MONASH University

APPLICATION OF ARTIFICIAL INTELLIGENT (AI) TECHNIQUES TO PREDICT THE PERFORMANCE OF SOLAR PV PLANTS

RIVYESCH RANJAN

Supervisor: Dr. Arshad Adam Salema

A Thesis
submitted in partial fulfillment of the requirements for the
Degree in Bachelor of Engineering (Mechanical)
Faculty of Engineering
Monash University

June 2021

Certificate of Originality

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any other degree or diploma at Monash University or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at Monash University or elsewhere is explicitly acknowledged in the thesis.

I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.

Rivyesch Ranjan

…………………………………………………………..

(Rivyesch Ranjan)

Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Arshad for the continuous support and many insights provided during the numerous consultation sessions over the past year. His assistance and guidance went a long way in helping me successfully complete this research project. I would also like to acknowledge Mr. Jai Mangal Singh for the guidance given in relation to the DL mode programming. Next, I would like to thank Monash for the education imparted over the last four years and also for providing the data used in this study.

Finally, my deep and sincere gratitude to my family for their continuous and unparalleled love, help and support. I am forever indebted to my father for giving me the opportunities and experiences that have made me who I am today.

Abstract

Problem: The widespread use of solar energy in the global power grid has facilitated the urgent need for accurate solar energy prediction models to minimise the negative impacts of photovoltaics (PV) on electricity and energy systems. Forecast models that can predict the PV output accurately will not only improve the performance of the system, but also aid operational decisions.

Objectives: The objective of this research is to develop an ANN and LSTM network to predict the performance of Monash University's grid-connected solar PV plants and then compare and analyse the predictive performance of the two AI models.

Methodology: Two models were proposed, namely a simplified Long Short-Term Memory (LSTM) network and a stacked LSTM. Additionally, an Artificial Neural Network (ANN) was also developed to act as a benchmark model for comparison purposes. Real meteorological data for the year 2019 was used as input to these models. The input features used were solar irradiance and module temperature. Only four months' worth of data were studied, and a set of the proposed and benchmark model were created for each of the four months.

Results: Through various optimisation processes such as data processing, model fitting, hyperparameters tuning and metric evaluation, the results show that both proposed LSTM models outperform the ANN benchmark model. The simplified LSTM model for each of the months displays a better ability to respond to fluctuations and follows the trend of the actual power generation profile more closely. The simplified LSTM and stacked LSTM had a nRMSE of 0.0980 ± 0.0187 and 0.0981 ± 0.0196 respectively while the ANN had an average nRMSE of 0.1092 ± 0.0233. Since the error of the two proposed LSTM models being very close, the optimal model was found to be the simplified LSTM due to a much lower computational training time.

Conclusion: The results suggests that LSTMs in general are superior to ANNs for time series regression problems. The optimal model found out of the three models compared was the simplified LSTM model. In future, the developed AI models can be used for real-time forecasting of power of solar PV plants in Malaysia or similar geographical locations.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Background

In the last two decades, power production from renewable energy sources has spread around the world due to the ever-growing demand for electricity and the necessity to progressively phase out conventional power generation methods, mainly involving natural gas and coal. There remains a lot of untapped potential in the field of renewable energy (RE). Figure 1 below shows the large increase in the use of RE over the past decade. It is estimated that renewable power capacity globally will expand by 50% between 2019 and 2024, led by solar energy [1].

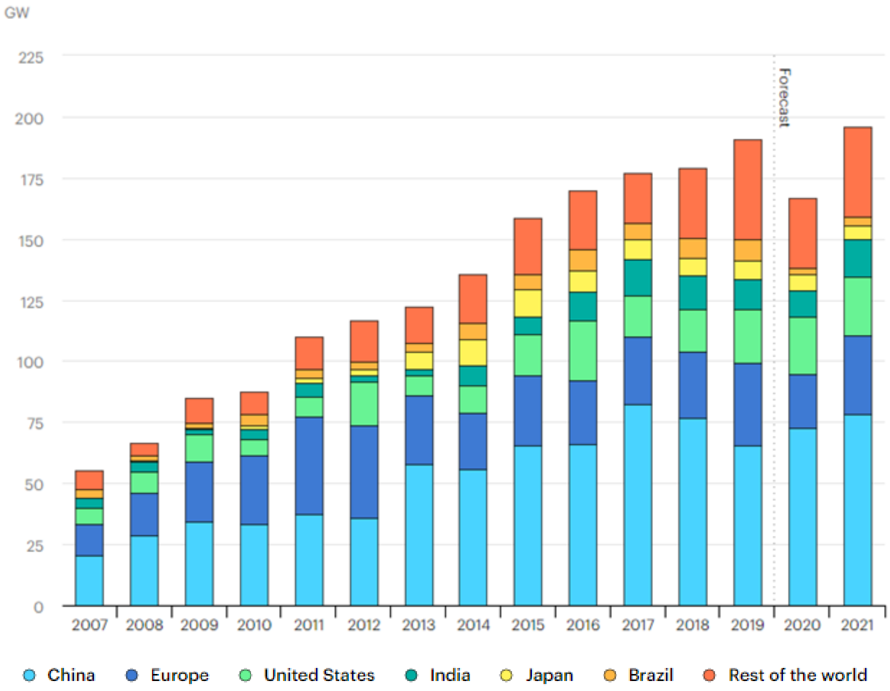


Figure 1: Renewable electricity capacity additions and forecast [2]

Solar energy has become a competitive and viable alternative to fossil fuels. Solar energy is the radiant energy from the sun. On average the solar radiation intensity on the Earth’s surface is 1367 W/m^2 and the total global absorption of this electromagnetic energy is roughly 1.8 x 10^{11} MW [3]. In simple terms, this is sufficient to meet all power requirements worldwide and the main advantage is that it is limitless in nature. There are two types of solar power plants, namely, solar thermal systems and solar photovoltaic (PV). The latter is the concern and focus of this research project. Solar photovoltaics (PV) works on the principle that free electrons found in the solar cell are excited by the incident photons in sunlight. These cells are made of embedded semiconductors which cause a charge build up which yields electricity [4]. Over the past decade solar PV capacity has risen tremendously and shows no sign of slowing down as seen in Figure 2. Solar PV is expected to account for about 60% of the anticipated growth in RE [1].

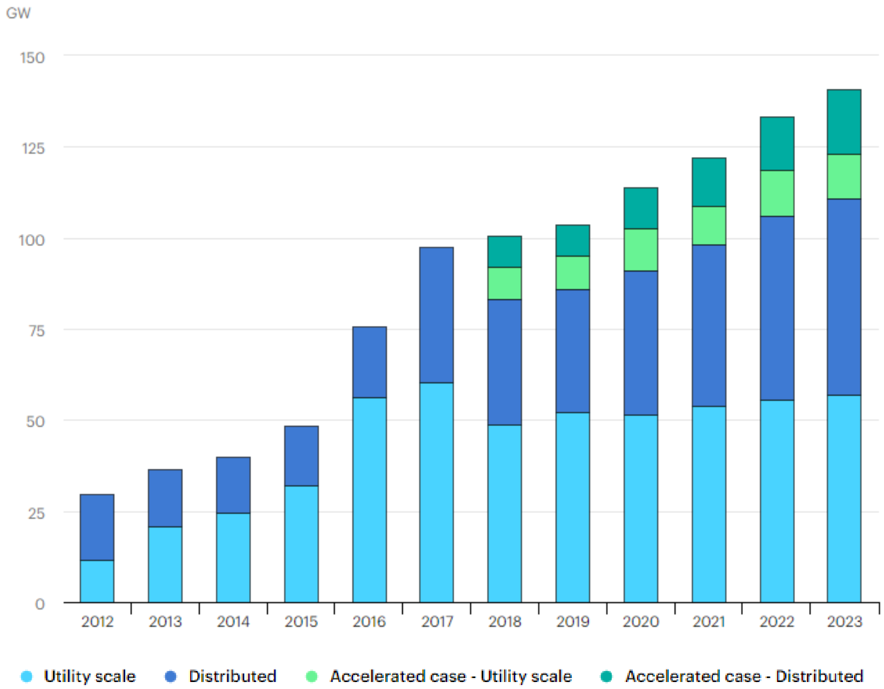


Figure 2: Net solar PV capacity additions [5]

Due to the high level of uncertainty and variability associated with RE sources, there is a significant demand for reliable and accurate forecasting methods. It is necessary due to the dynamic nature of PV output which greatly depends on weather conditions that are by nature highly uncertain. The weather parameters include solar irradiance, air temperature, cloud variation, wind speed, relative humidity, etc. To this day no model has been able to predict precisely their time evolution since their dynamics have not been modelled exhaustively yet. Besides that, the weather forecast that are used commonly as inputs are highly uncertain and can be dynamic in nature as it is heavily reliant on climate and location. This uncertainty becomes a strong barrier for integration of solar PV with the grid as it negatively affects the balance between supply and demand [6].

Solar forecasting techniques can be categorised into three main categories which include image-based methods, numerical weather prediction (NWP), and statistical and machine learning (ML) methods. The focus of this research study is concerned with statistical and ML methods. Among the methods mentioned, ML methods which are a subset of AI techniques have become extremely popular due to the recent advances in technology and wide availability of powerful ML toolboxes and software packages [7]. This type of method uses historical data mainly along with any other meteorological input variables to train models capable of generating forecast based on the relevant inputs.

A forecasting model that is relatively accurate has various advantages such as planning and set-up of solar plant, managing distribution networks and power reserve, and ensuring grid stability [8]. Forecast models that can predict the PV output well will not only improve the system performance, but also aid operational decisions. The principal benefit is it can prevent over-voltage which happens when the PV generation is larger than demand. This has the effect of over-loading grid equipment and causing permanent damage to motors, electronics, etc.

Major factors that significantly affect the forecast model’s performance in determining the solar PV power are time-horizon and time resolution, geographic location, weather conditions, availability and quality of the data [9]. It has been determined that the errors for most models developed are significant, reaching average values between 15 to 20 % [10]. Despite this, of the possible methods, artificial intelligence (AI) has been shown to outperform other methods and is the best available option out there.

1.2 Problem Statement

The problem with AI methods, including machine learning (ML) and deep learning (DL), is that they are relatively new and despite all its advantages there are numerous shortcomings that have yet to be solved completely. So far from the literature review done, there is no one single generic prediction method or model that can accurately forecast for all cases. Each system is unique and forecasting models must be optimised based on the available data. A main problem primarily consists of the possible configuration and architecture of the model. Although previous researchers have developed AI models, many have focused only on ANN. Very few have considered predicting performance of solar power inspired by LSTM models, especially in this geographical region. Lastly, no AI forecast model has been applied to Monash's grid connected solar PV plant.

1.3 Objectives and Scope of Study

The objective of this research are as follows:

- To develop artificial intelligent models to predict the performance of Monash university grid-connected solar PV plants.
- To apply, compare and analyse performance of ANN and LSTM artificial intelligent models

Two AI models which are an ANN and LSTM network will be developed for the short-term prediction of Monash's grid-connected solar PV plant. These models are then validated and tested. The performances of the developed models are compared and analysed. Three popular performance evaluation metrics which are mean absolute error (MAE), root mean square error (RMSE) and the coefficient of determination (R^2) are used to determine the accuracy of each model's prediction.

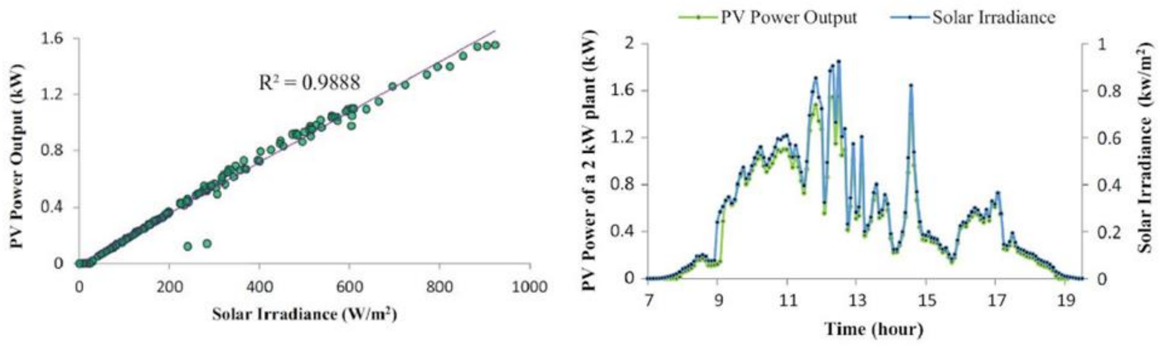
CHAPTER 2

LITERATURE REVIEW

2.1 Factors affecting Solar PV Forecasting Performance

This section describes some of the significant findings from literature reviewed on the factors that have a strong influence on the accuracy of solar PV forecasting. There are a few factors that should be considered in order to develop solar PV power forecasting models with a maximum accuracy and minimum error.

The selection of the most appropriate and relevant meteorological parameters has been found to greatly impact the model's performance. By using more irrelevant meteorological parameters, the performance of the model has been observed to degrade. This finding was validated by both Koca et al. as well as Behrang et al. Both of these research developed and compared the performance of numerous models using different combinations of input variables which ultimately led to different output [11, 12]. Whilst different models act differently on different combinations of input parameters, most researchers are in agreement that the power output of solar panels is primarily dependent on the amount solar irradiation received [13, 14]. Solar irradiation is the electromagnetic radiation energy from the sun. From past studies, it has been proven that there is a direct relationship with the amount of solar power generated by the PV panel. Furthermore, as seen in Figure 3(a) below, it is evident that solar irradiation has one of the highest coefficients of determination compared to other meteorological parameters. To put into perspective, the coefficient of determination of solar irradiance is 0.988 while for atmospheric temperature it was found to be only 0.3776 [15]. A recent paper published agreed that solar irradiance is the most important feature but disagreed on the importance of temperature. Here it was determined from the data that the correlation coefficient of temperature was 0.9506, which is extremely high and makes it a good feature to use [16].



(a) (b)

Figure 3: (a) Correlation between solar irradiance and solar PV power output, (b) Solar irradiance and PV output curve for a specific day [15]

A study conducted in Makkah City found that the solar irradiation ranges between 4.22 kWh/m^2/day and 7.4 kWh/m^2/day. The peak solar radiation measured occurred in June. The months January, November and December exhibited relatively low solar irradiation. The results highlight the PV system produces surplus power from January to July whereas for the months August to December there was a deficit in the energy produced [17]. The inconsistency of the power produce by the PV system is a result of the solar irradiation varying not only daily but also on average monthly. The study mentioned above in [15] also analysed the variation in PV power from daybreak to dusk and showed that the magnitude of power was greatest at noon. This was a consequence of the solar irradiance intensity reaching its peak around noon as seen in Figure 3(b). It is also important to note that no correlation is found during nightfall as there simply isn’t any solar irradiation. Fouilloy et al. found that models such as ANN do indeed suffer in terms of performance when the weather variability is high [18].

In a different study the climatic effect on solar irradiation is analysed. A neural network (NN) model was used to forecast the power on three separate types of days with weather patterns classified into sunny, cloudy, and rainy. The results were taken on each day at the same time to facilitate a fair comparison. It was found that the power output was 10 kW, 3.0 kW and 0.1 kW on the sunny, cloudy and rainy day respectively [19]. This is a reasonable finding since on cloudy and rainy days there will be less solar irradiation penetrating the PV system as compared to on a sunny day.

Finally, two other important factors are geographical location and forecasting time horizon. Research by Premalatha et al. illustrated that the behaviours of models behave differently according to the geographical locations. This affect is brought about due to there being unique or different climatic conditions in different geographical regions of the world. For this particular study, the results and the findings were inferred by training and testing models with five different geographical locations in India [20]. As for the forecasting time horizon, the review paper by inferred from past literature that the performance of the model increases with short-time ahead forecasting and vice versa, and that this almost always holds true regardless of the model or the data granularity [21].

2.2 Solar PV Prediction Modelling

Over the year numerous machine learning and deep learning techniques have been developed and widely used for forecasting energy modelling and forecasting [22]. Most researchers have favoured the use of Artificial Neural Networks (ANN) in the field of PV power prediction use due to the non-linearity of meteorological data [23]. Wang et al. created a back propagation neural network (BPNN) to forecast the short-term solar radiation based on a time series data [24]. A few studies have shown that by utilising more hidden layers, essentially ‘deepening’ the model, the ANN is able to represent complex functions more effectively than RNN with fewer hidden layers [25, 26]. Like ANN, Support vector machines (SVM) is also widely used by researchers in the field of PV power prediction. It is a popular technique mainly used in prediction, classification and regression analysis [27]. The authors in [28] used a SVR model to make a one day-ahead forecast and found reasonable success when testing under different weather conditions. A Support vector regression (SVR) was applied to a small-scale PV plant in Malaysia. The performance showed that it performs well in tropical climates with a relative low RMSE [29].

In a study done in Korea, six-layer feedforward deep neural network is used for a grid connected solar PV to forecast 24 hours ahead. The only parameter used is weather forecast and despite this it still performed better than other models used previously. This did not hold true when the model was run during summer and cloudy weather as the error of the forecast was simply too great [30]. In a different study, research was done where different deep neural networks were compared and analysed. The important finding from the research was that the size of the available dataset has the greatest influence on the models' accuracy. The results found demonstrated that deep learning networks have good potential as the models were found to be highly stable and robust [31].

Some researched boldly claim that deep convolutional neural network (DCNN) is a superior model compared to other models. Combined with the right data processing techniques it gives very effective models that are highly reliable [32]. A branch of DL called deep belief network (DBN) showed remarkable ability in learning patterns continuously till it reached the output stage. It is then able to draw up distinctive patterns found in the dataset. Another paper also explored the use of DBN along with two types of recurrent neural networks (RNN). Promising results were seen for RNNs [33]. In the authors used two variants of RNN, namely gated recurrent unit (GRU) and LSTM. The comparison of the performance with RNN highlighted that GRU and LSTM are more suitable for time series predictions compared to a simple RNN [34]. There are many other subsets of DL such as convolutional neural network (CNN) with the capability to mimic the organisation of animal visual cortex. Its pattern recognition ability could be a game changer when it comes to classifying and predicting metrological parameters [4].

2.3 Artificial Neural Network (ANN)

An ANN model to predict the output solar radiation for a region in Malaysia is trained using five inputs which are air temperature, minimum temperature, maximum temperature relative humidity and wind speed. A multilayer perceptron (MLP) with back propagation topology used. The training algorithm used is Levenberg-Marquardt back propagation due to its better learning rate and speed. 15 months of data is normalised then separated into training, testing and validation in the following proportions 70:15:15. The weights of the neurons are set randomly, and a hyperbolic tangent transfer function is used. For this particular case there are 20 neurons in the first hidden layer and 10 in the second. The accuracy of the model is only 65.4 % based of the regression value. The best performance for mean squared error (MSE) is 0.11169 at epoch 14 which is still considered to be very high. Nevertheless, the ANN model trained is still capable of predicting the output solar radiation, albeit less accurately [35].

In [36] an optimised nonparametric ANN model was used to accurately make day-ahead forecasts for PV systems. PV operational and meteorological data acquired for an entire year belonging to the University of Cyprus was used to train the models. A feedforward supervised learning process with back-propagation algorithm is designed to minimise the overall error. A regulation term controls the complexity of the neural network and prevents overfitting. To determine the best combination of inputs the Pearson Correlation Coefficient was calculated to determine the importance of each variable on the output. Different input parameter combinations were tried and after training it was found that the combination that included all the input variables had the lowest error. Training done with 70 % of the sample and the remaining equally split between validation and testing yielded the best performance. It is also proven that the model trained using randomly selected samples performed better than those that used continuous samples. The best accuracy for the validation set was obtained when the number of hidden neurons was between 12 to 16, with 12 units in a single layer being chosen to reduce the complexity of the model. For the optimal ANN design an average nRMSE of 0.76% was achieved. By using an ensemble method, the average nRMSE was reduced to 0.7%, thus further optimising the ANN model.

A one-day ahead PV forecasting of the energy production on an hourly basis using Artificial Neural Network has been done. A multi-layered feed forward topology network with back propagation algorithm is used due to its simplicity for implementation. The dataset from a 10 KW PV system is used to train models, with the 7 normalised inputs fed in. A trial-and-error method finds the optimal number of neurons in the hidden layer to be five. A sigmoid function is chosen as the transfer function of the neurons. An entire year worth of data makes up the entire time series data and this is categorised into different seasons to account for the varying weather conditions, before being fed into the model for training. An expression is developed and used to extend the power output forecast for PV systems of different energy ratings [37].

Ding et al proposed a ANN-based approach that uses feed-forward neural network to predict the power output of PV system directly by reference to historical data of PV system. An improved back-propagation learning algorithm is used instead of the standard BP to improve convergence. The key difference is that in the improved BP algorithm the learning step rate is adjusted accordingly along with the weights. To improve the forecast accuracy, similar day selection algorithm is used where the closest historical record which has the similar weather condition with the forecast day is used as input. The input data consists of historical power data and weather data that were normalised before being used to train the model. It was observed that the PV system varies widely in different weather types, with the curve being smooth on a sunny day while on a rainy or snowy day it appears to fluctuate. On sunny days the MAPE is found to be 10.66% whereas on rainy days the MAPE shows a larger error of 18.89 %. Overall, it showed the forecast works better for sunny days as there is relatively less fluctuations [38].

An ANN model accompanied by a clear sky model for input data validation is used to forecast the energy of the PV plant for the next day, with hourly resolution. Both weather and the output power measured on the PV systems historical data are used as inputs to the ANN. A clear sky solar radiation model (CSRM) was used to validate the reliability of each fifteen-minute sample. The model has a multi-layer perceptron (MLP) structure and utilises an error back propagation training method. A model with the following characteristics is used: 9 neurons in the first layer, 7 neurons in the second layer, and 3000 iterations for each trial. It had a good compromise in terms of efficiency and computational time effectiveness. Generally, the longer the training set size the lower are the errors. Both the quantity and quality of the samples in the training set are critical to the forecasting reliability and accuracy. The best forecast days are those that are sunny whereas the error increases during the unstable days when the weather conditions are very cloudy. Most relevant errors are noticed to have occurred during sunrise and sunset [39].

Omar et al. chose a multilayer perceptron (MLP) feed-forward artificial neural network that employs a series of historical meteorological weather data to make prediction on the output power. The time horizon of choice is 24 hours ahead. The inputs used are day, hour, measured temperature, irradiance from clear sky model, cloud cover, wind velocity, pressure and humidity. The global horizontal irradiance is said to be highly correlated with the power output of the plant. Optimisation was done by varying the number of neurons and the number of hidden layers. The optimal model that had the lowest error has only one hidden layer comprising of 225 neurons. Finally, an ensemble technique is used to combine the predictions and a key observation made is that the ensemble network was able to achieve superior results in comparison with a single network of the same configuration. It had a nMAE and nRMSE of 12.4% and 17.71% respectively [40].

In a different research a statistical tool called Design of Experiments (DOE) was used to choose the input variable that have the most influence on the desired output and with this knowledge fewer number of runs need to be performed in the simulation process. Data pre-processing mainly comprises of normalising the data and feature scaling. The exogenous time series data available included cloudiness, hours of insolation, temperature, precipitation and humidity. Hours of insolation had the highest correlation index at 0.811. 32 trials were run for the model set up and through trial-and-error method, the optimal model was found to have two hidden layers with three neurons in the first layer and six in the second layer. 100 epochs were run with a learning rate of 0.1 using the Levenberg-Marquardt training algorithm. The training was done using the sequential data series making use of only the PV generation series and all data had been feature scaled normalised. When an ensemble technique is applied the mean MAPE drops to 4.7% [10].

In [41] an ANN model that takes in four meteorological variables along with the historical data for the year 2002 as inputs to the model. The variables are number of days of the year, maximum air temperature, humidity and average atmospheric pressure. A back propagation algorithm is used to minimise the error signal and to calculate the updated weight of the layers that make up the model. Back propagation is one of the most widely used structure for solar radiation forecasting. The weight and biases of the network are updated using the Levenberg Marquardt algorithm which is found to be the fastest training algorithm in MATLAB’s neural network toolbox for supervised learning methods. For the training process, the available data is divided into 70% training, 15% validation and the remaining for testing the model. The performance is evaluated based on the MSE, with a lower MSE indicating the model has minimum errors. Overall, the model showed reasonable accuracy in predicting the solar radiation with a regression of at least 94.4% obtained. In order to further reduce the MSE other intelligent techniques such as Particle Swarm Optimisation, Genetic Algorithm and Bees algorithm could be explored and tested.

In the study by Demirdelen et al. a multi-layered feed forward neural network structure is used. Two separate algorithms, namely Firefly Algorithm (FA) and Particle Swarm Optimisation (PSO) are applied to train the network coefficients. A correlation analysis performed showed two important findings. The first is that there are strong correlations between the solar radiation, not only in consecutive hours but also during consecutive days. Secondly, the correlation between two consecutive days for the same hour is stronger than that between the current hour and two hours ahead in the same day. For this study, the former two days’ solar radiation data at the time of prediction and the data at the current time are used as the input to the prediction model. The input layer is fed with data of ambient temperature, solar radiation, PV panel temperature for the years 2015 and 2016. For each method used, eight neurons are present in the hidden layer. The results for the three models showed that ANN-FA had the lowest MAPE at 19.4149, followed closely by ANN-PSO at 19.5525 and lastly the simple ANN at 22.2098. The regression was also calculated with ANN-FA having the best at 0.97993. It is concluded that the best method for PV power forecasting is obtained using ANN-FA method as it removes the problem of sticking to local minimum values which is an issue for the ANN-PSO method. It was seen that traditional ANN was not able to predict using a few input data alone [42].

Ncane and Saha employed a fuzzy logic and artificial neural network that was developed using MATLAB. A comparison done showed centroid defuzzification methods having triangular membership function (MFs) give the least percentage error. The fuzzy logic system trained using normalised values of irradiance and temperature yielded a fairly low error of only 1.924 %. An ANN was built using a multilayer feedforward backpropagation network trained with Levenberg-Marquardt algorithm. Normalised values of solar irradiance and temperature have been used as inputs to the model that outputs the solar PV plant output power. The neural network uses 30 neurons and the dataset has been split into a 60:20:20 ratio. The average error for the ANN method was found to be 2.626 % which proves it is capable to mimic the actual solar PV plant behaviour. Both methods are successful and reliable as their results follow the actual solar PV array signature [43].

2.4 Long-Short Term Memory (LSTM) Network

Jebli et al. studied the efficiencies of three DL models suitable for forecasting time series data, namely RNN, LSTM and GRU. Prior research has found that Adam optimiser is the best optimiser and Tanh activation function shows good fitting capabilities. Results show that the RNN and LSTM both have similar performance in terms of accuracy and can be considered as reliable models for time-series regression problems [44]. Bao et al. research despite being developed for different application also gets results that agree on the suitability of LSTM for time series prediction but disagree with the use of RNN for such problems [34]. Huang et al. compared MLP and LSTM prediction models. The study showed that increasing the input sample size leads to complex hyperparameters of the neural networks. Up to a certain point, the prediction accuracy improves as the model complexity increases. It was concluded that the LSTM model is slightly better than the MLP due to its better memory in training historical data [45].

A simplified LSTM algorithm in [46] for the forecast of one day-ahead solar power generation uses a moving window technique to read the past observations. The model uses RMSE as loss calculation in addition to a dropout layer to prevent overfitting. It is observed from the dataset that the signal variations are cyclic and has a strong correlation with the sun activity. The LSTM model produces the best results achieved for sunny and low light weather profiles which have smoother lines. Park et al. also proposes a LSTM model. Fine tuning of parameters was done by varying the number of hidden layers and hidden neurons whilst keeping the initial learning rate constant. The study demonstrated that the use of multiple layers can further deepen the learning as compared to a single-layer model. The graphs plotted indicate that the single-LSTM model and multi-LSTM model both trend in almost the same pattern. However, the results also infer that the larger the amount of solar PV power generation, the lower the accuracy of the single-LSTM model while the opposite relationship is true for multi-layer LSTM model [47].

The paper by Konstantinou et al. proposed a stacked LSTM network that predicts the PV power output for 1.5 hours ahead or six timesteps. The model only uses endogenous data which is split in such a way that the first 80% of observations were used for training and the rest for testing. To make predictions the previous 3.2 days' worth of data or 192 timesteps before the target is used. Results indicate that the actual power output of at least the previous day affects the trend of the predicted power for any given day. To better evaluate the prediction accuracy k-fold cross-validation is introduced [48]. Hossain and Mahmood created two forecasting algorithms each consisting of two stacked LSTM hidden layers. The first model forecast a single step ahead PV power whereas the second is capable of forecasting intraday rolling horizons. The optimal input sequence length is found to be 12-time steps. Increasing the input sequence length any further will have a detrimental effect on the accuracy. Both models' accuracy changes by the season and the results indicate a strong correlation to the solar irradiance. The second model developed proves that adding more predictors can effectively improve the performance and that the error is significantly lower for smaller forecast horizons [49].

In [50] the periodicity dependence of an LSTM model in capturing time correlation is identified as a research gap yet to be studied. Hence, an LSTM-FC with double branches is proposed where the LSTM obtains the temporal correlation of PV power generation, and the FC layers improve the mapping relationship between features. The outputs of the two branches are then weighted to get the final prediction result. The dataset consists of PV power generation data and meteorological data such as temperature, humidity, wind direction and wind speed for one entire year. A test carried out finds LOF to be the most suitable outlier detection algorithm. The results after training indicate that the RMSE decreases as the length of the input used for training increases, but the decline is no longer significant after a certain lagging time. The plots verify the idea that using periodic data can improve model accuracy as the expected and predicted curves are basically fitted with only small amounts of errors between them. The error is minimal at RMSE and nRMSE of 2.5606 and 0.1743, respectively. Kwon et al. makes use of LSTM and fully connected (FC) layers in tandem with each other to make predictions of the hourly load for the next day of a power system in Korea. The historical data input is fed to a LSTM layer combined with a FC layer are combined to give the prediction data for the target day. This data is then used in turn as an input to another sole FC layer to produce the final output. The moving window method was used to select the training set. Through experimentation the optimal window size for the LSTM layer was selected. Adam optimiser was found not only to be the most computationally efficient but also gave the best results, and hence was used in the training process. As a whole, the LSTM layer was used to extract features from the historical data while the FC layers were used to form relationship with the target load. The combination of LSTM layer with FC layer had a positive influence on the predictive ability with an annual average MAPE of 1.49% and 1.52% for the year 2017 and 2018 respectively [51].

The authors Chen et al. proposed an RCC-LSTM that uses only previous PV power data and meteorological data. The similar time periods in the dataset are identified and combined into a training data set using the RCC algorithm. Due to the RCC classification method the model does not require large training data set which means the calculation cost of the model is reduced and real-time predictions can be made faster. The RCC-LSTM model has an average training time-cost 28.84% lower than that of LSTM [52]. Elsewhere, [53] compared three types of neural networks which are LSTM, GRU and RNN. Each model was hybridised by combining it with genetic algorithm (GA) to find the most appropriate hyperparameters such as number of window size and number of units in each hidden layer. All three models consisted of three hidden layers to facilitate a fair comparison. The dataset was divided according to seasons and one model was built for each of them. Both GRU and

LSTM are excellent in dealing with long-term dependency in data. While the GRU has higher efficiency and lower complexity, LSTM is better at treating long-distance relationship sequence data due to its ability to remember sequence data from long distance relationships. The results found that LSTM-GA handles larger data size better than GRU-GA. The use of dropout helps prevent the problem of over-fitting. Lastly, the incorporation of GA ensured higher accuracy in MSE and MAE as the best choice of hyperparameters were selected.

2.5 Summary

Figure 4 below illustrates all the findings over the past few years on the trends of researchers in relation to forecasting solar PV power. It is evident that machine learning models have been most prominently used. Despite the numerous papers on deep learning techniques for forecasting solar PV power, it still represents only a small proportion of the research done and highlights the need for more research in this field of AI. As stated in the prior sections, ANN has been the most popular method chosen for solar forecasting applications and the time horizon many use is short-term. Finally, the last pie chart shown emphasises that solar PV production in Malaysia remains extremely low in comparison with other countries.

Literature review done identified some of the key factors affecting the performance of solar PV power forecasting. The forecasting time horizon and feature selection were some of the most important. Majority of researchers agree that solar irradiance is the most crucial input feature in forecasting solar PV power. Many of the papers discussed above in relation to LSTM models have been done very recently. Some of the papers reviewed have compared developed LSTM models to other deep learning models and machine learning models, to determine if LSTM are better suited for time series forecasting in solar PV applications. Literature indicates that there is optimism among researchers that LSTM has the potential to improve the forecasting accuracy of solar power output. Some researchers have already developed models that did in fact improve the performance in comparison with other models. It can be gathered that a lot of researchers are still to this day investigating and analysing the performance of LSTM. However, this also suggests that the current state of knowledge about the forecasting abilities of LSTM is not sufficient enough to take advantage of the untapped potential of LSTM model in the domain of solar power output forecasting applications. This is the reason for there being many papers published recently in the past one or two years on the development of simplified and stacked LSTM models.

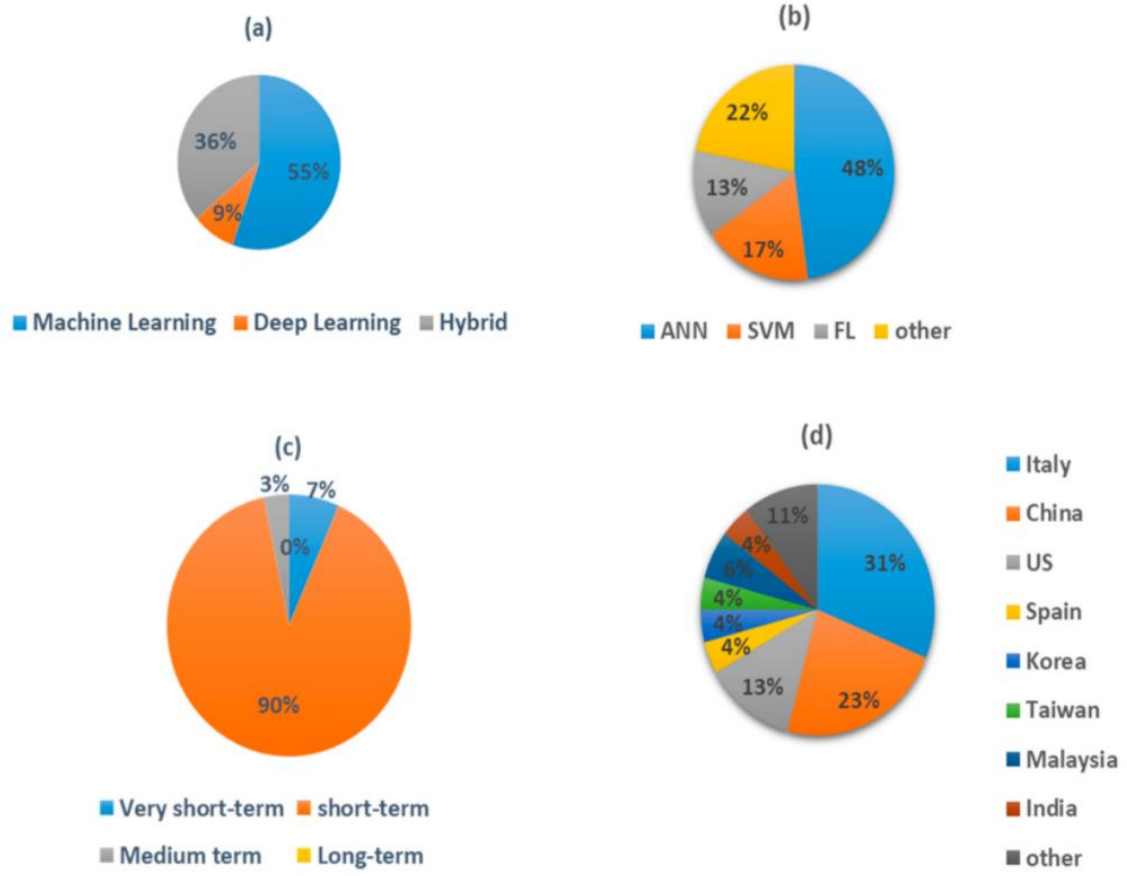


Figure 4: (a) AI-based PV power forecasting methods, (b) Most common DL techniques, (c) time scale horizons, and (d) countries involved in PV forecasting [54]

CHAPTER 3

METHODOLOGY

3.1 Installation Site and PV System Description

Monash University Malaysia, as part of its initiative towards a sustainable future, has installed a grid-connected solar PV plant on the rooftop of its buildings. This installation consists of 646 monocrystalline PV modules encompassing a total area of 2169 m^2. The PV modules are accumulated into five units, and each unit is connected to an inverter [55]. Table 1 summarises the important information of the solar PV system. The AC output of each inverter along with other influential parameters such as solar irradiation, module temperature and ambient temperature were recorded by the data logger installed at 5-min intervals. This data was then transferred to a webserver wirelessly. The data for the year 2019 was then acquired and then used for the purposes of this study.

Table 1: Physical and technical specification of the solar PV system at Monash Malaysia

PV Module Details	Specifications
Number of PV modules	646
Cell Type	Mono
Rated maximum power (P_{max})	360
Module efficiency (%)	18.5
Number of inverters	5
Data Logger	Solar-Log 2000

3.2 Dataset

The dataset used in this research project is provided by Monash University Malaysia for its newly installed rooftop solar PV power plant as shown in Figure 5. Data is available for the entire calendar year of 2019 and has a resolution of five minutes. The data consists of past observations of measured weather data and PV power generation data. This is real-world data that will be used as historical data in the ANN and LSTM forecast models to predict the future solar energy that will be produced by the PV system. Two key observations are immediately evident upon plotting the graph of output power against time. Each day the signal varies in a manner that can be characterised with a bell curve, where the peak power is produced around noon. There is an obvious correlation of output power with the sun activity. Furthermore, the signal variations are periodic, repeating every single day.

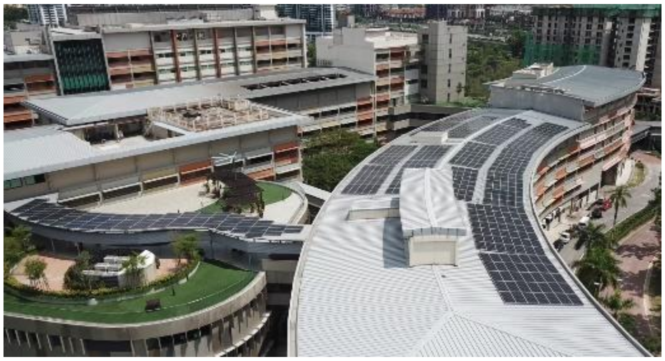
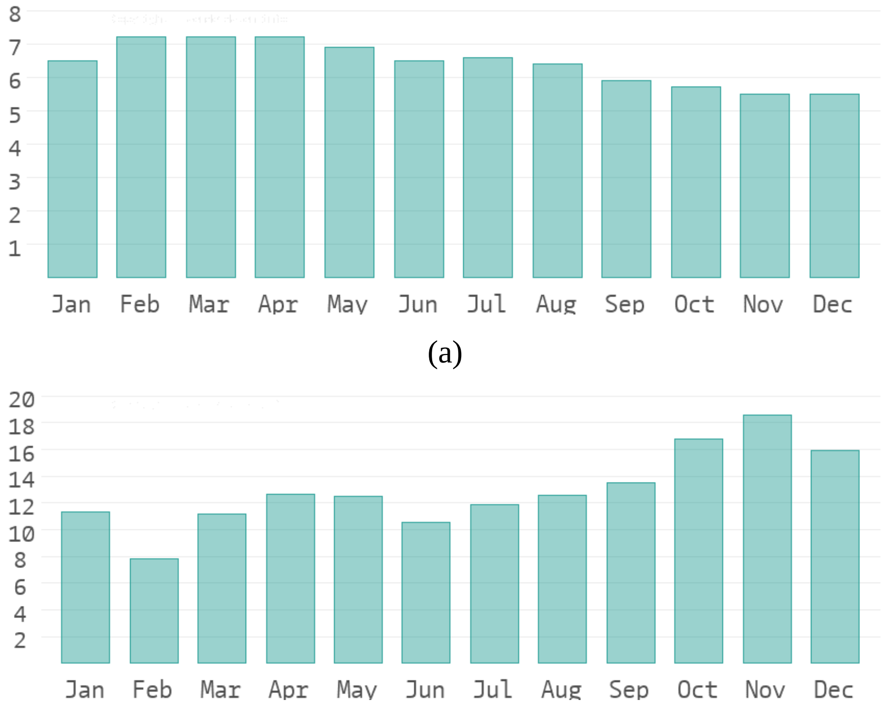


Figure 5: Monash University Malaysia solar PV plant located on sections of the roof

The colossal size of the dataset which would require extensive amounts of computational memory and time for training, validation and testing of the model. Hence, the study has been limited to just four months for which an individual model would be designed for each month. The months of interest are February, June, August and November. These months have been chosen specifically to provide a comprehensive study on the models' performances in different weather conditions. The month of February had the most sunshine and least rainfall recorded while November had the most rainfall and least sunshine recorded. June was slightly sunny while August was slightly raining. This information was found from an official online source for which the graphs are presented in Figure 6 below.



(a)

(b)

Figure 6: (a) Hours of sunshine per day in Malaysia 2019, (b) Rainy days per month in Malaysia 2019 [56]

3.3 Simulation Environment

In this experiment, the hardware environment includes Windows 10, 8GB RAM memory, and Intel Core i7-7500U CPU processor. The software environment used was MATLAB R2020b which is ideal for iterative analysis and design processes. In MATLAB, the Deep Learning Toolbox and Statistics and Machine Learning Toolbox were used.

3.4 Data Pre-processing

Initially the data in Excel is examined to identify the data recorded and the characteristics of the data. The data is then imported into MATLAB. Since the dataset provides the output power measured by five separate inverters, it is necessary to create a new column that consist of the total AC power produced which is essentially the summation of all the power recorded by the five inverters for every timestep. The historical data recorded has a variety of features but not all of them are useful for the prediction. Based on the literature reviews, the important variables that have strong influence on the output power is imported as individual months into MATLAB.

3.4.1 Night-time Values

The dataset consists of observations recorded throughout the entire day by data loggers. As a result, many observations are found with a PV power output of zero. These readings correspond to night-time values when no solar PV power is produced due to an absence of sunlight. The inclusion of all these night-time observations which provide no significant value would affect the performance of the model and lead to a poorly trained model.

To circumvent this issue, only the observations for the time period of 07:00 to 19:55 are kept while the rest are discarded. This selection is made by inspecting the data to identify the earlier and latest time the sun is out for any given month at the site's location. Additionally, the selected timeframe was also made so that each day has at least one zero value.

3.4.2 Missing Data

In many datasets, missing data is common due to problems associated with fault sensors and systems that record and save the data. Incomplete data can lead to difficulties in forecasting, especially LSTM that requires continuous time-series data. A thorough check was done of the dataset to find any missing values that would need to be attended to. No missing data was found which is surprising but could easily be attributed to the brand-new system. If there were such missing data an interpolation method would be used to fabricate reasonable values.

3.4.3 Feature Extraction

The historical data recorded has a variety of features but not all of them are useful for the prediction. By only selecting the most appropriate features that can positively impact the learning, the number of variables is reduced, thus minimising the complexity of the training data and enhancing the efficiency and accuracy of the model. Pearson correlations, which is a popular statistical method shown in Equation 1, is used to identify the features that have the highest degree of correlation to the solar PV power output. The higher the correlation coefficient is, the stronger the correlation between the two variables [52]. The brief study was done by testing various combinations of predictors while the rest of the model configuration and settings were kept constant.

$$r_{x,y} = \frac{\sum(x-\mu_x)(y-\mu_y)}{\sum(x-\mu_x)^2 \times \sum(y-\mu_y)^2} \tag{1}$$

Where x is the meteorological data, y is the PV power generation and μ is the average value.

3.4.4 Feature Scaling

The features previously mentioned in the dataset are of different scales. Training the model with data that has not been scaled could potentially lead to a false prioritisation of some of the variables [57]. To avoid this, the features are scaled by normalising all the data so that each variable is within a range of 0 to 1. Literature has proven that this pre-processing method can have a significant, beneficial influence on the predicted output as it ensures the quality of the input data is less dispersed. Normalisation can be done with the formula given by

$$x_{normalised} = \frac{x - x_{minimum}}{x_{maximum} - x_{minimum}} \tag{2}$$

Where x is the real data of the features and the target power output, $x_{minimum}$ and $x_{maximum}$ is the minimum and maximum value respectively of the feature being normalised.

3.5 Training, Validation and Testing Dataset

Instead of splitting the data randomly which is usually the case, in this work the data is split sequentially or in order due to it being a time-series regression problem. Firstly, the data belonging to the last day of the dataset is removed and stored as the testing dataset to be used at the end. From the remaining data the training dataset consists of the first 85% data while the validation dataset occupies the last 15%. This can be easily visualised by referring to Figure 7 below. Since the validation dataset is not seen by the model during the training phase, it is used to tune the hyperparameters. This split ratio is done so as to avoid over fitting. Finally, the test dataset mentioned earlier is used to fairly test the model.

Whilst both validation and testing datasets are not seen by the model during the training phase, the validation dataset has been used to optimise the hyperparameters of the model such as the number of neurons in the hidden layers and the number of hidden layers. This fine-tuning process will ensure the best possible values of hyperparameters that minimise overall error. This also means that using the validation set to test the model would not be fair as the model's hyperparameters have been specifically tuned to perform well for the data in the validation set. Using the test dataset which is still a fully unknown data that has not been used in either the training or tuning process will provide an unbiased and fair evaluation of the optimised model.

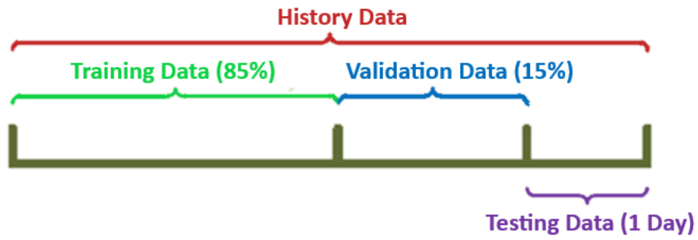


Figure 7: Specific division of the historical data

3.6 Development of ANN Predictive Model

An ANN is a machine learning technique that replicates the information processing mechanism of the human brain and its neurons. It has a unique capability to learn and approximate nonlinear functions with high fidelity and accuracy. The advantage of ANN lies in its self-training mechanism which compares the predicted and actual results and its self-learning ability to adjust its weights to minimize the error. This has made ANNs suitable for many diverse applications, especially in forecasting of future trends or events [58]. These models are becoming increasingly popular and are treated by many researchers as a benchmark [4].

The basic architecture of an ANN consists of an input layer, one or more hidden layers and an output layer. Each of these layers comprises of artificial neurons, the basic processing elements of this network which are linked to one another by unidirectional links. Each link has a weight attached to it. The output of neurons in each layer are fed forward to their next level until the entire output is obtained [59]. The chosen training procedure is the back-propagation (BP) algorithm which is one of the most powerful supervised learning algorithms. It involves fine-tuning the weights between neurons in the different layers based on the error rate obtained in the previous iteration. This is done over several iterations until the difference between the simulated output and actual output is under a certain fixed threshold [39]. The core calculation formulae of the ANN model are as follows:

$$a = \sum_{i=1}^{n} w_i \cdot x_i \tag{3}$$

$$y = f(a) = f\left(\sum_{i=1}^{n} w_i \cdot x_i - b\right) \tag{4}$$

$$y = f(a) = \frac{1}{1 + e^{-a}} \tag{5}$$

Where a is the weighted amount of the inputs (activation level), b is the bias and y is the activation function which gives the output of a neuron. Equation 5 is the activation function for a sigmoid function.

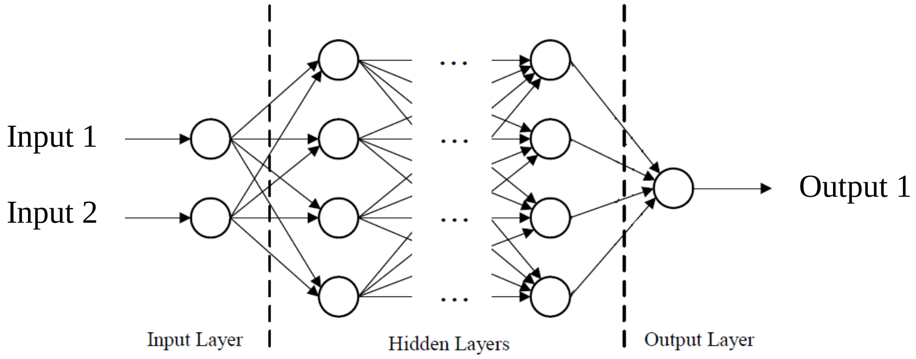


Figure 8: Schematic diagram of a multilayer ANN architecture

In MATLAB the Neural Network Time Series App is initiated with the built-in function ‘ntstool’. A nonlinear input-output type ANN was created which takes a specified number of past observations of selected inputs features to forecast the current output.

3.7 Development of LSTM Predictive Model

LSTM is a special kind of RNN designed to resolve gradient disappearance and gradient explosion problems encountered in RNN models [60]. It has numerous memory blocks connected through a succession of layers. Within the block, the LSTM cells consists of three types of gates, namely input gate, forget gate and output gate. These gates oversee the information update procedure, maintenance and deletion present in cell status [61]. They also preserve weights propagated through time and layers. The presence of memory blocks gives LSTM networks an edge over other existing methods as it addresses the gradient issues due to its ability to memorise network parameters for long durations. This makes LSTM suited to model input data with time-series characteristics such as sequential data. They exhibit excellent performance in dealing with nonlinear relationships and are a robust algorithm useful in efficiently solving long-range dependencies problem [51, 62]. This makes it suited to model input data with time-series characteristics such as sequential data and exhibit excellent performance in dealing with nonlinear relationships in a given dataset. For the purposes of this research only simplified LSTM models will be developed which essentially means the LSTM consists of only one hidden layer. The motivation behind this is related to the findings by Jozefowicz et al. that state variations made to a simplified LSTM do not significantly improve the performance for sequential tasks [63]. The output of LSTM is computed as shown in Equation 6 to Equation 11:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{6}$$

$$\check{c}_t = tanh\ (W_c \cdot [h_{t-1}, x_t] + b_c) \tag{7}$$

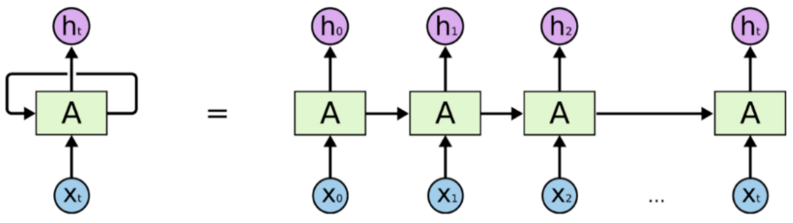
$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{8}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{9}$$

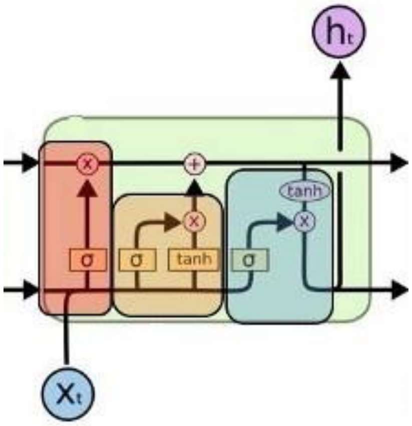
$$c_t = f_t \cdot c_{t-1} + i_t \cdot \check{c}_t \tag{10}$$

$$h_t = o_t \cdot tanh\ (c_t) \tag{11}$$

Where $\check{c}_t$, f_t, o_t, c_t, and h_t represents the input gate, cell input activation, forget gate, output gate, cell state, and the hidden state respectively. W_i, W_c, W_f and W_o represent their weight matrices respectively. b_i, b_c, b_f and b_o represent the biases. x_t is the input, h_{t-1} is the last hidden state, h_t is the internal state. σ is the sigmoid function.



(a)



(b)

Figure 9: (a) LSTM network architecture, (b) LSTM gates within an LSTM cell

3.8 Performance Evaluation Techniques of Models

Five commonly employed evaluation functions are used to verify the model performance. These are chosen as they have been found to be most suitable to the context of DL and regression problems. Each of the performance metrics below has its own specific target and provide different relevant information about the accuracy or error of the model. Hence, they are not comparable between them.

$$MAE = \frac{1}{n}\sum_{j=1}^{n}\left|y_j - \hat{y}_j\right| \tag{12}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2} \tag{13}$$

$$ME = \max_{1\le j\le n}\left|y_j - \hat{y}_j\right| \tag{14}$$

$$R^2 = 1 - \frac{\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}{\sum_{j=1}^{n}(y_j - \mu_{\hat{y}})^2} \tag{15}$$

where y_j is the actual output, $\hat{y}_j$ is the simulated output, $\mu_{\hat{y}}$ is the mean of the simulated output and n is the number of observations. All the values were calculated on a normalised scale basis. The equations are the same with the only difference being the inputs to the equation are normalised. So, all the metrics presented here are normalised error metrics.

RMSE is one of the most important evaluators as it describes the average spread of errors. It is a measure of the variation of the predicted values around the measured data, with a large positive RMSE indicating a large deviation and vice versa [64]. Most researcher tend to use some form of the RMSE in evaluating the performance of their forecast since it represents a real-world forecasting scenario more closely, where a relatively high weight is given to large errors and therefore have greater consequences [65]. This make it the ideal metric to be used as a model selection criterion during the optimisation phase to avoid models which make large errors. MAE is another commonly used metric in regression problems that computes the average of the absolute value of the error. It is much easier to interpret than RMSE due to its simplicity. Similar to RMSE, small values of MAE suggest better forecast. A concern with MAE is that many minor differences in small error can have a significant impact on the overall score [58]. In some case, R^2 may be more useful than either of the above and can better explain the performance of the model [50]. R^2 is a statistical metric that the proportion of the variance of dependent variables which are predicted and can be explained by independent variables [61]. Unlike, RMSE and MAE, higher values are better as it indicates the data is fit well within the corresponding forecasting framework.

CHAPTER 4

OPTIMISATION OF DEVELOPED MODELS

4.1 Feature Selection

The first step in developing an accurate model is to identify the features from the dataset that have the strongest correlation on the total AC power produced by the solar PV system. Those features that have been found to have a strong influence on solar PV power generated by previous research studies are extracted from the dataset, subject to availability. This is important as the study is limited to the available data and features such as relative humidity which have been proven by other researchers to be useful in the forecasts of solar PV were not measured and hence are not in the recorded dataset used.

The Person’s correlation of each of those features are found and presented in Table 2. It was observed from the findings that to solar irradiance has the strongest correlation to solar PV power generation. Those features which are related to temperature also have a strong correlation and hence impact the amount of solar PV power generation. Wind velocity had a very low value which implies it barely has any effect on the generation capabilities of the solar PV plant. Insolation was the only meteorological parameter that had an inverse correlation, albeit extremely low. The strength of the correlation is decided by the magnitude of the coefficient and not the polarity of the relationship, and hence insolation is another feature that is not important.

Table 2: Pearson's correlation coefficient of input features with total AC output

Meteorological Parameters	Pearson’s Correlation Coefficient
Irradiation	0.9861
Module Temperature	0.9229
Ambient Temperature	0.8425
Wind Velocity	0.0323
Insolation	-0.1079

From Table 2 it is clear the ranking of each parameter’s importance for the forecast of solar PV power. However, past studies done by researchers have shown that taking all the parameters above a pre-determined set threshold or taking the top two or three parameters do not necessarily lead to a model with the least error. Instead, different combinations of the relevant features that have a high Pearson’s coefficient should be tested out via trial-and error method to determine the best possible combination of features to be used as the inputs to the model. From Table 3 below, the best possible combination of features to use as predictors to the model was found to be solar irradiance and module temperature. This happens to be the combination that utilises the two parameters with the highest and second-highest Pearson’s correlation coefficient. The test results show that despite ambient temperature also having an incredibly high Pearson’s correlation coefficient, the inclusion of it to create a predictor with three input features will deteriorate the performance of the model. Furthermore, it would increase model complexity and lead to larger computational time.

Table 3: Sensitivity study of different input feature combinations

Selected Input Features	nMAE	nRMSE	R^2
Irradiation	0.2318	0.3259	-0.6276
Irradiation & Module Temperature	0.0067	0.0912	0.8726
Irradiation & Ambient Temperature	0.0217	0.0954	0.8605
Irradiation & Module Temperature & Ambient Temperature	0.0089	0.0929	0.8677

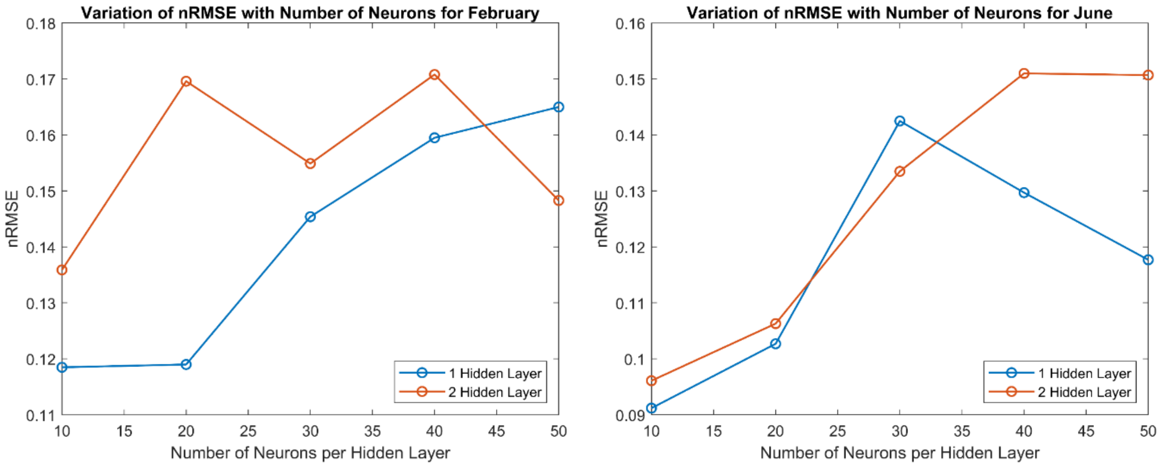
4.2 Optimisation of ANN models

The two main factors that affect the accuracy of the prediction are the number of neurons in each hidden layer and number of hidden layers [35]. These two network configuration parameters were varied one at a time to determine the optimal network settings.

Since the aim of this research was to develop a real-time forecasting model, the number of hidden layers was limited to two hidden layers. This is consistent with commercial ANNs that commonly incorporate anywhere between one or two hidden layers [66]. The reason for this is the complexity and computational burden that are a consequence of using too many hidden layers. Hence, one or two hidden layers is the ideal compromise between accuracy and computational time.

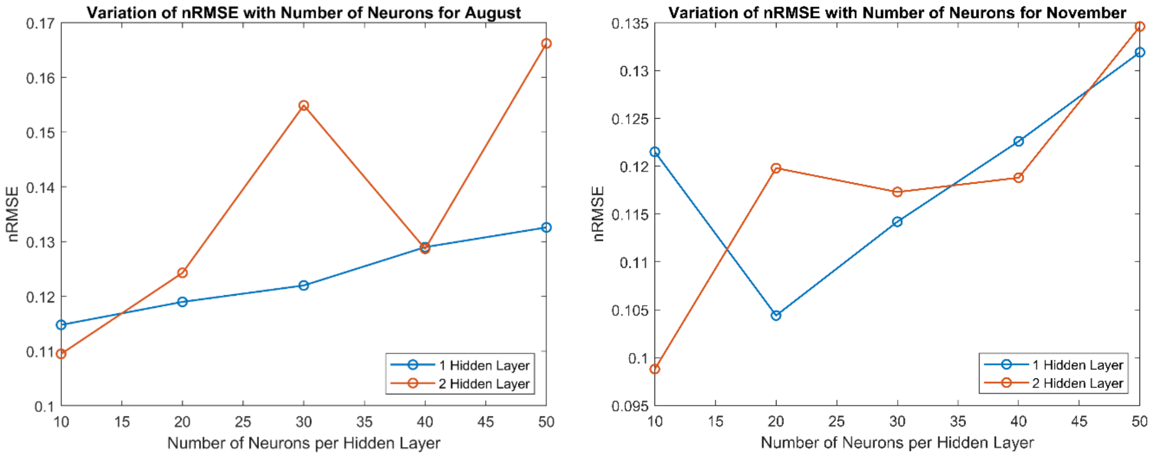
Figure 10 below show the variation of the RMSE as the number of neurons change. As explained in the methodology section the optimisation is performed based on the RMSE performance metric criterion, and this is the reason the graphs shown are only for RMSE. From the graphs we observe that the ANN gives the lowest error when the number of neurons per hidden layer is low. This is the case for the models with just a lone single hidden layer and the models with two hidden layers. For the ANN models created for the months of February and June, it is clear that a single hidden layer model configuration results in the lowest error, and that the lone hidden layer should be made up of 10 neurons. While the graphs for February and June does suggest that the error could further decrease at neuron numbers greater than 50 that were not tested during the optimisation stage, the decision to select 10 neurons is a valid option as it takes into account both low errors and computational efficiency with regard to the time taken for training. At neuron numbers greater than 50 the training would take extremely long making it less ideal for real-time forecasting applications. Besides that, at low neuron numbers the errors were already extremely low.

For the ANN models created for August and November, a similar pattern is found where low number of neurons in the one or more hidden layers result in a much lower error. There are two key differences between the models for these two months and the models for the months discussed prior that can be inferred from the plots below. The first is for both August and November, the ANN models showed that the lowest nRMSE and hence the best performance accuracy would be achieved by using models with two hidden layers instead of one hidden layer. Secondly, the graphs in Figure 10 depict that as the number of neurons per hidden layer increases, there is a clear increase in the error of the forecast models. Even beyond 50 neurons per hidden layer there is no indication that the models' error would fall. The optimal configuration is a two hidden layer model consisting of 10 neurons per hidden layer for both the models created for each of the two months data, August and November.



(a)

(b)



(c) (d)

Figure 10: (a) Study of variation of neurons and hidden layers for February ANN model, (b) Study of variation of neurons and hidden layers for June ANN model, (c) Study of variation of neurons and hidden layers for August ANN model, and (d) Study of variation of neurons and hidden layers for November ANN model

The detailed quantitative assessment results obtained during the optimisation stage is presented in the Table 4. These performance metrics have been calculated for the validation dataset as it is this dataset that was used during the all the optimisation phase regardless of the type of model. Table 4 summarises the optimal configuration each model requires in terms of number of layers and number of neurons. It is observed that the ANN model for the month of June was the most accurate, followed by the model for November. The model for February had the highest error in terms of RMSE.

Table 4: Optimised configuration of ANN model and performance metrics evaluation for validation dataset for each month's model

Month	Number of Layers	Number of Neurons	nMAE	nRMSE	Maximum Error	R^2
February	1	10	0.0285	0.1185	0.4659	0.8546
June	1	10	0.0090	0.0912	0.4729	0.8791
August	2	10	0.0094	0.1095	0.4878	0.8412
November	2	10	0.0063	0.0988	0.6000	0.8503

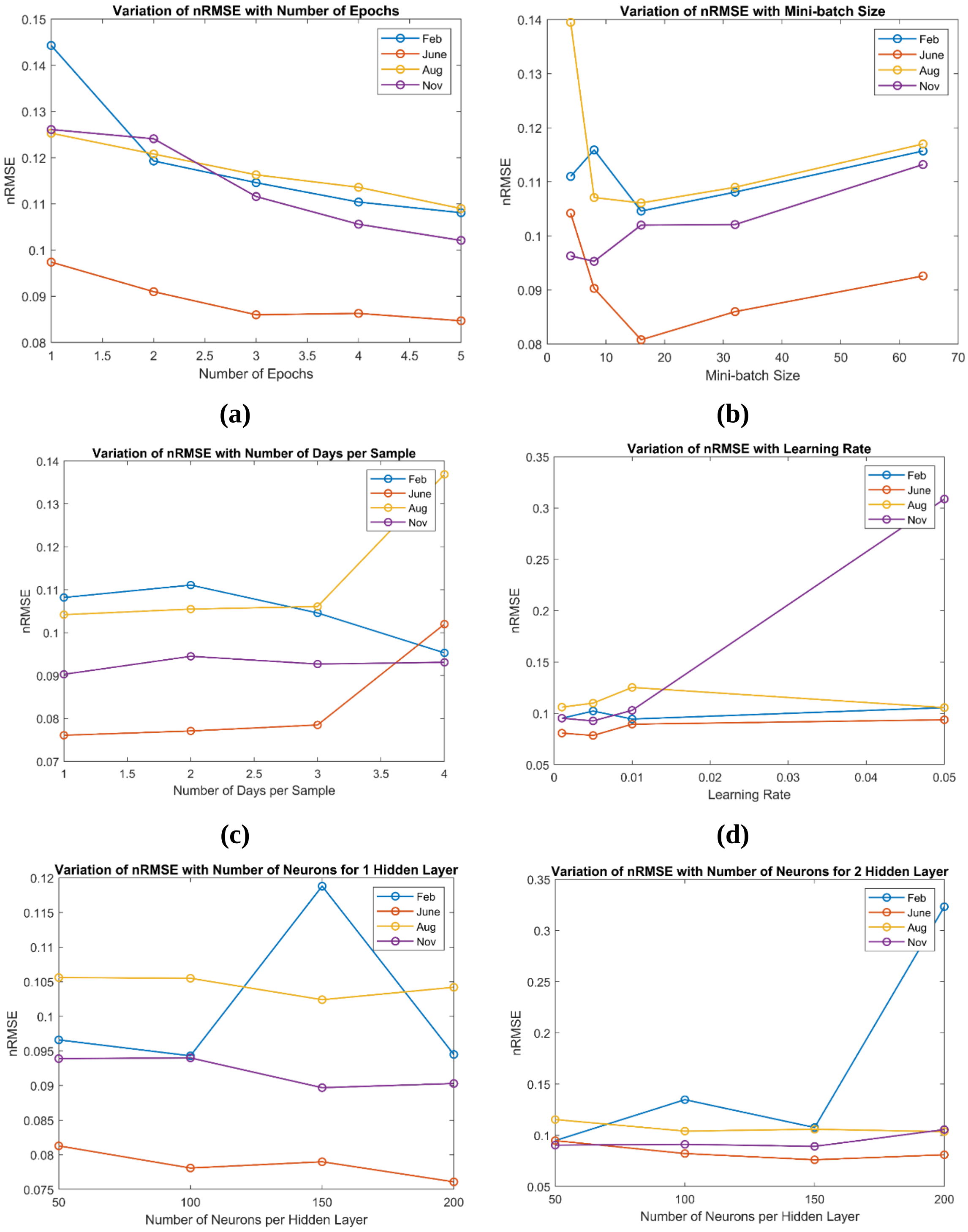
4.3 Optimisation of LSTM models

The performance of an LSTM model is influenced by several learning variables. Hence, it is crucial to tune the hyperparameters of every LSTM model created so that optimal results can be obtained. The LSTM models developed are limited to two types, namely a simplified ‘vanilla’ LSTM with a single hidden layer and a stacked LSTM with two hidden layers. With the combination of input features already determined, a base model is first created. This is done for consistency so that the hyperparameter tuning of the many models created for the four months begin with the same model settings. The configuration settings of this base model can be viewed in Table 5 below. Then the optimisation is done on the base model and one at a time each hyperparameter is varied while the others are kept fixed. The hyperparameters that will be tuned are sequence length, learning rate, mini batch size, dropout ratio, number of hidden units and number of hidden layers. The increase in the number of hidden layers is done towards the end after all other hyperparameters have been varied and the optimal selected. Finally, to ensure that the models trained have not been overfitted a study of commonly used dropout ratios were done. A few dropout ratios were applied to the optimal configuration of the simplified and stacked LSTM, that had been determined from the other hyperparameter tuning already done. The optimiser chosen was Adam optimiser and was kept the same for all four modes. Past literature has shown that it is a computationally efficient algorithm capable of finding the optimal solution through the adjustment of the learning rate [67].

Table 5: Baseline model initial configuration

Hyperparameters	Value
Number of Hidden Units	200
Number of Hidden Layers	1
Mini-batch Size	32
Number of Epochs	3
Initial Learning Rate	0.001
Input Sequence Length (days)	3
Dropout Ratio (%)	0

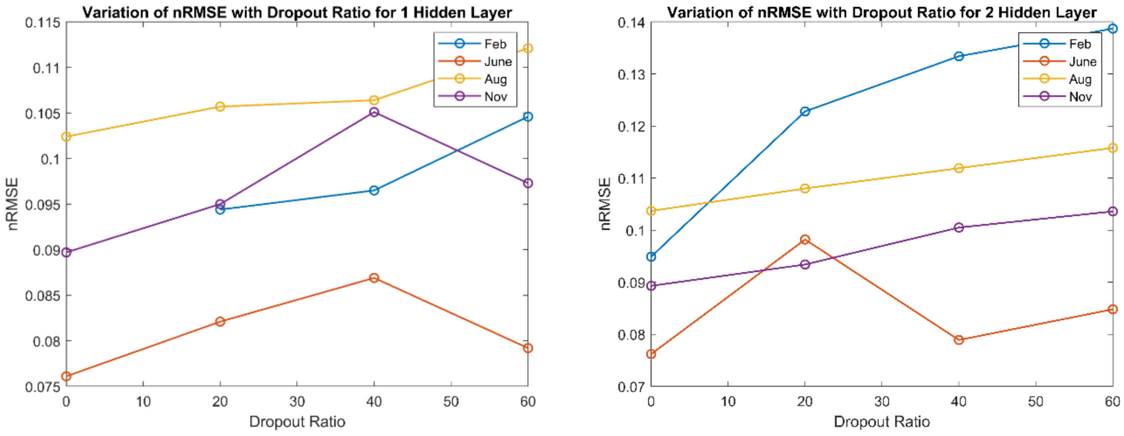
Figure 11 displayed below again demonstrate the optimisation that was done taking into account the nRMSE. From the figures displayed below, a few important trends can be identified. Firstly, all the models exhibited a decrease in error as the number of epochs was increased. Smaller mini-batch size proved to reduce the error of the model by a considerable extent. For few of the models the optimal mini-batch size was eight while for others it was 16. As for the input length sequence all the models with the exception of the model created for the February data showed reduced RMSE error when using a 1-day input sequence length. This of course corresponds to 156 timesteps provided as part of the input. For February's LSTM model the error was significantly lower when the input sequence length was 7-days. Some commonly used initial learning rate were tried on the models. Figure 11(d) infers that small learning rates lead to a more accurate model. Then the variation of the number of hidden units in the simplified LSTM showed that generally the more hidden units there are in the hidden layer, the lower the forecasting error of the model. It should also be stated that the graph does demonstrate that optimal number of hidden units is very specific to each model developed and no general rule can be used. This is highlighted by the simplified LSTM model for February where at 150 hidden units there is a sharp and sudden spike in the error of the model. We notice for the stacked LSTM models developed, at larger number of hidden units the error is generally lower, although similar to before the February model exhibited a sharp increase in error when the number of hidden units was changed from 150 units to 200 units. For all models created the introduction of a dropout ratio did not improve the accuracy of the forecast. Hence, it was concluded that the models were not overfitted during the training process and no dropout is required.


Variation of nRMSE with Number of Epochs
nRMSE
Number of Epochs
0.15
0.14
0.13
0.12
0.11
0.1
0.09
0.08
1
1.5
2
2.5
3
3.5
4
4.5
5
Feb
June
Aug
Nov
Variation of nRMSE with Mini-batch Size
nRMSE
Mini-batch Size
0.14
0.13
0.12
0.11
0.1
0.09
0.08
0
10
20
30
40
50
60
70
Feb
June
Aug
Nov
Variation of nRMSE with Number of Days per Sample
nRMSE
Number of Days per Sample
0.14
0.13
0.12
0.11
0.1
0.09
0.08
0.07
1
1.5
2
2.5
3
3.5
4
Feb
June
Aug
Nov
Variation of nRMSE with Learning Rate
nRMSE
Learning Rate
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0.01
0.02
0.03
0.04
0.05
Feb
June
Aug
Nov
Variation of nRMSE with Number of Neurons for 1 Hidden Layer
nRMSE
Number of Neurons per Hidden Layer
0.12
0.115
0.11
0.105
0.1
0.095
0.09
0.085
0.08
0.075
50
100
150
200
Feb
June
Aug
Nov
Variation of nRMSE with Number of Neurons for 2 Hidden Layer
nRMSE
Number of Neurons per Hidden Layer
0.35
0.3
0.25
0.2
0.15
0.1
0.05
50
100
150
200
Feb
June
Aug
Nov

(a) **(b)**

(c) **(d)**

(e) **(f)**



(g) **(h)**

Figure 11: For all months models (a) Study of variation of epochs, (b) Study of variation of mini-batch, (c) Study of variation of input sequence length, (d) Study of variation of learning rate, (e) Study of variation of number of hidden units for simplified LSTM, (f) Study of variation of number of hidden units for stacked LSTM, (g) Study of variation of dropout for simplified LSTM, (a) Study of variation of dropout for stacked LSTM

Even though the convergence graphs presented above show the hyperparameter settings and model configuration that would give the lowest error for every model created, this does not necessarily mean it is the optimal configuration. This is because the optimal configuration would take into account the computational time and complexity of the model too. Whilst it can be inferred with certainty from Figure 11(a) above that using more epochs will further reduce the error of the model, this would also result in extensive training time taken. From the optimisation process it was found that certain hyperparameters such as initial learning rate, input sequence length and number of epochs have greater influence on the time it takes the model to train than other hyperparameters tested. Table 6 below highlights the optimised hyperparameters for each model taking into consideration on the best possible trade-off between error and computational time.

Table 6: Optimised configuration of LSTM models and performance metrics evaluation for validation dataset for each month's models

Hyperparameters	February	June	August	November
Number of Epochs	5	3	5	5
Mini-batch Size	16	16	16	8
Learning Rate	0.01	0.005	0.001	0.005
Input Sequence Length (days)	4	1	1	1
Number of Hidden Units (Simplified LSTM)	100	200	150	150
Number of Hidden Units (Stacked LSTM)	50	150	200	150
Dropout (%)	0	0	0	0

The performance metrics calculated for the optimised models are presented below in Table 7. This quantitative assessment was done for the validation dataset. A quick look suggest that the stacked LSTM model does not really improve the accuracy of the forecast by reducing the RMSE. However, the error values are relatively similar even though the simple LSTM performs better. Only for the month of June did the stacked LSTM give a lower RMSE.

Table 7: Optimised configuration of LSTM models and performance metrics evaluation for validation dataset for each month's models

Month	Model	nMAE	nRMSE	Maximum Error	R^2
February	Simple LSTM	0.0206	0.0943	0.5850	0.9098
	Stacked LSTM	0.0080	0.0949	0.5179	0.9086
June	Simple LSTM	0.0076	0.0761	0.4243	0.9160
	Stacked LSTM	0.0139	0.0762	0.4417	0.9158
August	Simple LSTM	0.0045	0.1024	0.4426	0.8613
	Stacked LSTM	0.0049	0.1037	0.4523	0.8578
November	Simple LSTM	0.0062	0.0897	0.5655	0.8767
	Stacked LSTM	0.0112	0.0893	0.5951	0.8777

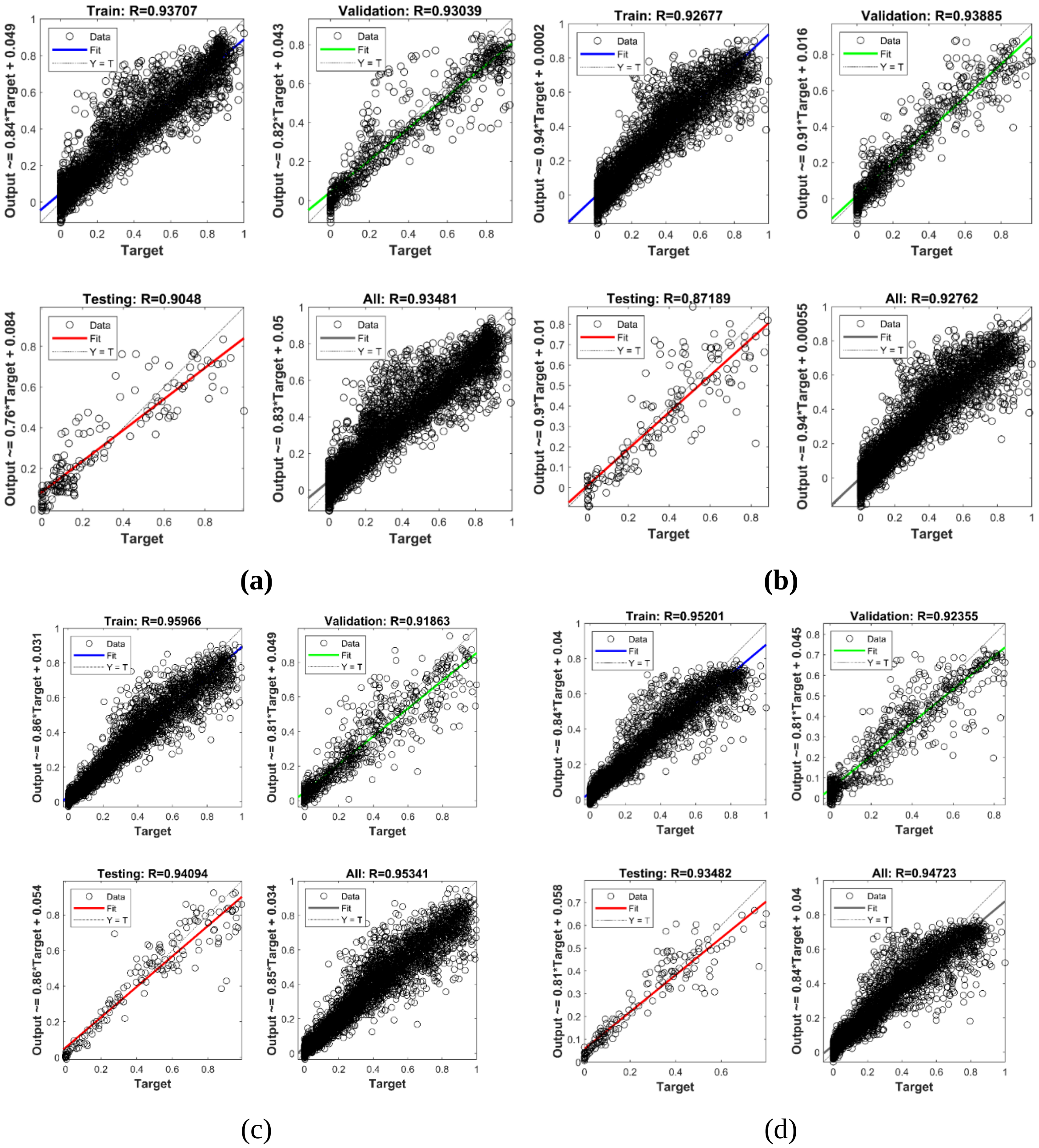
CHAPTER 5

RESULTS AND DISCUSSION

5.1 Regression plots for optimised ANN models

Regression is the comparison between the simulated prediction and the target value. A higher regression indicates a stronger fit. For ANN development, regression is the main plot that indicates how good the model has fit the data.

The key results are that for the months of February and June which are extremely sunny and sunny respectively, the ANN predicts many negative power values. This is not the case or is minimal for the other two months. Referring to the testing plot for each month it is clear that the ANNs for August and November perform better as it has a higher regression. In these plots the markers are closer to the dashed grey line which is the ideal scenario where prediction corresponds exactly to the target. The best ANN performance was for August based on the regression plot as it has the highest regression and the closest line of best fit to the true line.



(a) **(b)**

(c) (d)

Figure 12: Regression of the training, validation and testing stage for the months (a) February, (b) June, (c) August, and (d) November

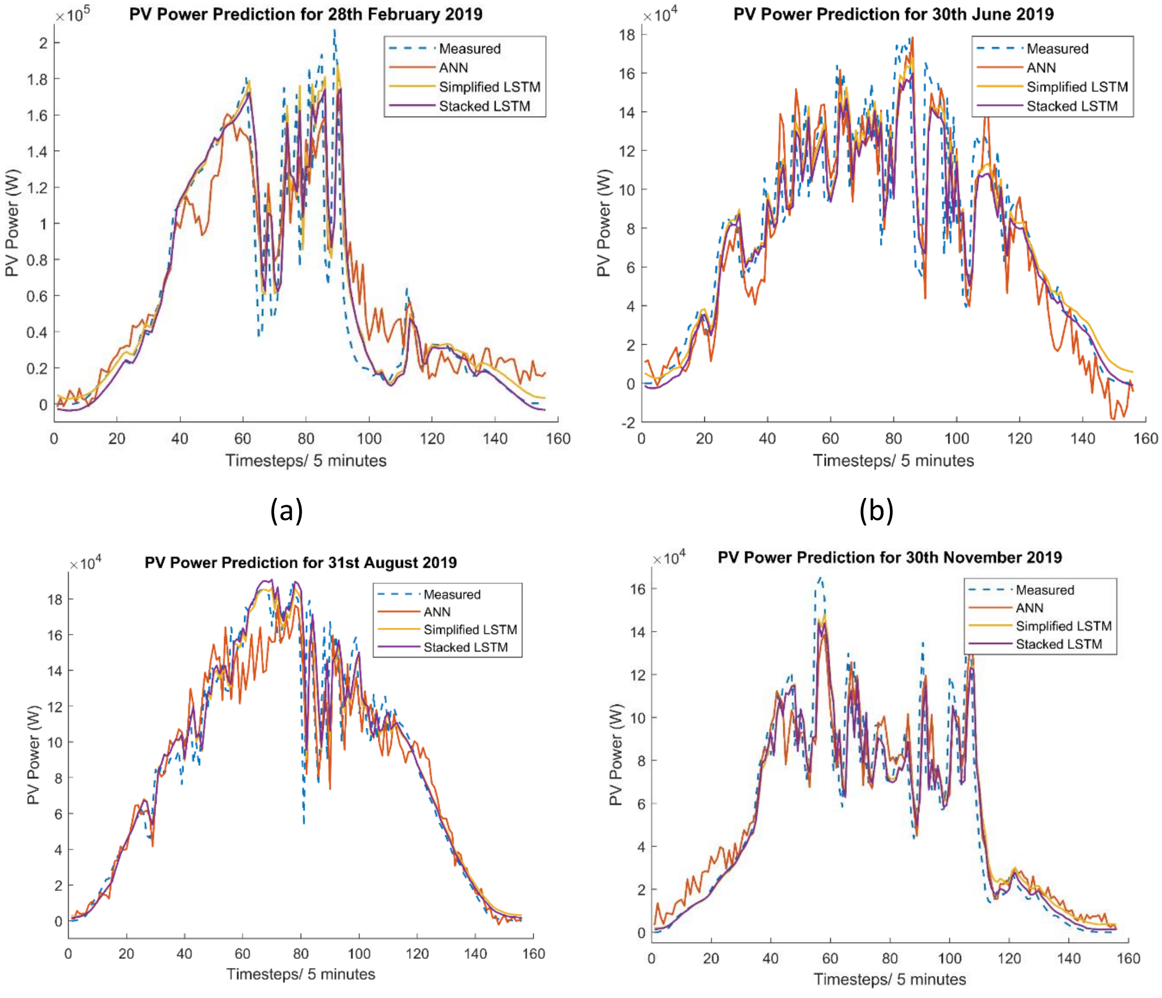
5.2 Comparison of Solar PV Power Prediction using the ANN and LSTM models

In this section, the results of the models for each of the four months are presented and discussed. The experimental results are depicted graphically in Figure 13 below to provide a qualitative assessment. It is evident that LSTM networks are superior to ANN models when it comes to dealing with time series data. It is apparent for all the months studied that the measured value and the predicted value of the LSTM models are basically fitted, and the error between them is relatively small. The graphs moved in a similar pattern and overall was able to capture the variations in the output power signal very well. This is true for both the simplified LSTM as well as the stacked LSTM. The forecast signals produced were relatively stable, not exhibiting unwarranted and constant fluctuations. These characteristics agree well with the findings by Konstantinou et al. They too found that the LSTM model was able to capture the overall behaviour of the real power output and follow the trends. Also, the prediction is similar even during days with sharp fluctuations [48].

The same is not true for the ANN model where the predicted and measured graphs were contrastingly different. Whilst the overall general trend of the measured output power signal is replicated, the prediction graph is simply too erratic. For most of the months studied, the ANN prediction graph exhibited constant fluctuations, with there often being changes to the direction of the gradient despite there being no change in the slope of the measure graph. Only for the month of August and November was the prediction of the ANN very similar to that of the actual power signal. Besides the constant fluctuation and instability of the prediction, all the models except the model for the month of November, predicted negative values of power at either the start of the day or towards the end of the day. This issue is less prominent for the months of February and August but still nonetheless present. For the month of June, the ANN model predicted large negative values which is problematic as the model cannot be relied upon to make acceptable and dependable power forecast. These findings proves that the ANN is inferior to LSTM when it comes to model fitting.

Despite the LSTM network proving to be the better and more accurate model, it too like the ANN model displayed a certain amount of error. From Figure 13 it is observed that the accuracy of the simplified LSTM and stacked LSTM deteriorates at extremely high PV power measurements, often predicting lower than the actual measured values. For February and June, the simplified LSTM was slightly better at forecasting accurately the peaks or relatively high output power. From the graph itself, it is difficult to determine with certainty which type of LSTM models suffers the worse from this behaviour for the other two months.

All the simplified LSTM models also tends to not go back down to a PV power of zero at the end of the day as it should. The graphs indicates that while the simplified LSTM prediction curve decreases as the end of the day approaches it still remains a little way off from the measured curve. For the month of November and especially August, the simplified LSTM was as close to zero as it gets, at the start and at the end of the day. For those two months, the error at either end of the day was minimal. With regard to the stacked LSTM two of the four months, namely February and June, the model predicted negative values of output power. This behaviour is similar to that of the ANN models discussed above.



(a)

(b)

(c)

(d)

Figure 13: Comparison of ANN, simplified LSTM and stacked LSTM predicted PV power output with the actual measured PV power output for test set for the month (a) February, (b) June, (c) August, and (d) November

Although the graphical examination is essential, it does not permit quantitative assessment. To further examine and analyse the proposed simplified LSTM model and stacked LSTM model, four prediction performance metrics are applied to the testing data of each of the four months. All the values presented here are on a normalised error metric scale. The normalised values are easier to interpret and provide a more meaningful representation of the results.

From the results presented in Table 8, the LSTM models performs better in terms of all the performance metrics for all months tested. On average the nRMSE of the ANN is 0.1092 with a standard deviation of 0.0233 for the four months investigated whilst the average nRMSE of the simplified LSTM model is 0.0980 with a standard deviation of 0.0187. The same pattern can be observed when comparing the average nMAE of both models. The average nMAE of ANN is 0.0171 ± 0.0081 while for the simplified LSTM is 0.0041 ± 0.0027. The lower average values of nMAE and nRMSE for the simplified LSTM indicates the difference in the prediction and the actual power is smaller which results in a more accurate forecasting model. In terms of R^2, the higher values noticed for LSTM models is a positive sign as it implies a better fit and therefore higher prediction accuracy. The average R^2 of simplified LSTM is 0.8546 ± 0.0514 which is larger in comparison to ANN for which an average R^2 of 0.8208 ± 0.0725 is determined.

The relatively low errors of the LSTM models can be attributed to the large number of timesteps used in the input sequence. According to Konstantinou et al. the larger number of timesteps is useful for LSTM cells to remember longer-term trends in the data [48]. The paper compared the results obtained with another paper by Abdel-Nasser and Mahmoud [68]. The differences in results between the two papers were attributed to the difference in number of timesteps that are used as inputs. Furthermore, another difference for the difference in performance found was the size of the networks used. Konstantinou et al. states that the slightly larger network used contributed to better performance than the smaller ones presented in Ahmed-Nasser and Mahmoud’s work. A paper published by Aslam et al. compared deep learning models such as LSTM and GRU with a machine learning model random forest regression (RFR). The deep learning models consistently outperformed the

RFR model. RMSE for the year 2018 was 6.5817 for RFR which is higher than 6.3750 for LSTM. Overall, state-of-the-art deep learning models are better than machine learning models such as ANN and RFR due to its ability to learn long term-dependencies in time series data. Machine learning models are simply not very efficient in solving these types of problems [69].

By deepening the LSTM to create a stacked LSTM, it is found that the average nRMSE, nMAE and R^2 is 0.0981 ± 0.0196, 0.0109 ± 0.0079 and 0.8555 ± 0.0507 respectively. In comparison with the simplified LSTM model developed, the nRMSE and nMAE increases although the overall R^2 is improved, albeit only slightly. A paper published by Kyeong et al. investigates the performance of single and multiple hidden layer LSTM models. They found through their study that using the multiple LSTM model makes it possible to achieve more accurate prediction. Their research determined that using the multi-LSTM model reduced the error by 0.6 % [47]. However, this contradicts the results above where it is seen that for a stacked LSTM, a multiple hidden layer LSTM model, the error nRMSE and nMAE actually increases. Our findings show that the simplified LSTM has the lowest average error. Research done by Gao et al. states that increasing the number of hidden layers does improve the training model's prediction accuracy. However, this does not necessarily translate to better prediction accuracy when the model is put on the test data. This is due to the model possibly being over-fitted and overall increasing the number of hidden layers may not be beneficial to forecasting performance [70]. The finding by Gao et al. agrees with the results obtained above where the increase in number of layers does not improve the forecasting performance.

In terms of the improvements achieved by both LSTM network over the ANN model, the highest improvements are achieved for the months of June and August. These two months were in the middle of the spectrum in terms of weather conditions, with June being slightly sunny and August being slightly rainy. The results infer that LSTM networks represents the overall trend in the data much better for dataset that consists of great deal of combination between data belonging to a variety of weather types. Nevertheless, the nMRSE for June and August for the simplified and stacked LSTM are still much higher than the nRMSE for

November. This implies that the error of the models is much higher for days with mixed weather conditions. For months February and November which are classified under one of the extremes, namely very sunny or very rainy, ANN models and LSTM models show very little differences and are both adept at modelling the input data, although LSTM is still more accurate and provides a forecast with smaller error.

This agrees with the research by Yu et al. where it was determined that on mixed days the MAE and RMSE of the forecasting models increases compared to sunny and cloudy days. This essentially means the models' prediction accuracy deteriorates. However, the LSTM still has the lowest error and follows the actual output trend most closely compared to the other models tested which included back-propagation neural network (BPNN) and SVR. For sunny and cloudy days, the LSTM's error is lower. The RMSE for a dataset in Atlanta for sunny days, cloudy days and mixed days were 28.65 W/m^2, 29.18 W/m^2 and 68.89 W/m^2 respectively. The only difference is that in the results found by Yu et al. the lowest error for the LSTM was for sunny days whereas in our research the lowest error was for the month with rainy or cloudy days [71].

Table 8: Comparison of performance metrics for ANN, simplified LSTM and stacked LSTM for test dataset for each month's models

Month	Model	nMAE	nRMSE	Maximum Error	R^2
February	ANN	0.0169	0.1204	0.5097	0.8111
	Simplified LSTM	0.0055	0.1107	0.5370	0.8404
	Stacked LSTM	0.0048	0.1125	0.5073	0.8350
June	ANN	0.0285	0.1321	0.6055	0.7223
	Simplified LSTM	0.0035	0.1141	0.4894	0.7892
	Stacked LSTM	0.0221	0.1133	0.4900	0.7940
August	ANN	0.0126	0.1060	0.4898	0.8827
	Simplified LSTM	0.0068	0.0938	0.4303	0.9066
	Stacked LSTM	0.0108	0.0951	0.4033	0.9048
November	ANN	0.0103	0.0781	0.3461	0.8671
	Simplified LSTM	0.0005	0.0734	0.3257	0.8821
	Stacked LSTM	0.0058	0.0715	0.3272	0.8882

Lastly, it should be noted that the time taken for training each individual model was recorded as is given in Table 9 below. ANNs require significantly less time to train compared to either of the two LSTM models developed. This finding about the computation time is true for other researchers. Massaoudi et al. trained various models for comparison purposes including LSTM and ANN models. Of the models created, ANN offers the least computation time. For two datasets, the LSTM took 3.63 times longer compared to the ANN [72].

It was found through the optimisation process that the selection of the optimal hyperparameters greatly reduces the training time and makes it practical for real-time forecasting. The LSTM models for all months besides February shows reasonably acceptable training time. For the month of February, the training time is much higher due to the model requiring an input sequence length amounting to four days’ worth of data for each future prediction as compared to the one-day input sequence length for the rest of the models. Comparing the training time taken for the simplified LSTM and stacked LSTM, it is clear that the stacked LSTM takes longer to train due to the added complexity of the model as a result of the introduction of another hidden layer with numerous hidden units. This ultimately means there are more calculations that the model's algorithm needs to perform during the learning process, and this is reflected in the increased training time. Overall, the time taken to train the stacked models are still acceptable.

Table 9: Comparison of computation time for training of the optimised ANN, simplified LSTM and stacked LSTM models in seconds

Model	February	June	August	November
ANN	75	84	108	30
Simplified LSTM	1361	166	260	382
Stacked LSTM	1336	318	664	690

In summary, the simplified LSTM is the optimal model as in majority of the months investigated it had a lower RMSE and MAE, indicating an overall lower error. The maximum error of both the simplified and stacked models were relatively close and reasonably low, and hence is not a deciding factor here. Since the training time of the simplified LSTM is generally much lower than the stacked LSTM and involves lower computational burden, the simplified LSTM is the best choice of model for the solar PV power output forecasting at the solar PV plant at Monash University Malaysia.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

With the growing deployment of solar energy into modern grids, the need for SPV power generation forecasting has become increasingly important due to the intermittent nature of weather. An accurate and reliable forecast model would deal with the volatility and uncertainty associated with solar PV systems. This article proposed an ANN and LSTM network for short term SPV output power forecasting by considering solar irradiation and module temperature as input features. The study discussed in detail the various pre-processing steps undertaken to prepare the raw data such as normalisation and Pearson's correlation feature selection. In addition, both the ANN and LSTM were optimised before obtaining the results and comparisons were made. Three error metrics, nMAE, nRMSE and R^2 were used to test and measure the accuracy of each algorithm.

A visual examination of the results plotted suggest that the LSTM network reacts better to each fluctuation and follows the trend of the actual output power signal more closely as compared to the ANN model. The results also reflect that as the amount of solar irradiation changes, the amount of solar PV power generation changes in an identical pattern. However, the accuracy of both the ANN and LSTM suffer when the solar PV power generation is large. It was found that fine-tuning the predictive models enhanced the accuracy of the forecast tremendously. The nRMSE was much lower for both LSTM compared to ANN regardless of the month tested. The average nRMSE for ANN, simplified LSTM and stacked LSTM were 0.1092, 0.0980 and 0.0981 respectively. In addition, the average R^2 for ANN, simplified LSTM and stacked LSTM were 0.8208, 0.8546 and 0.8556 respectively. It can be concluded that LSTM showed very high accuracy and low errors. Overall, it is superior to ANN models in dealing with time series regression problems. With regards to the type of LSTM model, the results suggest that the simplified LSTM is the optimal model as it has the lowest errors and takes less time to train making it ideal for real time solar PV power forecasting.

As for the future works that can be done to improve the LSTM forecasting model, different architectures of LSTM models could be utilised such as bidirectional LSTM and convolutional LSTM. Besides that, significant improvements could be achieved by hybridisation with other optimisation methods such as PSO and ant-colony optimisation or with other DL methods such as convolutional neural networks and auto encoder. Lastly, the forecast model could be extended to longer term PV forecasting.

References

[1] E. Ogliari and A. Nespoli, "Photovoltaic Plant Output Power Forecast by Means of Hybrid Artificial Neural Networks," in *A Practical Guide for Advanced Methods in Solar Photovoltaic Systems*: Springer, 2020, pp. 203-222.

[2] IEA. "Renewable electricity capacity additions, 2007-2021, updated IEA forecast." https://www.iea.org/data-and-statistics/charts/renewable-electricity-capacity-additions-2007-2021-updated-iea-forecast (accessed 6 June 2020.

[3] A. S. B. M. Shah, H. Yokoyama, and N. Kakimoto, "High-precision forecasting model of solar irradiance based on grid point value data analysis for an efficient photovoltaic system," *IEEE Transactions on Sustainable Energy*, vol. 6, no. 2, pp. 474-481, 2015.

[4] R. Ahmed, V. Sreeram, Y. Mishra, and M. Arif, "A review and evaluation of the state-of-the-art in PV solar power forecasting: Techniques and optimization," *Renewable and Sustainable Energy Reviews*, vol. 124, p. 109792, 2020.

[5] IEA. "Net solar PV capacity additions, main and accelerated case, 2012-2023." https://www.iea.org/data-and-statistics/charts/net-solar-pv-capacity-additions-main-and-accelerated-case-2012-2023 (accessed.

[6] M. Rana and A. Rahman, "Multiple steps ahead solar photovoltaic power forecasting based on univariate machine learning models and data re-sampling," *Sustainable Energy, Grids and Networks*, vol. 21, p. 100286, 2020.

[7] G. M. Yagli, D. Yang, and D. Srinivasan, "Automatic hourly solar forecasting using machine learning models," *Renewable and Sustainable Energy Reviews*, vol. 105, pp. 487-498, 2019.

[8] S. Aslam, H. Herodotou, N. Ayub, and S. M. Mohsin, "Deep Learning based Techniques to Enhance the Performance of Microgrids: A Review," in *2019 International Conference on Frontiers of Information Technology (FIT)*, 2019: IEEE, pp. 116-1165.

[9] A. Nespoli *et al.*, "Day-ahead photovoltaic forecasting: A comparison of the most effective techniques," *Energies*, vol. 12, no. 9, p. 1621, 2019.

[10] M. Moreira, P. Balestrassi, A. Paiva, P. Ribeiro, and B. Bonatto, "Design of experiments using artificial neural network ensemble for photovoltaic generation forecasting," *Renewable and Sustainable Energy Reviews,* vol. 135, p. 110450, 2021.

[11] M. Behrang, E. Assareh, A. Ghanbarzadeh, and A. Noghrehabadi, "The potential of different artificial neural network (ANN) techniques in daily global solar radiation modeling based on meteorological data," *Solar Energy,* vol. 84, no. 8, pp. 1468-1480, 2010.

[12] A. Koca, H. F. Oztop, Y. Varol, and G. O. Koca, "Estimation of solar radiation using artificial neural networks with different input parameters for Mediterranean region of Anatolia in Turkey," *Expert Systems with Applications,* vol. 38, no. 7, pp. 8756-8762, 2011.

[13] H. Z. Al Garni, A. Awasthi, and M. A. Ramli, "Optimal design and analysis of grid-connected photovoltaic under different tracking systems using HOMER," *Energy conversion and management,* vol. 155, pp. 42-57, 2018.

[14] M. Mishra, J. Nayak, B. Naik, and A. Abraham, "Deep learning in electrical utility industry: A comprehensive review of a decade of research," *Engineering Applications of Artificial Intelligence,* vol. 96, p. 104000, 2020.

[15] U. K. Das *et al.*, "Forecasting of photovoltaic power generation and model optimization: A review," *Renewable and Sustainable Energy Reviews,* vol. 81, pp. 912-928, 2018.

[16] T. Saengsuwan, "Prediction Model for Solar PV Rooftop Production," *Journal of Renewable Energy and Smart Grid Technology,* vol. 15, no. 2, pp. 16-25, 2020.

[17] M. A. Ramli, A. Hiendro, K. Sedraoui, and S. Twaha, "Optimal sizing of grid-connected photovoltaic energy system in Saudi Arabia," *Renewable Energy,* vol. 75, pp. 489-495, 2015.

[18] A. Fouilloy *et al.*, "Solar irradiation prediction with machine learning: Forecasting models selection method depending on weather variability," *Energy,* vol. 165, pp. 620-629, 2018.

[19] C. Chen, S. Duan, T. Cai, and B. Liu, "Online 24-h solar power forecasting based on weather type classification using artificial neural network," *Solar energy,* vol. 85, no. 11, pp. 2856-2870, 2011.

[20] N. Premalatha and A. Valan Arasu, "Prediction of solar radiation for solar systems by using ANN models with different back propagation algorithms," *Journal of applied research and technology,* vol. 14, no. 3, pp. 206-214, 2016.

[21] P. Singla, M. Duhan, and S. Saroha, "A comprehensive review and analysis of solar forecasting techniques," *Frontiers in Energy,* pp. 1-37, 2021.

[22] C. Wan, J. Zhao, Y. Song, Z. Xu, J. Lin, and Z. Hu, "Photovoltaic and solar power forecasting for smart grid energy management," *CSEE Journal of Power and Energy Systems,* vol. 1, no. 4, pp. 38-46, 2015.

[23] H. K. Yadav, Y. Pal, and M. M. Tripathi, "Photovoltaic power forecasting methods in smart power grid," in *2015 Annual IEEE India Conference (INDICON),* 2015: IEEE, pp. 1-6.

[24] Z. Wang, F. Wang, and S. Su, "Solar irradiance short-term prediction model based on BP neural network," *Energy Procedia,* vol. 12, pp. 488-494, 2011.

[25] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine,* vol. 29, no. 6, pp. 82-97, 2012.

[26] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," *arXiv preprint arXiv:1312.6026,* 2013.

[27] S. Sobri, S. Koohi-Kamali, and N. A. Rahim, "Solar photovoltaic generation forecasting methods: A review," *Energy Conversion and Management,* vol. 156, pp. 459-497, 2018.

[28] U. K. Das *et al.*, "SVR-based model to forecast PV power generation under different weather conditions," *Energies,* vol. 10, no. 7, p. 876, 2017.

[29] K. A. Baharin, H. Abdul Rahman, M. Y. Hassan, and C. K. Gan, "Short-term forecasting of solar photovoltaic output power for tropical climate using ground-based measurement data," *Journal of renewable and sustainable energy,* vol. 8, no. 5, p. 053701, 2016.

[30] J. Son, Y. Park, J. Lee, and H. Kim, "Sensorless PV power forecasting in grid-connected buildings through deep learning," *Sensors,* vol. 18, no. 8, p. 2529, 2018.

[31] K. Wang, X. Qi, and H. Liu, "A comparison of day-ahead photovoltaic power forecasting models based on deep learning neural network," *Applied Energy,* vol. 251, p. 113315, 2019.

[32] H. Wang *et al.*, "Deterministic and probabilistic forecasting of photovoltaic power based on deep convolutional neural network," *Energy conversion and management,* vol. 153, pp. 409-422, 2017.

[33] K. Almohammadi, H. Hagras, D. Alghazzawi, and G. Aldabbagh, "A survey of artificial intelligence techniques employed for adaptive educational systems within e-learning platforms," *Journal of Artificial Intelligence and Soft Computing Research,* vol. 7, no. 1, pp. 47-64, 2017.

[34] W. Bao, J. Yue, and Y. Rao, "A deep learning framework for financial time series using stacked autoencoders and long-short term memory," *PloS one,* vol. 12, no. 7, p. e0180944, 2017.

[35] S. A. B. Jumaat, F. Crocker, M. H. Abd Wahab, and N. H. B. M. Radzi, "Investigate the photovoltaic (PV) module performance using Artificial Neural Network (ANN)," in *2016 IEEE Conference on Open Systems (ICOS),* 2016: IEEE, pp. 59-64.

[36] S. Theocharides, G. Makrides, V. Venizelou, P. Kaimakis, and G. Georghiou, "PV production forecasting model based on artificial neural networks (ANN)," in *33rd Eur. Photovolt. Sol. Energy Conf,* 2017, pp. 1830-1894.

[37] A. K. Sahoo and S. K. Sahoo, "Energy forecasting for grid connected MW range solar PV system," in *2016 7th India international conference on power electronics (IICPE),* 2016: IEEE, pp. 1-6.

[38] M. Ding, L. Wang, and R. Bi, "An ANN-based approach for forecasting the power output of photovoltaic system," *Procedia Environmental Sciences,* vol. 11, pp. 1308-1315, 2011.

[39] S. Leva, A. Dolara, F. Grimaccia, M. Mussetta, and E. Ogliari, "Analysis and validation of 24 hours ahead neural network forecasting of photovoltaic output power," *Mathematics and computers in simulation,* vol. 131, pp. 88-100, 2017.

[40] M. Omar, A. Dolara, G. Magistrati, M. Mussetta, E. Ogliari, and F. Viola, "Day-ahead forecasting for photovoltaic power using artificial neural networks ensembles,"

in *2016 IEEE International Conference on Renewable Energy Research and Applications (ICRERA)*, 2016: IEEE, pp. 1152-1157.

[41] P. Kumar, N. Sing, and M. Ansari, "Solar radiation forecasting using artificial neural network with different meteorological variables," *Communication and Computing Systems-Prasad (et al)*, pp. 9781315364094-88, 2017.

[42] T. Demirdelen, I. O. Aksu, B. Esenboga, K. Aygul, F. Ekinci, and M. Bilgili, "A new method for generating short-term power forecasting based on artificial neural networks and optimization methods for solar photovoltaic power plants," in *Solar photovoltaic power plants*: Springer, 2019, pp. 165-189.

[43] Z. Ncane and A. Saha, "Forecasting Solar Power Generation Using Fuzzy Logic and Artificial Neural Network," in *2019 Southern African Universities Power Engineering Conference/Robotics and Mechatronics/Pattern Recognition Association of South Africa (SAUPEC/RobMech/PRASA)*, 2019: IEEE, pp. 518-523.

[44] I. Jebli, F.-z. Belouadha, M. I. Kabbaj, and A. Tilioua, "Deep Learning based Models for Solar Energy Prediction," *ASTESJ*, Journal Article vol. 6, no. 1, pp. 349-355, 2021. [Online]. Available: internal-pdf://ASTESJ_060140.pdf.

[45] D. Huang *et al.*, "Prediction of Solar Photovoltaic Power Generation Based on MLP and LSTM neural networks," in *2020 IEEE 4th Conference on Energy Internet and Energy System Integration (EI2)*: IEEE, pp. 2744-2748.

[46] C.-H. Liu, J.-C. Gu, and M.-T. Yang, "A Simplified LSTM Neural Networks for One Day-Ahead Solar Power Forecasting," *IEEE Access*, vol. 9, pp. 17174-17195, 2021.

[47] M. K. Park, J. M. Lee, W. H. Kang, J. M. Choi, and K. H. Lee, "Predictive model for PV power generation using RNN (LSTM)," *Journal of Mechanical Science and Technology*, vol. 35, no. 2, pp. 795-803, 2021.

[48] M. Konstantinou, S. Peratikou, and A. G. Charalambides, "Solar Photovoltaic Forecasting of Power Output Using LSTM Networks," *Atmosphere*, vol. 12, no. 1, p. 124, 2021.

[49] M. S. Hossain and H. Mahmood, "Short-term photovoltaic power forecasting using an LSTM neural network," in *2020 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, 2020: IEEE, pp. 1-5.

[50] Y. Li, F. Ye, Z. Liu, Z. Wang, and Y. Mao, "A Short-Term Photovoltaic Power Generation Forecast Method Based on LSTM," *Mathematical Problems in Engineering,* vol. 2021, 2021.

[51] B.-S. Kwon, R.-J. Park, and K.-B. Song, "Short-term load forecasting based on deep neural networks using LSTM layer," *Journal of Electrical Engineering & Technology,* vol. 15, pp. 1501-1509, 2020.

[52] B. Chen, P. Lin, Y. Lai, S. Cheng, Z. Chen, and L. Wu, "Very-short-term power prediction for pv power plants using a simple and effective rcc-lstm model based on short term multivariate historical datasets," *Electronics,* vol. 9, no. 2, p. 289, 2020.

[53] W. Bendali, I. Saber, B. Bourachdi, M. Boussetta, and Y. Mourad, "Deep Learning Using Genetic Algorithm Optimization for Short Term Solar Irradiance Forecasting," in *2020 Fourth International Conference On Intelligent Computing in Data Sciences (ICDS)*, 2020: IEEE, pp. 1-8.

[54] A. Mellit, A. Massi Pavan, E. Ogliari, S. Leva, and V. Lughi, "Advanced Methods for Photovoltaic Output Power Forecasting: A Review," *Applied Sciences,* vol. 10, no. 2, p. 487, 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/2/487.

[55] M. Z. Saleheen, A. A. Salema, S. M. M. Islam, C. R. Sarimuthu, and M. Z. Hasan, "A target-oriented performance assessment and model development of a grid-connected solar PV (GCPV) system for a commercial building in Malaysia," *Renewable Energy,* vol. 171, pp. 371-382, 2021.

[56] "Climate and temperature development in Malaysia." https://www.worlddata.info/asia/malaysia/climate.php (accessed 06-May-2021.

[57] D. Thara, B. PremaSudha, and F. Xiong, "Auto-detection of epileptic seizure events using deep neural network with different feature scaling techniques," *Pattern Recognition Letters,* vol. 128, pp. 544-550, 2019.

[58] A. R. Pazikadin, D. Rifai, K. Ali, M. Z. Malik, A. N. Abdalla, and M. A. Faraj, "Solar irradiance measurement instrumentation and power solar generation forecasting based on Artificial Neural Networks (ANN): A review of five years research trend," *Science of The Total Environment,* vol. 715, p. 136848, 2020.

[59] M. Seyedmahmoudian *et al.*, "State of the art artificial intelligence-based MPPT techniques for mitigating partial shading effects on PV systems–A review," *Renewable and Sustainable Energy Reviews,* vol. 64, pp. 435-455, 2016.

[60] K. Wang, X. Qi, and H. Liu, "Photovoltaic power forecasting based LSTM-Convolutional Network," *Energy,* vol. 189, p. 116225, 2019.

[61] U. Kumar, S. Mishra, and S. Madichetty, "An Efficient SPV Power Forecasting using Hybrid Wavelet and Genetic Algorithm based LSTM Deep Learning Model," in *2020 21st National Power Systems Conference (NPSC)*, 2020: IEEE, pp. 1-6.

[62] S. Sengupta *et al.*, "A review of deep learning with special emphasis on architectures, applications and recent trends," *Knowledge-Based Systems,* vol. 194, p. 105596, 2020.

[63] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *International conference on machine learning*, 2015: PMLR, pp. 2342-2350.

[64] T. Khatib and W. Elmenreich, "A model for hourly solar radiation data generation from daily solar radiation data using a generalized regression artificial neural network," *International Journal of Photoenergy,* vol. 2015, 2015.

[65] A. du Plessis, J. Strauss, and A. Rix, "Short-term solar power forecasting: Investigating the ability of deep learning models to capture low-level utility-scale Photovoltaic system behaviour," *Applied Energy,* vol. 285, p. 116395, 2021.

[66] M. Negnevitsky and A. Intelligence, "A guide to intelligent systems," *Artificial Intelligence, 2nd edition, pearson Education,* 2005.

[67] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[68] M. Abdel-Nasser and K. Mahmoud, "Accurate photovoltaic power forecasting models using deep LSTM-RNN," *Neural Computing and Applications,* vol. 31, no. 7, pp. 2727-2740, 2019.

[69] M. Aslam, J.-M. Lee, M. R. Altaha, S.-J. Lee, and S. Hong, "AE-LSTM Based Deep Learning Model for Degradation Rate Influenced Energy Estimation of a PV System," *Energies,* vol. 13, no. 17, p. 4373, 2020.

[70] M. Gao, J. Li, F. Hong, and D. Long, "Short-term forecasting of power production in a large-scale photovoltaic plant based on LSTM," *Applied Sciences*, vol. 9, no. 15, p. 3192, 2019.

[71] Y. Yu, J. Cao, and J. Zhu, "An LSTM short-term solar irradiance forecasting under complicated weather conditions," *IEEE Access*, vol. 7, pp. 145651-145666, 2019.

[72] M. Massaoudi *et al.*, "An effective hybrid NARX-LSTM model for point and interval PV power forecasting," *IEEE Access*, vol. 9, pp. 36571-36588, 2021.

Appendix

Appendix A: ANN Optimisation

Table 10: Performance metrics of validation dataset during optimisation stage for month of February

Number of Neurons	nMAE	nMSE	nRMSE	Max Error	R^2
1 Hidden Layer					
10	0.0285	0.0140	0.1185	0.4659	0.8546
20	0.0045	0.0142	0.119	0.4979	0.8123
30	0.0174	0.0211	0.1454	0.5919	0.7811
40	0.0682	0.0254	0.1595	0.5167	0.7365
50	0.0314	0.0272	0.1650	0.5666	0.7179
2 Hidden Layers					
10	0.0218	0.0185	0.1359	0.4897	0.8087
20	0.0395	0.0288	0.1696	0.6401	0.7020
30	0.0365	0.0240	0.1549	0.5726	0.7515
40	0.0490	0.0292	0.1708	0.8321	0.6979
50	0.0002	0.0220	0.1483	0.6745	0.7724

Table 11: Performance metrics of validation dataset during optimisation stage for month of June

Number of Neurons	nMAE	nMSE	nRMSE	Max Error	R^2
1 Hidden Layer					
10	0.0090	0.0083	0.0912	0.4729	0.8791
20	0.0002	0.0106	0.1027	0.4909	0.8468
30	0.0368	0.0203	0.1425	0.5695	0.7050
40	0.0547	0.0168	0.1297	0.5465	0.7558

50	0.0144	0.0138	0.1177	0.4043	0.7990
2 Hidden Layers					
10	0.0162	0.0092	0.0961	0.4206	0.8660
20	0.0146	0.0113	0.1063	0.5101	0.8361
30	0.0507	0.0178	0.1335	0.4972	0.7412
40	0.0160	0.0228	0.1510	0.4990	0.6689
50	0.0151	0.0227	0.1507	0.6628	0.6702

Table 12: Performance metrics of validation dataset during optimisation stage for month of August

Number of Neurons	nMAE	nMSE	nRMSE	Max Error	R^2
1 Hidden Layer					
10	0.0030	0.0132	0.1148	0.4418	0.8254
20	0.0045	0.0142	0.1190	0.4979	0.8123
30	0.0201	0.0149	0.1220	0.5038	0.8028
40	0.0090	0.0167	0.1290	0.6396	0.7794
50	0.0324	0.0176	0.1326	0.5139	0.7672
2 Hidden Layers					
10	0.0094	0.0120	0.1095	0.4878	0.8412
20	0.0063	0.0154	0.1243	0.5545	0.7954
30	0.0350	0.0240	0.1549	0.5335	0.6823
40	0.0135	0.0166	0.1287	0.5727	0.7804
50	0.0603	0.0276	0.1662	0.6374	0.6341

Table 13: Performance metrics of validation dataset during optimisation stage for month of November

Number of Neurons	nMAE	nMSE	nRMSE	Max Error	R^2
1 Hidden Layer					

10	0.0087	0.0148	0.1215	0.4995	0.7735
20	0.0258	0.0109	0.1044	0.4339	0.8330
30	0.0096	0.0130	0.1142	0.4603	0.8000
40	0.0039	0.0150	0.1226	0.4048	0.7694
50	0.0092	0.0174	0.1319	0.6009	0.7331
2 Hidden Layers					
10	0.0063	0.0098	0.0988	0.6000	0.8503
20	0.0027	0.0144	0.1198	0.6626	0.7799
30	0.0162	0.0138	0.1173	0.4985	0.7890
40	0.0062	0.0141	0.1188	0.3876	0.7837
50	0.0279	0.0181	0.1346	0.5895	0.7220

Appendix B: LSTM Optimisation

Table 14: Performance metrics of validation dataset during optimisation stage for month of February

Number of Epochs	nMAE	nMSE	nRMSE	Max Error	R^2
1	0.0414	0.0208	0.1443	0.5464	0.7944
2	0.0176	0.0142	0.1193	0.5076	0.8595
3	0.0051	0.0131	0.1146	0.4953	0.8702
4	0.0126	0.0122	0.1104	0.4911	0.8797
5	0.0006	0.0117	0.1081	0.5225	0.8847
Mini-batch Size	nMAE	nMSE	nRMSE	Max Error	R^2
4	0.0375	0.0123	0.1110	0.6452	0.8783
8	0.0413	0.0134	0.1159	0.6448	0.8674
16	0.0223	0.0109	0.1046	0.5340	0.8919
32	0.0006	0.0117	0.1081	0.5225	0.8847
64	0.0193	0.0134	0.1157	0.4712	0.8679
Learning Rate	nMAE	nMSE	nRMSE	Max Error	R^2
0.001	0.0103	0.0091	0.0953	0.5716	0.9080
0.005	0.0142	0.0105	0.1023	0.6526	0.8940
0.01	0.0220	0.0089	0.0945	0.5549	0.9095
0.05	0.0291	0.0112	0.1056	0.6667	0.8869
Input Sequence Length (days)	nMAE	nMSE	nRMSE	Max Error	R^2
1	0.0043	0.0117	0.1082	0.6083	0.8788
2	0.0208	0.0123	0.1111	0.5682	0.8776
3	0.0223	0.0109	0.1046	0.5340	0.8919
4	0.0103	0.0091	0.0953	0.5716	0.9080
Number of Hidden Units (Simplified LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
50	0.0296	0.0093	0.0966	0.5858	0.9053
100	0.0206	0.0089	0.0943	0.5850	0.9098

150	0.0665	0.0141	0.1188	0.6820	0.8570
200	0.0220	0.0089	0.0945	0.5549	0.9095
Number of Hidden Units (Stacked LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
50	0.0080	0.0090	0.0949	0.5179	0.9086
100	0.0716	0.0182	0.1349	0.6339	0.8155
150	0.0499	0.0116	0.1077	0.5452	0.8824
200	0.0765	0.1045	0.3233	0.4745	-0.0596
Dropout Ratio (Simplified LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
0	0.0206	0.0089	0.0943	0.5850	0.9098
20	0.0025	0.0089	0.0944	0.5952	0.9096
40	0.0103	0.0093	0.0965	0.5692	0.9056
60	0.0223	0.0109	0.1046	0.5340	0.8919
Dropout Ratio (Stacked LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
0	0.0080	0.0090	0.0949	0.5179	0.9086
20	0.0287	0.0151	0.1228	0.5648	0.8470
40	0.0354	0.0178	0.1334	0.5626	0.8196
60	0.0400	0.0192	0.1387	0.5869	0.8050

Table 15: Performance metrics of validation dataset during optimisation stage for month of June

Number of Epochs	nMAE	nMSE	nRMSE	Max Error	R^2
1	0.0113	0.0095	0.0974	0.4814	0.8627
2	0.0022	0.0083	0.091	0.4544	0.8802
3	0.0033	0.0074	0.086	0.4342	0.8929
4	0.0178	0.0074	0.0863	0.4599	0.8923
5	0.0147	0.0072	0.0847	0.4532	0.8929
Mini-batch Size	nMAE	nMSE	nRMSE	Max Error	R^2
4	0.0203	0.0109	0.1042	0.4783	0.8428

8	0.0374	0.0082	0.0903	0.4811	0.8820
16	0.0021	0.0065	0.0808	0.4348	0.9056
32	0.0033	0.0074	0.0860	0.4342	0.8929
64	0.0041	0.0086	0.0926	0.4851	0.8758
Learning Rate	nMAE	nMSE	nRMSE	Max Error	R^2
0.001	0.0021	0.0065	0.0808	0.4348	0.9056
0.005	0.0087	0.0062	0.0785	0.4217	0.9107
0.01	0.0337	0.0080	0.0895	0.3840	0.8841
0.05	0.0440	0.0088	0.0938	0.3714	0.8727
Input Sequence Length (days)	nMAE	nMSE	nRMSE	Max Error	R^2
1	0.0076	0.0058	0.0761	0.4243	0.9160
2	0.0076	0.0059	0.0771	0.4276	0.9146
3	0.0087	0.0062	0.0785	0.4217	0.9107
4	0.0548	0.0104	0.1020	0.3464	0.8520
Number of Hidden Units (Simplified LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
50	0.0196	0.0066	0.0813	0.4597	0.9040
100	0.0084	0.0061	0.0781	0.4413	0.9116
150	0.0237	0.0062	0.0790	0.4953	0.9096
200	0.0076	0.0058	0.0761	0.4243	0.9160
Number of Hidden Units (Stacked LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
50	0.0456	0.0090	0.0949	0.5097	0.8695
100	0.0335	0.0068	0.0823	0.3986	0.9018
150	0.0139	0.0058	0.0762	0.4417	0.9158
200	0.0088	0.0066	0.0811	0.4574	0.9046
Dropout Ratio (Simplified LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
0	0.0076	0.0058	0.0761	0.4243	0.9160
20	0.0274	0.0067	0.0821	0.3934	0.9022
40	0.0383	0.0076	0.0869	0.3775	0.8905

60	0.0161	0.0063	0.0792	0.4033	0.9090
Dropout Ratio (Stacked LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
0	0.0139	0.0058	0.0762	0.4417	0.9158
20	0.0554	0.0096	0.0982	0.3528	0.8602
40	0.0060	0.0062	0.0789	0.4240	0.9098
60	0.0169	0.0072	0.0848	0.4114	0.8958

Table 16: Performance metrics of validation dataset during optimisation stage for month of August

Number of Epochs	nMAE	nMSE	nRMSE	Max Error	R^2
1	0.0150	0.0157	0.1253	0.5071	0.8015
2	0.0144	0.0146	0.1208	0.5267	0.8156
3	0.0090	0.0135	0.1163	0.5052	0.8289
4	0.0137	0.0129	0.1136	0.4947	0.8368
5	0.0044	0.0119	0.1090	0.4455	0.8499
Mini-batch Size	nMAE	nMSE	nRMSE	Max Error	R^2
4	0.0224	0.0195	0.1395	0.5824	0.7540
8	0.0259	0.0115	0.1071	0.4101	0.8550
16	0.0045	0.0113	0.1061	0.4536	0.8575
32	0.0044	0.0119	0.1090	0.4455	0.8499
64	0.0077	0.0137	0.1170	0.4817	0.8270
Learning Rate	nMAE	nMSE	nRMSE	Max Error	R^2
0.001	0.0045	0.0113	0.1061	0.4536	0.8575
0.005	0.0381	0.0121	0.1100	0.4007	0.8471
0.01	0.0648	0.0157	0.1253	0.5215	0.8015
0.05	0.0000	0.0112	0.1057	0.4622	0.8586
Input Sequence Length (days)	nMAE	nMSE	nRMSE	Max Error	R^2
1	0.0216	0.0109	0.1042	0.4195	0.8563
2	0.0182	0.0111	0.1055	0.4323	0.8575

3	0.0045	0.0113	0.1061	0.4536	0.8575
4	0.0042	0.0187	0.1369	0.6004	0.7626
Number of Hidden Units (Simplified LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
50	0.0033	0.0112	0.1056	0.4501	0.8524
100	0.0134	0.0111	0.1055	0.4376	0.8529
150	0.0045	0.0105	0.1024	0.4426	0.8613
200	0.0216	0.0109	0.1042	0.4195	0.8563
Number of Hidden Units (Stacked LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
50	0.0375	0.0133	0.1155	0.4954	0.8234
100	0.0047	0.0109	0.1042	0.4561	0.8563
150	0.0147	0.0112	0.1060	0.4356	0.8513
200	0.0049	0.0108	0.1037	0.4523	0.8578
Dropout Ratio (Simplified LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
0	0.0045	0.0105	0.1024	0.4426	0.8613
20	0.0017	0.0112	0.1057	0.4590	0.8522
40	0.0039	0.0113	0.1064	0.4652	0.8503
60	0.0159	0.0126	0.1121	0.5048	0.8338
Dropout Ratio (Stacked LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
0	0.0049	0.0108	0.1037	0.4523	0.8578
20	0.0059	0.0117	0.1080	0.4540	0.8457
40	0.0195	0.0125	0.1119	0.4418	0.8342
60	0.0266	0.0134	0.1158	0.4519	0.8225

Table 17: Performance metrics of validation dataset during optimisation stage for month of November

Number of Epochs	nMAE	nMSE	nRMSE	Max Error	R^2
1	0.0062	0.0159	0.1261	0.504	0.7552

2	0.0297	0.0154	0.1241	0.4701	0.7628
3	0.0215	0.0125	0.1116	0.5874	0.8081
4	0.0147	0.0112	0.1056	0.5843	0.8281
5	0.0067	0.0104	0.1021	0.5674	0.8393
Mini-batch Size	nMAE	nMSE	nRMSE	Max Error	R^2
4	0.0146	0.0093	0.0963	0.5526	0.8571
8	0.0001	0.0091	0.0953	0.5959	0.8601
16	0.0259	0.0104	0.1020	0.5700	0.8398
32	0.0067	0.0104	0.1021	0.5674	0.8393
64	0.0068	0.0128	0.1132	0.5546	0.8026
Learning Rate	nMAE	nMSE	nRMSE	Max Error	R^2
0.001	0.0001	0.0091	0.0953	0.5959	0.8601
0.005	0.0038	0.0086	0.0927	0.5667	0.8675
0.01	0.0369	0.0106	0.1030	0.6258	0.8366
0.05	0.2873	0.0953	0.3088	0.7752	-0.4865
Input Sequence Length (days)	nMAE	nMSE	nRMSE	Max Error	R^2
1	0.0048	0.0082	0.0903	0.5603	0.8749
2	0.0139	0.0089	0.0945	0.5206	0.8624
3	0.0038	0.0086	0.0927	0.5667	0.8675
4	0.0010	0.0087	0.0931	0.5918	0.8690
Number of Hidden Units (Simplified LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
50	0.0230	0.0088	0.0939	0.5903	0.8647
100	0.0279	0.0088	0.0940	0.6112	0.8645
150	0.0062	0.0080	0.0897	0.5655	0.8767
200	0.0048	0.0082	0.0903	0.5603	0.8749
Number of Hidden Units (Stacked LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
50	0.0182	0.0082	0.0905	0.5845	0.8743
100	0.0080	0.0083	0.0913	0.6342	0.8721

150	0.0112	0.0080	0.0893	0.5951	0.8777
200	0.0222	0.0112	0.1057	0.5669	0.8287
Dropout Ratio (Simplified LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
0	0.0062	0.0080	0.0897	0.5655	0.8767
20	0.0184	0.0090	0.0950	0.6107	0.8616
40	0.0183	0.0111	0.1051	0.5898	0.8305
60	0.0201	0.0095	0.0973	0.6049	0.8547
Dropout Ratio (Stacked LSTM)	nMAE	nMSE	nRMSE	Max Error	R^2
0	0.0112	0.0080	0.0893	0.5951	0.8777
20	0.0130	0.0087	0.0934	0.5741	0.8663
40	0.0233	0.0101	0.1005	0.5918	0.8451
60	0.0077	0.0107	0.1036	0.5814	0.8352

Appendix C: MATLAB Code for ANN Development

```
%only importing August 2019 data
file='August 2019.xlsx';
opts1 = detectImportOptions(file);
opts1.SelectedVariableNames =
{'Var1','Irradiation_W_m2_','ModuleTemperature___C_','AmbientTemperature_
__C_','TotalPac'};

Aug = readtable(file,opts1);
[filterAug ia] = rmmissing(Aug);
AugCheck = ia(ia(:,1)==1); % expecting 29x3 = 87 rows to be removed

h = hour(filterAug.Var1);
m = minute(filterAug.Var1);

tabAug = filterAug((h >= 7) & (h < 20), :);
matrixAug = tabAug(:,2:end);
datasetAug = table2array(matrixAug);
[row col] = size(datasetAug);

% normalize between range 0 to 1
dataNormalized = normalize(datasetAug,"range");
maxValues = max(datasetAug);
minValues = min(datasetAug);

% extracting the features from the dataset
irradiation = dataNormalized(:,1);
moduleTemp = dataNormalized(:,2);
ambientTemp = dataNormalized(:,3);
totalPower = dataNormalized(:,end);

Database = [irradiation';moduleTemp'];
Target = totalPower';
%%
X = tonndata(Database,true,false);
T = tonndata(Target,true,false);

% Choose a Training Function
trainFcn = 'trainlm';  % Levenberg-Marquardt backpropagation.

% Create a Time Delay Network
inputDelays = 1:156;
hiddenLayerSize = 30;
net = timedelaynet(inputDelays,hiddenLayerSize,trainFcn);

% Prepare the Data for Training and Simulation
% The function PREPARETS prepares timeseries data for a particular
network,
% shifting time by the minimum amount to fill input states and layer
% states. Using PREPARETS allows you to keep your original time series
data
% unchanged, while easily customizing it for networks with differing
% numbers of delays, with open loop or closed loop feedback modes.
[x,xi,ai,t] = preparets(net,X,T);

% Setup Division of Data for Training, Validation, Testing
trainIndex = round(0.85*(length(t)-156));

net.divideFcn = 'divideind';  % Divide data randomly
net.divideMode = 'time';  % Divide up every sample
net.divideParam.trainInd = 1:trainIndex;
net.divideParam.valInd = trainIndex+1:length(t)-156;
net.divideParam.testInd = length(t)-156+1:length(t);

% Choose a Performance Function
net.performFcn = 'mse';  % Mean Squared Error

% Choose Plot Functions
net.plotFcns = {'plotperform','plottrainstate', 'ploterrhist', ...
    'plotregression', 'plotresponse', 'ploterrcorr', 'plotinerrcorr'};

% Train the Network
[net,tr] = train(net,x,t,xi,ai);

layer1neuron30 = net;
save layer1neuron30

% Test the Network
y = net(x,xi,ai);
e = gsubtract(t,y);
performance = perform(net,t,y)

% View the Network
view(net)

% Validating the Network
valSim = net((x(:,tr.valInd)),xi,ai);
valSim = cell2mat(valSim);
valTarget = t(tr.valInd);
valTarget = cell2mat(valTarget);
n = length(valSim);

% performance metric results:
% normalised
nMAE = abs((sum(valTarget-valSim))/n)
nMSE = (sum((valTarget - valSim).^2))/n
nRMSE = sqrt(nMSE)
nMaxError =  max(valTarget-valSim)

% un-normalised
valSimActual= valSim.*(maxValues(end)-minValues(end)) + minValues(end);
valTargetActual = valTarget.*(maxValues(end)-minValues(end)) +
minValues(end);

MAE = abs((sum(valTargetActual-valSimActual))/n)
MSE = (sum((valTargetActual - valSimActual).^2))/n
RMSE = sqrt(MSE)
```

```
MaxError =  max(valTargetActual-valSimActual)
ei = valTargetActual-valSimActual;
SSR = sum(ei.^2);
SST = sum((valTargetActual-mean(valTargetActual)).^2);
RSquare = 1-(SSR/SST)

% plot the comparison between actual and forecasted values
figure
hold on
plot(valTargetActual,'b')
plot(valSimActual,'r')
xlabel("Timesteps")
ylabel("Solar PV Power")
title("Forecast on Test Data Un-normalised")
legend(["Observed" "Forecast"])
hold off
f1 = gcf;
exportgraphics(f1,'Thesis_PowerComparison.png','Resolution',600)

% plot the RMSE graph
figure
stem(valTargetActual - valSimActual)
xlabel("Timestep")
ylabel("Error")
title("RMSE")
title("RMSE = " + RMSE)
f2 = gcf;
exportgraphics(f2,'Thesis_RMSE.png','Resolution',600)

% plot the MSE graph
figure
stem((valSimActual - valTargetActual).^2)
xlabel("Timestep")
ylabel("Error")
title("MSE")
title("MSE = " + MSE)
f3 = gcf;
exportgraphics(f3,'Thesis_MSE.png','Resolution',600)

% plot the R^2 graph
figure
plot(valTarget,valSim,'*')
hold on
xlabel('Target')
ylabel('Output')
p = polyfit(valTarget,valSim,1);
f = polyval(p,valTarget);
plot(valTarget,f,'r')
title(sprintf('Regression line y = %0.2f*Target + %0.2f',p(1),p(2)))
hold off
f4 = gcf;
exportgraphics(f4,'Thesis_R2.png','Resolution',600)
%%
% Testing the Network
testSim = net(x(:,tr.testInd),xi,ai);
testSim = cell2mat(testSim);
```

```
testTarget = t(tr.testInd);
testTarget = cell2mat(testTarget);
n = length(testSim);

% performance metric results:
% normalised
nMAE = abs((sum(testTarget-testSim))/n)
nMSE = (sum((testTarget - testSim).^2))/n
nRMSE = sqrt(nMSE)
nMaxError =  max(testTarget-testSim)

% un-normalised
testSimActual= (testSim.*(maxValues(end)-minValues(end)) +
minValues(end))';
testTargetActual = (testTarget.*(maxValues(end)-minValues(end)) +
minValues(end))';

MAE = abs((sum(testTargetActual-testSimActual))/n)
MSE = (sum((testTargetActual - testSimActual).^2))/n
RMSE = sqrt(MSE)
MaxError =  max(testTargetActual-testSimActual)
ei = testTargetActual-testSimActual;
SSR = sum(ei.^2);
SST = sum((testTargetActual-mean(testTargetActual)).^2);
RSquare = 1-(SSR/SST)

% plot the comparison between actual and forecasted values
figure
hold on
plot(testTargetActual,'b')
plot(testSimActual,'r')
xlabel("Timesteps")
ylabel("Solar PV Power")
title("Forecast on Test Data Un-normalised")
legend(["Observed" "Forecast"])
hold off

% plot the RMSE graph
figure
stem(testTargetActual - testSimActual)
xlabel("Timestep")
ylabel("Error")
title("RMSE")
title("RMSE = " + RMSE)

% plot the MSE graph
figure
stem((testSimActual - testTargetActual).^2)
xlabel("Timestep")
ylabel("Error")
title("MSE")
title("MSE = " + MSE)

% plot the R^2 graph
figure
plot(testTarget,testSim,'*')
```

```
hold on
xlabel('Target')
ylabel('Output')
p = polyfit(testTarget,testSim,1);
f = polyval(p,testTarget);
plot(testTarget,f,'r')
title(sprintf('Regression line y = %0.2f*Target + %0.2f',p(1),p(2)))
hold off

trainSim = net((x(:,tr.trainInd)),xi,ai);
trainSim = cell2mat(trainSim);
trainTarget = t(tr.trainInd);
trainTarget = cell2mat(trainTarget);

valSim = net((x(:,tr.valInd)),xi,ai);
valSim = cell2mat(valSim);
valTarget = t(tr.valInd);
valTarget = cell2mat(valTarget);

testSim = net(x(:,tr.testInd),xi,ai);
testSim = cell2mat(testSim);
testTarget = t(tr.testInd);
testTarget = cell2mat(testTarget);

allSim = net(x,xi,ai);
allSim = cell2mat(allSim);
allTarget = t;
allTarget = cell2mat(allTarget);

plotregression(trainTarget, trainSim, 'Train', valTarget, valSim,
'Validation', testTarget, testSim, 'Testing',allTarget,allSim,'All')

f5 = gcf;
exportgraphics(f5,'Thesis_RegressionComparison.png','Resolution',600)
```

Appendix D: MATLAB Code for LSTM Development

```
clear all; clc; close all;

%only importing June 2019 data
file='August 2019.xlsx';
opts1 = detectImportOptions(file);
opts1.SelectedVariableNames =
{'Var1','Irradiation_W_m2_','ModuleTemperature___C_','AmbientTemperature_
__C_','TotalPac'};

June = readtable(file,opts1);
[filterJune ia] = rmmissing(June);
JuneCheck = ia(ia(:,1)==1); % expecting 29x3 = 87 rows to be removed
%%
h = hour(filterJune.Var1);
m = minute(filterJune.Var1);

tabJune = filterJune((h >= 7) & (h < 20), :);
matrixJune = tabJune(:,2:end);
datasetJune = table2array(matrixJune);
[row col] = size(datasetJune);

% normalize between range 0 to 1
dataNormalized = normalize(datasetJune,"range");
maxValues = max(datasetJune);
minValues = min(datasetJune);

% extracting the features from the dataset
irradiation = dataNormalized(:,1);
moduleTemp = dataNormalized(:,2);
ambientTemp = dataNormalized(:,3);
totalPower = dataNormalized(:,end);

all_days = tabJune.Var1.Day;

%to sort the data based on the dates
 for i = 1:max(all_days)
     ind = (all_days ==i);
     rows = find(any(ind==1,2));
     days{i} = tabJune(rows(1):rows(end),:);
 end

 numTimeStepsDay = height(days{1});
%%
% database creation for LSTM inputs
numDays = 3; % number of days
sequenceLength = numDays*numTimeStepsDay; % number of time steps used in
each sample
database = {};
powerDatabase = {};
resultDatabase = [];
```

```
count = 0;
for i = 1:1:row-sequenceLength
    count = count+1;
    a = irradiation(i:i+sequenceLength-1)'; % first feature (most
important) which is also desired output
    b = moduleTemp(i:i+sequenceLength-1)'; % second feature

    database{count,1} = [a;b];
end

[rt ct] = size(database);

% database creation for LSTM outputs
count = 0;
for i = 1:1:row-sequenceLength
    count = count+1;
    d = totalPower(i:i+sequenceLength-1)'; % solar PV power
    powerDatabase{count,1}=[d];
end

count = 0;
for i=1:rt-1
    count = count+1;
    e = powerDatabase{count+1,1};
    resultDatabase(count,1) = e(1,end);
end
%%
% segmenting the database into training, validation and testing sets
% training : 85%
% validation : 15%
% testing : 1 days

numTimeStepsTest = numTimeStepsDay*1;
numTimeStepsTrain = round(0.85*(rt-numTimeStepsTest));
numTimeStepsVal = rt-numTimeStepsTest-numTimeStepsTrain;

XTrain = {};
YTrain = [];

for i=1:numTimeStepsTrain+1
    XTrain{i,1}=database{i,1}; % training set predictors (inputs)
end

YTrain = resultDatabase(1:numTimeStepsTrain+1); % training set responses
(outputs)


% LSTM Network Creation
numFeatures = 2;
numHiddenLayers = 200;
numResponses = 1;

Layers = [sequenceInputLayer(numFeatures),...
    lstmLayer(numHiddenLayers,'OutputMode','last'),...
    fullyConnectedLayer(numResponses),...
    regressionLayer];

miniBatchSize = 32;
Epoch = 3;

options = trainingOptions ('adam',...
    'ExecutionEnvironment','auto',...
    'MaxEpochs',Epoch,...
    'MiniBatchSize',miniBatchSize,...
    'InitialLearnRate',0.001,...
    'Plots','training-progress');
%%
% initialises the network to predict t+1 response given the inputs (only
1
% timestep forward output)
CaseBatch4 = trainNetwork(XTrain,YTrain,Layers,options);

if exist('CaseBatch4.mat','file')
    fprintf('The Variables Exist. Loading...\n');
else fprintf('Variables Do Not Exist. Creating Variables...\n');
    save CaseBatch4
end
%%
% validation stage

% update network state with observed values
count = 0;
for i=numTimeStepsTrain+1:numTimeStepsTrain+numTimeStepsVal
    count = count+1;
    XValidation{count,1} = database{i,1}; % testing set predictors
(inputs)
end

YValidation =
resultDatabase(numTimeStepsTrain+2:numTimeStepsVal+numTimeStepsTrain);

[UpdatedNet YPredVal] =
predictAndUpdateState(CaseBatch4,XValidation,'MiniBatchSize',16,'Executio
nEnvironment','cpu');

% Ypred predicts an additional timestep in the future past that of the
last time of August

% unnormalize the prediction
YPredValActual = YPredVal.*(maxValues(end)-minValues(end)) +
minValues(end);
YValidationActual = YValidation.*(maxValues(end)-minValues(end)) +
minValues(end);
YTrainActual = YTrain.*(maxValues(end)-minValues(end)) + minValues(end);

n = length(YValidation);

% performance metric results:

% normalised
```

```
nMAE = abs((sum(YValidation-YPredVal(1:end-1)))/n)
nMSE = (sum((YValidation - YPredVal(1:end-1)).^2))/n
nRMSE = sqrt(nMSE)
nMaxError =  max(YValidation-YPredVal(1:end-1))
ei = YValidation-YPredVal(1:end-1);
SSR = sum(ei.^2);
SST = sum((YValidation-mean(YValidation)).^2);

% un-normalised
MAE = abs((sum(YValidationActual-YPredValActual(1:end-1)))/n)
MSE = (sum((YValidationActual - YPredValActual(1:end-1)).^2))/n
RMSE = sqrt(MSE)
MaxError =  max(YValidationActual-YPredValActual(1:end-1))
RSquare = 1-(SSR/SST)

% graphical results:

% plot the comparison between actual and forecasted values
figure
hold on
plot(YValidationActual,'b')
plot(YPredValActual,'r')
xlabel("Timesteps")
ylabel("Solar PV Power")
title("Forecast on Test Data Un-normalised")
legend(["Observed" "Forecast"])
hold off

% plot the RMSE graph
figure
stem(YValidationActual - YPredValActual(1:end-1))
xlabel("Timestep")
ylabel("Error")
title("RMSE")
title("RMSE = " + RMSE)

% plot the MSE graph
figure
stem((YPredValActual(1:end-1) - YValidationActual).^2)
xlabel("Timestep")
ylabel("Error")
title("MSE")
title("MSE = " + MSE)

% plot the R^2 graph
figure
plot(YValidation,YPredVal(1:end-1),'*')
hold on
xlabel('Target')
ylabel('Output')
p = polyfit(YValidation,YPredVal(1:end-1),1);
f = polyval(p,YValidation);
plot(YValidation,f,'r')
title(sprintf('Regression line y = %0.2f*Target + %0.2f',p(1),p(2)))
hold off
%%
```

```
% % Testing Stage
%
% update network state with observed values
count = 0;
for i=numTimeStepsTrain+numTimeStepsVal+1:rt
    count = count+1;
    XTest{count,1} = database{i,1}; % testing set predictors (inputs)
end

YTest = resultDatabase(numTimeStepsTrain+numTimeStepsVal+1:rt-1);

[UpdatedNet YPredTest] =
predictAndUpdateState(Case200,XTest,'MiniBatchSize',16,'ExecutionEnvironm
ent','cpu');

% % Ypred predicts an additional timestep in the future past that of the
last
% % time of June
%
% % unnormalize the prediction
YPredTestActual = YPredTest.*(maxValues(end)-minValues(end)) +
minValues(end);
YTestActual = YTest.*(maxValues(end)-minValues(end)) + minValues(end);
%
n = length(YTest);

% performance metric results:

% normalised
nMAE = abs((sum(YTest-YPredTest(1:end-1)))/n)
nMSE = (sum((YTest - YPredTest(1:end-1)).^2))/n
nRMSE = sqrt(nMSE)
nMaxError =  max(YTest-YPredTest(1:end-1))
ei = YTest-YPredTest(1:end-1);
SSR = sum(ei.^2);
SST = sum((YTest-mean(YTest)).^2);


% un-normalised
MAE = abs((sum(YTestActual-YPredTestActual(1:end-1)))/n)
MSE = (sum((YTestActual - YPredTestActual(1:end-1)).^2))/n
RMSE = sqrt(MSE)
MaxError =  max(YTestActual-YPredTestActual(1:end-1))
% ei = YTestActual-YPredTestActual(1:end-1);
% SSR = sum(ei.^2);
% SST = sum((YTestActual-mean(YTestActual)).^2);
% nRSquare = 1-(SSR/SST)
RSquare = 1-(SSR/SST)
% graphical results:

% plot the comparison between actual and forecasted Testues
figure
hold on
plot(YTestActual,'b')
plot(YPredTestActual,'r')
```

```
xlabel("Timesteps")
ylabel("Solar PV Power")
title("Forecast on Test Data Un-normalised")
legend(["Observed" "Forecast"])
hold off

% plot the RMSE graph
figure
stem(YTestActual - YPredTestActual(1:end-1))
xlabel("Timestep")
ylabel("Error")
title("RMSE")
title("RMSE = " + RMSE)

% plot the MSE graph
figure
stem((YPredTestActual(1:end-1) - YTestActual).^2)
xlabel("Timestep")
ylabel("Error")
title("MSE")
title("MSE = " + MSE)

% plot the R^2 graph
figure
plot(YTest,YPredTest(1:end-1),'*')
hold on
xlabel('Target')
ylabel('Output')
p = polyfit(YTest,YPredTest(1:end-1),1);
f = polyval(p,YTest);
plot(YTest,f,'r')
title(sprintf('Regression line y = %0.2f*Target + %0.2f',p(1),p(2)))
hold off
```

Appendix E: Reflections on Program Outcomes (PO) Achievement

Program Outcomes	**Reflections**
PO1 Mechanical Engineering Knowledge: Apply knowledge of mathematics, natural science, engineering fundamentals and specialisation in Mechanical engineering to the solution of complex engineering problems	Mechanical engineering knowledge such as mathematics, artificial intelligence (AI) and coding in MATLAB has been applied. Furthermore, data mining and data analysis skills were applied.
PO2 Problem Analysis: Identify, formulate, survey research literature and analyse complex Mechanical engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences and engineering sciences	A comprehensive literature review was performed to identify and understand the current state of knowledge in this specific topic or field. Then research gaps and weaknesses were identifies and the problem statement was developed.
PO3 Design/Development of Solutions:Design solutions for complex Mechanical engineering problems and design systems, components or processes that meet specified needs.	The research involved solving the forecasting problem of the Monash solar PV plant which had not been done before. Through careful planning a solution was designed using a few different artificial intelligent techniques/models.
PO4 Research-based Investigation: Conduct investigations of complex Mechanical engineering problems using research-based knowledge and research methods including design of experiments, (analysis and interpretation of data, and	Since the artificial intelligent techniques and data pre-processing were required to complete this project, a lot of knowledge and methods were obtained from research papers, books and other reliable online sources.

synthesis of information to provide valid conclusions.	
PO5 Modern Tool Usage: Create, select and apply appropriate techniques, resources, and modern engineering and IT tools, including prediction and modelling, to complex Mechanical engineering problems, with an understanding of the limitations	Majority of the work was completed in MATLAB and Microsoft Excel. In MATLAB the machine learning and deep learning toolbox were used to develop the AI models. Besides that, Microsoft Word was utilized for proper documentation and Microsoft PowerPoint was used for presenting.
PO6 Engineer and Society: Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to professional engineering practice and solutions to complex Mechanical engineering problems	The motivation behind this research project is to aid the operational decision of Monash in relation to the solar PV and also improve system performance. This would ultimately allow Monash to have a higher penetration of renewable energy used, thus contributing to sustainability and the environment.
PO7 Environment and Sustainability: Understand and evaluate the sustainability and impact of professional engineering work in the solution of complex Mechanical engineering problems in environmental contexts.	The motivation behind this research project is to aid the operational decision of Monash in relation to the solar PV and also improve system performance. This would ultimately allow Monash to have a higher penetration of renewable energy used, thus contributing to sustainability and the environment.
PO8 Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of engineering practice.	Constructive feedbacks from supervisor were taken aboard for self-improvement and learning. No work was plagiarized and sufficient acknowledgment were given to

	ideas and contributions that were taken externally.
PO9 Communication: Communicate effectively on complex Mechanical engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions	Communication was done fully online either via Zoom calls or Whatsapp text messaging. Other than that, emails were sent to the supervisor. All this was to get clarifications, feedback and give updates on progress
PO10 Individual and Team work: Function effectively as an individual, and as a member or leader in diverse teams and in multi-disciplinary settings	Research task was carried out independently. Only collaboration was with supervisor, and that was limited to advice, guidance, and feedback.
PO11 Lifelong Learning: Recognise the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change	Understand that the state of knowledge around the world is continuously changing and to keep up to date have to treat it as a lifelong learning process
PO12 Project Management and Finance: Demonstrate knowledge and understanding of engineering management principles and economic decision-making and apply these to manage projects	At the initial phase, a research plan and a Gantt chart consisting of the timelines were created. This was followed as best as possible, although some amount of flexibility was expected due to unforeseen issues.