NY, USA, 2005; pp. 317–322.
9. Vasel, A.; Iakovidis, F. The effect of wind direction on the performance of solar PV plants. *Energy Convers. Manag.* **2017**, *153*, 455–461. [CrossRef]
10. Gokmen, N.; Hu, W.; Hou, P.; Chen, Z.; Sera, D.; Spataru, S. Investigation of wind speed cooling effect on PV panels in windy locations. *Renew. Energy* **2015**, *90*, 283–290. [CrossRef]
11. Solcast, A. Solcast API Documentation. 2022. Available online: https://docs.solcast.com.au/#get-tmy-p50 (accessed on 21 January 2024).
12. Raju, N.V.G.; Lakshmi, K.P.; Scholar, V.M.J.; Kalidindi, A.; Padma, V. Study the Influence of Normalization/Transformation process on the Accuracy of Supervised Classification. In Proceedings of the 2020 Third International Conference on Smart Systems and Inventive Technology, Tirunelveli, India, 20–22 August 2020.
13. Grąža, S. Prieš 7 m. įrengtos saulės elektrinės generacija. 2020. Available online: https://www.saulesgraza.lt/saules-elektrines-generacija (accessed on 2 April 2022).
14. Taylor, S.J.; Letham, B. Forecasting at scale. *Am. Stat.* **2018**, *72*, 37–45. [CrossRef]
15. Shewalkar, A.; Nyavanandi, D.; Ludwig, S.A. Performance evaluation of deep neural networks applied to speech recognition: RNN, LSTM and GRU. *J. Artif. Intell. Soft Comput. Res.* **2019**, *9*, 235–245. [CrossRef]
16. Yamak, P.T.; Yujian, L.; Gadosey, P.K. A comparison between arima, lstm, and gru for time series forecasting. In Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence, Sanya, China, 20–22 December 2019; pp. 49–55.
17. Cahuantzi, R.; Chen, X.; Güttel, S. A comparison of LSTM and GRU networks for learning symbolic sequences. In *Science and Information Conference*; Springer: Cham, Swizterland, 2023; pp. 771–785.
18. Lu, L.; Shin, Y.; Su, Y.; Karniadakis, G.E. Dying ReLU and Initialization: Theory and NumericalExamples. *arXiv* **2019**, arXiv:1903.06733.
19. Chicco, D.; Warrens, M.J.; Jurman, G. The coefficient of determination R-squaredis more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. *PeerJ Comput. Sci.* **2021**, *7*, e623. [CrossRef] [PubMed]
20. Wu, N.; Green, B.; Ben, X.; O'Banion, S. Deep transformer models for time series forecasting: The influenza prevalence case. *arXiv* **2020**, arXiv:2001.08317.