International Journal on Engineering Technology (InJET)

www.kec.edu.np/kec-journal/
Volume 3, issue no. 1, Dec 2025, Page 36 - 45
Received Date: Jul 29, 2025
Accepted Date: Sep 23, 2025
ISSN: 3021-940X (Print)
DOI: https://doi.org/10.3126/injet.v3i1.87012

Spatiotemporal $PM_{2.5}$ Estimation in Kathmandu Using Deep Learning with OpenMeteo and NASA MERRA-2 Data: Performance Benchmarking Against Machine Learning Model

Kshitiz Raj Paudyal[1, *], Rajad Shakya[2], Jesis Upadhayaya[3]

[1]IOE Thapathali Campus, Kathmandu, Nepal, kshitizrajpaudyal@gmail.com
[2]IOE Thapathali Campus, Kathmandu, Nepal, rshakya.8063@tcioe.edu.np
3IOE Thapathali Campus, Kathmandu, Nepal, jesisupadhyayal@gmail.com

Abstract

In rapidly urbanizing regions like the Kathmandu Valley, air pollution, specifically fine particulate matter ($PM_{2.5}$), represents an increasing environmental and public health challenge. Traditional air quality monitoring systems are inherently limited in spatiotemporal scope, limiting the ability to conduct long-term air-quality assessments. This study aims to develop a framework for high-resolution historical $PM_{2.5}$ concentration reconstruction from previously limited spatiotemporal $PM_{2.5}$ datasets. To achieve this goal, we will incorporate historical Open-Meteo weather data and NASA MERRA-2 satellite reanalysis data as dependencies. The predictive models will be evaluated, namely the Deep Neural Network (DNN) and Extreme Gradient Boosting (XGBoost) two models using both hourly and daily datasets, based on qualitative and quantitative study metrics. Using the hourly Open-Meteo dataset, the DNN model achieved the highest accuracy ($R^2 = 0.8725$, RMSE = 18.23 µg/m³) and the XGBoost model performed best ($R^2 = 0.7827$, RMSE = 12.81 µg/m³) using the daily dataset. More generally, through evaluation, it appears that data quality and resolution can outweigh the effect of the algorithm in predicting $PM_{2.5}$. This framework demonstrates considerable promise for capturing nonlinear and temporal dependencies within air pollution dynamics to conduct high-fidelity $PM_{2.5}$ reconstruction. Findings support a data-informed basis for environmental and urban planning, forecasting of environmental issues, and public health intervention in the Kathmandu Valley.

Keywords: deep learning, air quality reconstruction, Kathmandu Valley

1. Introduction

In the year 2025, the air quality in Kathmandu Valley experienced a drastic decline. PM2.5 and AQI (Air Quality Index) levels remained at levels classified as extremely unhealthy throughout January–March and reached levels above international health limits on occasion. The ignition of wildfires in Nepal on April 1, 2025, aggravated the situation where the Valley became one of the most polluted cities in the world with a PM2.5 concentration of 365 µg/m³, which is classified as hazardous. An article published on April 3, 2025 stated that the air quality was reported as "unhealthy" for 75 of the previous 90 days. The AQI values from nearby monitoring stations peaked at 273 (Shankapark), 248 (Ratnapark), 235 (Bhaisipati), and at 235 (Khumaltar)with values exceeding international thresholds, it worsened the air quality . (ICIMOD, 2025) Although these numbers are a bit alarming, we will need to analyze the respective contributions of each of these sources (wildfires, industrial discharges, vehicular discharges, etc.) to the concentration of different air pollutants. Air pollution has become one of the challenging problems in urban regions around the world. In the Kathmandu Valley, air quality has worsened because of rapid, uncontrolled urban growth, more vehicles on the road, chaotic construction activities, and emission from industrial activities, though further analysis is required to quantify their individual contributions. One major pollutant is fine particulate matter that measures 2.5 micrometers or smaller ($PM_{2.5}$). This very small particle can present heightened health risks as it may pass into the lungs and enter the bloodstream - resulting in a host of respiratory-related illnesses.

Despite the growing concern, consistent long-term records of $PM_{2.5}$ concentrations in Kathmandu have been missing, especially for the decades before the 2010s when air quality monitoring systems were not in place. The lack of data has limited the ability to examine long-term air quality trends, limiting the knowledge of public health risks and the capacity for evidence-based environmental policy development.

To tackle this issue, we suggested a data-based method to recreate historical hourly $PM_{2.5}$ levels in the Kathmandu Valley. We used meteorological reanalysis data from NASA's Modern-Era Retrospective Analysis for Research and Applications version 2 (MERRA-2) and also ground-based $PM_{2.5}$ measurements from the US Embassy air monitoring station. This allowed us to train a deep learning model that can estimate $PM_{2.5}$ levels over a broad time range. Similarly, this model is compared with machine learning algorithms like Extreme Gradient Boost (XGBOOST) to ensure accuracy. With parameters that include a continuous historical $PM_{2.5}$ time series from 2017 to 2021, the hope is to address a substantial void in the air quality records of the Kathmandu Valley. This will provide a dataset for understanding long term pollution trends and seasonal variation, enabling and supporting management of the environment, policy making, as well as public health studies in the Kathmandu Valley.

2. Related Work

The predominant focus of many modeling activities around the world has been air pollution, particularly air pollution caused by fine particulate matter ($PM_{2.5}$). In an effort to improve forecasting capabilities and fill gaps in historical information, researchers applied disparate statistical and Machine Learning approaches to the problem.(Ayus, 2023) in their article, name "*Comparison of machine learning and deep learning techniques for prediction of air pollution: a case study from China*", provide a comparison of predictive methods that use deep learning frameworks as opposed to traditional machine learning framework models for air pollution prediction. With extremely large datasets Ayus et al., showed that deep learning frameworks perform better than traditional machine learning models in predicting $PM_{2.5}$ concentrations due in part to being able to capture non-linear temporal relationships in $PM_{2.5}$ concentrations better than traditional machine learning models. In Nepal, (Khanal, 2023) reported a comparative study in Nepal entitled "*Comparative Analysis of Time Series Forecasting Models for predicting $PM_{2.5}$ level is Kathmandu*", comparing SARIMA, Prophet and XGBoost. SARIMA (Seasonal Autoregressive Integrated Moving Average) is a traditional forecasting method for time-series data that is valid for data with clear seasonal data; however, it may struggle with non-linear relationships and multiple, complex variable interactions. Prophet was developed by Facebook to model time-series data, again with seasonal patterns and holidays; Prophet still suffers from the same limitation as SARIMA by not capturing complex non-linearities that are often prevalent in environmental data, such as predicting PM2.5 concentrations. XGBoost is a gradient boosting method, designed to allow for these complex, non-linear relationships and external features. XGBoost has the potential to be a stronger and more resilient prediction method for PM2.5 concentrations under conditions that change. The authors concluded that XGBoost provided better forecast accuracy compared to traditional statistical models, however, its predictive capabilities were limited to short-term forecasting and did not attempt to reconstruct historical trends as limited by the available ground data. In addition, (Bhatta, 2023) published "*Reconstructing PM2.5 Data Record for the Kathmandu Valley Using a Machine Learning Model*", which described a machine learning model developed to estimate the hourly $PM_{2.5}$ values that occurred in the past using MERRA-2 meteorological reanalysis data as input. Their investigation filled gaps in the observational record, particularly before the introduction of air quality monitoring stations in Kathmandu, but the model was limited to traditional machine learning methods which may struggle with the complexity of the temporal signal over several decades. This work moves on from those foundations, and conducted a comparative analysis of DNN and XGBoost on both daily and hourly datasets, something that has not been undertaken as a comparative analysis in the case of Kathmandu. In using deep learning (DNN) as opposed to traditional machine learning approaches, we are able to capture the complex non-linear relationships present in atmospheric data more effectively.

3. Related Theory

The basis of our study is found in non-linear regression, ensemble learning, and deep feedforward neural networks built for temporal environmental data. The AdamW optimizer is an efficient gradient descent method that not only uses adaptive learning rates, but also uses decoupled weight decay to improve generalization. Instead, XGBoost is a state-of-the-art ensemble method (Chen, 2016) that uses gradient-boosting decision trees. XGBoost is a more effective ensemble method for tabular data with a limited number of rows, and with interpretable contributions from the features. For most real practices, DNNs work excellently with massive data sets that have massive features while, structures of data of few records are useful for XGBoost. Let us represent the input dataset with a matrix:

$$X \in R^{n \times d} \quad \text{(Equation 1)}$$

Where n is the number of samples (time steps), and d is the number of features per timestep (e.g., weather parameters, lags, encoded time features). The target output vector is:

$$y \in R^{n \times x} \quad \text{(Equation 2)}$$

The deep learning model learns a function $f: R^d \rightarrow R$ such that:

$$\hat{y} = f(x; \theta) \quad \text{(Equation 3)}$$

Where ŷ is the predicted $PM_{2.5}$ value, and θ denotes the set of model parameters. We used a feedforward Deep Neural Network (DNN) (LeCun, 2015) with layers structured as:

$$f(x) = W_L\, \sigma_{L-1}(W_{L-1}\, \sigma_{L-2}(\cdots \sigma_1(W_1 x + b_1) \cdots) + b_{L-1}) + b_L \quad \text{(Equation 4)}$$

Here, W_i and b_i are the weight matrices and biases for the i-th layer, and σ_i represents the activation function, typically the ReLU (Rectified Linear Unit):

$$\sigma(x) = (0, x) \quad \text{(Equation 5)}$$

We used various techniques to improve training stability and promote generalization. Batch normalization was employed to normalize inputs (Ioffe, 2015) . We also applied regularization through decoupled weight decay, which enabled faster converge.

$$\widehat{y_i} = \sum_{k=1}^{K} f_k(x_i)\,,\ f_k \in F \quad \text{(Equation 6)}$$

Where each f_k is a regression tree, and F is the space of all possible trees. The model is trained to minimize a regularized objective:

Here each f_k is a regression tree, and F is the space of all possible trees. The model is trained to minimize a regularized objective:

$$L(\theta) = \sum_{i=1}^{n} l(y_i, \widehat{y_i}) + \sum_{k=1}^{K} \Omega(f_k) \quad \text{(Equation 7)}$$

Where l is the loss function and Ω penalizes model complexity to avoid overfitting.

Which mathematical approach best models the temporal dynamics of air pollution?

Deep learning models, especially deep neural networks (DNNs) frequently outperform traditional, non-deep-learning methods such as XGBoost (Ayus, 2023) for $PM_{2.5}$ reconstruction, particularly when trained on a higher-resolution dataset with a large amount of detail such as real ground station measurements. The deep structure helps DNNs capture complex nonlinear relations and temporal dependencies inherent in localized environmental datasets, leading to improved $PM_{2.5}$ estimates.

However, as is often the case with deep learning, the performance benefit tends to fade when working with summarized data instead of fine-grained data, such as the NASA MERRA-2 reanalysis data. When working with coarser data, XGBoost (or other traditional methods) may also be able to perform similarly or, in some cases, better than deep learning methods, because they do not have the capacity to learn and fit the fine-grained data as effectively as they would in a finer data condition.

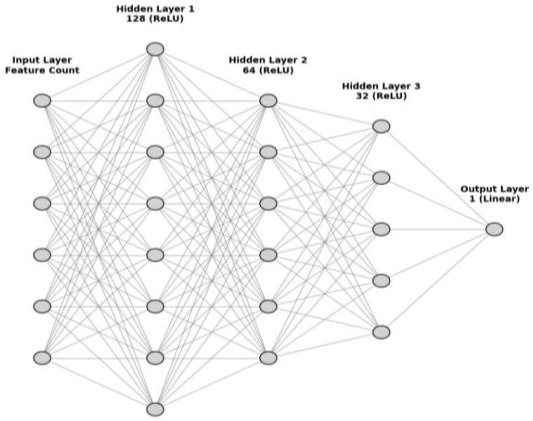

Figure 1. DNN Architecture

The image shows the architecture of the deep neural network (DNN) employed in this study, including an input layer, a linearly activated output layer, and three hidden layers, or "blocks," consisting of 128, 64, and 32 nodes, respectively.

4. Dataset Description

We used two important datasets in the study as inputs to our deep neural network to reconstruct $PM_{2.5}$ concentrations in the Kathmandu Valley. The first dataset was acquired from Open-Mateo. We used hourly data from 2017-2021. It encompassed temperature at 2 m above ground level, relative humidity, dew point temperature, apparent temperature, surface pressure, precipitation, cloud cover, shortwave radiation, wind speed and direction at 10 m above ground level, wind gusts, vapor pressure deficit, duration of sunshine, and categorical weather codes. (OpenMeteo, 2023)

The second dataset was NASA's MERRA-2 reanalysis data, from 1980-2025, which was originally in hourly format. It contained varied atmospheric variables including cloud pressure and temperature, atmospheric thickness at pressure levels, vertical velocity, planetary boundary layer height, surface pressure, humidity and specific humidity at different levels, sea level pressure, temperature profiles of different altitudes, concentrations of ozone and toxic gases, total column water vapor, tropopause height, surface temperature, wind components at different heights, and lifting condensation level height. (Buchard, 2017)

We used the readings of $PM_{2.5}$ concentrations from the US Embassy and US Diplomatic Post air quality monitoring stations in Kathmandu from 2017-2021, for the ground truth $PM_{2.5}$ concentrations. The averaged $PM_{2.5}$ values from both stations represented the ground truth values that were consistent and reliable data for training and testing the model allowing the DNN to reconstruct better. (OpenDataNepal, 2025)

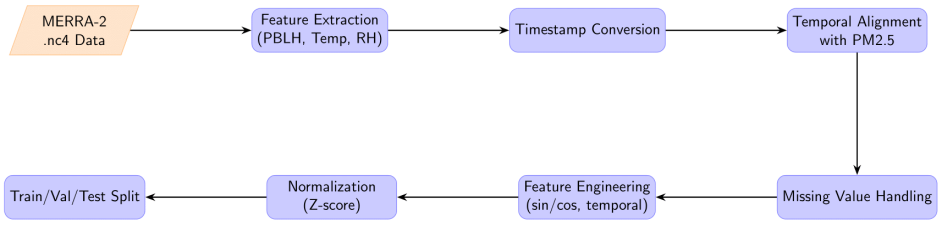

Figure 2. Data Preprocessing Pipeline

This figure shows the process of preparing the data to be used in the deep neural network model, including feature extraction, converting to timestamps, temporal alignment with PM2.5 data, missing value imputation, and normalization, which prepares the data for model training. Preprocessing included linear interpolations of missing observations, chronological sorting, and lag features for the hourly dataset (1-3 hours) as well as the daily dataset (1-3 days) where it was a concern of the temporal nature of the data for air quality forecasting. Cyclical feature encoding using sine and cosine transformations were only applied to the hours and months feature of the hourly dataset, as the cyclical nature of these features must be preserved in the forecasting. Both datasets were standardized with Z-score normalization using StandardAero, and the pre-processed data was saved to NumPy arrays for efficient batch training of both DNN and XGBoost models. As a result, the

datasets were clean, temporally informative, and scaled correctly for $PM_{2.5}$ reconstruction in Kathmandu Valley.

5. Methodology

The study engaged a clearly defined experimental pipeline overlaying a series of procedural steps for data preparation, feature engineering, model training, and model evaluation.

5.1 Data Preprocessing

Upon preliminary inspection of the datasets, the $PM_{2.5}$ and meteorological time series datasets displayed numerous missing data phenomena. If not addressed, missing data could produce biases or break temporal dependencies. For this study, missing data was addressed using a linear interpolation along the temporal axis. Linear interpolation was selected because it maintains continuity of short gaps and avoids artificial noise from spline or polynomial methods. It preserves temporal continuity, without adding abrupt artificial deviations. For the hourly dataset, small gaps (<3 hours) of missing data were interpolated reliably. In the case of the daily dataset, only complete days were considered legitimately interpretable, as we wanted to preserve consistency in the analysis of trends.

5.2 Data Scaling

Data Scaling to provide numerical stability and faster convergence numerical values for all features and target variables were standardized using Z-score normalization:

$$X' = \frac{X - \mu}{\sigma} \qquad \text{(Equation 8)}$$

where: X' is the normalized variable, X is the original input, μ is the mean, and σ is the standard deviation.

Z-score normalization was preferred over min-max scaling because it maintains outlier structure and is more stable for meteorological data. Each input feature (X) and target output (y) were standardized separately using StandardScaler from the scikit-learn library. Further, standardizers were saved in order to reuse them later in the inverse transformation of the predicted $PM_{2.5}$ values for performance evaluations.

5.3 Feature Engineering

To include temporal dependencies and atmospheric inertia, lag features were added as lagged values of $PM_{2.5}$ and meteorological parameters from the previous 1, 2, and 3 hours (in the case of the hourly dataset) or days (in the case of the daily dataset) for each timestamp. This allowed the models to learn from previous pollutant and weather conditions, which are highly predictive of the current $PM_{2.5}$ concentrations.

Furthermore, to include periodic trends, cyclical features were derived using sine and cosine transformations of time variables. For the hourly dataset, the hour of day and month of year were transformed as follows:

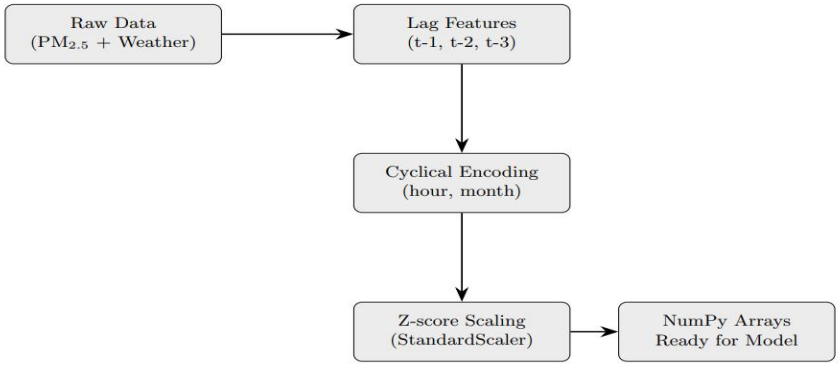

Figure 3. Feature Engineering Pipeline

This visual display illustrates the method of preparing the dataset for modeling, which involved the generation of lagged data, cyclical encoding of time features (hour and month), Z-score normalization, and also restructuring the dataset as NumPy arrays for modeling.

$$hour_{sin} = sin\left(\frac{2\pi \cdot hour}{24}\right),\ hour_{cos} = cos\left(\frac{2\pi \cdot hour}{24}\right) \quad \text{(Equation 9)}$$

$$month_{sin} = sin\left(\frac{2\pi \cdot month}{12}\right),\ month_{cos} = cos\left(\frac{2\pi \cdot month}{12}\right) \quad \text{(Equation 10)}$$

Equations 9 and 10 represent the cyclical encoding of time variables (hour and month) using sine and cosine transformations. This method preserves the periodic nature of time, ensuring that values such as 23:00 and 00:00 (or December and January) are close in feature space, something ordinary numeric encoding fails to capture. These transformations assisted in the neural network model learning the cyclical characteristics of the atmospheric conditions over time.

5.4 Model Construction and Training

Models were produced in two formats, Deep Neural Networks (DNNs) and XGBoost regressors. The DNNs were implemented using TensorFlow's Keras API (Abadi, 2016). All layers in the architecture were dense layers with ReLU activations with batch normalization, and dropout was used to avoid over-fitting. The AdamW optimizer was used as it embraces adaptive learning rates, and has incorporated weight decay (as opposed to L_2 penalties) to improve generalization.

Callbacks included EarlyStopping (patience = 20) and ReduceLROnPlateau (patience = 10, factor = 0.5) to assist training maximum learning while minimizing overfitting. XGBoost models were tuned using grid search over visualization hparams namely max_depth, learning_rate, and n_estimators. Training-validation-testing splits were performed chronologically in keeping with time series's real-world forecasting patterns with 70-15-15% proportions.

- *Patience*: In the field of machine learning, patience is defined as the number of iterations (or epochs) to hold off on improving a model's performance. Practice is usually implemented in early stopping, which is a measure to help avoid overfitting the training process, by halting the training process if model performance on a validation set does not improve, after a specified number of epochs.
  If **patience = 10,** the model will continue training for 10 more epochs even if tgarhe validation performance does not improve. If no improvement is seen after 10 epochs, the training stops.
- *Hparams*: **hparams** is shorthand for **hyperparameters**, which are parameters that are set before training a machine learning model and are not learned from the data. Some of them include learning rate, batch size, number of layers or neurons.

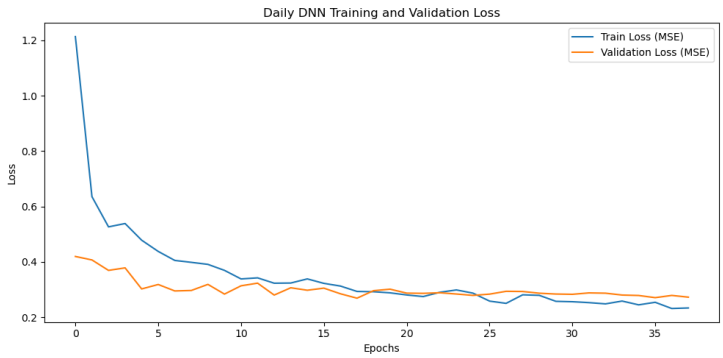

Figure 4. DNN Training and Validation Losses on Daily OpenMeteo Dataset

- *N_estimators:* indicates how many individual or "estimators" to use in an ensemble model, like Random Forest or Gradient Boosting. In an ensemble model, estimators are combined together to improve the predictive power of the model through a reduction in variance (Random Forest) or bias (Gradient Boosting).
- *Hyperparameter Optimization and Model Robustness:* In order to confirm that the performance of the model was not simply a matter of the arbitrary selection of parameters, we tuned both the DNN and the hyperparameters of the XGBoost model during grid search and validation-based optimization. The DNN was evaluated for different hidden layer depths (2-5 layers) and number of neurons,1 (32-256) and 2 a variety of batch sizes (32-128) and learning rates (10^{-3}-10^{-5}). The final model's architecture (128-64-32) was chosen based on validation R^2 and no signs of overfitting. Likewise, for XGBoost, the parameters max_depth, n_estimators, learning_rate, and subsample

were tuned to best balance both bias and variance. The systematic tuning of hyperparameters supports that the results reported are reliable.

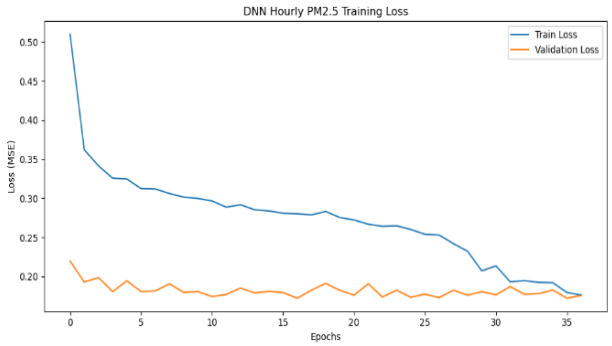

Figure 5. DNN Training and Validation Losses on Hourly OpenMeteo Dataset

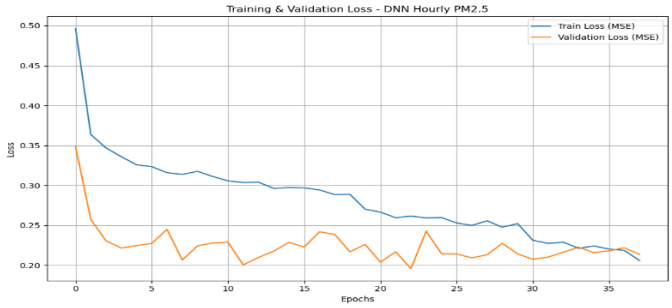

Figure 6. DNN Training and Validation Losses on Hourly NASA MERRA-2

This figures above show the training and validation losses (Mean Squared Error, MSE) of the **DNN model** over the course of training. The training loss decreases rapidly, indicating that the model is learning, while the validation loss decreases more gradually and stabilizes, showing the model's generalization capability.

5.5 Model Evaluation

The models were evaluated using the following metrics on the unseen test set:

- *R^2 Score*: $R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$
- *Root Mean Square Error (RMSE)*: $\text{RMSE} = \sqrt{\frac{1}{n}\sum(y_i - \hat{y}_i)^2}$
- *Mean Absolute Error (MAE)* :$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$
- *Mean Difference*:$\Delta = \frac{1}{n}\sum(\hat{y}_i - y_i)$

These metrics were calculated after inverse transforming the predictions back to $PM_{2.5}$ (µg/m³) in original units. We report performance for each model-dataset combination (DNN-hourly, XGBoost-hourly, DNN-daily, XGBoost-daily).

5.6 Sensitivity and Comparative Analysis

To assess how robust the model is, we performed experiments again varying the input feature settings. When removing lag features, R^2 was lower by about 0.04 average across all models, demonstrating a contribution to temporal awareness. Training the DNN without cyclical encoding of the hour and month features also decreased predictive stability, especially with hourly data. These tests confirm the importance of the engineered temporal features and bolster our claims in Section 6.

6. Results & Discussions

Table 1. Comparative Evaluation of DNN vs XGBoost on Hourly and Daily Datasets (Open-Meteo Vs Satellite Data NASA-MERRA2).

Metric	DNN Hourly (NASA Data)	DNN Hourly (Open-Meteo)	XGBoost Hourly (Open-Meteo)	DNN Daily (Open-Meteo)	XGBOOST Daily (Open-Meteo)
R^2	0.7673	0.8725	0.8014	0.7641	0.7827
RMSE (µg/m³)	24.6358	18.2348	22.7577	13.3492	12.8126
MAE (µg/m³)	11.5239	9.1317	8.8460	9.7499	8.9456
Mean Difference (µg/m³)	-5.2387	-3.4006	-0.1550	-1.6883	4.1579

The performance metrics (R^2, RMSE, MAE, Mean Difference) on hourly and daily datasets for DNN and XGBoost with OpenMeteo and NASA MERRA-2 data can be seen in the table above. In general, XGBoost model outperformed DNN models on all metrics, with distinct effectiveness at capturing variability in daily forecast. Deep learning approaches can achieve success over traditional machine learning methods like XGBoost, but only when high resolution, ground-based features can supply high-quality forecasts. When choosing a prediction model, it seems that the data input quality has more bear re-leaved on the algorithm.

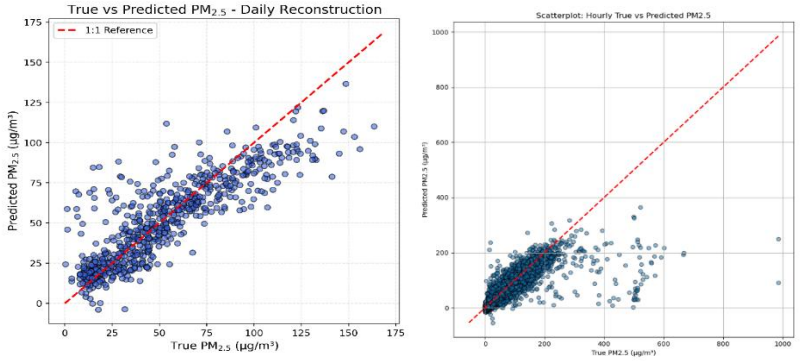

Figure 7. Scatter-Plot of True Vs Predicted Daily Open-Meteo (Left) Vs Hourly NASA MERRA-2(Right) $PM_{2.5}$

Figure 7 illustrates the scatterplot comparison between the **true PM2.5 values** and the **predicted PM2.5 values** for both **daily** and **hourly** data of different resolution. The OpenMeteo dataset, which has high-resolution, surface measurements, performs very well for daily PM2.5 forecasting, as demonstrated in the left scatterplot where the predicted values and true values are closely matched. This strong association demonstrates the model has the ability to capture longer-term trends and seasonal variations in air quality data. Ground data will be more effective for daily forecasting. On the other hand, while the NASA MERRA-2 is beneficial for environmental modeling at larger scales, the hourly PM2.5 prediction results in a more scattered points (right plot), reflecting greater prediction error. The MERRA-2 is satellite-based, coarser resolution data, which cannot reflect the finer grain fluctuations synonymous with hourly data, which is less suitable for tasks with high frequency forecasting. This shows, with regards to forecasting PM2.5 for high frequency and high-resolution time scales, that high resolution ground data, like OpenMeteo, is shown to be more accurate than satellite data, such as NASA MERRA-2, which are better suited for larger scales, as well as longer term study.

Additionally, our results highlight the DNN models' sensitivity to noise and potential overfitting, especially with smaller datasets. Nevertheless, both architectures demonstrated strong predictive capabilities. Preliminary tests on the Open-Meteo (hourly and daily) and NASA MERRA-2 (hourly) datasets are done and are expected to produce promising long-term reconstructions.

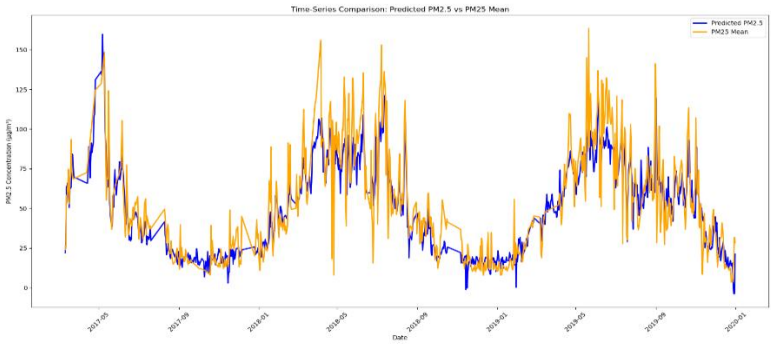

Figure 8. Time-Series Comparison: Predicted $PM_{2.5}$ with Actual $PM_{2.5}$ readings

The time-series plot provides a comparison of Predicted $PM_{2.5}$ (blue line) with $PM_{2.5}$ Mean (orange line) from the period 2017-03-11 to 2019-12-31. In general, the Predicted PM2.5 appears to closely track the actual readings obtained from the ground ($PM_{2.5}$ Mean), especially during episodes of high pollution associated with either seasonal fluctuations caused by wildland fire events or pollution related to urban activity. The peaks in the lines are aligned throughout the plot indicating that gross trends and larger variation in $PM_{2.5}$ concentration is being modeled appropriately. However, smaller instances appear throughout the plot, where the predicted $PM_{2.5}$ does deviate from actual predictions indicating some level of model bias and/or temporal mismatch. These instances may provide evidence that the model is not capturing smaller fluctuations and/or minor trends in the data either due to poor model specifications to adjust for certain shocks every two years such as seasonal or cyclical variation.

In comparison to existing systems and some previously released methods, our framework shows obvious advantages. (Khanal, 2023) employed SARIMA, Prophet, and XGBoost to forecast $PM_{2.5}$ over short-term periods. However, their reliance on temporal resolution (a sequence of measurements) and linear assumptions resulted in R^2 values ranging from below 0.75. Additionally, Bhatta (2023) identified $PM_{2.5}$ data from MERRA-2 reanalysis through machine learning methods that showed reasonable estimation, but reduced performance when it came to estimating short-term variability. Our DNN model trained on high resolution Open-Meteo data produced an R^2 of 0.87 on hourly data; moreover, we showed the model's capability to detect finer temporal patterns. Our results, even under coarser NASA MERRA-2 input, proved competitive with traditional ML models. All of these results confirm that using ground-based meteorological data and a deep-learning architecture yield a substantially higher level of precision compared to existing reconstruction or forecasting methodologies, especially for data-poor areas like the Kathmandu Valley.

7. Conclusion & Future Work

Overall, both the DNN and XGBoost models demonstrated decent predictive performance for each dataset, but their strengths differ by data resolution and temporal scale. The DNN performed slightly better based on the hourly Open-Meteo dataset ($R^2 = 0.8725$, RMSE = 18.23 µg/m³) than XGBoost ($R^2 = 0.8014$, RMSE = 22.76 µg/m³), suggesting that the DNN was able to pick up on short-term variability better than XGBoost when there were dense, ground-based inputs available. In contrast, the XGBoost outperformed the DNN when using the daily Open-Meteo data ($R^2 = 0.7827$, RMSE = 12.81 µg/m³), sustaining more model stability and generalizability across the longer time period, even with minor, positive mean bias (+4.16 µg/m³). The results using MERRA-2 satellite data were consistently weaker relative to the ground-based input datasets, which again stressed the importance of high estimates of surface-based features for skillful $PM_{2.5}$ forecasting. To sum up, model performance cannot solely rely on algorithm alternative choice. Instead, the performance relies on a combination of levels of performance precision of input variables, as well as input frequency and some physical representativeness in turns affected by the resolution. It appears XGBoost would work best for daily operation reporting and forecasting, while DNN's should set-up high frequency, near real-time monitoring. Future work will focus on expanding the model's application to longer multi-year datasets and larger spatial domains to assess its scalability and temporal robustness across varying climatic conditions.

The research work can be found here: https://github.com/Kshitiz726/spatiotemporal-pm25-estimation

References

Abadi, M. A. A. B. P. e. a., 2016. TensorFlow: Large-scale machine learning on heterogeneous systems. *arXiv preprint.*

Ayus, I. N. N. &. G. D., 2023. Comparison of machine learning and deep learning techniques for prediction of air pollution: A case study from China. *Asian Journal of Atmospheric Environment.*

Bhatta, S. a. Y. Y. Y., 2023. Reconstructing PM2.5 Data Record for the Kathmandu Valley Using a Machine Learning Model. *Atmosphere,* 14(7), p. 1073.

Buchard, V. R. C. A. d. S. A. M. e. a., 2017. The MERRA-2 Aerosol Reanalysis, 1980 Onward: Assessment of Aerosol Properties. *Journal of Climate,* 30(2), pp. 319-340.

Chen, T. &. G. C., 2016. A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

Choi, J. L. S. a. L. J., 2020. Deep learning-based PM2.5 forecasting model using LSTM and multiple meteorological variables. *Atmospheric Environment.*

ICIMOD, 2025. *Kathmandu choked on polluted air for 75 of the last 90 days..* [Online]
Available at: https://www.icimod.org/press-release/kathmandu-choked-on-polluted-air-for-75-of-the-last-90-days/
[Accessed 27 June 2025].

Ioffe, S. &. S. C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32nd International Conference on Machine Learning (ICML).*

Khanal, I. e. a., 2023. *Comparative Analysis of Time Series Forecasting Models for Predicting PM2.5 Level in Kathmandu.* s.l., Institute of Engineering.

LeCun, Y. B. Y. &. H. G., 2015. Deep learning. *Nature,* 521(7553), pp. 436-444.

Loshchilov, I. &. H. F., 2019. Decoupled weight decay regularization. *Proceedings of the International Conference on Learning Representations (ICLR).*

OpenDataNepal, 2025. *Air Quality in Kathmandu.* [Online]
Available at: https://opendatanepal.com/dataset/air-quality-data-in-kathmandu

OpenMeteo, 2023. *OpenMeteo: Free Weather Data for All..* [Online]
Available at: https://open-meteo.com/
[Accessed 2025].