Hybrid Energy inspired DC Micro-grid System using Machine Learning

Samruddhi Khairnar, Balasaheb Tarle, Sakshi Jadhav, Kunika Narnaware, Pranjal Shewale
Department of Computer Engineering, MVPS's KBT College of Engineering, Nashik-422013, India
kbtug20170@kbtcoe.org, tarle.balasaheb@kbtcoe.org

***Abstract*—Traditional power plants make use of the manual demand and generation matching processes, causing time delays and wastage of electric power. Hence, a smart hybrid power plant management system is necessary, to minimize the wastage of electricity by controlling the integration of various energy sources, according to the observed power demand of the users. In this paper, an LSTM-based machine learning approach for energy demand forecasting is discussed, including model architecture, hyper-parameter tuning, evaluation details. The mean squared error of the trained model is 0.0053 on load demand data from the Dataset-of-HRP-38-test-system by Zhenyu Zhou [4] and 0.0013 on data collected from a simulated fan load. These reliable estimates can be made use of by hybrid powerplants to control the generation of electric power according to the estimated demand. Further, a naive algorithm for power scheduling is explored.**

***Index Terms*—Energy Demand Forecasting, Hybrid Power plant Controller, Energy Management System, Internet of Things, Machine Learning.**

I. Introduction

The installed capacity of thermal power in India is 237.26 GW, accounting for 60 % of the total electricity generation in India [3]. The rest of which is derived from non-renewable energy sources. 64 % of the consumers are located in rural areas, where renewable energy is abundant from biomass, wind, etc. These resources can be integrated with the main grid as per their availability, beyond which, electricity can be generated as per the estimated load demand of the users, conventionally, aiming to minimize the reliance on the conventional energy sources.

In the study by Liu et al. [1], the authors investigate the impact of distributed energy resources (DERs) on traditional power plants and dispatch control centers. They introduce a simplified LSTM algorithm for one day-ahead solar power forecasting, achieving an average RMSE of 0.512. The authors made use of data collected from PhotoVoltaic (PV) systems to train models to forecast one day-ahead solar power generation.

The study by Shibl et al. [2] proposes a two-stage machine learning based energy dispatch management system for hybrid power plants (HPPs), aiming to optimize renewable energy integration alongside backup sources. The first stage aims to forecast the output power of renewable energy sources, as well as the load demand. The second stage aims at coordinating the output power of the reserve and backup sources.

II. Model Architecture

A. LSTM Overview

Long Short Term Memory is a type of Recurrent Neural Network (RNN), used for learning sequences and time series data. It resolves the vanishing gradient issue of traditional RNNs. LSTM networks have cells that can store information over long time periods, allowing them to learn and remember patterns in time series data. These cells have an internal cell state and three gates (forget, input, and output gates).

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

1. **Forget Gate** (f_t): Enables an LSTM cell to selectively forget a certain amount of previously stored information

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

2. **Input Gate** (i_t): Enables an LSTM cell to selectively add new information at each time step.

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

3. **Candidate Cell State** $(\tilde{C}_t)$: It is the proposed update to the cell state at a given time step, calculated using the current input and the previous hidden state (output).

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

4. **Cell State** (C_t): Represents the long-term memory of the LSTM cell, allowing it to retain information over long sequences.

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

5. **Output Gate** (o_t): Determines what should be the output of the LSTM cell at any given time step.

$$h_t = o_t \cdot \tanh(C_t)$$

6. **Hidden State** (h_t): Captures the information learnt by an LSTM cell at a specific time step, as its current output.

- $\tilde{C}_t, C_t, h_t$ are the candidate, cell, and hidden states
- x_t is the incoming input at time step t
- W_f, W_i, W_C, W_o are weight matrices of the respective gates
- b_f, b_i, b_C, b_o are bias vectors used by the gates
- $\tanh$, σ are the hyperbolic tan, sigmoid activation functions
- $\tanh(z) = \dfrac{e^z - e^{-z}}{e^z + e^{-z}}$ and $\sigma(z) = \dfrac{1}{1 + e^{-z}}$

B. Data Preprocessing

The load demand data for compiling the model architecture was procured from the Dataset-of-HRP-38-test-system by Z. Zhou [4]. The load demand data was present in the form of hourly reading columns, which were flattened into a 1-D array of 8760 readings, followed by using a rolling window approach to generate a supervised learning dataset for LSTM.

$$[1, 2, 3, 4, 5] \Rightarrow [6]$$
$$[2, 3, 4, 5, 6] \Rightarrow [7]$$
$$[3, 4, 5, 6, 7] \Rightarrow [8]$$

Eg: Time series data = [1,2,3,4,5,6,7] and window size = 5. Five records from the data become a record of input data to be passed and 6^{th} record becomes the output of the model. The last record, 8, wasn't present in the dataset. Likewise, trained LSTM models can predict next data points using historical readings. The dataset is then reshaped into a 3D matrix of shape (inputs, time-steps, features) as shown below.

$$[[[1], [2], [3], [4], [5]]]$$

C. Data Normalization

Normalization is a technique used to scale up/down data to the range of [0,1]. Normalization brings all features on a comparable scale and makes it easier for models to train on the data and generalize better.

$$X_{\text{norm}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}$$

D. Splitting of the Dataset

The windowed and normalized data was divided into three disjoint sets - 80%, 10%, 10%, sequentially - **Training set**: used to train the model, **Validation set**: used to analyze the model's performance on unseen data while training, **Testing set**: used to evaluate the model's performance. The model is fine-tuned by the optimizer - **Adam** in our case, at every iteration, such that the validation error is minimized.

E. Hyperparameter Tuning and Model Evaluation

The following hyperparameters were tweaked and the models were evaluated on the testing set using the MSE metric.

1) No. of LSTM units (using a single LSTM layer).

Units	20	30	40	50	60	70	80	84
MSE	7.72	3.43	3.21	2.91	2.06	1.91	1.99	1.92

2) Dropout: percentage of random connections to be dropped in each layer (to avoid overfitting).

Dropout	0.1	0.2	0.3	0.4
MSE	1.91	4.19	3.43	4.33

3) Batch size: No. of data points after which the gradients are backpropagated.

Batch Size	16	32	48	60
MSE	1.81	1.861	2.15	1.97

4) Window size: The size of the rolling window.

Window Size	24	48	60	84
MSE	1.81	2.48	1.63	2.63

5) Layers: No. of hidden layers in the network.

Layers	1	2	3
MSE	1.63	2.09	2.21

TABLE I
FINALIZING THE BEST PERFORMING HYPERPARAMETERS

Parameter	Units	Dropout	Batch Size	Window Size	Layers
Best Value	70	0.1	16	60	1

On selecting the best value for each of the mentioned hyperparameters, the above table was obtained. After fitting 1000 epochs of a model with these parameters on the training dataset, the MSE (Mean Squared Error) obtained on the testing dataset is 0.0053.

$$\text{Mean Squared Error} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- n: Number of data records
- y_i: Actual value of the record i
- $\hat{y}_i$: Predicted value of the record i

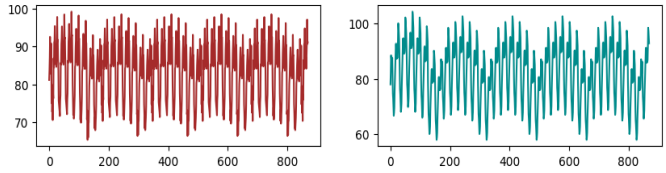

Fig. 1. Predicted (Left) and Actual (Right) Load Demand in GW vs Time

III. Load Demand Forecasting

A. Data Acquisition

The finalized hyperparameters were further used to fit an LSTM model to the data gathered from an actual fan load of 12V and maximum current rating of 1A, with a simulated behavior of changing speeds, controlled by a PWM (Pulse Width Modulation) speed regulator. The voltage dropped across the fan's ends and the line current was measured using sensors, at regular intervals, to obtain a total of 1475 readings.

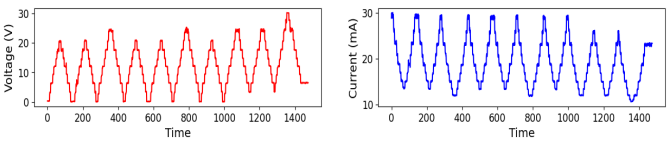

Fig. 2. Voltage and Current sensor readings

The sensor readings were calibrated to accurately represent the true voltage and current values. The voltage and current

values were then substituted in the below formula to obtain the load (power) demand readings.

$$\text{Power (W)} = \text{Voltage (V)} * \text{Current (mA)} * 0.001$$

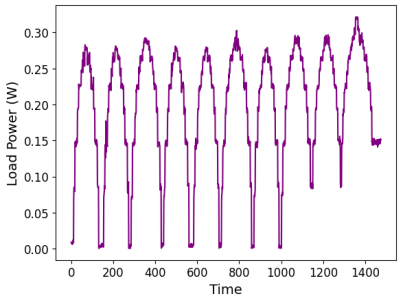

Fig. 3. Load Demand readings

B. Data Preprocessing

The load power readings were flattened into a 1-D series and a rolling window of size 60 was applied on to the data to transform it into a supervised learning matrix of sequences, using the approach discussed earlier. Further, the dataset was split into 80%, 10%, 10%, as the training, validation and testing sets followed by normalization.

C. Model Fitting and Evaluation

The best-performing hyperparameters listed in Table I were used to fit an LSTM model to the dataset, using a learning rate of 0.0001. The MSE of the trained model on the testing data was 0.0013. Hence, the model was successful in providing reliable estimates for the load demand pattern.

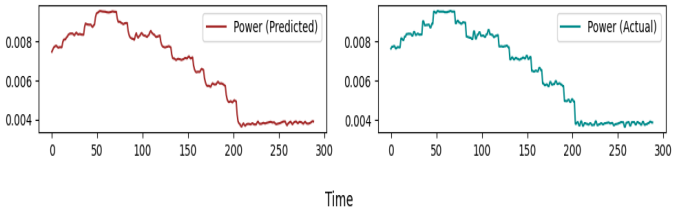

Fig. 4. Load Power (Predicted vs Actual) (W)

IV. Analysis of Solar Data

A. Data Acquisition

The modeling of load behavior was followed by the analysis of solar power generation data. A 12V, 10W solar panel was connected to a dummy motor load (12 V, 1.5A maximum current rating). The voltage generated across the ends of the panel, the line current in the circuit, illumination, temperature, and humidity at the site were measured using sensors, at regular intervals. A total of 3892 readings were collected. The aim of the data collection was to use current (which depends on the load) as the target feature, which can be predicted using the remaining features. Hence, given the - panel voltage, illumination, temperature, humidity - at any instant of the day,

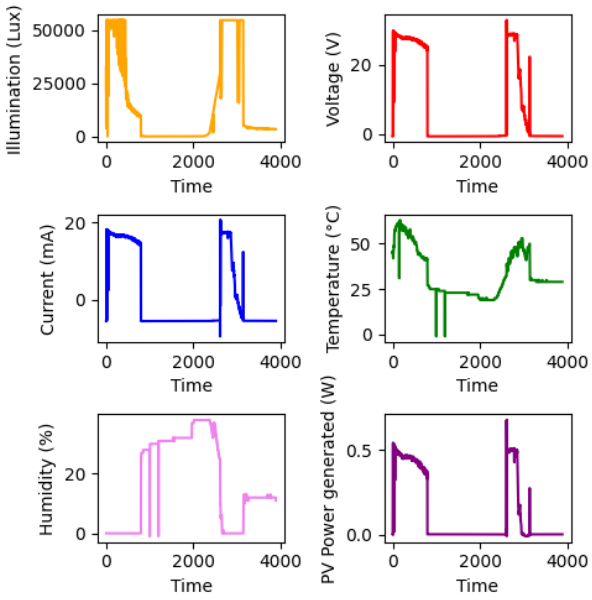

Fig. 5. Solar Power Generation readings

the line current can be predicted (even when no load is connected to the panel) and multiplied with the voltage to obtain the estimated solar power generation.

B. Data Preprocessing

The dataset was cleaned and split into 80%, 10%, and 10%, as the training, validation, and testing sets and normalized.

C. Model Training and Evaluation

A 2-layered MLP (Multi-Layer Perceptron) was trained on the dataset with the line current as the target feature. The layers contained 10 dense units each and ReLU (Rectified Linear Unit) activation function was applied at each layer. A learning rate of 0.0001 and the Adam optimizer were used to fit 100 epochs on the dataset.

$$\text{ReLU}(z) = \max(0,z)$$

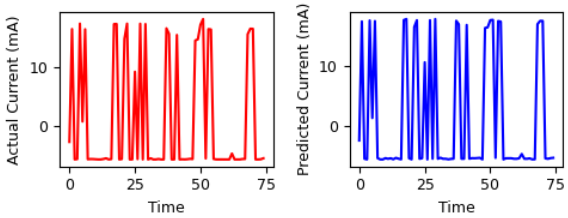

Fig. 6. Solar Line Current prediction

The trained model had an MSE of 0.31 on the testing set.

V. Energy Management System

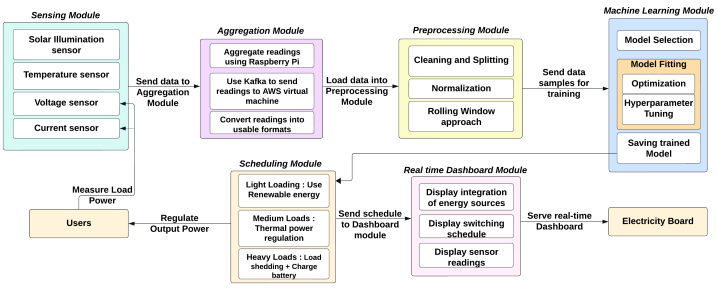

Fig. 7. System Architechture

Algorithm 1: Naive Power Scheduling Algorithm

Input: Sensors, Window size=60, Burst time=30 minutes
Output: None

```
(1)  while True do
(2)      Illumination, Voltage, Temperature, Humidity ← Get
           readings(Sensors);
(3)      Current ← Solar model.predict();
(4)      Estimated solar power ← Voltage * Current (mA) * 0.001;
(5)      Size ← length(Historical Load readings);
(6)      Windowed readings ← Historical Load readings[Size - window
           size : Size];
(7)      Estimated demand ← Load model.predict(Windowed readings);
(8)      Battery power ← Sense battery charging();
(9)      Conventional power ← Estimated demand - Estimated solar
           power - Battery power;
(10)     User List ← Get all users in the grid();
(11)     Demand List ← Sense demand when switching on(User List);
(12)     Sort(Demand List, Arrival time, Power demand);
(13)     foreach demand in Demand List do
(14)         if demand.max ≤ Solar power then
(15)             Solar power -= demand.max;
(16)             demand.assign(Solar line);
(17)         end
(18)         else if demand.max ≤ Conventional power then
(19)             Conventional power -= demand.max;
(20)             demand.assign(Conventional line);
(21)         end
(22)         else if demand.max ≤ Battery power then
(23)             Battery power -= demand.max;
(24)             demand.assign(Battery line);
(25)         end
(26)         else
(27)             pass;
                 // Implement Load shedding OR Connect
                    to battery backup line
(28)         end
(29)     end
(30)     Clear demand list();
(31)     Turn off assigned demands after(burst time);
(32)     Actual demand ← Sense line current() * Sense voltage drop();
(33)     View Actual vs Predicted Load on Dashboard();
(34) end
```

A. System Overview

The proposed system (Fig. 7) can be deployed in Hybrid power plants, to automate the power generation in real-time, as per the estimated load demand.

1) **Sensing Module:**
   a) An illumination sensor: for sunlight readings.
   b) A temperature and humidity sensor: weather data.
   c) A combination of voltage and current sensors, to obtain load demand readings.
2) **Aggregation Module:** Aggregation of sensor readings using a Raspberry Pi, sending them to a virtual machine running on the AWS Cloud using Apache Kafka.
3) **Preprocessing Module:** Cleaning and normalization of the aggregated data; applying a rolling window.
4) **Machine Learning Module:** Fitting an LSTM model to the dataset, tuning its hyperparameters, validating its accuracy using the MSE metric, saving the model.
5) **Power Scheduling Module:** Switches the power sources' line connections at the users' end so each user gets connected to either the solar or conventional source.
6) **Dashboard Module:** Displays the actual power generation and load demand versus the estimates provided.

B. Power Scheduling Algorithm

Power scheduling is performed by regulating the switches on the user line (busbar) as discussed in Algorithm 1. Each user is assigned to one of the power sources - solar, battery backup, or conventional power source, depending on the power available with each source. The following are three operating cases, considered in the scheduling algorithm.

1) **When Demand $<$ Renewable Generation**: The system allots renewable power to all users and extra power is used for charging backups.
2) **Renewable Generation $<$ Power Demand $<$ Power Plant Capacity**: Conventional Power $=$ Estimated demand $-$ Renewable Generation (output is controlled as per the load demand estimates). Uncertain loads can be handled using power backups/batteries.
3) **Power Plant Capacity $<$ Power Demand**: Load shedding is carried out in the intended user's power line.

VI. Results

The LSTM model trained on load demand data from the Dataset-of-HRP-38-test-system by Zhenyu Zhou [4] had a mean squared error of 0.0053 on the testing set, which comprised 10% of the readings from the entire dataset. The hyperparameters finalized from this training were used to train another model on data collected from a simulated fan load. The trained model resulted in an MSE of 0.0013 on the testing set.

VII. Conclusion

Hence, the domain of energy demand forecasting was explored, and an LSTM model was trained on the collected load demand dataset to estimate the power demand of the users in the future. Another neural network was trained to estimate the solar power generated at any time instant using data collected from an actual solar panel. A prototype for a Hybrid DC Microgrid was implemented, and power scheduling was done using a naive approach that makes use of the estimated load demand and solar power to regulate the conventional power. The users are allotted power from either of the sources, solar or conventional, depending on the power availability of each.

REFERENCES

[1] 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. 18967-18975, Jan. 22, 2021. DOI: 10.1109/ACCESS.2021.3053638.

[2] M. M. Shibl, L. S. Ismail, and A. M. Massoud, "An Intelligent Two-Stage Energy Dispatch Management System for Hybrid Power Plants: Impact of Machine Learning Deployment," *IEEE Access*, vol. 11, pp. 11212-11223, Feb. 6, 2023. DOI: 10.1109/ACCESS.2023.3243097.

[3] "Power Sector in India," India Brand Equity Foundation, Dec. 2023. Available online: https://www.ibef.org/industry/power-sector-india

[4] Zhenyu Zhuo, "DATASET-OF-HRP-38-TEST-SYSTEM," IEEE DataPort, DOI: 10.21227/ggy4-7497, May 17, 2022.

[5] B. Mohandes, M. Wahbah, M. S. El Moursi, and T. H. M. El-Fouly, "Renewable Energy Management System: Optimum Design and Hourly Dispatch," *IEEE Transactions on Sustainable Energy*, vol. 12, no. 3, pp. 1615, July 2021.

[6] H. Karami, M. J. Sanjari, S. H. Hosseinian, and G. B. Gharehpetian, "An Optimal Dispatch Algorithm for Managing Residential Distributed Energy Resources," *IEEE Transactions on Smart Grid*, vol. 5, no. 5, pp. 2360, September 2014.

[7] M. Tan, S. Yuan, S. Li, Y. Su, H. Li, and F. He, "Ultra-Short-Term Industrial Power Demand Forecasting Using LSTM Based Hybrid Ensemble Learning," *IEEE Transactions on Power Systems*, vol. 35, no. 4, pp. 2937, July 2020.

[8] J.-B. Fiot and F. Dinuzzo, "Electricity Demand Forecasting by Multi-Task Learning," *IEEE Transactions on Smart Grid*, vol. 9, no. 2, pp. 544, March 2018.

[9] S. S. Arnob, A. I. M. S. Arefin, A. Y. Saber, and K. A. Mamun, "Energy Demand Forecasting and Optimizing Electric Systems for Developing Countries," *IEEE Access*, vol. 11, pp. 11212-11223, Feb. 27, 2023. DOI: 10.1109/ACCESS.2023.3250110.

[10] L. Hernandez, C. Baladrón, J. M. Aguiar, B. Carro, A. J. Sanchez-Esguevillas, J. Lloret, and J. Massana, "A Survey on Electric Power Demand Forecasting: Future Trends in Smart Grids, Microgrids and Smart Buildings," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1460-1489, Third Quarter 2014.

[11] D. H. Vu, K. M. Muttaqi, A. P. Agalgaonkar, A. Zahedmanesh, and A. Bouzerdoum, "Recurring Multi-layer Moving Window Approach to Forecast Day-ahead and Week-ahead Load Demand Considering Weather Conditions," *Journal of Modern Power Systems and Clean Energy*, vol. 10, no. 6, November 2022.

[12] N. Uremović, M. Bizjak, P. Sukić, G. Štumberger, B. Žalik, and N. Lukač, "A New Framework for Multivariate Time Series Forecasting in Energy Management System," *IEEE Transactions on Smart Grid*, vol. 14, no. 4, July 2023.

[13] X. Wen and W. Li, "Time Series Prediction Based on LSTM-Attention-LSTM Model," *IEEE Access*, vol. 9, pp. 15-23, May 2023. DOI: 10.1109/ACCESS.2023.3276628.

[14] M. Tan, S. Yuan, S. Li, Y. Su, H. Li, and F. He, "Ultra-Short-Term Industrial Power Demand Forecasting Using LSTM Based Hybrid Ensemble Learning," *IEEE Transactions on Power Systems*, vol. 35, no. 4, pp. 2937, July 2020.

[15] J. Mubiru, "Predicting total solar irradiation values using artificial neural networks," *Renew. Energy*, vol. 33, no. 10, pp. 2329–2332, Oct. 2008.