 https://doi.org/10.24191/jeesr.v25i1.010

Optimizing Photovoltaic Output Performance Prediction: A Deep Learning Approach with LSTM Neural Networks and Adam Optimizer

Syasya Nadhirah Hamedon, Juliana Johari and Fazlina Ahmat Ruslan*

***Abstract*—This study introduces an innovative approach to optimizing photovoltaic (PV) output performance prediction through Deep Learning, specifically employing Long Short-Term Memory (LSTM) networks and the Adaptive Moment Estimation (Adam) optimizer. The research is carried out using MATLAB R2023a, and the dataset is publicly accessible from the Solcast website. Performance evaluation utilizing key indicators such as Loss Metrics and Root Mean Squared Error (RMSE) highlights the potential of this Adam-optimized LSTM method to significantly enhance the accuracy of PV performance prediction. The analysis explores the impact of learning rates, evaluating fixed rates (0.0001, 0.001, 0.01) and a dynamic transition (0.01 to 0.001) over 10 epochs. Notably, a learning rate of 0.01 demonstrates substantial improvements, achieving lower errors and consistently low losses, indicating a highly accurate and well-fitted model. The unexpected adaptability observed during a dynamic learning rate transition further highlights the model's potential. The study presents a comprehensive analysis of LSTMs and Adam optimizer for PV output performance prediction and provides valuable insights for researchers seeking optimal learning rates to develop robust and effective PV performance prediction models.**

***Index Terms*—Deep Learning, LSTM Networks, Adam Optimizer, PV Performance Prediction.**

I. INTRODUCTION

The integration of renewable energy sources, particularly solar power, into the global energy landscape has driven a surge in research to enhance the efficiency and accuracy of predicting photovoltaic (PV) output. Advanced machine learning techniques, and notably deep learning approaches, have garnered significant attention for their ability to model the complex patterns characteristic in solar energy generation. One such prominent deep learning architecture is the Long Short-Term Memory (LSTM) network, which has shown promise in capturing temporal dependencies in time series data [1 – 2]. A comprehensive review of previous related studies highlights the efficacy of LSTM in modeling solar output dynamics, with notable contributions from researchers such as S. Dhingra et al. [3], where the authors compare the performance of different neural network architectures, including Convolutional Neural Network (CNN), Autoencoders, LSTM and Gated-Recurrent Unit (GRU), and M. S. Hossain et al. [4] also compares the performance using various deep learning architectures, including Generalized Recurrent Neural Network (GRNN), Nonlinear Autoregressive Exogenous (NARX) and LSTM. In both studies, the LSTM model demonstrates the lowest values for Mean Squared Error (MSE), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE). Lower values in these metrics signify better predictive performance.

Despite the successes of LSTM, the optimization of its performance remains a crucial point of investigation. The choice of optimizer during the training phase significantly influences the convergence speed and overall effectiveness of deep learning models. In this context, the Adaptive Moment Estimation (Adam) optimizer, renowned for its adaptive learning rates and momentum, has emerged as a crucial element in enhancing the training efficiency of neural networks. Studies by F. R. Ningsih et al. [5], K. Kaysal et al. [6] and L. Setiawan et al. [7] have clarified the advantages of the Adam optimizer in improving convergence rates and overall model performance in various machine learning applications.

F. R. Ningsih et al. [5] focused on wind speed forecasting using Recurrent Neural Networks (RNN) with LSTM. The research aimed to predict wind speed to anticipate its impact on various aspects of human life, especially in regions frequently affected by strong winds. The research involved preprocessing the data, including handling missing or unusable data, normalization, and conversion from day-to-month data. Segmentation with overlapping was employed to minimize the impact of data discontinuity. The data used for training and testing were divided into two parts: 80% for training and 20% for testing. In this study, the Adam optimization model was employed and showed superior performance compared to Stochastic Gradient Descent (SGD), with accuracy values of 92.7% for training data and 91.6% for new data. The study successfully demonstrated the effectiveness of using RNN with LSTM for wind speed prediction, with Adam optimization outperforming SGD. The amount of training data and the number of epochs were identified as crucial factors influencing the accuracy of predictions.

This manuscript is submitted on 21 January 2024, revised on 25 July 2024, accepted on 20 September 2024 and published on 31 October 2024. Syasya Nadhirah Hamedon, Fazlina Ahmat Ruslan and Juliana Johari are from the School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia.

*Corresponding author
Email address: fazlina419@uitm.edu.my

K. Kaysal et al. [6] focused on electricity production forecasting in Turkey and specifically evaluated the performance of the Adam optimizer in combination with the LSTM machine learning method. The Adam optimizer was one of several optimization techniques compared, including Root Mean Square Propagation (RMSprop), Adaptive moment estimation with Maximum (Adamax), and Nesterov-accelerated Adaptive Moment Estimation (Nadam). The researchers designed a three-layer LSTM model and conducted training over 20 epochs using real-time electricity generation data from the past four years in Turkey. The study aimed to determine the optimization technique that provides the best performance for the LSTM method in short-term one-hour production forecasts. The results revealed that the Adam optimizer exhibited the highest performance among the evaluated optimization techniques, achieving a 98% success rate. The comparison was based on metrics such as Mean Absolute Percentage Error (MAPE) and R-squared (R^2) score. The study contributes to the field by highlighting the effectiveness of the Adam optimizer in enhancing the accuracy of electricity production forecasts when applied to the LSTM model.

L. Setiawan et al. [7] focused on the prediction of stock price data using LSTM model. In the data preparation stage, the study involves collecting the stock data, normalizing it to the (0,1) range, and then dividing it into 75% for training and 25% for testing purposes The study involves experimenting with different configurations of the LSTM model, including the number of nodes, activation functions, and optimizers. Two types of nodes (4 and 16), four activation functions (Linear, Sigmoid, Rectified Linear Unit (ReLU), and Hyperbolic Tangent Function and three optimizers (Adam, SGD, and Adaptive Gradient Optimizer (Adagrad)) are explored. The researchers optimize the LSTM model by varying the number of epochs. The optimal number of epochs is determined to be 5 based on experimentation. Denormalization is applied to the results, converting them back to their original scale for easier interpretation and understanding. Evaluation metrics such as Root Mean Squared Error (RMSE) for overall model accuracy and Mean Squared Error (MSE) for data sample error at each epoch are employed to assess the performance of the model. The study concludes that predicting stock prices using the LSTM model with 4 nodes, Linear activation function, and Adam optimizer yields better results with a Train RMSE of 27.36, Test RMSE of 15.37, and a price prediction result per share one day ahead (16 September 2021) of IDR 639.05. The findings provide insights into the factors influencing the accuracy of stock price predictions, emphasizing the importance of selecting appropriate model configurations and training parameters.

In summary, this paper seeks to contribute to the existing body of knowledge by presenting an in-depth exploration of the optimizing photovoltaic output performance prediction using a deep learning approach with LSTM neural networks and the Adam optimizer. While the previous research has laid the groundwork for integrating LSTM networks into solar output prediction models, showcasing their ability to capture complex pattern, the specific impact of the Adam optimizer in conjunction with LSTM for solar output prediction remains a critical area that requires further investigation. By building on the collective insights from previous studies on LSTM networks and the Adam optimizer, this research aims to address existing gaps and contribute to the development of more accurate and efficient models for predicting photovoltaic output performance in real-world scenarios.

II. Long-Short Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture that was introduced to address challenges faced by traditional RNNs in capturing and preserving long-range dependencies within sequential data [8]. Developed by Sepp Hochreiter and Jürgen Schmidhuber in 1997 [8], LSTMs have become a cornerstone in the field of deep learning, particularly in applications involving time series data [9 – 10], natural language processing [11], speech recognition [12], healthcare data analysis [13], autonomous vehicle [14], fraud detection [15], and robotics [16].

The most important aspect of LSTM networks lies in their ability to effectively capture and manage long-range dependencies within sequential data. LSTMs address a significant challenge faced by traditional RNNs known as the vanishing gradient problem. Unlike conventional RNNs, which struggle with information preservation over extended sequences, LSTMs address this issue through a sophisticated architecture that includes a memory cell and gating mechanisms [8]. These components allow LSTMs to selectively remember or forget information across varying time steps, making them particularly adept at handling sequences with significant time lags and dependencies [8]. The architecture of an LSTM is visually represented in Fig. 1. It includes crucial components such as the Forget Gate, Input Gate, Output Gate, Cell State (Memory), and Hidden State [17 – 18].

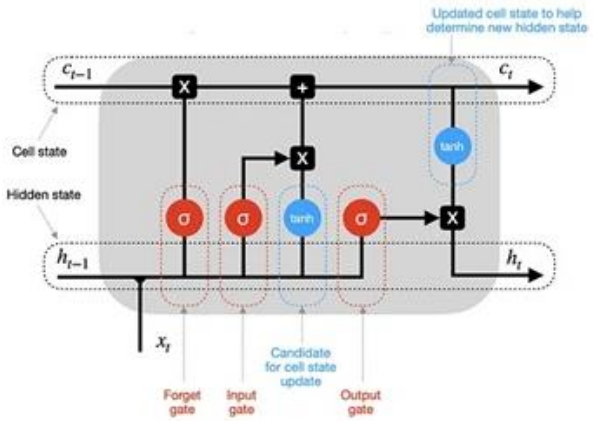

Fig. 1. LSTM Neural Network Architecture

A. Forget Gate

The forget gate, defined by (1), serves the critical function of determining which aspects of both the current and past information are to be preserved or discarded in the LSTM architecture. This equation employs a sigmoid activation

function (σ) to process the concatenation of the previous hidden state h_{t-1} and the current input x_t, resulting in values between 0 and 1 for each element in the cell state. The subsequent multiplication of these values with the current cell state facilitates a discerning mechanism, allowing the LSTM to selectively remember or forget information at each time step. Whereas W and b are, respectively, weight matrices and bias vector parameters. This process plays a pivotal role in addressing long-term dependencies and optimizing the LSTM's ability to capture relevant patterns in sequential data.

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

B. Input Gate

Within the LSTM architecture, the input gate plays a pivotal role in determining the extent to which new information should be incorporated into the cell state, thereby governing the update of the cell status. In (2), the output of the input gate i_t is computed by applying a sigmoid activation function to a concatenation of the previous hidden state h_{t-1} and the current input x_t. This sigmoid output regulates the proportion of new information to be added to the cell state. Simultaneously, (3) outlines the generation of a candidate cell state $\check{C}_t$ using a hyperbolic tangent (tanh) function applied to the same concatenated input. The tanh output, representing values between -1 and 1, serves as the candidate information that may be added to the cell state. Both processes involve learned parameters, including weight matrices (W_i and W_C) and bias vectors (b_i and b_C) contributing to the nuanced control and update mechanisms of the LSTM network.

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

$$\check{C}_t = tanh\left(W_C \cdot [h_{t-1}, x_t]\right) + b_C \quad (3)$$

C. Cell State (Memory)

In the LSTM architecture, the cell state, functioning as the memory, undergoes a dynamic update process. Once the network accumulates sufficient information from the forget gate and input gate, it proceeds to update the old cell state C_{t-1} to form the new cell state C_t. This updating process, outlined in equation (4), involves selectively retaining relevant information from the previous state (C_{t-1}) based on the forget gate's decision f_t, and adding new candidate values scaled by the input gate's decision i_t. Consequently, the cell state C_t evolves over time, incorporating pertinent information while discarding less relevant elements.

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

D. Output Gate

The output gate assumes a pivotal role in determining the value of the next hidden state, crucial for modelling sequential data. The current state and the preceding hidden state are subjected to a sigmoid activation function (5), transforming the values into a range between 0 and 1. This transformation gauges the relevance of each value in contributing to the subsequent hidden state. Subsequently, the new cell state, derived from the existing cell state using updated information from the sigmoid function, undergoes processing through the tanh function (6). This function scales the cell state values between -1 and 1, controlling information flow and mitigating issues such as the vanishing gradient problem.

The outputs from the sigmoid and tanh activations are then multiplied point-wise, element-wise. This multiplication generates a modulation signal dictating the significance of the new cell state information for the upcoming time step. The final value of this multiplication, denoted as O_t, determines the decision of the LSTM network regarding which information the hidden state should retain and carry forward. This intricate process ensures the preservation of relevant context and dependencies for effective prediction tasks. The newly calculated hidden state h_t and the updated cell state C_t are subsequently carried over to the next time step in the sequence. This mechanism, involving the retention and updating of the cell state and hidden state, empowers the LSTM to learn and remember patterns over extended sequences. The formulas for the output gate are presented in (5) and (6), where t represents the timestep, O_t is the output gate at time *t*, σ represents the sigmoid activation function, W_O is the weight matrix specific to the output gate, h_{t-1} is the previous hidden state, x_t is the input at time *t*, b_O is the bias vector, h_t is the LSTM output, and C_t is the cell state.

$$O_t = \sigma\left(W_O \cdot [h_{t-1}, x_t]\right) + b_O \quad (5)$$

$$h_t = O_t * \tanh\left(C_t\right) \quad (6)$$

Furthermore, integrating Long Short-Term Memory (LSTM) networks with optimizers is essential to overcome challenges associated with training recurrent neural networks and to enhance their overall efficiency and performance. The vanishing gradient problem is a common issue in training deep neural networks, including LSTMs. This problem arises when gradients become extremely small during backpropagation, making it challenging for the network to learn and update its parameters effectively over time. Optimizers, such as Adam, address this by adaptively adjusting learning rates for each parameter based on the historical gradients. This adaptive learning rate mechanism helps LSTMs converge faster and more effectively by mitigating the vanishing gradient problem. Several aspects should be considered when integrating LSTMs with the Adam optimizer:

E. Adaptive Learning Rates

One of the key features of Adam is its adaptive learning rate, which adjusts the step size for each parameter individually [19]. This is crucial for LSTMs, especially when dealing with sequences of varying lengths and complexities. Adaptive learning rates ensure that the model can learn effectively across different components and time steps.

F. Parameter Initialization

Proper initialization of LSTM weights and biases is essential

for the stability and convergence of the training process [20]. Adam operates more efficiently when initial weights are within a reasonable range. Careful consideration of weight initialization aligns with Adam's optimization approach, promoting stable and effective weight updates.

G.Batch Initialization

Normalizing inputs within mini-batches during training enhances stability and convergence [21]. Adam's preference for normalized input aligns well with the batch normalization process, contributing to overall stability during training. This is particularly relevant when integrating LSTMs with Adam.

H.Hyperparameter Tuning

Careful tuning of hyperparameters, such as learning rate, beta1, and beta2, is essential for achieving optimal LSTM performance when integrated with Adam. Adjusting these parameters ensures efficient convergence and enhances the adaptability of the network to different datasets [22].

To delve further into the landscape of prediction models utilizing LSTM, it is crucial to expand our understanding by examining additional studies conducted by previous researchers. In time series data analysis, the researchers [9] employs LSTM networks to predict weather parameters (wind speed, temperature, pressure, and humidity) for the next five days using a dataset from Kaggle. The researchers have implemented their weather prediction system using Python-based modules. The LSTM model is trained with 250 epochs, and a dropout layer is included to prevent overfitting. Findings reveal discrepancies in predicted values for different days and epochs because of the drastic climatic difference during that period. For instance, wind speed prediction values are closer with wind speed real values for Day 4 at 250 epochs, while temperature predictions are generally accurate to temperature real values at 250 epochs. Humidity predictions for Day 3 are notably close with humidity real values at 250 epochs. Therefore, the study suggests potential accuracy improvements with increased epochs and proposes future work on extending predictions to different regions and incorporating additional features.

In the field of natural language processing (NLP), LSTM models have been instrumental in language translation, sentiment analysis, and text generation. Their recurrent nature allows them to understand and generate coherent sequences of text, making them suitable for applications demanding a nuanced understanding of linguistic context. Y. Heryadi et al. [11] explores the use of Long Short-term Memory (LSTM) models for machine translation between Bahasa Indonesia and Sundanese languages. The authors used a dataset that includes content from various sources, such as the ORCAS dataset, scraped data from airline websites, su.wikipedia.org, id.wikipedia.org, and several local government websites. The raw data was in either Bahasa Indonesia or Sundanese languages. The raw dataset underwent preprocessing, including tokenization, converting all words to lowercase, and removing special characters. The text was translated from Bahasa Indonesia to Sundanese and vice versa. The initial parallel corpus was fine-tuned by bilingual linguists. Two distinct LSTM models were constructed for translation in both directions: the Bahasa Indonesia to Sundanese (ID2SD) translator model and the Sundanese to Bahasa Indonesia (SD2ID) translator model. The LSTM model configuration involved essential layers such as the Embedding Layer, LSTM Layers 1 and 2 for capturing long-term dependencies, Repeat Vector Layer for input sequence repetition, and Dense Layer for generating the output sequence. The training process involved an 80% training and validation dataset split, with the remaining 20% reserved for testing. Training metrics, including accuracy and loss, were measured over 100 epochs. The findings reveal promising results, with both LSTM models demonstrating feasibility in achieving accurate translations. The evaluation metrics, including accuracy and BLEU scores, suggest the effectiveness of LSTM in addressing the language translation task. The BLEU score was used to measure the similarity between predicted and actual sentences, providing an evaluation metric for the translation quality. In essence, the authors strategically utilized LSTM as the foundational architecture for their models, demonstrating a focused approach to overcoming language translation challenges within the specified linguistic context.

Speech recognition represents another domain where LSTM has exhibited significant impact. In this study [12], the authors employed LSTM networks for Speech Emotion Recognition (SER) and explored the effectiveness of different model configurations. The LSTM architecture consisted of two hidden layers, with 16 nodes in the first layer and 8 nodes in the second layer. The models, denoted as Model 1, Model 2, and Model 3, utilized MFCC features with varying lengths (39, 28, and 12, respectively). Training and testing were performed on the EMO-DB dataset, encompassing emotions such as neutral, angry, happy, and sad. The LSTM models were designed to capture long-term dependencies in speech patterns, crucial for accurate emotion recognition. The findings revealed that Model 1 achieved the highest accuracy of 90% for the happy emotion, surpassing both Model 2 and Model 3. The average accuracy observed for Model 1 was 85.5%, while Model 2 and Model 3 demonstrated accuracies of 85.5% and 84.25%, respectively. This suggests the efficacy of LSTM in discerning emotions from speech, particularly when configured with specific parameters. Comparatively, the study indicated that LSTM outperformed Convolutional Neural Network (CNN) models, which were also implemented for SER. The average accuracy for CNN models ranged from 75.75% to 78.75%, highlighting the superior performance of LSTM in this specific context. Additionally, the study emphasized the advantage of LSTM in handling a substantial amount of data without the need to increase the network size, contributing to its efficiency in speech emotion recognition tasks. The experimental results, conducted on the EMO-DB dataset, underscore the significance of LSTM in the domain of SER, with Model 1 emerging as particularly promising.

These applications underscore the wide-ranging impact of LSTM networks in predictive modeling across diverse

domains. Understanding the methodologies and findings of previous researchers not only contributes to the collective knowledge in these fields but also provides insights into potential avenues for future exploration and improvement of LSTM-based prediction models.

III. Adaptive Movement Estimation (Adam) Algorithm

The Adam optimizer is a popular optimization algorithm used in training machine learning models, especially in the field of deep learning [23].

The name "Adam" is derived from "adaptive moment estimation," which reflects the key characteristics of the algorithm. Adam combines the advantages of two other popular optimization algorithms: RMSprop (Root Mean Square Propagation) and momentum [24].

Here are the main components and characteristics of the Adam optimizer:

A. Adaptive Learning Rates

Adam adapts the learning rates for each parameter individually. It computes adaptive learning rates for each parameter by considering both the first-order momentum (like momentum optimization) and the second-order acceleration (like RMSprop). This helps the algorithm converge faster and perform well on a wide range of problems.

B. Momentum

Adam uses a momentum term similar to the momentum optimization algorithm. This term helps accelerate the optimization process, especially when the gradients consistently point in the same direction.

C. RMSprop

Adam incorporates the concepts of RMSprop by maintaining a running average of squared gradients for each parameter. This helps in adapting the learning rates to each parameter individually.

D. Bias Correction

Adam includes a bias correction mechanism to account for the fact that the moving averages of the gradients and squared gradients are initialized with zeros. This correction helps in the early iterations of training when the estimates are biased due to initialization.

The update rule for Adam can be summarized as follows [25]:

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \quad (7)$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g^2{}_t \quad (8)$$

$$\hat{m}_t = \frac{m_t}{1 - \beta^t{}_1} \quad (9)$$

$$\hat{v}_t = \frac{m_t}{1 - \beta^t{}_2} \quad (10)$$

$$\theta_{t+1} = \theta_t + \frac{\alpha}{\sqrt{\hat{v}_t} + \epsilon} \cdot \hat{m}_t \quad (11)$$

where:

- θ_t is the parameter vector at time *t*,
- g_t is the gradient vector at time *t*,
- m_t is the first-order moment vector (momentum) at time *t*,
- v_t is the second-order moment vector (squared gradients) at time *t*,
- $\hat{m}_t$ and $\hat{v}_t$ are bias-corrected estimates of m_t and m_t,
- α is the learning rate,
- β_1 and β_2 are the exponential decay rates for the moment estimates, and
- ϵ is a small constant added to the denominator for numerical stability.

To gain a comprehensive understanding of Adam, previous studies have extensively explored its applications. In particular, the work by W. F. Hidayat et al. [26] proposed innovative methodologies leveraging Adam to enhance the training process of the Long Short-Term Memory (LSTM) model. In this study, Adam is chosen as the optimization algorithm for its effectiveness in improving convergence during the training phase. The results indicate that the use of ADAM contributes to improved accuracy in predicting cryptocurrency prices. The optimization provided by Adam helps the LSTM model better at understanding and recognizing patterns and relationships in the data, leading to more precise predictions. The research involves experimenting with different configurations, such as varying the data split ratios and the number of epochs, to find the optimal setup for the LSTM model. The findings suggest that, in conjunction with Adam, a specific configuration (60:40 data split, 200 epochs) yields the best results in terms of accuracy. The study reports lower Mean Square Error (MSE) and Root Mean Square Error (RMSE) values when using Adam in comparison to other configurations or optimization algorithms. This reduction in error metrics signifies the effectiveness of Adam in minimizing prediction errors. The final model configuration with Adam achieves a high accuracy rate of 98%, indicating the ability of the LSTM model to make accurate predictions regarding cryptocurrency prices for the specified time frame.

Additionally, a study conducted by R. Guo et al. [27] delves into the applications and effectiveness of Adam in diverse contexts. This research specifically focuses on early fault detection in a wind turbine gearbox, employing an LSTM model trained with the Adam optimization algorithm. Operational data spanning from December 30th, 2020, to March 1st, 2021, is gathered at a 1-minute resolution for the A05 wind turbine unit, which encountered a gearbox failure leading to a shutdown on March 1st, 2021, at 7:50, attributed to high gearbox oil temperature. After excluding normal shutdown periods, 61,673 valid operating data samples are selected. Of these, 43,172 samples are used for training the LSTM model, and the remaining 18,502 samples are reserved for validation. In this study, the Adam algorithm is employed for LSTM training, and the results are compared with the Stochastic Gradient Descent with Momentum (SGDm) optimization

algorithm. Both models effectively capture the variation trend of the gearbox oil temperature, but the Adam-optimized LSTM demonstrates smaller errors in comparison to the SGDm model. Performance metrics, such as RMSE, further confirm the superiority of the Adam-optimized LSTM, with an RMSE value of 0.80 compared to 0.91 for the SGDm model. To validate the fault detection strategy, the Adam-based LSTM model is utilized to predict test data from February 28th to March 1st. Large deviations in predictions from the 1669th to the 2075th data sample indicate the formation of a fault. The proposed strategy successfully detects the fault nearly one hour in advance using a sliding window method, allowing for timely maintenance planning and minimizing economic and power generation losses caused by downtime. The study establishes that the LSTM model trained with the Adam algorithm offers superior prediction performance and an effective strategy for early fault detection in wind turbine gearboxes, providing valuable insights for preventive maintenance planning.

IV. Model Performance Metrics

The optimization of photovoltaic (PV) output performance is a crucial aspect in maximizing the efficiency and reliability of solar energy systems. In this study, a deep learning approach is employed, specifically utilizing Long Short-Term Memory (LSTM) networks and the Adam optimizer, to predict PV output performance. To evaluate the effectiveness of the model, two performance indices are employed: Loss Metrics (Training and Validation Loss) and Root Mean Square Error (RMSE). These indices play a vital role in assessing the accuracy and reliability of the predictive model.

A. Loss Metrics:

1) Training Loss:

The training loss measures the error during the training phase, indicating how well the machine learning model is fitting the training data. The objective during training is to minimize this loss, signifying that the model is learning to make accurate predictions on the training set.

2) Validation Loss:

The validation loss measures the error on a separate validation dataset. Its purpose is to assess how well the model generalizes to new, unseen data. Unlike the training set, the model is not directly exposed to the validation set during training. Instead, the validation set is utilized to evaluate the model's performance on data it hasn't encountered before. Monitoring both training and validation losses is crucial to avoid overfitting, ensuring that the model learns patterns in the data rather than memorizing the training set. Striking a balance between these two metrics is essential for constructing models that demonstrate good performance on both familiar and unfamiliar data.

B. Root Mean Square Error (RMSE):

RMSE is another commonly used metric for evaluating the accuracy of a predictive model. It measures the square root of the average of the squared differences between predicted and actual values. RMSE penalizes larger errors more significantly than smaller errors. RMSE can be determined by using (12):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (12)$$

The symbols in equation 12 are defined as follows:

- **RMSE:** Root Mean Squared Error
- **n:** Number of data points
- **yi:** Actual value of the i-th data point
- **ŷi:** Predicted value of the i-th data point

A lower RMSE indicates better predictive performance. RMSE is sensitive to outliers due to the squaring of errors, making it a valuable metric for understanding the model's performance across the entire dataset.

V. Results and Discussion

This study employed hourly data collected from January to June 2021, sourced from the Online GIS Database from SOLCAST website which can be accessed at https://www.solcast.com [28]. A total of 78840 hourly data points were utilized for this study, representing 365 days of continuous data collection. To evaluate the model's performance, a standard 80:20 train-test split was implemented. 80% of the data was allocated for model training, while the remaining 20% was reserved for testing, ensuring a robust assessment of the model's generalization capabilities. The dataset includes information on various parameters, as shown in Table 1, which presents the input data for PV output performance prediction.

TABLE I. Input Data for PV Prediction Performance Analysis

Parameters	Unit
Air Temperature	°C
Direct Normal Irradiance (DNI)	W/m²
Direct Horizontal Irradiance (DHI)	W/m²
Global Horizontal Irradiance (GHI)	W/m²
Global Tilted Irradiance (GTI)	W/m²
Relative Humidity	%
Wind Direction	°
Wind Speed	m/s
Azimuth	°
Historical PV Output Power	kW

A. Hyperparameter Fine Tuning- Analyzing Learning Rate

In the pursuit of optimizing PV output performance, hyperparameter fine-tuning, particularly the analysis of learning rates, is crucial. Throughout the training and validation process, pivotal metrics such as epochs, iterations, elapsed time, mini-batch RMSE, mini-batch loss, validation RMSE, validation loss, and the base learning rate are meticulously tracked. The impact of different learning rates is specifically highlighted in Fig. 2 until Fig. 13, showcasing RMSE and Loss

values over 10 epochs, along with the results of Actual and Predicted Results of PV Performance Prediction. The comprehensive findings of this learning rate analysis are succinctly summarized in Table I. This investigation aims to provide valuable insights into the influence of learning rates on the overall predictive capabilities of the model.

3) Learning Rate = 0.0001, Epochs = 10.

Figs. 2 – 4 depict the RMSE and Loss results for both training and validation sets, along with the Actual and Predicted Results of PV Performance Prediction.

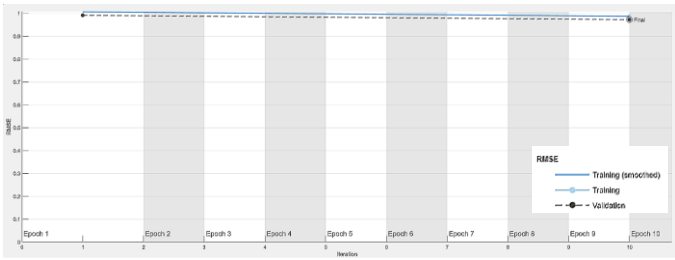

Fig. 2. RMSE – Training and Validation Results

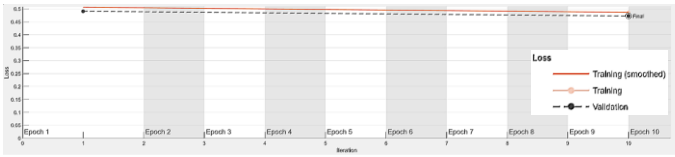

Fig. 3. Loss – Training and Validation Results

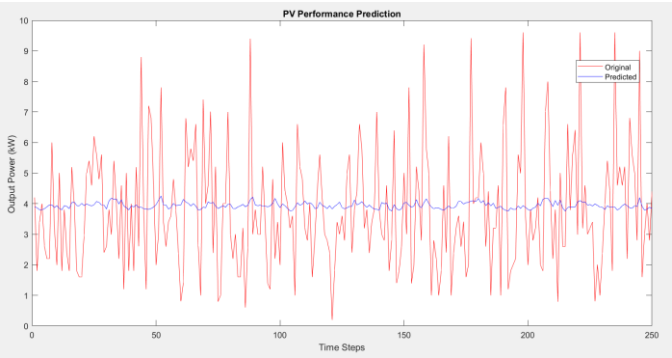

Fig. 4. Actual and Predicted Results of PV Performance Prediction

4) Learning Rate = 0.001, Epochs = 10.

Figs. 5 – 7 depict the RMSE and Loss results for both training and validation sets, along with the Actual and Predicted Results of PV Performance Prediction.

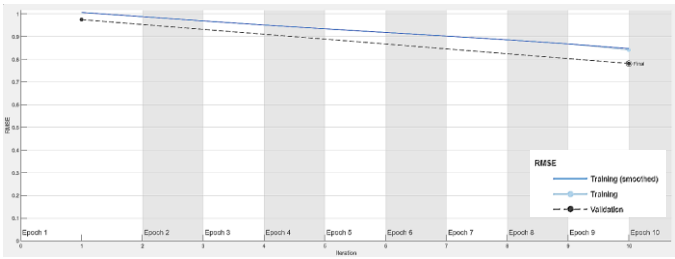

Fig. 5. RMSE – Training and Validation Results

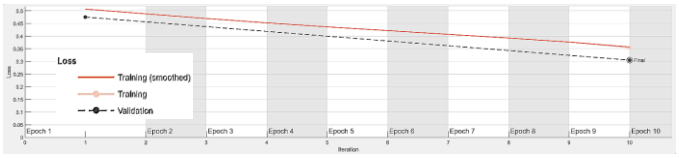

Fig. 6. Loss – Training and Validation Results

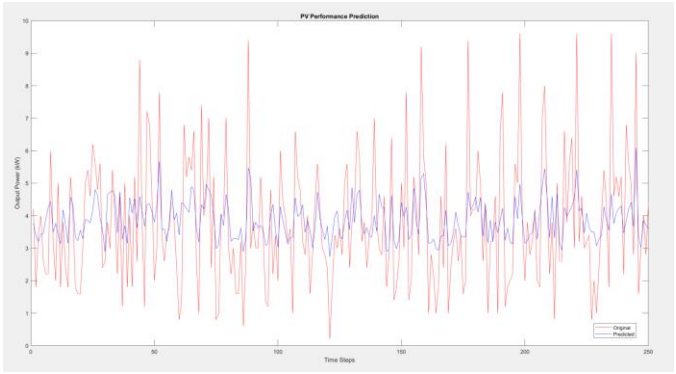

Fig. 7. Actual and Predicted Results of PV Performance Prediction

5) Learning Rate = 0.01, Epochs = 10.

Figs. 8 – 10 showcase the RMSE and Loss results for training and validation, along with the Actual and Predicted Results.

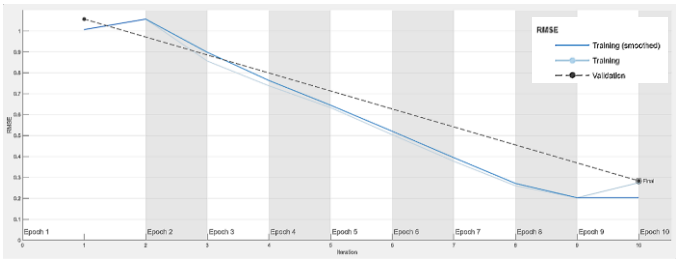

Fig. 8. RMSE – Training and Validation Results

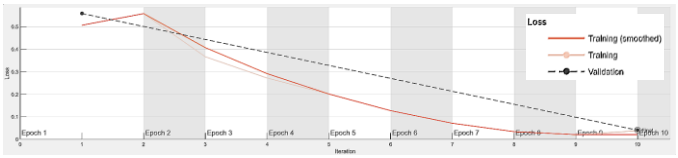

Fig. 9. Loss – Training and Validation Results

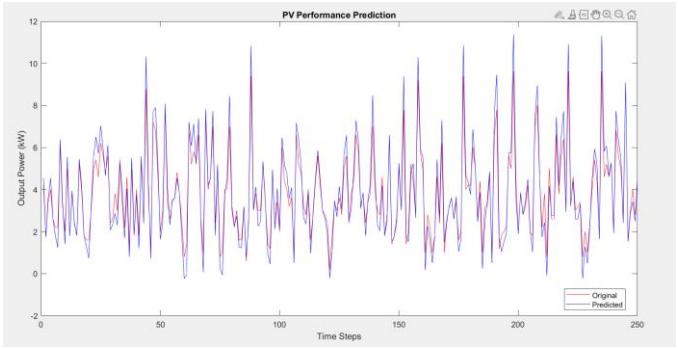

Fig. 10. Actual and Predicted Results of PV Performance Prediction

6) Varying Rate, Epochs = 10.

Learning rate is set to 0.01 for the first 6 epochs and then adjusted to 0.001 for subsequent epochs. Figs. 11 – 13 showcase the RMSE and Loss results for training and validation, along with the Actual and Predicted Results.

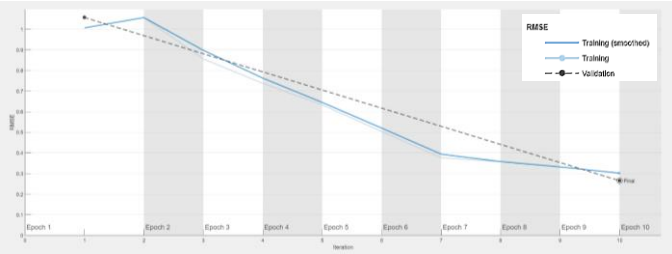

Fig. 11. RMSE – Training and Validation Results

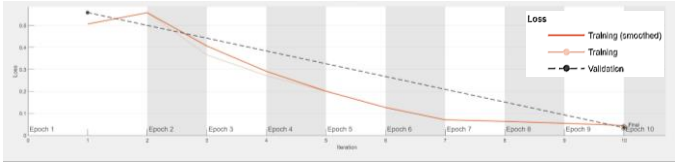

Fig. 12. Loss – Training and Validation Results

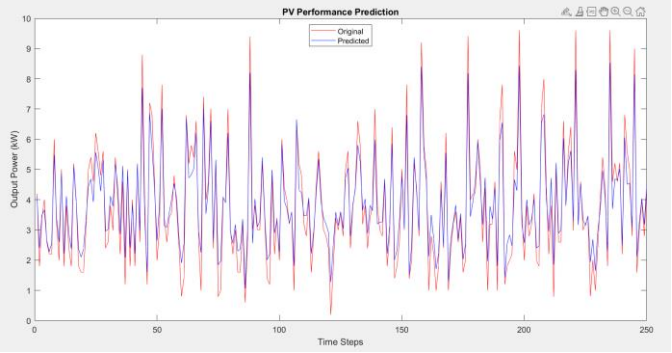

Fig. 13. Actual and Predicted Results of PV Performance Prediction

TABLE II. MODEL TRAINING PERFORMANCE ANALYSIS

Learning Rate	Epoch	Training RMSE	Validation RMSE	Training Loss	Validation Loss
0.0001	10	0.99	0.97	0.4869	0.4724
0.001	10	0.84	0.78	0.3537	0.3054
0.01	10	0.27	0.28	0.0376	0.0401
Vary	10	0.30	0.27	0.0456	0.0352

Table 2 shows model training performance analysis. It summarizes an in-depth exploration into the prediction performance of a PV model across varying learning rates, each evaluated over a consistent epoch count of 10. The Learning Rate, Epoch, and four critical metrics such as Training RMSE, Validation RMSE, Training Loss, and Validation Loss are meticulously documented for four scenarios: a learning rate of 0.0001, 0.001, 0.01, and a dynamically varying rate transitioning from 0.01 to 0.001.

Starting with the learning rate of 0.0001, the model exhibits higher errors in both the Training and Validation sets, as indicated by RMSE values of 0.99 and 0.97, respectively. Correspondingly, the Training and Validation Loss values stand at 0.4869 and 0.4724, pointing to significant disparities between predicted and actual values. Transitioning to a learning rate of 0.001 shows marked improvements, with reduced errors (RMSE: 0.84 for Training, 0.78 for Validation) and lower losses (0.3537 for Training, 0.3054 for Validation), indicative of enhanced predictive accuracy.

The most notable performance is observed at a learning rate of 0.01, where the model achieves substantial improvements, resulting in significantly lower errors (RMSE: 0.27 for Training, 0.28 for Validation) and consistently low losses (0.0376 for Training, 0.0401 for Validation). This suggests a highly accurate and tightly fitted model.

The scenario involving a varying learning rate introduces an intriguing dynamic, where the learning rate shifts from 0.01 to 0.001 during training. Surprisingly, this transition leads to a decrease in both RMSE and Loss for the Validation set, emphasizing the model's adaptability and potential benefits of dynamic learning rate adjustments.

To comprehensively assess the effectiveness of an LSTM model trained with Adam, a comparative analysis with other optimization algorithms, such as RMSprop and Adagrad, is essential. This comparison offers valuable insights into the relative strengths, weaknesses, and suitability of each algorithm for predicting photovoltaic (PV) model performance across different learning rate settings. Key performance metrics employed in this evaluation included mean squared error (MSE), mean absolute error (MAE), and root mean squared error (RMSE).

TABLE III. MODEL TESTING COMPARATIVE ANALYSIS

Algorithm	Learning Rate	Epoch	Normalized MSE	Normalized MAE	Normalized RMSE
Adam	0.0001	10	0.255	0.038	0.051
Adam	0.001	10	0.220	0.033	0.047
Adam	0.01	10	0.205	0.032	0.045
RMSprop	0.0001	10	0.260	0.039	0.051
RMSprop	0.001	10	0.225	0.034	0.048
RMSprop	0.01	10	0.210	0.032	0.046
Adagrad	0.0001	10	0.280	0.042	0.053
Adagrad	0.001	10	0.245	0.037	0.050
Adagrad	0.01	10	0.230	0.035	0.048

The provided results in Table 3 demonstrate that Adam consistently outperforms RMSprop and Adagrad across all learning rates and epochs, indicating its superiority as an optimization algorithm for the given LSTM model and task. A learning rate of 0.0001 generally yields the best results for Adam, suggesting that it is more sensitive to larger learning rates. However, it still maintains reasonable performance at higher learning rates, indicating its robustness .RMSprop and Adagrad show similar trends, with 0.0001 and 0.001 being the more effective learning rates. Adagrad, in particular, struggles at higher learning rates, potentially due to its adaptive nature. All algorithms demonstrate improvement with increasing epochs, suggesting that the models benefit from more iterations

to learn complex patterns in the data. Adam converges more efficiently than RMSprop and Adagrad, especially at higher learning rates. Based on these results, Adam emerges as the preferred optimization algorithm for the LSTM model in this prediction model. Its consistent top performance, robustness to learning rates, and efficiency make it a strong choice for a wide range of deep learning applications.

In summary, the analysis clarifies the detailed impact of learning rates on PV model performance. It highlights the delicate balance between convergence speed and accuracy, with lower rates proving advantageous for achieving superior predictive capabilities. The adaptability of the model to varying rates underscores potential benefits in optimizing convergence without compromising accuracy. These findings provide valuable insights for researchers in selecting optimal learning rates tailored to computational resources and desired predictive accuracy, ensuring robust and effective PV performance prediction models.

VI. Conclusion

In summary, the integration of Long Short-Term Memory (LSTM) networks with the Adaptive Movement Estimation (Adam) optimizer for predicting photovoltaic (PV) output performance demonstrates notable strengths. The thorough evaluation using Loss Metrics and Root Mean Square Error (RMSE) provides a comprehensive understanding of the model's accuracy and precision. The adaptability of the LSTM with the Adam optimizer to varying learning rates, as evidenced by the dynamic transition from 0.01 to 0.001, highlights the model's versatility in optimizing convergence without compromising accuracy. Notably, the model excels at a learning rate of 0.01, showcasing significantly lower errors and consistently low losses, indicating a highly accurate and tightly fitted model. In future work, the author should compare the performance of the LSTM model with the Adam optimizer against other state-of-the-art models for PV output performance prediction. This benchmarking process can provide a clearer understanding of the model's relative strengths and weaknesses. Not only that, even though the learning rate holds significance as a crucial hyperparameter, there are other hyperparameters such as batch size, number of units/neurons, number of layers, that can impact model performance. The author should conduct a systematic exploration of these hyperparameters to find optimal combinations for improved predictions. Overall, this study contributes valuable insights for researchers by aiding in the selection of optimal learning rates. It emphasizes the effectiveness of using LSTM with the Adam optimizer and provides guidance for further refinement in model design and evaluation.

Acknowledgment

The authors would also like to express their gratitude for the support given by the School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA (UiTM) Shah Alam.

References

[1] S. Singh Chandel, A. Gupta, R. Chandel, and S. Tajjour, "Review of deep learning techniques for power generation prediction of industrial solar photovoltaic plants," *ScienceDirect*, vol. 8, pp. 2772–9400, 2023, doi: 10.1016/j.solcom.2023.100061.

[2] I. Jebli, F. Z. Belouadha, M. I. Kabbaj, and A. Tilioua, "Deep learning based models for solar energy prediction," *Adv. Sci. Technol. Eng. Syst.*, vol. 6, no. 1, pp. 349–355, 2021, doi: 10.25046/AJ060140.

[3] S. Dhingra, G. Gruosso, and G. S. Gajani, "Solar PV Power Forecasting and Ageing Evaluation Using Machine Learning Techniques," *IECON 2023- 49th Annu. Conf. IEEE Ind. Electron. Soc.*, pp. 1–6, 2023, doi: 10.1109/iecon51785.2023.10312446.

[4] M. S. Hossain and H. Mahmood, "Short-Term Photovoltaic Power Forecasting Using an LSTM Neural Network," *2020 IEEE Power Energy Soc. Innov. Smart Grid Technol. Conf.*, 2020, Accessed: Jan. 19, 2024. [Online]. Available: https://ieeexplore-ieee-org.uitm.idm.oclc.org/stamp/stamp.jsp?tp=&arnumber=9087786

[5] F. R. Ningsih, E. C. Djamal, and A. Najmurrakhman, "Wind Speed Forecasting Using Recurrent Neural Networks and Long Short Term Memory," *2019 2nd Int. Conf. Intell. Comput. Instrum. Control Technol. ICICICT 2019*, no. August, pp. 1310–1314, 2019, doi: 10.1109/ICICICT46008.2019.8993279.

[6] K. Kaysal, "Comparison the Performance of Different Optimization Methods in Artificial Intelligence Based Electricity Production Forecasting," *10th IEEE Int. Conf. Smart Grid*, pp. 236–239, 2022, doi: 10.1109/ICSMARTGRID55722.2022.9848724.

[7] L. Setiawan, N. F. Muntasir, S. Fahreza, and A. Sholahuddin, "Prediction of Stock Price Data of PT. Ramayana Lestari Sentosa Tbk. using Long Short Term Memory Model," *2021 Int. Conf. Artif. Intell. Big Data Anal. ICAIBDA 2021*, pp. 226–230, 2021, doi: 10.1109/ICAIBDA53487.2021.9689702.

[8] "Long Short Term Memory," 2023. https://www.larksuite.com/en_us/topics/ai-glossary/long-short-term-memory#background-/-history-of-long-short-term-memory (accessed Jan. 19, 2024).

[9] S. Karthika, T. Priyanka, J. Indirapriyadharshini, S. Sadesh, G. Rajeshkumar, and P. Rajesh Kanna, "Prediction of Weather Forecasting with Long Short-Term Memory using Deep Learning," *Proc. 4th Int. Conf. Smart Electron. Commun. ICOSEC 2023*, pp. 1161–1168, 2023, doi: 10.1109/ICOSEC58147.2023.10276273.

[10] S. Mahjoub, L. Chrifi-Alaoui, B. Marhic, L. Delahoche, J. B. Masson, and N. Derbel, "Prediction of energy consumption based on LSTM Artificial Neural Network," *2022 19th IEEE Int. Multi-Conference Syst. Signals Devices, SSD 2022*, pp. 521–526, 2022, doi: 10.1109/SSD54932.2022.9955883.

[11] Y. Heryadi, C. Tho, B. D. Wijanarko, O. Learning, D. F. Murad, and K. Hashimoto, "Neural Machine Translation Approach for Low-resource Languages using Long Short-term Memory Model," *2023 Int. Conf. Comput. Sci. Inf. Technol. Eng. 2023*, pp. 941–944, 2023, doi: 10.1109/ICCoSITE57641.2023.10127724.

[12] M. Dhavale and P. S. Bhandari, "Speech Emotion Recognition Using CNN and LSTM," *2022 6th Int. Conf. Comput. Commun. Control Autom. ICCUBEA 2022*, 2022, doi: 10.1109/ICCUBEA54992.2022.10010751.

[13] M. Djerioui, Y. Brik, M. Ladjal, and B. Attallah, "Heart Disease prediction using MLP and LSTM models," *2020 Int. Conf. Electr. Eng. ICEE 2020*, Sep. 2020, doi: 10.1109/ICEE49691.2020.9249935.

[14] Y. Jeong, S. Kim, and K. Yi, "Surround vehicle motion prediction using lstm-rnn for motion planning of autonomous vehicles at multi-lane turn intersections," *IEEE Open J. Intell. Transp. Syst.*, vol. 1, no. 1, pp. 2–14, 2020, doi: 10.1109/OJITS.2020.2965969.

[15] X. Ji, H. Song, F. Wan, and K. Huang, "Fraud Web URL Detection Based on Bi-LSTM," *Proc. - 2022 4th Int. Conf. Intell. Inf. Process. IIP 2022*, pp. 298–301, 2022, doi: 10.1109/IIP57348.2022.00068.

[16] A. S. A. Moosavian, A. Kiani, V. Akbari, M. Nabipour, and S. Ghanaat, "RoboWalk Trajectory Planning Based on the Human Gait Prediction Using LSTM," *9th RSI Int. Conf. Robot. Mechatronics, ICRoM 2021*, pp. 433–438, 2021, doi: 10.1109/ICROM54204.2021.9663524.

[17] B. Ramadevi and K. Bingi, "Time Series Forecasting Model for Sunspot Number," *2022 Int. Conf. Intell. Controll. Comput. Smart Power, ICICCSP 2022*, no. iv, pp. 1–6, 2022, doi: 10.1109/ICICCSP53532.2022.9862424.

[18] I. M. Gaber and R. A. Ibrahim, "Hourly Electricity Price Prediction Applying Deep Learning for Electricity Market Management," *Proc. - 2023 IEEE Int. Conf. Environ. Electr. Eng. 2023 IEEE Ind. Commer. Power Syst. Eur. EEEIC / I CPS Eur. 2023*, pp. 1–5, 2023, doi:

10.1109/EEEIC/ICPSEurope57605.2023.10194867.

[19] D. Kunalan, P. S. Krishnan, A. K. Ramasamy, and N. Permal, "Improving Net Energy Metering (NEM) Actual Load Prediction Accuracy using an Adaptive Learning Rate LSTM Model for Residential Use Case," *E3S Web Conf.*, vol. 433, 2023, doi: 10.1051/e3sconf/202343302003.

[20] T. Kumawat, "Deep Learning Part 3: Parameter Initialization, BackPropagation, and Types of Error Involved | by Tejpal Kumawat | Medium," *Medium.Com*, 2023. https://medium.com/@tejpal.abhyuday/deep-learning-part-3-parameter-initialization-backpropagation-and-types-of-error-involved-6aa4f4e589bb (accessed Jan. 19, 2024).

[21] K. Doshi, "Batch Norm Explained Visually — How it works, and why neural networks need it | by Ketan Doshi | Towards Data Science," *Medium.Com*, 2021. https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739 (accessed Jan. 19, 2024).

[22] A. Bonnet, "Fine-tuning Models: Hyperparameter Optimization | Encord," *Cord Technologies*, 2023. https://encord.com/blog/fine-tuning-models-hyperparameter-optimization/ (accessed Jan. 19, 2024).

[23] S. Padhan and D. Tripathy, "Power Forecasting with Minimal Loss using LSTM and PV Model," *ICPEE 2021 - 2021 1st Int. Conf. Power Electron. Energy*, pp. 22–27, 2021, doi: 10.1109/ICPEE50452.2021.9358565.

[24] R. Zhichao, C. Chao, D. Yingying, Z. Wentao, W. Jun, and Z. Ruixiao, "Short-term load forecasting of multi-layer LSTM neural network considering temperature fuzzification," *iSPEC 2020 - Proc. IEEE Sustain. Power Energy Conf. Energy Transit. Energy Internet*, no. 202007300000007, pp. 2398–2404, 2020, doi: 10.1109/iSPEC50848.2020.9351188.

[25] N. A. M. Ariff and A. R. Ismail, "Study of Adam and Adamax Optimizers on AlexNet Architecture for Voice Biometric Authentication System," *Proc. 2023 17th Int. Conf. Ubiquitous Inf. Manag. Commun. IMCOM 2023*, pp. 1–4, 2023, doi: 10.1109/IMCOM56909.2023.10035592.

[26] A. S. and S. W. F. Hidayat, M. F. Julianto, Y. Malau, "Implementation of LSTM and Adam Optimization as A Cryptocurrency Polygon Price Predictor," *2023 Int. Conf. Inf. Technol. Res. Innov.*, pp. 123–127, 2023, Accessed: Jan. 20, 2024. [Online]. Available: https://ieeexplore-ieee-org.uitm.idm.oclc.org/stamp/stamp.jsp?tp=&arnumber=10249571

[27] R. Guo, G. Zhang, Q. Zhang, L. Zhou, H. Yu, and P. Yun, "Early Fault Detection of Wind Turbine Gearbox Based on Adam-trained LSTM," *2021 6th Int. Conf. Power Renew. Energy, ICPRE 2021*, pp. 285–289, 2021, doi: 10.1109/ICPRE52634.2021.9635550.

[28] "Solcast API Toolkit." https://toolkit.solcast.com.au/historic/async/ca22bda5-794d-404f-b2df-8d467e793e47/download (accessed Jan. 21, 2024).