Short Term Solar Power Forecasting with Long-Short Term Memory (LSTM) Networks

Nathanael Cadicamo
Symbolic Systems
cadicamo@stanford.edu

Riya Karumanchi
Computer Science
riyakaru@stanford.edu

Irawadee Thawornbut
Computer Science
irawadee@stanford.edu

Jessica Yang
Human Biology
jyang066@stanford.edu

Abstract

This project focuses on enhancing solar power output forecasting using Long Short-Term Memory (LSTM) networks, a specialized form of recurrent neural networks (RNNs) adept at time-series analysis. Addressing the need for accurate short-term solar power production forecasts, we preprocessed historical weather and solar power data from two solar plants in India to align weather conditions with power generation on an hourly basis. We developed a custom LSTM model that builds off TensorFlow's standard LSTM by conducting extensive hyperparameter optimization to refine our model and enhance it's predictability for solar power output. Our results demonstrate our LSTM model's superior accuracy over a baseline linear regression model, highlighting LSTM's effectiveness in capturing the complex dynamics of solar power generation. Our project, with its code available on GitHub, could ultimately lead to more reliable and efficient solar power forecasting and management of solar energy resources.

1 Introduction

This project aims to predict solar power output using historical weather and solar power generation data. The primary goal is to develop a model that can accurately forecast short-term solar power production, which is crucial for efficient energy management and planning in solar power plants. We explore the use of Long Short-Term Memory (LSTM) neural networks, a type of recurrent neural network (RNN) known for their effectiveness in handling time series data.

2 Literature Review

There has been extensive literature on short-term solar power forecasting in recent years, particularly those utilizing Long Short-Term Memory (LSTM) networks, due to growing reliance on renewable energy sources. Gensler et al. (2016) showcased the superior forecasting performance of deep learning algorithms, including LSTM, compared to Artificial Neural Networks (ANNs) and other reference models. Several papers on the problem of solar power prediction appear across the literature that found LSTM performed better than simpler neural networks such as ANNs, physical models, and other reference models [1] [2] [3] [4]. Voyant et al. (2017) built upon Gensler's work by providing a comprehensive overview of various methodologies used in solar radiation and power forecasting. This paper discussed the strengths and limitations of various approaches including time series, stochastic, machine learning, and hybrid models to provide a broader perspective on solar power prediction techniques [5].

Harrou et al. (2020) categorized solar power forecasting methods into four distinct classes: statistical, machine learning, physical models, and hybrid approaches. The paper emphasized the suitability of LSTM models for photovoltaic (PV) solar power production forecasting due to their ability to model time-dependent data, reporting that relatively small batch sizes and large epochs were successful [2]. Focusing on short-term PV power forecasting, Hossain et al. (2020) then highlighted LSTM's efficacy in modeling temporal changes in data that led to reduced forecasting errors compared to other methods. This research is particularly relevant for our project, as it directly applies LSTM for the kind of solar power forecasting we undertake. In Liu et al. (2021), a simplified LSTM model tailored for one day-ahead solar power forecasting demonstrated notable success with using intra-hour ramping effectively under various weather scenarios with a RMSE score of 0.512, outperforming traditional Multilayer Perceptron (MLP) models in accuracy and suggesting LSTM may be promising for short-term forecasting [1].

There have been many hybrid LSTM models used for weather predictions and forecasting that

have built upon the principles generated from previous research, including Suleman et al. (2022) using Spatial Feature Attention Long Short-Term Memory (SFA-LSTM) model to capture accurate spatial and temporal relations for temperature prediction, Venkatachalam et al. (2023) using Transductive Long Short-Term Memory (T-LSTM) model for weather forecasting [6] [7].

Overall, the approaches in these key papers are complementary to our approach. Our data is on the scale of 15 minute intervals, and thus, we hope to find success with intra-hour ramping as with Liu et al. (2021), and further, our broad-level LSTM approach is comparable with each paper. However, we will build a custom LSTM model by conducting extensive hyperparameter optimization to refine our model, so in this way, we stand out from this selection of the literature and can identify the most optimal configurations of hyperparameters that significantly enhance the model's predictability for solar power output.

3 Data

We utilize two pairs of files taken from Kaggle with each pair consisting of one solar power generation dataset and one sensor readings dataset collected over a 34-day period from two solar power plants in India. The sensor data is gathered from a single array of sensors placed at the plant, while the power generation datasets are gathered from inverters, each of which has multiple lines of solar panels attached to it. Each solar power generation dataset has around $69,000$ time-intervals that each contain the date and time, plant ID, source key, DC-power output, AC-power output, daily yield, and total yield. Each sensor reading dataset has approximately 3100 time-intervals that each contain data on date and time, plant ID, source key, ambient temperature, module temperature, and irradiation.

To preprocess the data for our LSTM model, we read the sensor reading CSV file into a pandas DataFrame `df_weather`, converted the `'DATE_TIME'` column in `df_weather` to Python's datetime format for easy extraction of date and time components, and added new columns `'DATE'` and `'HOUR'` by extracting the date and hour from the `'DATE_TIME'` column. We then grouped the data by `'DATE'` and `'HOUR'`, and calculated the maximum and minimum ambient temperatures (`'AMBIENT_TEMPERATURE'`) and irradiation (`'IRRADIATION'`) for each group. The columns of our DataFrame were then reset to `['DATE', 'HOUR', 'MAX_AMBIENT_TEMP', 'MIN_AMBIENT_TEMP', 'MAX_IRRADIATION', 'MIN_IRRADIATION']`. Similarly, we read the power data CSV file into a DataFrame `df_power`, extracted `'DATE'` and `'HOUR'` columns, grouped the data by `'DATE'` and `'HOUR'`, and calculated the maximum `'DAILY_YIELD'` for each group.

To merge and align the two datasets, we merged the power and weather DataFrames using the `'DATE'` and `'HOUR'` columns, creating a combined dataset (`df_merged`) that aligns the weather conditions with the corresponding power yield for each hour of each day. Effectively, this preprocessing of our files allowed us to efficiently consolidate the weather and power generation data on an **hourly** basis, providing a more refined dataset to be used in our advanced time-series predictions using LSTM models. We then translated this data into sequences of 24 hours at a time for each continuous 24-hour block across the timescale of the data. Finally, we divided the dataset into the first 80% as the training dataset, the next 10% for the validation dataset, and the next 10% for the testing dataset. Note that this split is sequential, as required for time-series analysis.

4 Baseline: Linear Regression

We established a baseline model using linear regression to help us evaluate our LSTM model in later stages. Our approach involved (i) selecting key features from the daily weather data, specifically the maximum ambient temperature `high_temp`, minimum ambient temperature `low_temp`, and maximum irradiation `max_irradiation` (since the minimum irradiation for each day is 0; (ii) selecting the maximum total daily power yield of the plant, representing the daily solar power output, as our target variable for our model; and (iii) utilizing a linear regression model from the scikit-learn library, which learned a linear relationship between the selected weather features and the solar power output.

For Plant 1, without normalization, the learned weights were $[211.68, -74.65, 3210.23]$ with a bias of -729.28. With normalization, the weights adjusted to $[0.42, -0.06, 0.46]$ with a bias of -0.20. The mean absolute percent difference was 10.21% for the training set and 9.66% for the test

set. R-squared errors were 0.29 for the training set and 0.31 for the test set. Mean squared errors (MSE) were 0.05704 and 0.05064, respectively.

For Plant 2, without normalization, the learned weights were $[408.17, -19.15, 1241.56]$ with a bias of -6638.32. With normalization, the weights were $[0.90, -0.02, 0.14]$ with a bias of -1.31. The mean absolute percent difference was 6.79% for the training set and 6.69% for the test set. R-squared errors were 0.76 for the training set and 0.71 for the test set. MSE were 0.01675 and 0.01814, respectively.

The baseline model's performance varied between the two plants, but the overall results indicate that while linear regression provides a reasonable starting point, solar power prediction may require more sophisticated models such as LSTM to capture non-linear relationships, temporal dependencies in the data, and improve accuracy. The mean absolute percent differences and R-squared errors highlight the linear regression model's limitations in accurately capturing the variability in solar power output. Additionally, normalization appeared to significantly impact the model's weights and bias, suggesting that feature scaling is crucial for the linear regression model's performance. The next phase of our project thus involves developing and comparing an LSTM model against this baseline to assess performance gains.

5 Main Model: LSTM

Our main approach for solar power prediction centers on implementing a Long Short-Term Memory (LSTM) network to process time-series data and capture long-term dependencies for solar power prediction. We implement this with TensorFlow and perform an extensive hyperparameter search to optimize test loss on the model.

Our LSTM model consists of LSTM cells that process input sequences representing weather conditions and output the predicted solar power generation. Each LSTM cell is comprised of input gates, forget gates, output gates, a cell state, and a hidden state. The input to our model includes sequences of weather-related features such as ambient temperature, irradiation levels, and other relevant weather data, structured in sequence. The output is the predicted solar power generation for subsequent time intervals, where each sequence consists of 24-hour data points.

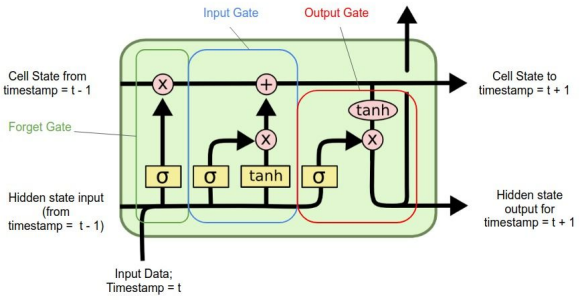

Figure 1: A depiction of an LSTM cell with associated gates and activation functions. (Source: Medium)

6 Evaluation Metric

To evaluate our model, we principally consider mean squared error (MSE) on the test loss. We examine this metric over the large hyperparameter combination space (1900 unique hyperparameter combinations for each plant) to minimize test loss. We then compare MSE values from LSTM to our baseline linear regression.

7 Results and Analysis

Our baseline linear regression model demonstrated a varying degree of predictive accuracy for two different power plants after compressing all our data into full-day units and only applying linear regression. We got MSE (on normalized data) for the test set on plant 1 and 2 of 0.0506 and 0.0181, respectively. This analysis reveals linear regression's fundamental limitations for predicting solar power output from weather data, as linear models entirely ignore the time-series dependencies inherent to weather data, and factors like irradiation and temperature that affect solar output in ways that are not strictly additive—often involving complex, multiplicative, or threshold-based interactions.

In contrast, our TensorFlow LSTM error rates were, at best, lower by an order of magnitude. Initially, these test loss MSE rates were 0.006251 and 0.012048 for Plants 1 and 2 on hyperparameters of 100 epochs, batch size 20, number of layers 50. After optimizing the hyperparameters, however, a process that will be further explained in the latter parts of this section, we saw a low normalized MSE loss value of 0.004715 and 0.009615 for Plant 1 and Plant 2, respectively, which on average was 130.97% lower than the error for linear regression, and 24.3281% lower than the test loss error without hyperparameter optimization. On the Tensorflow-based LSTM network we built, it is evident that these types of networks are more

well-suited for this task over more reductive, linear regression models. This is primarily due to their ability to model temporal dependencies and handle more complex, non-linear relationships. LSTMs particularly excel in capturing the sequential nature of weather data, as predicting solar power output is highly dependent on both current and past weather patterns. Unlike linear regression, LSTMs can learn and remember long-term dependencies. This makes them quite adept at understanding the nuances of daily and seasonal variations in sunlight and temperature, which directly impact solar energy production. Additionally, LSTMs are capable of managing the non-linear interactions between weather variables, such as how irradiation and ambient temperature synergistically affect solar panel output. This ability to process sequential data, combined with their efficacy for understanding data irregularities and their feature learning capabilities, makes LSTMs a more advanced and fitting choice for accurately forecasting solar energy output in a way that outperforms the linear regression baseline we built.

As mentioned, to fine-tune the model, our team ran a hyperparameter optimization program that tested a large combination (1900 unique combinations) of hyperparameters to identify those that produced the lowest test set error.

Table 1: Average of Best 25 Hyperparameter Values

	num_layers	num_epochs	batch_size	test_loss
Plant 1	65.20	155.60	23.20	0.004897
Plant 2	64.80	144.40	16.40	0.010472

Table 2: Overall Best Hyperparameters

	num_layers	num_epochs	batch_size	test_loss
Plant 1	70	160	10	0.004715
Plant 2	30	140	10	0.009615

Table 1: Test loss for the average of the best 25 combinations of hyperparameters by test loss.
Table 2: Overall best hyperparameters by test loss.

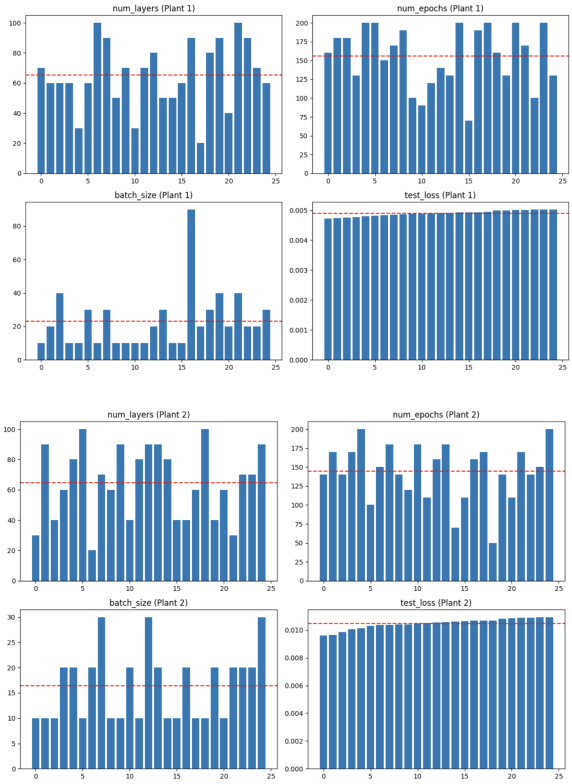

Figure 2: Plots of top 25 hyperparameter values by test loss. Note that the horizontal axis gives the position in rank for a hyperparameter by the associated test loss value. The red line indicates the average value expressed in Table 1.

Upon close examination of the hyperparameter tuning results, the optimal settings for both Plant 1 and Plant 2 indicate a preference for more complex LSTM models with a higher number of layers and a longer training process, as evidenced by the lower test losses associated with these values. The leftmost bars, representing the lowest test losses, suggest that around 70 layers for Plant 1 and slightly fewer for Plant 2, coupled with lower batch sizes and a substantial number of epochs (around 160 for Plant 1 and 140 for Plant 2, averaging to 150 epochs), are the most effective configurations. Specifically, the average number of layers and epochs is fairly similar between plants, while Plant 1 has, on average, a higher average batch size than Plant 2. This could indicate that Plant 1's model requires more training to converge to an optimal solution and it responds better to longer training and greater data without reducing performance. Since the best batch size for both plants is 10, this suggests that smaller batch sizes may lead to better performance, generalization, and noise reduction. However, batch size doesn't seem to

have the most consistent impact on test loss, suggesting that there may be other factors we need to investigate or that there is an optimal range that isn't clearly visible from our data. Furthermore, the test loss for both plants generally fell below the threshold line, implying the configurations we chose are generally effective.

The variability across the top 25 configurations for each parameter indicates a nuanced interaction between hyperparameters and model performance. While the red dashed line might suggest considering the average as a potential strategy, it's important to recognize that the best model performance is typically achieved through the synergistic effect of a specific combination of hyperparameters rather than their individual averages. Therefore, the optimal approach would likely involve selecting the best individual configurations as a basis and then fine-tuning around those values to enhance model performance. To approach fine-tuning, we could begin with the most promising individual combination, such as the configuration with around 70 layers and a higher epoch count for Plant 1, and experiment around batch size values (whose general pattern had the most disparity between plants), adjusting one hyperparameter at a time. This iterative fine-tuning process, guided by cross-validation to prevent overfitting, would allow us to hone in on the most effective and robust model parameters for predicting solar output given the characteristics of our dataset.

Moreover, in the course of our study, we observed a noteworthy variation in the test loss between Plant 1 and Plant 2 when applying a linear regression model, with Plant 1 exhibiting a considerably higher test loss compared to Plant 2. This outcome was suggestive of Plant 1's data being more complex or possibly more susceptible to certain weather variables that linear models fail to capture due to their inherent limitations in handling non-linear interactions over time. Upon the implementation of LSTM models, the test losses for both plants not only declined but also brought to light an intriguing reversal in the discrepancy pattern: Plant 2 displayed a higher test loss than Plant 1. This shift implies that while LSTM's ability to model temporal dependencies does enhance predictive accuracy overall, the nuances of each plant's data may necessitate a tailored approach in hyperparameter tuning to account for individual characteristics such as noise levels and weather volatility.

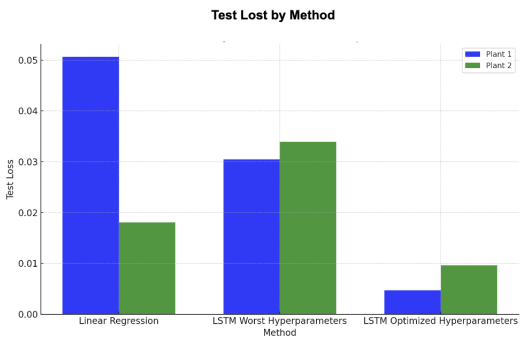

Figure 3: Test Loss (MSE) by ML Method Used.

The discrepancy in test losses observed between the plants underlines the necessity for model adaptability when applied to different solar power plants. For instance, the LSTM model's hyperparameters that are optimal for one plant may not necessarily translate to another due to differing environmental conditions and data characteristics. It is this adaptability and the need for fine-tuning that will form a cornerstone of our future work, ensuring that the model remains both accurate and generalizable across various plants and conditions.

8 Future Work

One key limitation of our model includes its reliance on historical data, especially in the context of climate change and evolving weather patterns. Our model is also heavily dependent on the quality and comprehensiveness of the input data, and there is a need to validate its performance in different settings to fully understand its scope and limitations.

Therefore, for future work, we hope to further optimize our model by finetuning our LSTM architecture and hyperparameters using a more granular search such as Bayesian optimization to more efficiently navigate the search space and possibly uncover better-performing model hyperparameters or configurations. Another direction for development includes incorporating additional predictive features–including wind speed, humidity, and climate phenomena like El Niño/La Niña cycles–that could significantly increase our model's predictive power. We also hope to further test our LSTM on diverse weather-related datasets from different geographical locations, encompassing a wider range of features and longer timespans. This will help us assess our model's adaptability, generalizability, and predictive accuracy across different scenarios. It would also be interesting to adapt our model for real-time prediction and explore our model's applicability to other forms of renewable energy

forecasting like wind power. We may also consider exploring the use of hybrid models to further boost our model's predictive accuracy.

9 Conclusion

This study advanced solar power output forecasting by employing Long Short-Term Memory (LSTM) networks, using historical weather data from two Indian solar plants. We demonstrated the limitations of linear regression models and developed a more effective TensorFlow-based LSTM model, which showed significantly better performance in forecasting accuracy. We achieved this through comprehensive hyperparameter optimization, enhancing our LSTM model's capability to understand the complex, time-dependent dynamics of solar energy data.

Future research directions include further refining our LSTM model with diverse weather datasets, incorporating additional environmental variables, and applying these models to other renewable energy sources for broader forecasting applications. The potential for real-time forecasting and the exploration of hybrid models combining LSTM with other algorithms also present future avenues for enhancing energy management practices. Our work opens new possibilities in renewable energy forecasting, potentially contributing to the development of more efficient and sustainable energy utilization.

Code

Please visit this GitHub repository to view code and data used for this project.

References

[1] Liu, Chun-Hung, Jyh-Cherng Gu, and Ming-Ta Yang. "A simplified LSTM neural networks for one day-ahead solar power forecasting." Ieee Access 9 (2021): 17174-17195.

[2] Harrou, Fouzi, Farid Kadri, and Ying Sun. "Forecasting of photovoltaic solar power production using LSTM approach." Advanced statistical modeling, forecasting, and fault detection in renewable energy systems 3 (2020).

[3] Gensler, André, et al. "Deep Learning for solar power forecasting—An approach using AutoEncoder and LSTM Neural Networks." 2016 IEEE international conference on systems, man, and cybernetics (SMC). IEEE, 2016.

[4] Hossain, Mohammad Safayet, and Hisham Mahmood. "Short-term photovoltaic power forecasting using an LSTM neural network and synthetic weather forecast." Ieee Access 8 (2020): 172524-172533.

[5] Voyant, Cyril, et al. "Machine learning methods for solar radiation forecasting: A review." Renewable Energy, Volume 105, 2017, Pages 569-582, ISSN 0960-1481.

[6] Suleman, Masooma Ali Raza, and S. Shridevi. "Short-term weather forecasting using spatial feature attention based LSTM model." IEEE Access 10 (2022): 82456-82468.

[7] Venkatachalam, K., et al. "DWFH: An improved data-driven deep weather forecasting hybrid model using Transductive Long Short Term Memory (T-LSTM)." Expert Systems with Applications 213 (2023): 119270.