

Article

Investigation of Load, Solar and Wind Generation as Target Variables in LSTM Time Series Forecasting, Using Exogenous Weather Variables

Thomas Shering¹, Eduardo Alonso^{1,*} and Dimitra Apostolopoulou² 

¹ Department of Computer Science, City, University of London, London EC1V 0HB, UK; tomshering@gmail.com

² Oxford Institute for Energy Studies, Oxford OX2 6FA, UK; dimitra.apostolopoulou@oxfordenergy.org

* Correspondence: e.alonso@city.ac.uk; Tel.: +44-20-7040-5060 (ext. 4049)

Abstract: Accurately forecasting energy metrics is essential for efficiently managing renewable energy generation. Given the high variability in load and renewable energy power output, this represents a crucial area of research in order to pave the way for increased adoption of low-carbon energy solutions. Whilst the impact of different neural network architectures and algorithmic approaches has been researched extensively, the impact of utilising additional weather variables in forecasts has received far less attention. This article demonstrates that weather variables can have a significant influence on energy forecasting and presents methodologies for using these variables within a long short-term memory (LSTM) architecture to achieve improvements in forecasting accuracy. Moreover, we introduce the use of the seasonal components of the target time series, as exogenous variables, that are also observed to increase accuracy. Load, solar and wind generation time series were forecast one hour ahead using an LSTM architecture. Time series data were collected in five Spanish cities and aggregated for analysis, alongside five exogenous weather variables, also recorded in Spain. A variety of LSTM architectures and hyperparameters were investigated. By tuning exogenous weather variables, a 33% decrease in mean squared error was observed for solar generation forecasting. A 22% decrease in mean absolute squared error (MASE), compared to 24-h ahead forecasts made by the Transmission Service Operator (TSO) in Spain, was also observed for solar generation. Compared to using the target variable in isolation, utilising exogenous weather variables decreased MASE by approximately 10%, 15% and 12% for load, solar and wind generation, respectively. By using the seasonal component of the target variables as an exogenous variable itself, we demonstrated decreases in MASE of 19%, 12% and 8% for load, solar and wind generation, respectively. These results emphasise the significant benefits of incorporating weather and seasonal components into energy-related time series forecasts.

Keywords: time series forecasting; exogenous variables; seasonal decomposition; LSTMs



Citation: Shering, T.; Alonso, E.; Apostolopoulou, D. Investigation of Load, Solar and Wind Generation as Target Variables in LSTM Time Series Forecasting, Using Exogenous Weather Variables. *Energies* **2024**, *17*, 1827. <https://doi.org/10.3390/en17081827>

Academic Editors: José Matas and Isabel Jesus

Received: 22 February 2024

Revised: 19 March 2024

Accepted: 27 March 2024

Published: 11 April 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

As the issue of climate change becomes ever more prominent, and the availability and geopolitical difficulties of fossil fuels become exacerbated, the need for a greater level of energy supply from renewable sources such as solar and wind is becoming a central global concern. Analysis and forecasting of energy-related time series is therefore a crucial field of research, with progress in this area representing a necessary step towards achieving a sustainable future. Forecasting time series involves predicting future data points in a set of temporally ordered values. Short-term forecasting, one hour ahead, was enacted as it is utilised in many applications such as wind power management, electricity market processing and load management [1]. As the field has advanced, statistical methods, classical machine learning methods, and more recently, deep learning methods have been proposed.

Within statistical methods, linear regression has shown reasonable results in short-term forecasting [2], although autoregressive integrated moving average (ARIMA) [3] is typically more suitable for energy-related forecasting since it does not assume a static relationship between variables. Classical machine learning techniques such as XGBoost have shown success for time series forecasting within the energy sector, and benefit from low computational times and high interpretability [4].

Solutions involving neural networks (networks of interconnected, digital nodes) have shown much promise in improving forecasting ability. For instance, neural networks have demonstrated superior accuracy over ARIMA within this research domain [5,6]. It has also been shown that simple multi-layer neuron models can outperform classical machine learning methods [7]. Even basic Artificial Neural Networks (ANNs), with a single hidden layer, have yielded solar generation predictions accurate enough for microgrid management [8]. It has been posited that statistical and classical machine learning methods cannot deal with volatile energy-related data as well as ANNs [9]. We suggest that the residual components of time series (less predictable noise in the data), which are likely to increase in magnitude due to climate change, contribute to this theory.

Recurrent neural networks (RNNs), that utilise deep learning, with more complex and layered neuron structures, have established a mechanism through which temporal relationships can be captured in connectionist weight matrices [10]. Therefore, they are especially relevant in time series forecasting. Hochreiter and Schmidhuber [11] proposed using long short-term memory units (LSTMs) which significantly reduce exploding and vanishing gradients that can be observed with conventional RNNs [12]. This issue involves individual neuron weights becoming either too large or tending to zero, significantly reducing the ability of a neural network to generalise data. The excellent ability of LSTM-based architectures to be selective and capture long-term dependencies has been demonstrated in hourly load, solar power and wind speed forecasting [13–15]. The choice to utilise LSTM architectures in this research was driven by their proven capacity for fast real-time forecasting ability. Their inherent capability to process sequential information and make predictions with minimal latency, using gating mechanisms and the ability to capture long-term temporal dependencies, aligns with the operational demands of real-time energy forecasting.

Many studies on short-term energy forecasting have achieved excellent results, although they did not utilise exogenous variables that could have improved forecast accuracy and generalisability even further. An exogenous variable is a time series that runs over the same time frame as the series being forecast, but its future values are not predicted. Rather, it is fed into the neural network alongside the variable being forecast in the hope of allowing the network to make more accurate predictions, adjusting its weight matrices in a more optimum way. As climate change continues to make weather patterns less predictable, data outliers will make time series forecasting more difficult [16]. This increases the need to actively record and utilise weather variables to enable more accurate predictions.

Another interesting extension of work that focuses purely on the neural network itself is the implementation of LSTM models that utilise seasonal decomposition for energy-related forecasting [17]. Seasonal decomposition divides a time series into its constituent components, which are usually: seasonal (recurring fluctuations), trend (overall directional changes or shifts in data magnitude), and residual (the noise not accounted for by the other components). This is indeed a highly promising area of research, with excellent generalisability and accuracy being reported, which strengthened our position to use LSTMs and conduct additional experimentation with seasonally decomposed components as exogenous variables. Ref. [17] note that anomalous weather conditions can cause degradation of forecasting accuracy, which further affirms our conviction to consider solutions that involve these variables explicitly in model training and weight updates, as exogenous variables.

Some recent progress has been realised, but comprehensive analysis of weather variable impact has not matured as a field. It has been shown that load and temperature correlate strongly and that inputting temperature as an exogenous variable provides improved

accuracies [18,19]. This is due in part to the use of temperature-controlling appliances such as air conditioning units and heaters, which increase load at more extreme temperatures. Our expanded suite of exogenous variables builds on this work. Distinctions between ‘cloudy’ and ‘sunny’ have been made by Lim et al. [18] for solar power generation, and whilst cloud cover is certainly a useful exogenous variable in this instance, research by Wojtkiewicz et al. [20] has suggested that a greater range of variables could have improved accuracies further. One-hour-ahead forecast errors were significantly reduced for solar irradiance when variables such as temperature and humidity were introduced in this study, even independently of cloud cover. This strengthened the case for investigating the impact of these variables on solar power generation. Using temperature, humidity, air pressure and other meteorological exogenous variables for one-hour-ahead wind speed forecasting has shown very promising results compared to using wind speed in isolation [21]. This study also used an LSTM architecture, giving clear motivation for investigating how these variables influenced wind power generation forecasting.

Understanding the impact of a greater range of exogenous weather variables (temperature, pressure, humidity, wind speed, rainfall in the last hour and cloud cover) on the target variables: load, solar generation and wind generation forecasting would be highly beneficial to the field, as the suite of exogenous variables used can have as much effect on forecast accuracy as the model or algorithms used themselves.

This paper aims to demonstrate that additional data collection and the use of exogenous variables are critical in accurate energy-related forecasting. Due to the current state of the literature described, there is still a need for a thorough exploration of the effect of these variables, specifically in energy-related time series forecasting. We have demonstrated that exogenous weather variable use can improve accuracy metrics by up to 30%, which provides clear justification for research in this area. This improved performance can assist in energy management to provide improved forecasts, more informed energy bidding and more efficient generation. Local authorities could also use greater forecasting accuracy to meet energy targets. It is hoped that other researchers may also benefit if our methods are combined with their own novel improvements. Contributions are summarised as follows:

1. Our results have shown that weather data, used as exogenous variables within an LSTM framework, can have a significant positive impact on forecast accuracy for electricity load, solar and wind generation. It has been shown that careful consideration of a range of such variables is crucial for successful energy-related forecasting operations.
2. Development of a framework for LSTM hyperparameter tuning for energy-related forecasting.
3. Novel use of the seasonal component of a time series as a separate exogenous variable has been demonstrated to have potential utility as a possible LSTM training aid, as this improved accuracy metrics by a substantial margin.

The remainder of the paper is organised as follows. In Section 2, the proposed methodological framework is developed. The results of the LSTM hyperparameter and model architecture optimisation are given in Section 3. Section 4 provides results for exogenous weather variable analysis, giving evidence of the associated positive impact on forecast accuracy. Section 5 presents results on using the seasonal component of a decomposed target time series as an exogenous variable for that target series itself. Overall, the results demonstrated increases forecast accuracies as a result of utilising exogenous variables, tuning hyperparameters and altering model design. Section 6 closes with the final conclusions.

2. Proposed Methodology

In this section, we describe the datasets used, data preprocessing, the data flow, LSTM architectures used, the seasonal decomposition process, evaluation metrics and initial hyperparameter optimisation.

2.1. Input Data

The dataset used contains hourly data, from 1 January 2015 00:00:00 to 31 December 2018 23:00:00, for energy generation, load and price, including separate fields for the different types of energy generation, for example, gas, coal, nuclear and renewable generation [22]. This file also contains forecasts for energy consumption and pricing. The forecasts are obtained from ENTSO-E, a public portal for transmission service operators (TSOs) data [23]. Since ENTSO-E facilitates the exchange and collection of data between TSOs, which record their data according to various compliances, the data collection process is deemed reliable. Whilst market factors may have influenced data reported by TSOs, it is not believed that this would have a significant impact on data integrity. An inescapable level of bias is present in the data collected since it does not account for rurally populated areas of Spain, although this omission provides greater evidence of the robustness of our model. Further work with a greater range of data collection points would be expected to improve forecast accuracies even further.

Hourly weather data over the same time period, for five different cities; Valencia, Madrid, Bilbao, Barcelona and Seville were also used [24]. These cities provide a reasonable cross-section of weather conditions across the major population centres of the country. The weather data contained the variables: temperature, pressure, humidity, wind speed, rainfall in the last hour and cloud cover. Weather data were aggregated for the five cities, based on their population size, since it is reasonable to assert that weather conditions in Madrid, for example, would have a greater influence on national electrical load figures. The weightings were: Valencia—0.2, Madrid—0.3, Bilbao—0.1, Barcelona—0.3 and Seville—0.1. Changing these weightings for further research would be facile and the allocation of weights is not likely to have impacted the results significantly. Studies have been carried out in order to find the optimum weighting of weather conditions from different geographical locations [25]; however, this was not a primary concern of this work.

The proposed framework was used to forecast the load, solar generation and onshore wind generation, i.e., these are our target variables. Exogenous weather variables studied were: temperature, rainfall in the last hour, humidity, cloud cover and wind speed. Boxplots of all target and exogenous variables are given in Appendix A Figure A1.

During data preprocessing, columns were checked for duplicates and null values. These constituted less than 0.1% of values and were filled using linear interpolation, where missing values were averaged between surrounding values. Since four years of data were available, the data were split with a train:validation:test ratio of 2:1:1. This split was carried out chronologically, with the first two years for training, the next year for validation, and the final year for testing. By using this split, it was ensured that data leakage was avoided as each model would only have access to the test data after completing training on entirely different (train and validation) datasets. The training dataset was used to optimise the model and the validation dataset was used to evaluate the accuracy and check for overfitting during training. Data was normalised so that all variables had equal initial weighting within the LSTM model. As part of the data preprocessing process, it was ensured that all variable timestamps were aligned. The flow of data is shown below in Figure 1.

The associated code for our research can be accessed at: https://github.com/tomshering/INM363_IP (accessed on 23 January 2024). This may also provide other researchers with source code that can be used to cross-validate hyperparameters and variables for time series forecasting with neural networks.

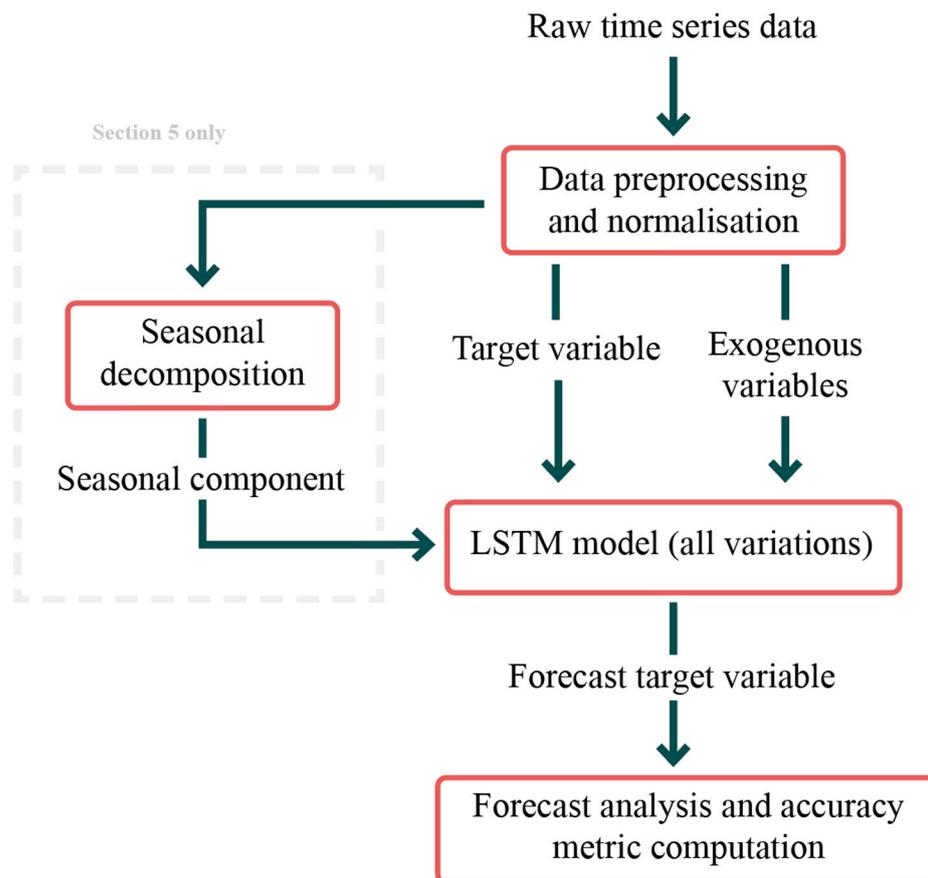


Figure 1. Data pipeline used in this research. The LSTM model is explained in Sections 2.2 and 2.3, with further details of hyperparameters used given in Section 3. The ‘Section 5 only’ data flow only applies to the research set out in Section 5 of this paper, with methods for this given in Section 2.4. Accuracy metrics computed are given in Section 2.5.

2.2. LSTM Basics

LSTM blocks [11] (Figure 2) work by utilising three gates, which control the flow of information from the memory cell, in the form of vectors, and each new input, to provide an output, which in our research was a single predicted value for the next data point of the target variable. The target and exogenous variables from each timestep were used to forecast the target variable value in the next timestep. No future timesteps were accessed by the model before they were forecast. The memory cell allows LSTM blocks to capture long-term dependencies between data points in a time series, whilst also allowing incoming information to influence output. The ability to comprehend all previous data, whilst also deciding which information to retain, and which to forget, made LSTM architectures appropriate for our research, where significant noise in variables such as wind generation may not always assist the model in forecasting future values.

In Figure 2, M_{t-1} represents the memory cell, a vector, at the previous time point; M_t represents the updated memory cell; h_{t-1} represents the previous hidden state; and h_t represents the updated hidden state, which was passed to the next LSTM block, if available (bottom right h_t) or passed as an output to a decoder (top h_t). By feeding exogenous variables into this architecture, the hidden state was able to capture features of this data and use it to more accurately forecast target variables. X_t represents the data input at time t , which in our case was the next value in the time series being processed. Blue + signs

indicate an additive combination of data. Sig. refers to the sigmoid function (Equation (1)) and tanh refers to the hyperbolic tangent function (Equation (2)).

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (1)$$

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (2)$$

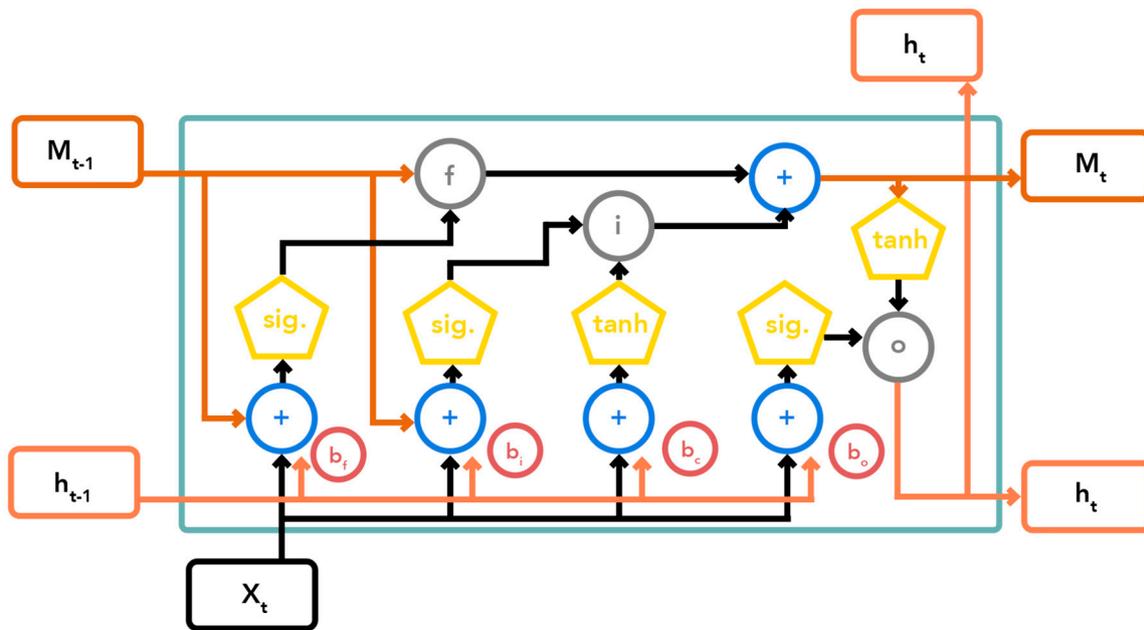


Figure 2. The data flow and operations within an LSTM block at time t .

The three gates of the LSTM block, forget, input and output, are denoted in grey circles as f , i and o , respectively, in Figure 2. Each gate involves a multiplicative operation that determines the extent to which incoming data is utilised downstream. This allows the model to take into account strong seasonal trends in variables such as solar generation and temperature, and any less obvious trends, without overfitting irrelevant data that naturally occurs in energy-related and environmental time series. Multiplication by 0 would represent a ‘closed’ gate, whereas a value of 1 would retain the full weighting of all information passed through. The forget gate determined which information from the previous hidden state should be erased. The input gate determined which information should be added to the current state, with the sigmoid function (Equation (1)) resolving which values to update and the tanh function (Equation (2)) creating a vector that is used as a temporary memory state. The sigmoid-filtered input information and the updated memory state were then combined at the output gate, which updated the hidden state.

The four biases shown in Figure 2 add to different data flows, in order to increase the level of customisability of an LSTM block. The b_f bias was added to the linear transformation at the forget gate, with a higher value resulting in less information being forgotten. The input gate bias, b_i , determined how much new information was added to the cell state; the cell state bias, b_c , influenced the weightings of candidate values being added to the temporary cell state, and the output bias, b_o , determined the extent to which the cell state should be added to the hidden state.

2.3. Proposed LSTM Architectures and Initialisation

Two models, one more complex than the other, were used. This accounted for the possibility that either simple models that do not over-extrapolate, or more complex mod-

els that can capture a greater depth of relationship between data points, could perform more successfully.

In order to test the impact of exogenous variables without adding unnecessary complexity, we used the model given in Figure 3, which we refer to as Model 1. Testing hyperparameters and increasing the number of layers in LSTM architectures at the same time was found to often dramatically reduce performance.



Figure 3. LSTM model 1.

The output from the LSTM block was passed through a linear neuron layer where some neurons are deactivated based on a dropout probability (see Section 3). The output from this layer was then passed through a decoder which converts the numerical output of the dropout layer into a predicted value for the next value of the target variable.

In order to develop the more complex model that captures hierarchical features, which we refer to as Model 2, we added more LSTM blocks. In this model, shown in Figure 4, data flowed through three LSTM blocks sequentially, going through a linear dropout layer after each block for regularisation. A decoder then converted the final output into a prediction for the next value in the target time series. This allowed later blocks to capitalise on short-term features apprehended by the earlier ones. Stacked LSTM models like this have been shown to perform well at capturing complex data series [26,27]. Therefore, the LSTM model shown in Figure 4 was used to establish whether a deeper model could provide greater accuracy and generalisation. Since a variety of exogenous weather variables were used, it was postulated that further neuron layers could capture a greater level of relationship between these and the target variables like load.

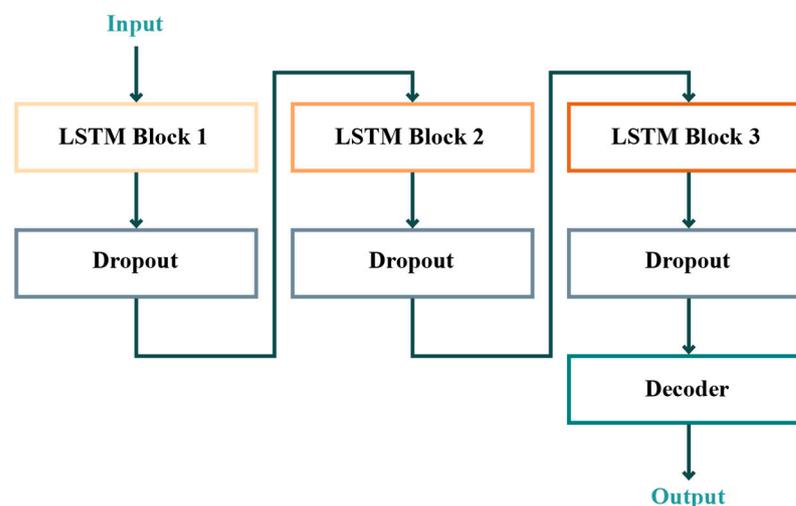


Figure 4. LSTM model 2.

Xavier weight initialisation (Equation (3)) was used as it has been reported as working well in similar applications by maintaining magnitudes of outputs in both directions [28]. A random value selected from a normal distribution, $U[a,b]$, with a lower bound of 'a' and an upper bound of 'b' was selected for each weight, where n represents the number of connections feeding into each neuron.

$$weight = U \left[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}} \right] \quad (3)$$

2.4. Seasonal Decomposition

Time series were seasonally decomposed using the ‘seasonal_decompose’ function, from the ‘statsmodel’ library [29]. The seasonal component of this decomposition gave a smooth, regularly fluctuating time series without noise, for each of the three target variables. Essentially, a new time series for each target variable was created, that corresponded to the original in timesteps, but the values of which only contained the seasonal component. These seasonal components were then fed into the Model 1 architecture (Figure 3) alongside the target variable, in order to assist in model training.

2.5. Evaluation Metrics

Test loss provided a quantitative evaluation of the difference between forecast and true values, using the unseen test data, and Mean Square Error Loss (MSE) was used as the loss function (Equation (4)). The mean average was computed for the set of squared differences between actual, y_i , and predicted values, \hat{y}_i , for all data points used, the number of which is given by N .

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (4)$$

Mean Average Scaled Error (MASE) and forecast bias were also used as evaluation metrics. Forecast bias reflects whether a model over- or under-predicts in an unbalanced fashion. This was useful to determine if the direction of predictions was skewed and was important to include as this information was not captured by other metrics. A value close to zero indicates a low bias.

$$MASE = \frac{\frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|}{\frac{1}{N-1} \sum_{i=2}^N |y_i - y_{i-1}|} \quad (5)$$

MASE [30] (Equation (5)), is calculated as the mean absolute error of forecast values, over the naïve forecast (each prediction taken as the previous value), excluding the first value (hence $i = 2$), since there is nothing to compare this with. The absolute error is computed by taking the absolute difference between the real and forecast values (numerator) and the real value and previous values (denominator). N represents the total number of data points in the time series; y_i represents the true value of the data point, \hat{y}_i represents the forecast value of the data point based on the model being tested, and y_{i-1} represents the value of the previous data point.

MASE was used as it gives good interpretability and compares the success of the model relative to fluctuations in the data itself. A value of 1 indicates that the model performs as well as a naïve forecast, and as the value tends to 0, the relative advantage of the model accuracy increases.

2.6. Initial Hyperparameter Optimisation

A selection of hyperparameters was initially optimised as follows. The given hyperparameter values were subsequently used for all experiments. These were not cross-validated as a full model optimisation, with every possible combination tested, because it was beyond the scope of the research.

- Optimiser weight decay = 1×10^{-6} . This penalises large weights (the strength of the signal that a neuron will pass on to the next neuron in the network) in the neural network to avoid overfitting (when a model is unable to generalise to unseen data because it has fitted too closely the training data).
- Batch size = 48. The number of data values in the time series processed before the model updates neuron weights in training. A lower value could potentially give more accurate predictions but the computational demand would increase. Furthermore, a lower value can make the model more ‘short-sighted’ to daily fluctuations, hence why a two-day period (48 h) was chosen.

- The number of layers within an LSTM block = 2. Two layers within the LSTM block allow the block to capture more dependencies between data points compared to one layer. More layers led to a decrease in accuracy as the model likely overfitted.
- A total of 200 epochs, the number of complete passes through all training data, during the training process, were used as this comfortably allowed convergence for training and validation loss. Therefore, discrepancies between training forecasts and the true values did not decrease further in a significant way after 200 passes.

3. Hyperparameter Cross-Validation and Model Evaluation

In this section, we provide the results from experimentation with different architectures and hyperparameter combinations. All exogenous weather variables stated in Section 2.1 were used in this phase, alongside the target variables.

The four hyperparameters trialled to optimise LSTM performance were as follows:

- Hidden dimension size refers to the number of neuron units inside the hidden layer of an LSTM unit. Hidden dimension size adjustment was important to test because it helps to strike a balance between capturing enough information and not overfitting.
- Learning rate, which affects how quickly neuron weights are updated during training, was selected as a key hyperparameter since the step size with which weights are updated is fundamental to convergence speed and accuracy, giving it a significant impact on performance. An optimal learning rate will provide a smooth progression towards the minimum test loss.
- Dropout, the probability of a neuron in the dropout layer being deactivated temporarily during model training, is an easily implementable regularisation technique. It allows the model to better generalise by not relying heavily on any individual neuron and its associated connections. A dropout probability of 0.5, for example, would mean that 50% of neurons would be likely to be temporarily deactivated, forcing the model to function with the remaining 50%.
- We experimented with three different optimisers: Adam, Root Mean Square Propagation (RMSProp) and Stochastic Gradient Descent (SGD), which are all commonly used algorithms that determine exactly how weights are updated during training. Optimiser functions can have a significant impact on model performance and convergence and depend greatly on other variables. As an example, the learning rate is a pivotal determinant of success for models using an SGD optimiser. Therefore, we thought it important to trial all possible hyperparameter combinations for each experiment.

The Adam optimiser is computationally efficient and utilises dynamic learning rate adjustment for different parameters depending on their utility. It has been used extensively by researchers in deep learning applications [31]. The Adam optimiser weight update procedure (Equation (6)) utilises two moving averages for every weight in the neural network; the first moment, \hat{m}_t , gives a smoothed estimate of the gradient, or direction of weight change, and the second moment, \hat{v}_t , gives a smoothed estimate of the magnitude of this gradient and is used with an exponent of 0.5 since it is a squared value. This allows v_t to capture the extent of gradient change independently of gradient direction. The mechanism through which the learning rate, α , affects weight update speed can be observed here, as a higher α would result in larger update magnitudes. This learning rate in itself is one of the hyperparameters that were tested through cross-validation, as described. A very small value, such as $\epsilon = 1 \times 10^{-8}$, is added to the denominator to avoid division by zero. The overall update quotient is subtracted from the original weight, w_t , to give the new weight, w_{t+1} .

$$w_{t+1} = w_t - \frac{\alpha \cdot \hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}} \quad (6)$$

The weight updating process is almost the same for RMSProp [32] (Equation (7)) as for Adam, which was partially developed from the former. Whereas Adam utilises a moving average of the gradients, RMSProp uses the current gradient, dw , directly. RMSProp also uses the learning rate, α , the moving average of squared gradients, v_{dw} , and a small scalar

value, ϵ , to update the current weight, w_t , to give a new weight, w_{t+1} . Success has been shown with RMSProp in time series forecasting [33]. Furthermore, RMSProp is affected more strongly by the weight change gradient compared to Adam, which introduces a notable point of difference for testing different variations of the optimiser algorithm.

$$w_{t+1} = w_t - \frac{\alpha \cdot dw}{\sqrt{v_{dw} + \epsilon}} \quad (7)$$

SGD weight updates (Equation (8)) operate under a simpler mechanism than the aforementioned optimiser algorithms and subtract the product of the learning rate, α , and the gradient, g_t , from the current weight, w_t , to give a new weight, w_{t+1} . We used mini-batch SGD, which uses a small batch of consecutive data points, selected randomly, to calculate the gradient, g_t . This can aid the model to avoid local minima, which are optimum weight values compared to nearby values but not the most optimal overall. SGD is simple and efficient, and has been used by researchers copiously since its introduction [34].

$$w_{t+1} = w_t - \alpha \cdot g_t \quad (8)$$

The intricate relationships between these hyperparameters meant that testing every possible combination was vital to maximising the forecasting accuracy and reliability of our LSTM models. Therefore, cross-validation was used to examine four hyperparameters, in every possible configuration, as given below, giving a total of $3^4 = 81$ experiments for each model (Table 1).

Table 1. Variations tested for each cross-validated hyperparameter.

Hyperparameter	Variation 1	Variation 2	Variation 3
Hidden Layer Dimension	100	200	400
Learning Rate	1.0×10^{-3}	1.0×10^{-4}	1.0×10^{-5}
Dropout Probability	0.2	0.5	0.8
Optimiser Algorithm	Adam	RMSProp	SGD

This section of work focussed on finding good forecasting methods for load, solar generation and wind generation, in terms of both hyperparameter combination and LSTM design and architecture. All five exogenous variables given in Section 2.1 were used during these experiments. This meant that the LSTM neural network had six input dimensions (one endogenous target variable and five exogenous weather variables), for the work given in Section 3, and variable dimensions for other experiments. For every LSTM model and hyperparameter combination, load, solar generation and wind generation were all forecast.

The following models were trialled for each hyperparameter cross-validation and target variable forecasting:

- Model 1 (Model 1);
- Model 1 + bidirectional LSTM (Model 1, bi);
- Model 1 + gradient clipping (Model 1, gc);
- Model 1 + bidirectional LSTM + gradient clipping (Model 1, bi, gc);
- Model 1 + gradient clipping + 4 LSTM block layers (Model 1, gc, nl4);
- Model 2 (Model 2);
- Model 2 + bidirectional LSTM (Model 2, bi);
- Model 2 + gradient clipping (Model 2, gc);

Bidirectional LSTMs allow data from the past and present to be analysed, by training the model in the backward direction as well as the forward, allowing more context to be captured for predictions [35]. Good results have been observed for comparable one-hour-ahead wind power forecasting using bidirectional LSTMs [36], as the effect of propagated errors is reduced with this additional functionality. Gradient norm clipping with a cap of $\|1\|$ for gradients was used, as this technique has been shown to reduce negative impacts

from exploding gradients [37]. Although the number of layers within the LSTM block was found to be optimum with the number of layers = 2, it was believed that with gradient clipping, the increased number of layers may be more likely to have a positive impact due to avoiding exploding gradients that can hamper larger architectures, and so four layers were used in some models. Our results show significant differences in accuracy between architectures (Figure 5).

For load forecasting, the Model 2 architecture with bidirectional functionality and an RMSProp optimiser gave the lowest MASE of 0.44 (Figure 5). Adding complexity, whether in the number of layers within an LSTM block, or the number of blocks themselves, decreased test loss and MASE, albeit with an increase in forecast bias. A greater diversity of possible pathways through the neural network likely allowed multiple ‘perspectives’ to be used for predictions, and allowed for greater comprehension, in terms of the weight matrices, of the data features. This includes more abstract data features that are only available at higher levels of representation [38].

For solar and wind generation forecasting, the simpler Model 1 architecture (Figure 3) yielded lower MASE of 0.38 and 0.84, respectively (Figure 5). This could be due to the model overfitting the data, especially for less predictable wind fluctuations. Deeper models can be difficult to train and the Model 2 architecture (Figure 4) may require larger amounts of data to provide better accuracies.

Whilst learning rates of 1×10^{-3} and 1×10^{-4} gave lower test losses and MASE values (Appendix A Table A1), it is noted that if training on a larger dataset, lower learning rates may improve post-training weight matrices. Whilst dropout proportions of 0.8 were tested, these did not capture sufficient information to benefit models. The non-adaptive SGD optimiser was also not able to outperform adaptive optimisers that can negotiate the gradient landscape better.

Forecast biases were small compared to the large values in the dataset, which is promising. Care should be taken to consider these though, as they could have ramifications for energy management. For example, underestimating load through a negative forecast bias could lead to power shortages.

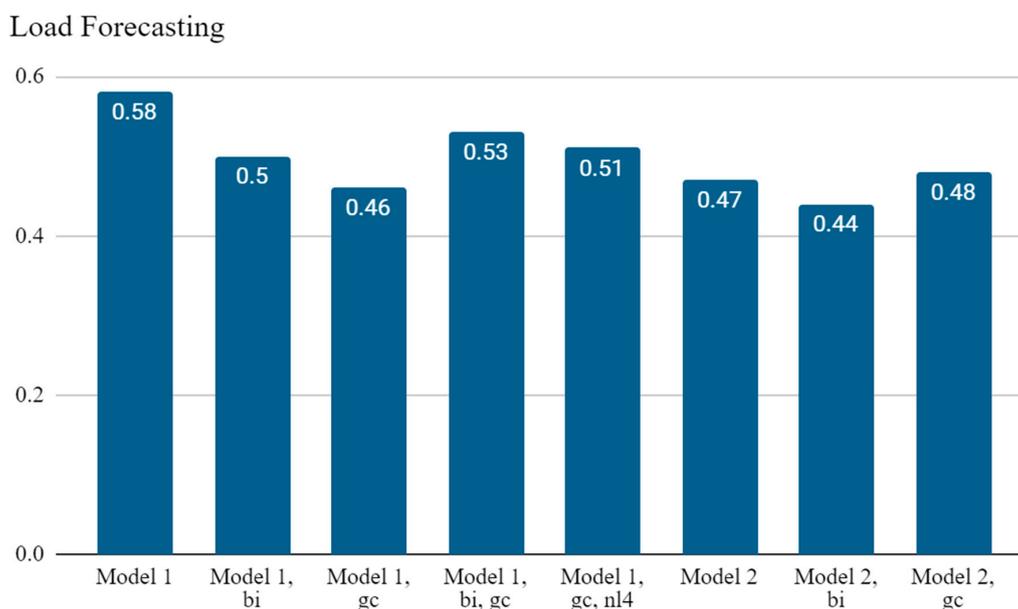


Figure 5. Cont.

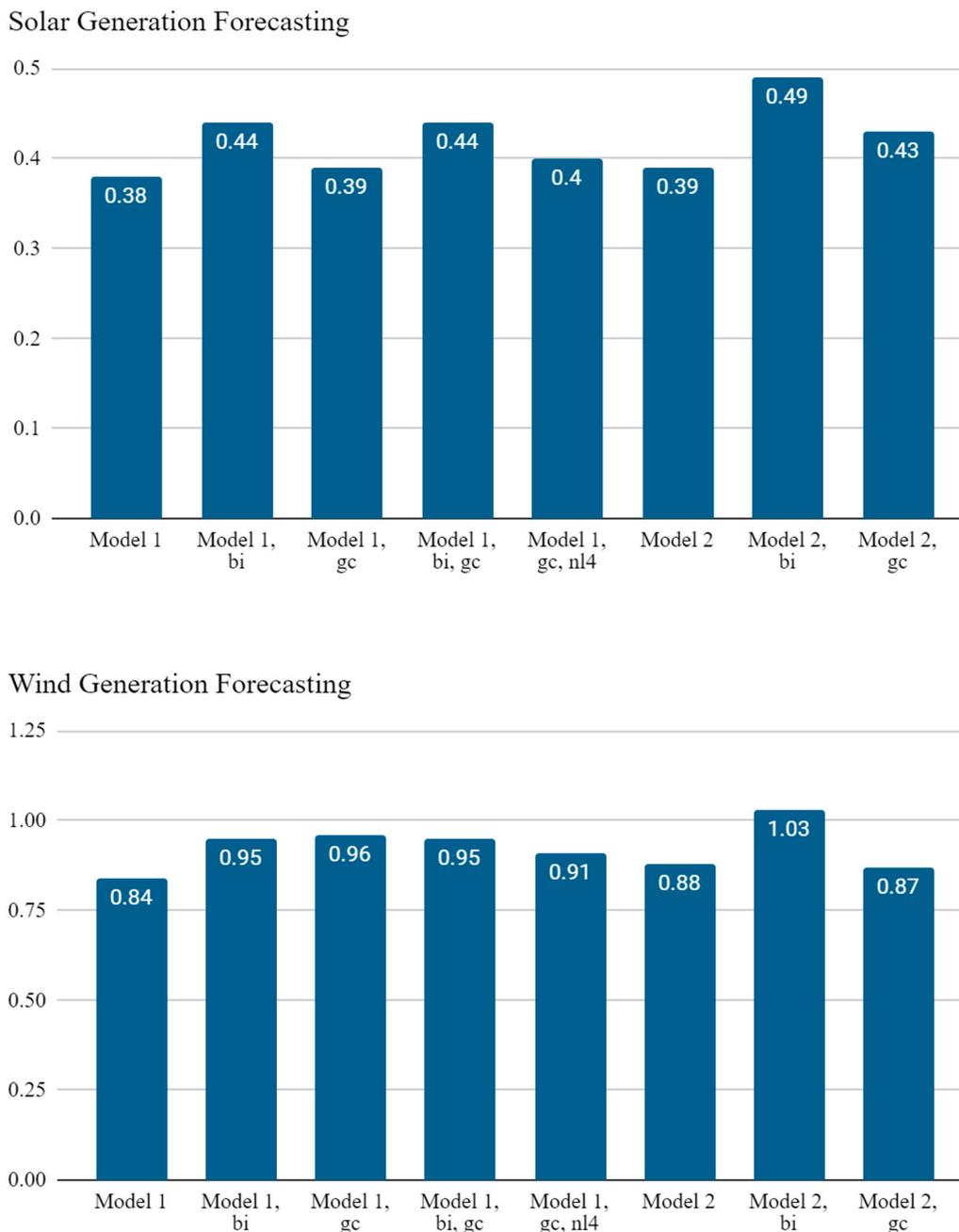


Figure 5. The best observed MASE (y -axis) found for each LSTM architecture, for load, solar and wind generation forecasting. All exogenous weather variables were used for each of these experiments. Each given MASE represents the lowest value found out of all hyperparameter combinations tested. See Appendix A Table A1 for optimum hyperparameter combinations and the corresponding test loss and forecast bias values.

4. Exogenous Variable Analysis

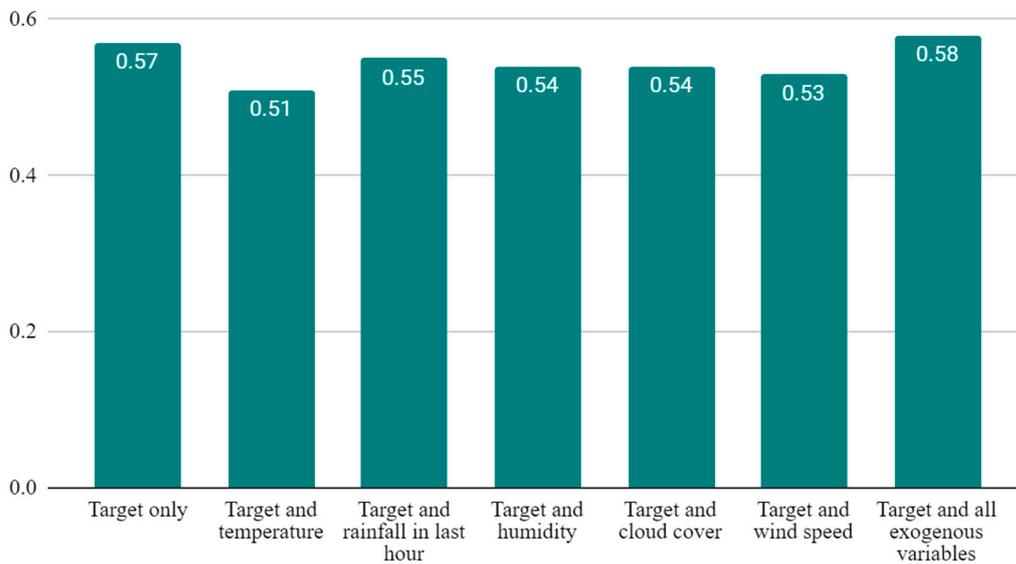
In this section we study the impact of individual exogenous variables, as well as the effect of using no exogenous variables at all, to demonstrate the improvement in accuracy metrics that these additional variables provide. In the foundational work set out in Section 3, all five exogenous variables were used. However, it is important to understand how these variables impact the forecasting of load, solar generation and wind generation independently. It is possible that some variables may add unnecessary noise without contributing to effective learning, whereas others may provide crucial cues for the neural network.

In the next stage of our investigation, hyperparameter combinations were still tested through cross-validation; however, the SGD optimiser function, a learning rate of 1×10^{-5} , and a dropout rate of 0.8 was discontinued due to consistently low performance.

Model 1 was used, with no additional design features, to provide greater interpretability of results and fewer contributing factors to dilute result findings. Using Model 1 also meant less risk of more complex models conflating irrelevant patterns with important ones and incorporating these into weight matrices.

As can be seen in Figure 6, all individual exogenous variables improved the MASE for load forecasting, although when used as a full suite, this slightly degraded performance, which could be attributed to conflicting residual components between the time series, or simply a greater level of complexity. Adding temperature, in isolation, had a significant impact. Many temperature control appliances have high power consumption, so this is understandable.

Load Forecasting



Solar Generation Forecasting

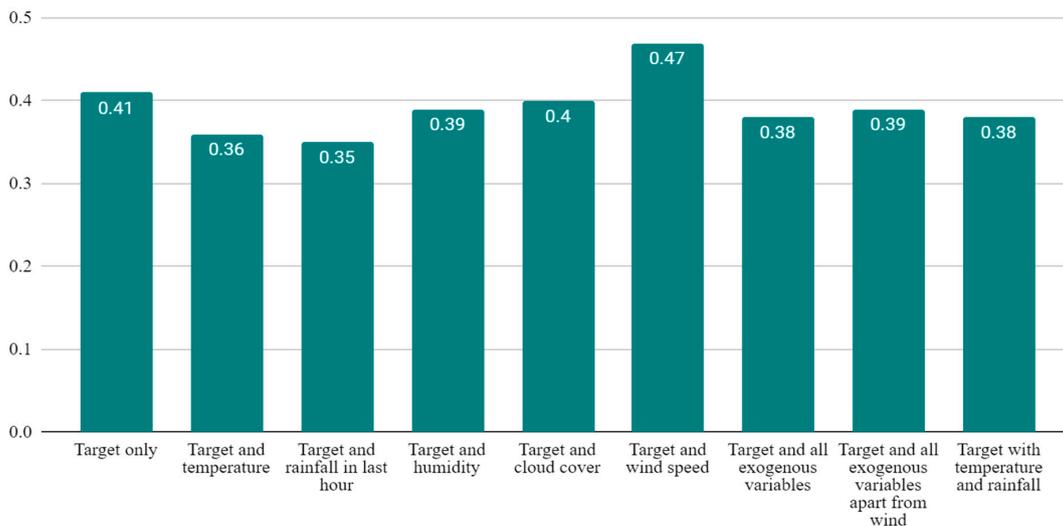


Figure 6. Cont.

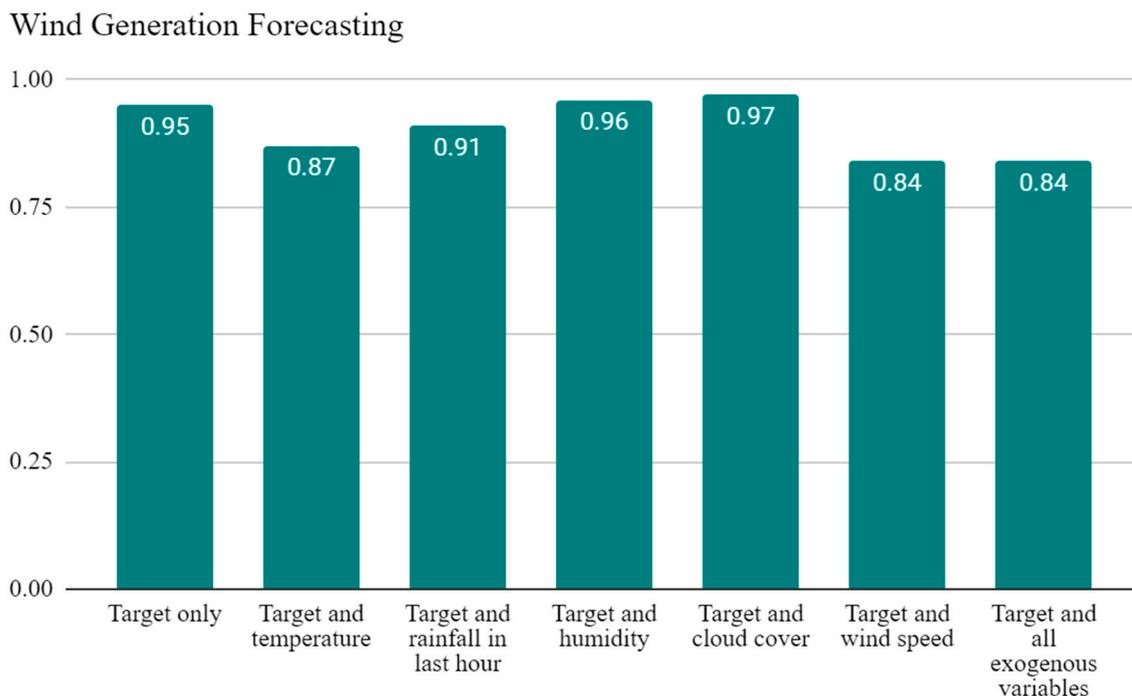


Figure 6. The best observed MASE (y -axis) found for each exogenous variable combination, for load, solar and wind generation forecasting. Each given MASE represents the lowest value found out of all hyperparameter combinations, which were cross-validated for each exogenous variable combination. See Appendix A Table A2 for optimum hyperparameter combinations and the corresponding test loss and forecast bias values.

Rain and temperature, when used as single additional exogenous variables, significantly decreased the MASE and test loss for solar generation (see Figure 7). It was surprising that cloud cover did not reduce the MASE and test loss by a greater amount. It has been shown that on cloudy days, solar generation forecasting typically decreases in accuracy, and so despite having a large impact on generation, it is possible that noise from clouds is difficult to correlate with solar generation fluctuations for the LSTM [39].

None of the individual exogenous variable combinations improved accuracy metrics for wind power generation against the experiments detailed in Section 3. Wind speed in isolation as an exogenous variable, or all-weather variables together, did produce lower MASE values compared to forecasting wind generation as a target variable with no exogenous variables, which is expected as wind speed correlates very strongly to wind power generation. Wind power generation proved the most challenging to forecast, and it is suggested that this is due to the weaker seasonal and cyclic components of this variable.

Overall, significant improvements were observed for several independent or combined weather variables being added to the LSTM as exogenous variables.

The main highlights from this section of work, with comparisons being made against the target variable being forecast in isolation, are as follows:

- ~10% reduction in MASE and a ~14% decrease in test loss when load was forecast with an exogenous temperature variable (Appendix A Table A2).
- ~15% and ~12% reductions in MASE, ~30% and ~33% reduction in test loss, for solar generation forecasting when paired with rainfall and temperature, respectively (Appendix A Table A2).
- ~12% reductions in MASE, ~6% and ~9% reductions in test loss, for wind generation combined with wind speed and all five variables, respectively (Appendix A Table A2).

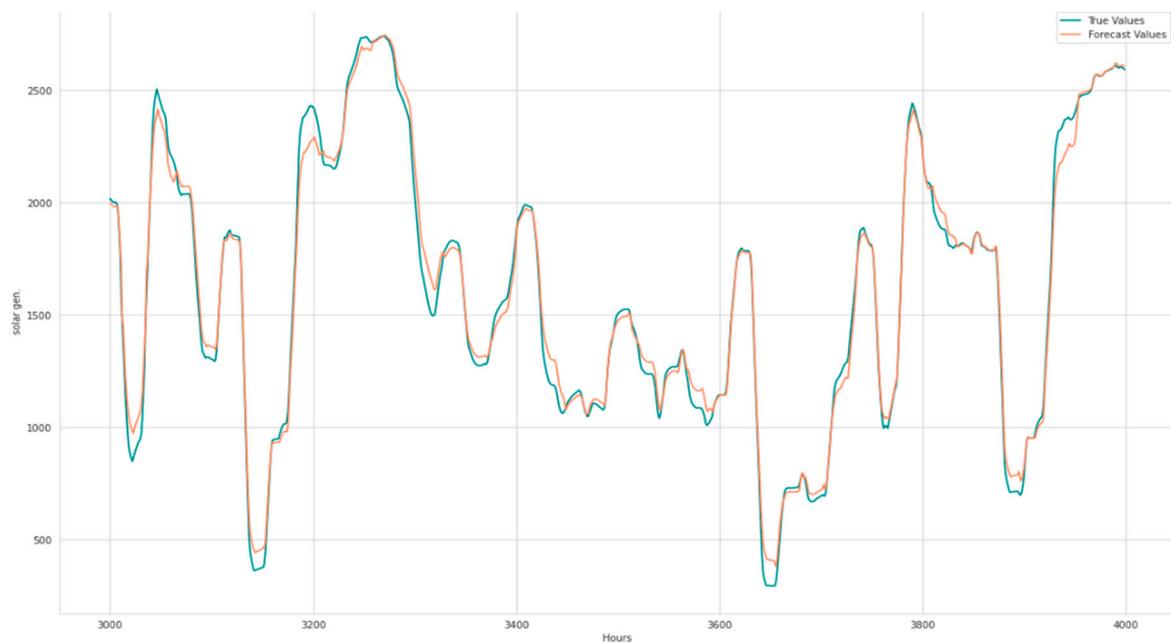


Figure 7. Forecast (orange) vs. true (blue) solar generation values, using the Model 1 LSTM and rainfall as a single exogenous variable, over a 1000 h period, which was randomly sampled from the test dataset, of length = 8760.

While it may seem obvious that a variable like wind speed would have an impact on wind generation, for example, the important finding here is that the history of past wind speeds, and the temporal patterns provided by this data in combination with past wind generation amounts, provides additional context, less susceptible to outliers, to give a greater level of forecasting accuracy than the wind generation itself. This same logic applies to all other experiments in this section. It is suggested that by allowing the LSTM to process weather conditions, within the hidden state, this increased context allowed more accurate load or generation forecasting. Figure 7 illustrates, as an example, the efficacy of our forecasting model.

5. Seasonal Decomposition as a Source of Exogenous LSTM Input

In this section, we present the results of our investigation using seasonally decomposed components as exogenous variables.

Seasonal decomposition is an attractive forecasting tool for energy-related series. This is because these data often show strong annual and daily cycles. It also includes subtle, long-term linear trends. Seasonal decomposition methods often work well as part of an ensemble, as they may not be able to forecast noise as effectively as other techniques. This has been shown to be effective with load forecasting [40].

Target time series were decomposed, and the seasonal component was then used as an exogenous variable for the LSTM, giving two inputs: the normalised target variable and the normalised seasonal component of that target variable. We did not use any data from the test dataset when training the model and the model was trained using the corresponding seasonal component of the training dataset only. The results of these experiments (Figure 8) were compared against the corresponding results for the target variable being forecast in isolation with no exogenous variables.

Forecasting with Residual Components

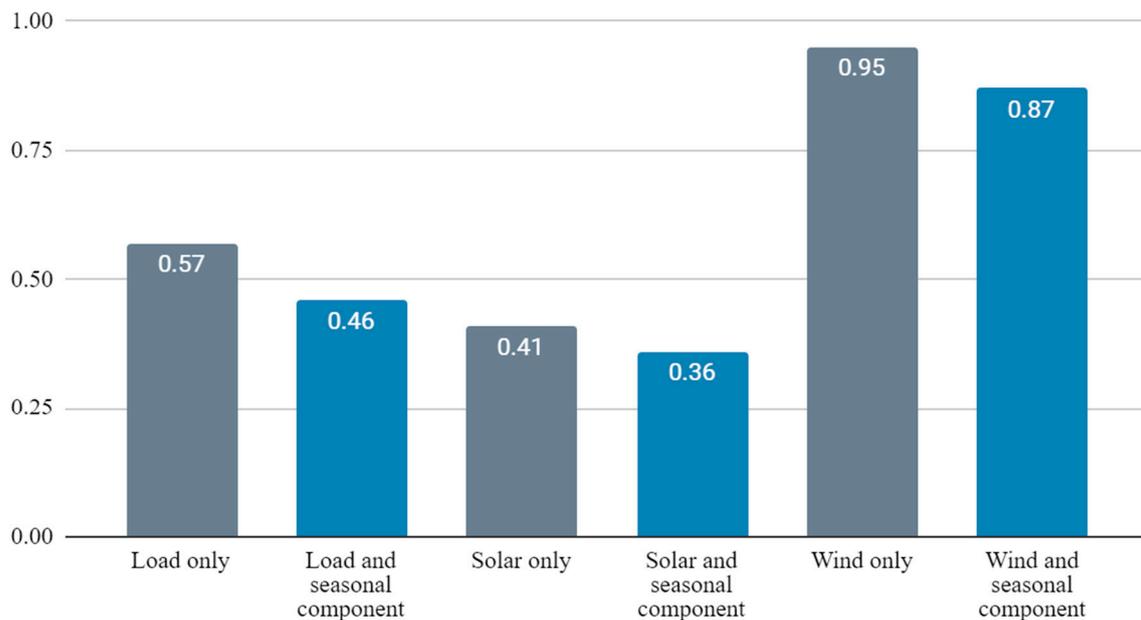


Figure 8. Forecasting each target variable without (grey) and with (blue) its seasonal component used as an exogenous variable. MASE values (y -axis) for optimum hyperparameter combinations are given. See Appendix A Table A3 for hyperparameter combinations and the corresponding test loss and forecast bias values.

Comparisons against the target variable being forecast in isolation, for the target variables forecast with their seasonal component as an exogenous variable are as follows (see Appendix A Table A3):

- ~19% reduction in MASE and ~29% reduction in test loss for load forecasting.
- ~12% reduction in MASE and ~19% reduction in test loss for solar generation forecasting.
- ~8% reduction in MASE and test loss for wind generation forecasting.
- ~12% reduction in MASE and a ~31% reduction in test loss for solar generation forecast with the seasonal component and temperature, all normalised.

The results from these experiments indicate that using a seasonal component of a time series alongside the target variable improves results dramatically. These improvements are not wholly surprising, since capturing the seasonal component of a time series would account for the greatest proportion of its absolute value.

Caution should be exercised when evaluating these findings though, since this method is essentially feeding a component of the time series into itself again. On the other hand, this method might find more niche applications in training neural networks on energy time series data, almost like ‘training wheels’ on a bike, perhaps during only a subset of early epochs. By using the seasonal component as an exogenous variable, it was hoped that this reinforcement would guide gradient descent closer to the global minimum. Since uniformly significant reductions in MASE and test loss were observed in all such experiments, it is proposed that in this regard, the combination of seasonal decomposition and neural network application was a success.

6. Conclusions

In this paper, we developed a framework for forecasting load, solar and wind generation using LSTM and five weather variables, i.e., temperature, rainfall, humidity, cloud cover and wind speed. As part of the framework, we proposed a methodology for tuning the LSTM design and hyperparameter optimisation. Adaptive optimiser functions (Adam,

RMSProp) have been shown to be more effective at dealing with complex time series compared to the non-adaptive SGD function. The importance of exogenous variable selection has been clearly highlighted within a time series forecasting landscape that perhaps does not fully take advantage of such considerations. The temperature has been found to have a significant impact on load; temperature and rainfall have been found to have a significant impact on solar generation; and temperature and wind speed have been found to have a significant impact on wind generation forecasting. It is proposed that the set of temperature, rainfall, wind speed, humidity and cloud cover offer a promising suite of exogenous variables that deserve further research. The use of seasonal components as exogenous variables also yielded promising results, with reductions in MASE observed for all three target variables.

The adoption of a three-block LSTM configuration with bidirectional functionality, as opposed to a single block, reduced MASE by 24% for load forecasting. Using temperature as an exogenous variable, as opposed to using the target variable in isolation for load forecasting, decreased the MASE by 10%. Whilst not directly comparable to the TSO forecasts, which were 24-h-ahead, our solar generation forecasts improved solar generation forecasts from this company by up to 22%. The use of exogenous weather variables to improve accuracy for solar generation decreased MSE test loss by up to 33% and MASE by approximately 15%. The use of the seasonal component as an exogenous variable reduced MASE for load, solar generation and wind generation by 19%, 12% and 8%, respectively. These improvements represent a lower average error margin and forecasts that are closer to the true values, which could provide several benefits to energy management operators and researchers.

Further work on extending the future time period through which the model can forecast, for example, a 24-h-ahead model, in one-hour increments, would be beneficial. Whilst this is not within the primary scope of our research, it would lessen this current constraint.

It is recommended that further research is conducted on the subject of data generation processes surrounding weather data, for use in energy-related time series forecasting. The collection of regularly recorded, hourly data on a range of weather metrics in each country, and the corresponding public access to this, could significantly improve time series forecasting outcomes. The study of the impact of other environmental variables such as solar zenith angle and air pressure would also add value to our research.

Furthermore, research into solely forecasting the residual component of a seasonally decomposed target variable, such as load, solar or wind generation, and then combining this forecast with algorithmically predicted seasonal and trend components, could yield even higher accuracies, since neural networks like LSTMs excel at deciphering more complex temporal patterns. Ding et al. [17] have shown that combining data decomposition and LSTMs provides excellent generalisability of data, allowing accurate predictions. This approach, combined with an explorative incorporation of exogenous weather variables, could improve such models further.

It is suggested that the findings in this report provide indications that the techniques investigated have the potential to significantly improve time series forecasting models within this area of research. If certain findings in this report, such as the use of specific exogenous variables with target variables, or using seasonal components as exogenous variables, could be combined with refined LSTM models produced by other researchers, the results could provide significant benefit to the research community and also the further adoption of renewable energy in the world.

Author Contributions: Conceptualization, T.S., E.A. and D.A.; Methodology, T.S., E.A. and D.A.; Software, T.S.; Validation, T.S., E.A. and D.A.; Data curation, T.S.; Writing—original draft, T.S.; Writing—review & editing, E.A., T.S. and D.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding authors.

Conflicts of Interest: Author Dimitra Apostolopoulou was employed by the company Oxford Institute for Energy Studies. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Appendix A

Table A1. The best-performing hyperparameter suites for each model design, for each target variable. Abbreviations or alternative nomenclature used: hd = number of units in the hidden dimension, lr = learning rate, drop. = LSTM dropout probability, opt. = optimiser algorithm, test_loss = test loss, mase = MASE, bias = forecast bias, $7.1 \times 10^{-4} = 7.1 \times 10^{-4}$ (for example), bi = bidirectional, gc = gradient clipped, nl4 = number of layers in LSTM blocks increased from 2 to 4.

Target Variable: Load	hd	lr (10^{-3})	Drop.	Opt.	Test Loss (10^{-3})	MASE	Bias (10^{-3})
Model 1	100	1	0.2	Adam	1.3	0.58	-0.75
Model 1, bi	100	1	0.2	RMSProp	1.0	0.5	-6.6
Model 1, gc	200	1	0.2	Adam	0.86	0.46	1.0
Model 1, bi, gc	100	1	0.5	Adam	1.2	0.53	-7.6
Model 1, gc, nl4	400	0.1	0.2	RMSProp	1.1	0.51	-3.5
Model 2	400	0.1	0.2	Adam	0.93	0.47	-1.3
Model 2, bi	200	1	0.2	Adam	0.84	0.44	-7.2
Model 2, gc	100	0.1	0.5	Adam	0.90	0.48	3.1
Target Variable: Solar Generation	hd	lr (10^{-3})	drop.	opt.	Test loss (10^{-3})	MASE	bias (10^{-3})
Model 1	100	1	0.2	Adam	0.96	0.38	-1.8
Model 1, bi	400	0.1	0.2	RMSProp	1.0	0.44	-7.9
Model 1, gc	100	1	0.5	Adam	0.99	0.39	-2.6
Model 1, bi, gc	100	1	0.5	RMSProp	1.0	0.44	-12
Model 1, gc, nl4	200	0.1	0.2	RMSProp	0.96	0.4	-8.0
Model 2	400	0.1	0.2	Adam	0.84	0.39	7.1
Model 2, bi	100	0.1	0.2	RMSProp	1.4	0.49	-1.6
Model 2, gc	400	0.1	0.2	Adam	0.86	0.43	13
Target Variable: Wind Generation	hd	lr (10^{-3})	Drop.	Opt.	Test Loss (10^{-3})	MASE	Bias (10^{-3})
Model 1	200	1	0.2	Adam	1.2	0.84	-0.47
Model 1, bi	200	0.1	0.8	RMSProp	1.3	0.95	0.55
Model 1, gc	400	0.01	0.5	Adam	1.3	0.96	0.051
Model 1, bi, gc	200	0.1	0.8	RMSProp	1.3	0.95	1.1
Model 1, gc, nl4	200	0.1	0.5	RMSProp	1.3	0.91	-1.5
Model 2	400	0.1	0.2	RMSProp	1.3	0.88	-1.7
Model 2, bi	100	0.1	0.2	RMSProp	1.3	1.03	-2.9
Model 2, gc	200	0.1	0.2	RMSProp	1.3	0.87	-0.20

Table A2. Different exogenous variable combinations, for all target variables, with corresponding hyperparameters for the experiment with best MASE. For rows denoted with a ‘*’ symbol, the second lowest MASE was chosen due to instability in validation loss over epochs for the hyperparameter combination with the lowest MASE.

Target Variable: Load	hd	lr (10^{-3})	Drop.	Opt.	Test Loss (10^{-3})	MASE	Bias (10^{-3})
Target only	200	1	0.5	RMSProp	1.2	0.57	2.8
Target and temperature	200	1	0.2	Adam	1.1	0.51	-0.55
Target and rainfall in last hour	200	1	0.2	RMSProp	1.2	0.55	-7.6
Target and humidity	200	1	0.2	RMSProp	1.2	0.54	-3.5
Target and cloud cover	200	1	0.5	RMSProp	1.2	0.54	-8.1
Target and wind speed	400	1	0.2	RMSProp	1.1	0.53	-7.0
Target and all exogenous variables	100	1	0.2	Adam	1.3	0.58	0.75
Target Variable: Solar Generation	hd	lr (10^{-3})	Drop.	Opt.	Test Loss (10^{-3})	MASE	Bias (10^{-3})
Target only	400	1.0	0.5	RMSProp	1.2	0.41	-13
Target and temperature	100	1.0	0.2	Adam	0.80	0.36	1.9
Target and rainfall in last hour	200	0.10	0.2	Adam	0.89	0.35	1.7
Target and humidity	100	1.0	0.2	RMSProp	1.1	0.39	-13
Target and cloud cover	100	1.0	0.2	RMSProp	1.2	0.40	-14
Target and wind speed *	100	1.0	0.5	RMSProp	1.3	0.47	-12
Target and all exogenous variables	100	1.0	0.2	Adam	0.96	0.38	-1.8
Target and all exogenous variables apart from wind	100	1.0	0.2	Adam	0.99	0.39	-4.9
Target with temperature and rainfall	100	1.0	0.2	Adam	0.93	0.38	-2.4

Table A2. Cont.

Target Variable: Wind Generation	hd	lr (10^{-3})	Drop.	Opt.	Test Loss (10^{-3})	MASE	Bias (10^{-3})
Target only	100	1	0.2	Adam	1.3	0.95	-7.7
Target and temperature	100	1	0.5	Adam	1.2	0.87	2.4
Target and rainfall in last hour	100	1	0.5	RMSProp	1.3	0.91	-5.9
Target and humidity *	200	1	0.2	Adam	1.3	0.96	-5.8
Target and cloud cover	200	1	0.2	RMSProp	1.4	0.97	-9.3
Target and wind speed	100	1	0.2	Adam	1.2	0.84	-4.3
Target and all exogenous variables	200	1	0.2	Adam	1.2	0.84	-0.47

Table A3. Experiments with the lowest MASE, using the seasonal component of the target variable as an exogenous variable for the LSTM.

Input Variables	hd	lr (10^{-3})	Drop.	opt.	Test loss (10^{-3})	MASE	Bias (10^{-3})
Load only	200	1	0.5	RMSProp	1.2	0.57	2.8
Load and seasonal component	200	1	0.2	Adam	0.85	0.46	7.2
Solar only	400	1	0.5	RMSProp	1.2	0.41	-13
Solar and seasonal component	200	0.1	0.2	Adam	0.97	0.36	0.12
Wind only	100	1	0.2	Adam	1.3	0.95	-7.7
Wind and seasonal component	100	1	0.2	Adam	1.2	0.87	-6.5

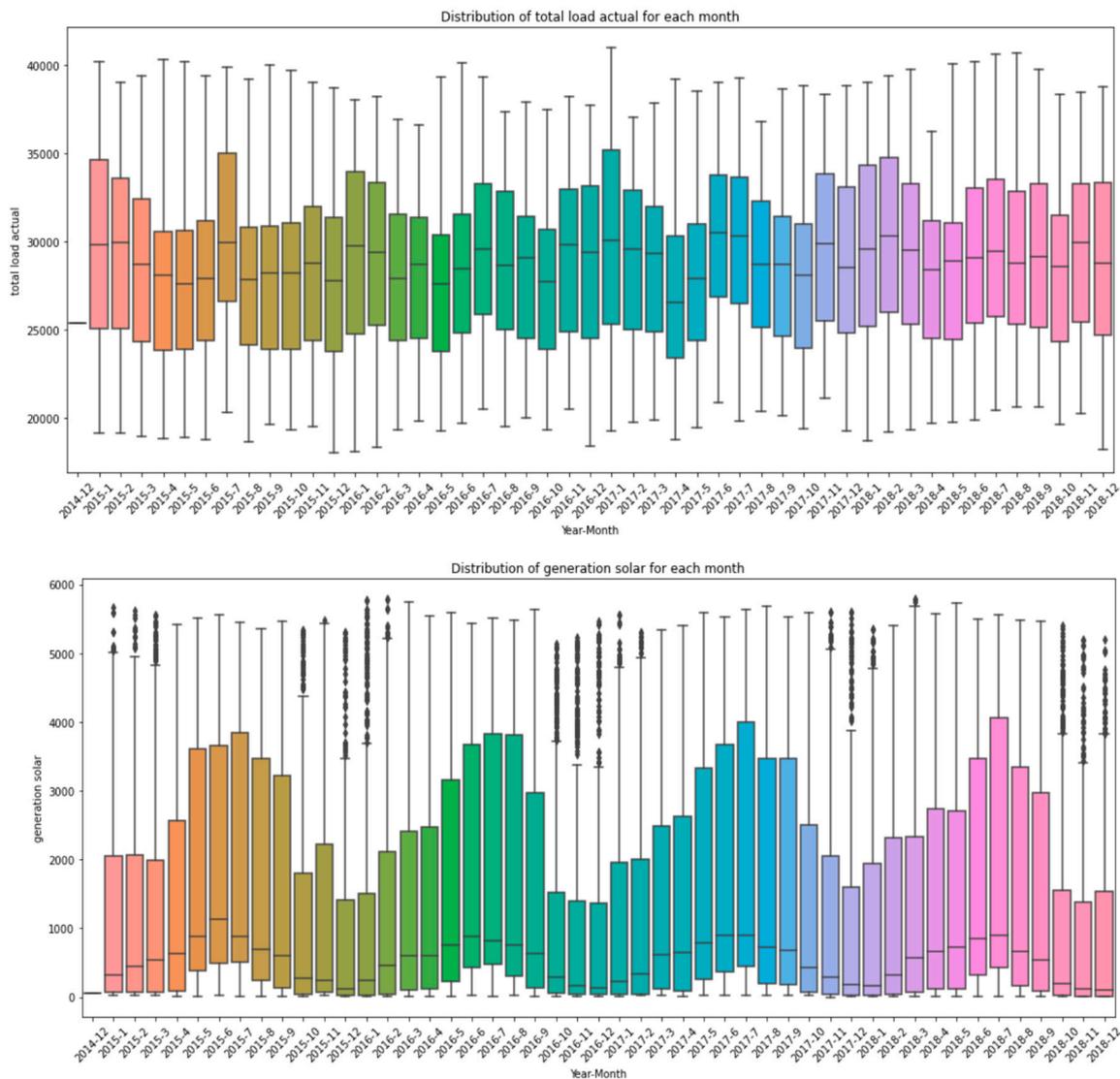


Figure A1. Cont.

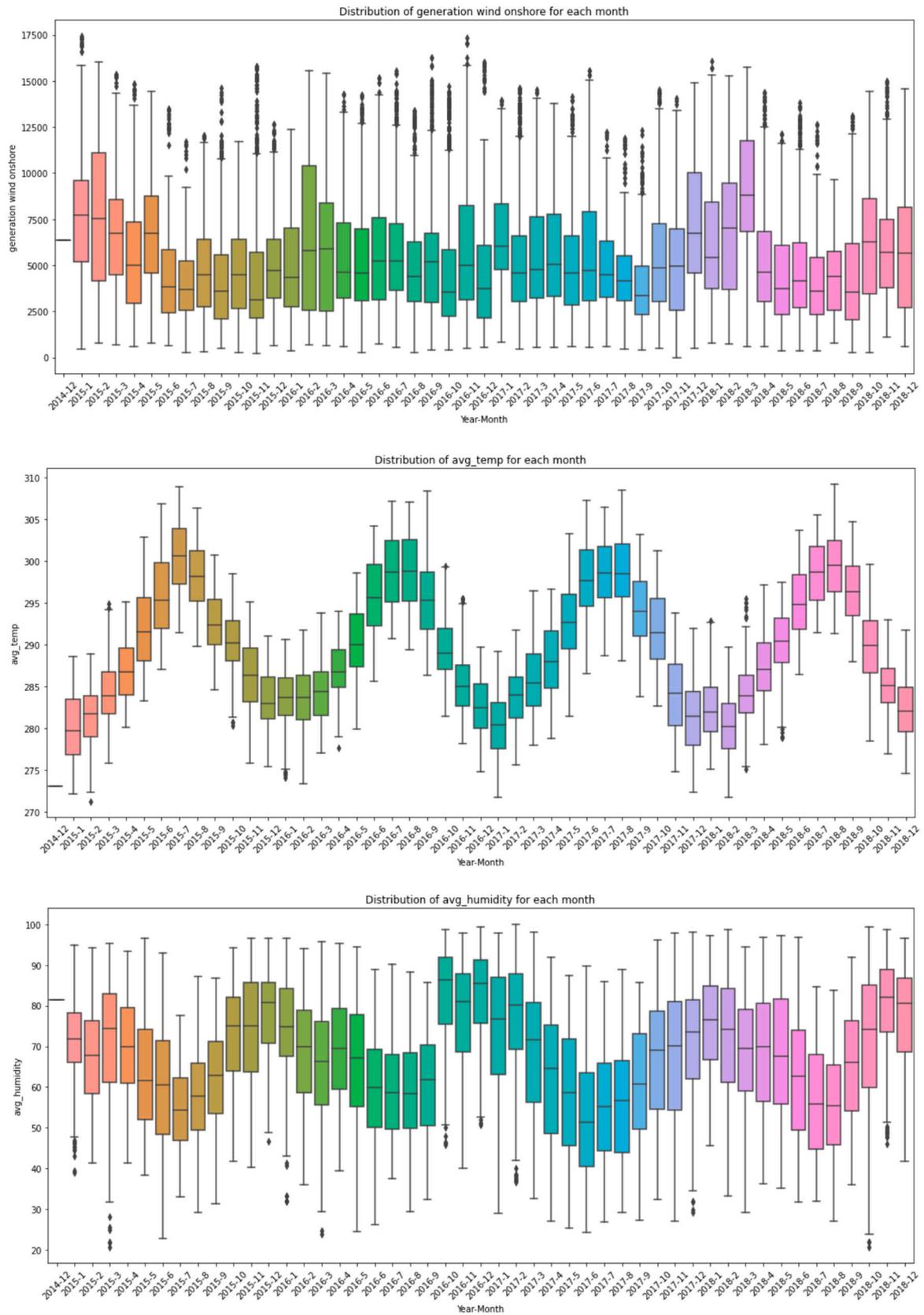


Figure A1. Cont.

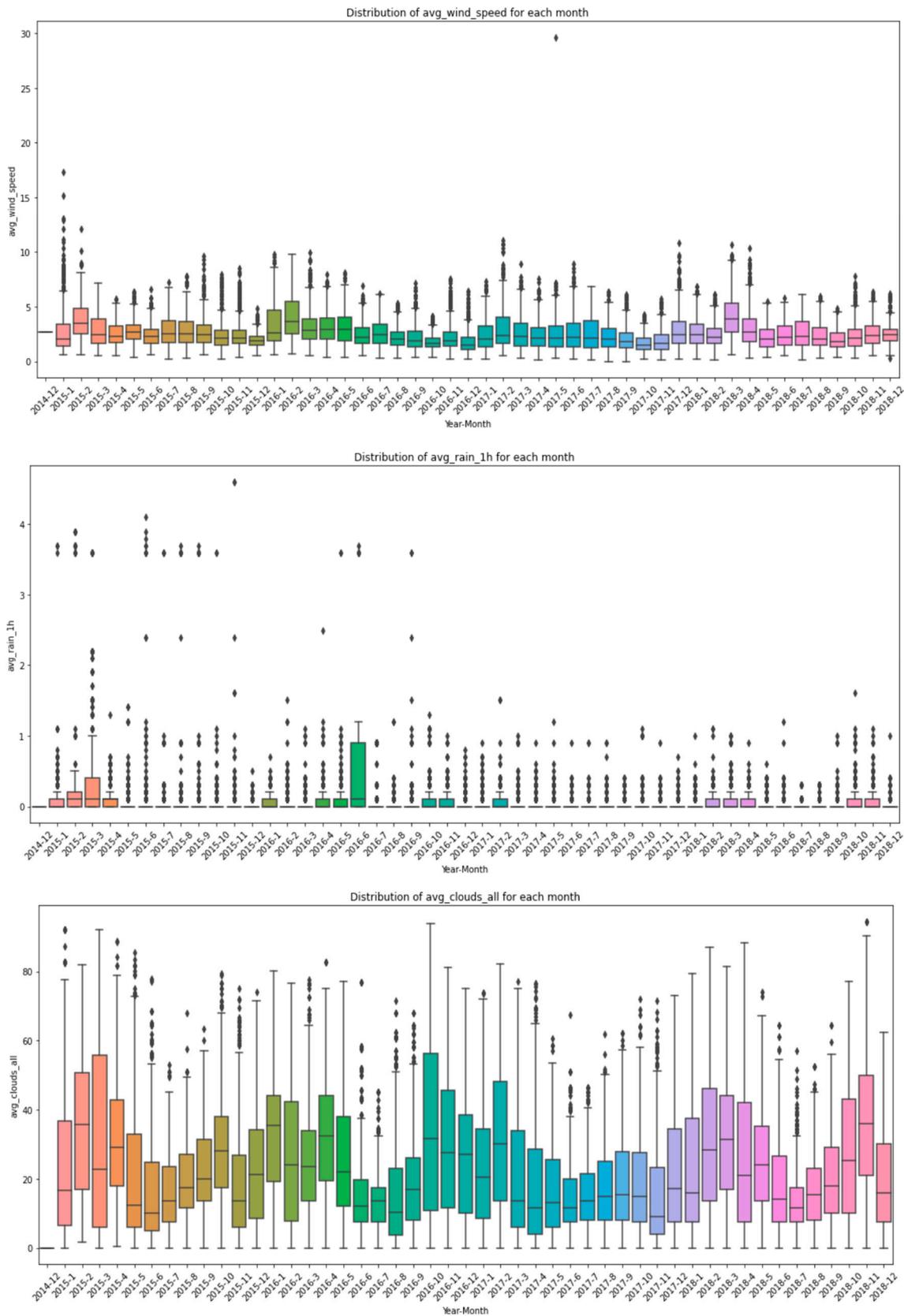


Figure A1. Boxplot graphs showing the distribution of the three target variables; load, solar generation (generation solar) and wind generation (wind onshore), and the 5 exogenous weather variables used; temperature (avg_temp), humidity (avg_humidity), wind speed (avg_wind_speed), rainfall in the last hour (avg_rain_1 h) and cloud cover (avg_clouds_all). Black diamonds represent outliers.

References

1. Mbuli, N.; Mathonsi, M.; Seitshiro, M.; Pretorius, J.H.C. Decomposition forecasting methods: A review of applications in power systems. *Energy Rep.* **2020**, *6*, 298–306. [[CrossRef](#)]
2. Dudek, G. Pattern-based local linear regression models for short-term load forecasting. *Electr. Power Syst. Res.* **2016**, *130*, 139–147. [[CrossRef](#)]
3. Ariyo, A.A.; Adewumi, A.O.; Ayo, C.K. Stock Price Prediction Using the ARIMA Model. In Proceedings of the 2014 UK Sim-AMSS 16th International Conference on Computer Modelling and Simulation, Cambridge, UK, 26–28 March 2014; pp. 106–112. [[CrossRef](#)]
4. Wang, Y.; Sun, S.; Chen, X.; Zeng, X.; Kong, Y.; Chen, J.; Guo, Y.; Wang, T. Short-term load forecasting of industrial customers based on SVM and XGBoost. *Int. J. Electr. Power Energy Syst.* **2021**, *129*, 106830. [[CrossRef](#)]
5. Elsaraiti, M.; Merabet, A. A Comparative Analysis of the ARIMA and LSTM Predictive Models and Their Effectiveness for Predicting Wind Speed. *Energies* **2021**, *14*, 6782. [[CrossRef](#)]
6. Tarmanini, C.; Sarma, N.; Gezegegn, C.; Ozgonenel, O. Short term load forecasting based on ARIMA and ANN approaches. *Energy Rep.* **2023**, *9* (Suppl. S3), 550–557. [[CrossRef](#)]
7. Markovics, D.; Mayer, M.J. Comparison of machine learning methods for photovoltaic power forecasting based on numerical weather prediction. *Renew. Sustain. Energy Rev.* **2022**, *161*, 112364. [[CrossRef](#)]
8. Rodríguez, F.; Fleetwood, A.; Galarza, A.; Fontán, L. Predicting solar energy generation through artificial neural networks using weather forecasts for microgrid control. *Renew. Energy* **2018**, *126*, 855–864. [[CrossRef](#)]
9. Wang, L.; Mao, M.; Xie, J.; Liao, Z.; Zhang, H.; Li, H. Accurate solar PV power prediction interval method based on frequency-domain decomposition and LSTM model. *Energy* **2023**, *262*, 125592. [[CrossRef](#)]
10. Elman, J.L. Finding structure in time. *Cogn. Sci.* **1990**, *14*, 179–211. [[CrossRef](#)]
11. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)]
12. Lu, S.; Zhu, Y.; Zhang, W.; Wang, J.; Yu, Y. Neural Text Generation: Past, Present and Beyond. *arXiv* **2018**, arXiv:1803.07133.
13. Elsaraiti, M.; Merabet, A. Solar Power Forecasting Using Deep Learning Techniques. *IEEE Access* **2022**, *10*, 31692–31698. [[CrossRef](#)]
14. Memarzadeh, G.; Keynia, F. A new short-term wind speed forecasting method based on fine-tuned LSTM neural network and optimal input sets. *Energy Convers. Manag.* **2020**, *213*, 112824. [[CrossRef](#)]
15. Torres, J.; Martínez-Álvarez, F.; Troncoso, A. A deep LSTM network for the Spanish electricity consumption forecasting. *Neural Comput. Appl.* **2022**, *34*, 10533–10545. [[CrossRef](#)] [[PubMed](#)]
16. Akhtar, S.; Shahzad, S.; Zaheer, A.; Ullah, H.S.; Kilic, H.; Gono, R.; Jasiński, M.; Leonowicz, Z. Short-Term Load Forecasting Models: A Review of Challenges, Progress, and the Road Ahead. *Energies* **2023**, *16*, 4060. [[CrossRef](#)]
17. Ding, S.; Zhang, H.; Tao, Z.; Li, R. Integrating data decomposition and machine learning methods: An empirical proposition and analysis for renewable energy generation forecasting. *Expert Syst. Appl.* **2022**, *204*, 117635. [[CrossRef](#)]
18. Kwon, B.-S.; Park, R.-J.; Song, K.-B. Short-Term Load Forecasting Based on Deep Neural Networks Using LSTM Layer. *J. Electr. Eng. Technol.* **2020**, *15*, 1501–1509. [[CrossRef](#)]
19. Machado, E.; Pinto, T.; Guedes, V.; Morais, H. Electrical Load Demand Forecasting Using Feed-Forward Neural Networks. *Energies* **2021**, *14*, 7644. [[CrossRef](#)]
20. Wojtkiewicz, J.; Hosseini, M.; Gottumukkala, R.; Chambers, T.L. Hour-Ahead Solar Irradiance Forecasting Using Multivariate Gated Recurrent Units. *Energies* **2019**, *12*, 4055. [[CrossRef](#)]
21. Xie, A.; Yang, H.; Chen, J.; Sheng, L.; Zhang, Q. A Short-Term Wind Speed Forecasting Model Based on a Multi-Variable Long Short-Term Memory Network. *Atmosphere* **2021**, *12*, 651. [[CrossRef](#)]
22. Jhana, N. Hourly Energy Demand Generation and Weather. Available online: <https://www.kaggle.com/datasets/nicholasjhana/energy-consumption-generation-prices-and-weather> (accessed on 12 July 2023).
23. ENTSO-E. Transparency Platform 2019. Available online: <https://transparency.entsoe.eu/dashboard/show> (accessed on 12 July 2023).
24. OpenWeather API. 2019. Available online: <https://openweathermap.org/api> (accessed on 12 July 2023).
25. Lin, J.; Ma, J.; Zhu, J.; Cui, Y. Short-term load forecasting based on LSTM networks considering attention mechanism. *Int. J. Electr. Power Energy Syst.* **2021**, *137*, 107818. [[CrossRef](#)]
26. Abdel-Nasser, M.; Mahmoud, K. Accurate photovoltaic power forecasting models using deep LSTM-RNN. *Neural Comput. Appl.* **2019**, *31*, 2727–2740. [[CrossRef](#)]
27. Mellit, A.; Pavan, A.M.; Lughi, V. Deep learning neural networks for short-term photovoltaic power forecasting. *Renew Energy* **2021**, *172*, 276–288. [[CrossRef](#)]
28. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Sardinia, Italy, 13–15 May 2010; pp. 249–256.
29. Statsmodel Documentation for the Seasonal_Decompose Function. Available online: https://www.statsmodels.org/dev/generated/statsmodels.tsa.seasonal.seasonal_decompose.html (accessed on 20 August 2023).
30. Hyndman, R.J.; Koehler, A.B. Another look at measures of forecast accuracy. *Int. J. Forecast.* **2006**, *22*, 679–688. [[CrossRef](#)]
31. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
32. Hinton, G. Coursera Neural Networks for Machine Learning, Lecture 6. 2018. Available online: https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf (accessed on 29 January 2024).

33. Sachdeva, A.; Jethwani, G.; Manjunath, C.; Balamurugan, M.; Krishna, A.V.N. An Effective Time Series Analysis for Equity Market Prediction Using Deep Learning Model. In Proceedings of the International Conference on Data Science and Communication (IconDSC), Bangalore, India, 1–2 March 2019; pp. 1–5. [[CrossRef](#)]
34. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2016**, arXiv:1609.04747.
35. Schuster, M.; Paliwal, K.K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **1997**, *45*, 2673–2681. [[CrossRef](#)]
36. Ko, M.-S.; Lee, K.; Kim, J.-K.; Hong, C.W.; Dong, Z.Y.; Hur, K. Deep Concatenated Residual Network With Bidirectional LSTM for One-Hour-Ahead Wind Power Forecasting. *IEEE Trans. Sustain. Energy* **2021**, *12*, 1321–1335. [[CrossRef](#)]
37. Pascanu, R.; Mikolov, T.; Bengio, Y. On the difficulty of training recurrent neural networks. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 7–19 June 2013; Volume 28, pp. 1310–1318.
38. Bengio, Y.; Courville, A.; Vincent, P. Representation Learning: A Review and New Perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1798–1828. [[CrossRef](#)]
39. Lim, S.C.; Huh, J.H.; Hong, S.K.; Park, C.Y.; Kim, J.C. Solar Power Forecasting Using CNN-LSTM Hybrid Model. *Energies* **2022**, *15*, 8233. [[CrossRef](#)]
40. Ribeiro, M.H.D.M.; da Silva, R.G.; Ribeiro, G.T.; Mariani, V.C.; Coelho, L.S. Cooperative ensemble learning model improves electric short-term load forecasting. *Chaos Solitons Fractals* **2023**, *166*, 112982. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.