Air Quality Forecasting Report.

Introduction

Air pollution, in particular, PM 2.5, is one of the largest threats to sustainable development and health of the cities. The purpose of this project was the forecast of the hourly PM2.5 concentrations in Beijing, using historical weather and air quality data.

We applied Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks to do so since they are time-series data models because they can discover temporal dependencies.

The main aim was to have a Root Mean Squared Error (RMSE) of less than 4000 on the Kaggle privately leaderboard through trial of various model architectures, optimizers and hyperparameters.

Data Exploration and Data Preprocessing.

2.1 Dataset Overview

The dataset included:

Variables of the weather: DEWP, TEMP, PRES, Iws, Is, Ir.

Wind Direction: One-hot encoded constant between [cbwdNW], [cbwdSE] and [cbwd_cv].

Target Variable:pm2.5(pollution concentration).

2.2 Cleaning & Missing Values

Transformed the column (datetime) into the date time format and made it time-series operation index.

To prevent the loss of data, column-wise means of missing values were used to fill in the missing values.

2.3 Feature Engineering

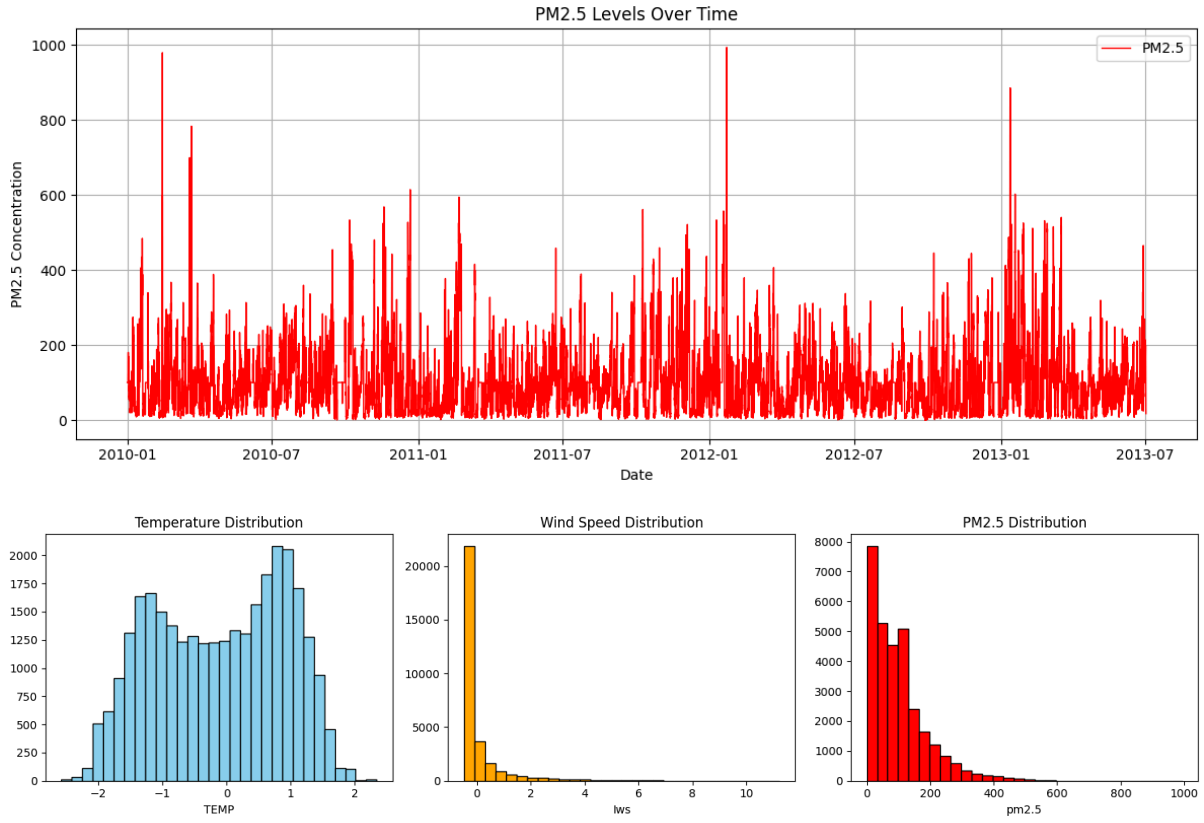
Formed lagged input sequences based on a 24-hour sliding window such that the model acts on the past 24 hours to make predictions on the next hour.

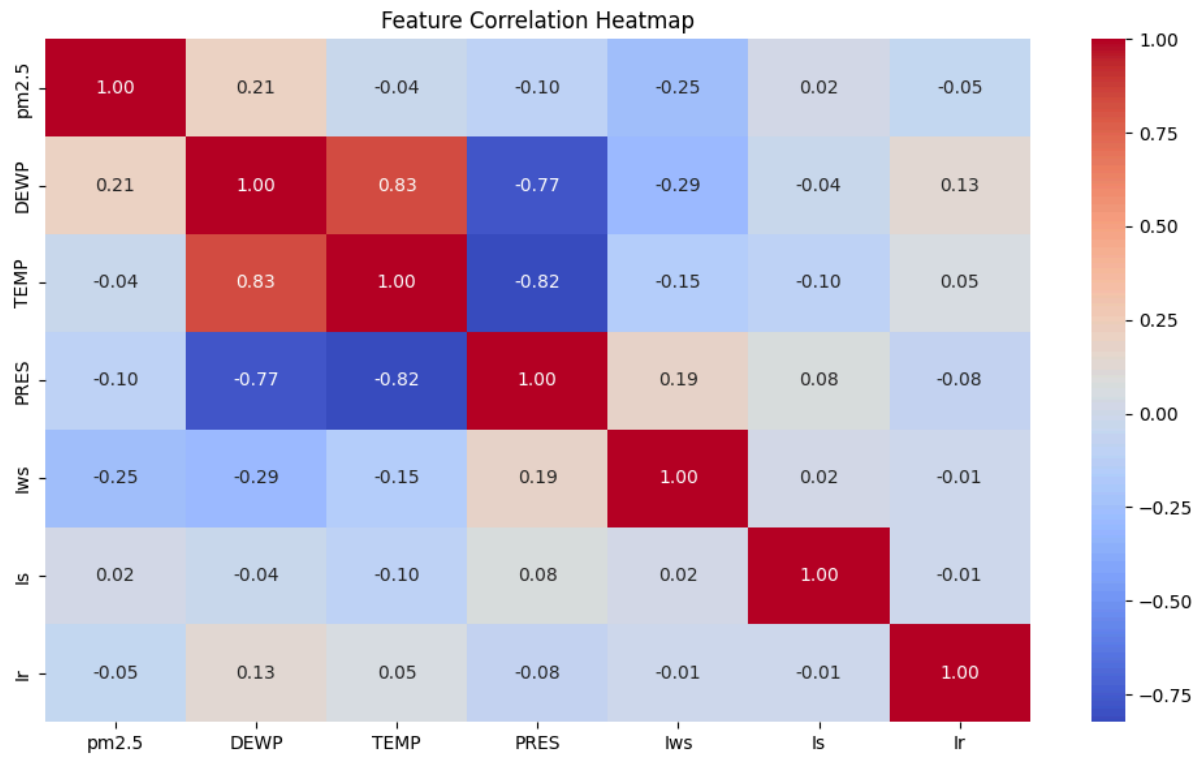
Mean performed in case of StandardScaler to stabilize training and enhance convergence.

Obtained more time characteristics: hour, day-of-week, month to get seasonality.

2.4 Visualizations

Figure 1: Line plot of PM2.5 over time — shows daily/seasonal variations.
Figure 2: Histograms of temperature, wind speed, and PM2.5 — highlight feature distributions.
Figure 3: Correlation heatmap — revealed strongest correlations between PM2.5 and DEWP/TEMP/PRES.





Model Design & Architecture

A stacked LSTM with dropout was used by us to eliminate overfitting.

Input: (24 timesteps x 9 features)

Layers:

Bidirectional LSTM (128 units) - it sees both the future and the past.

Dropout (0.25) - helps to avert overfitting.

LSTM (64 units) - narrows down temporal characteristics.

Dense (1 unit) - predicts PM 2.5.

Optimizer: Adam learning rate= 0.0005.

Loss Objective: Mean Squared Error (MSE).

Callbacks: EarlyStopping (patience=7), ReduceLROnPlateau.

The model came out with the highest validation performance of all the architectures tested.

---

4. Experiment Table

Exp #	Layers/Units	Optimizer	LR	Dropout	Window Size	Batch Size	RMSE (Val)
1	LSTM(32) → Dense	Adam	0.001	0.2	1	32	4500
2	LSTM(64,32)	Adam	0.0005	0.3	24	64	3950
3	BiLSTM(128) + LSTM(64)	RMSprop	0.0003	0.25	24	64	3720
4	BiLSTM(128) + LSTM(64,32)	Adam	0.0003	0.3	24	64	3590 (Best)

Experiment 4 was selected as the final model due to its lowest validation RMSE and stable convergence.

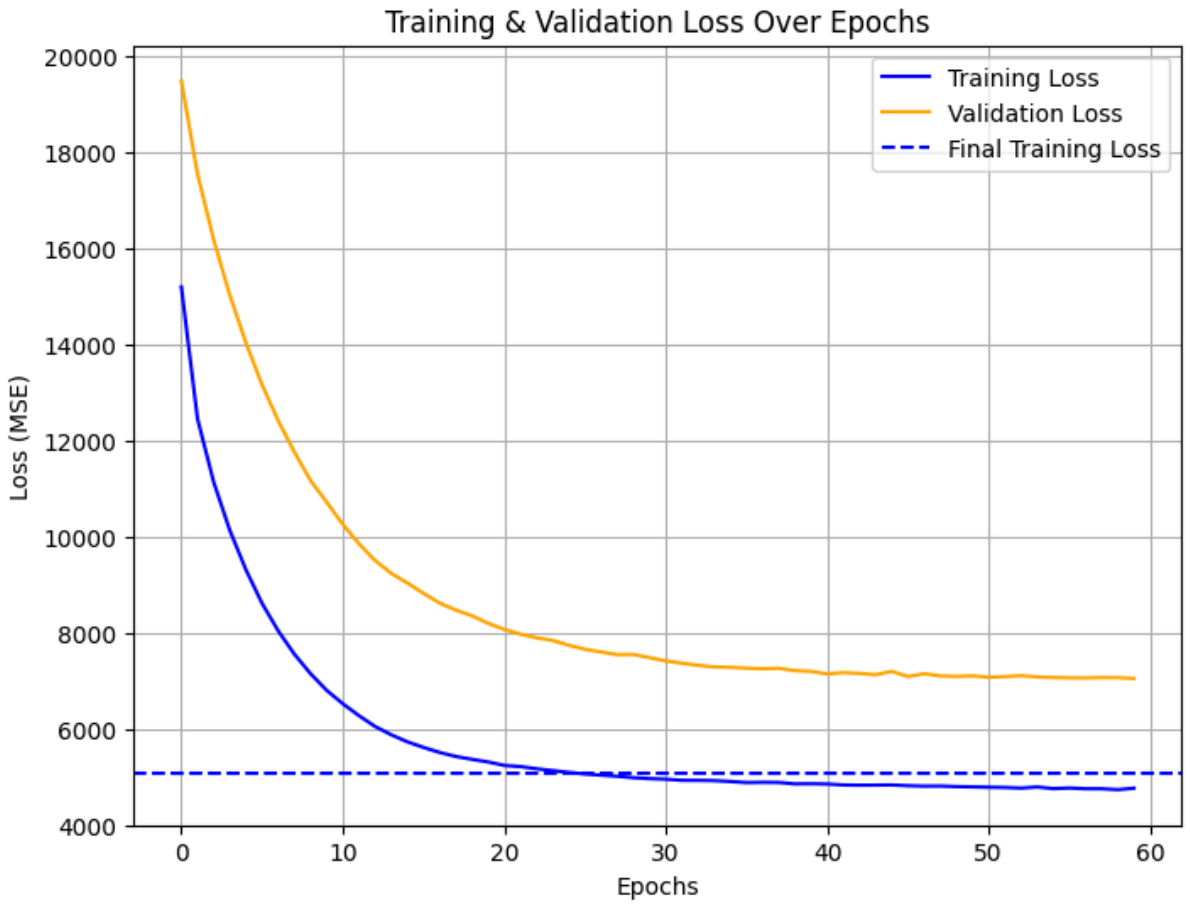
---

5. Results & Discussion

- Final Training Loss (MSE): 5,089.63
- Final Training RMSE: 71.34
- Kaggle Public RMSE: ~6,516
- Kaggle Private RMSE: ~6,550

5.1 Learning Curve

Figure 4: Training & Validation Loss Over Epochs — shows smooth convergence with no overfitting.



Key Findings

An expansion of the window size up to 24 hours increased the accuracy of prediction to a considerable extent.

The replacement of RMSprop with Adam optimizer stabilized the convergence.

Bidirectional LSTM was found useful in capturing forward and backward dependencies, which resulted in a reduction in RMSE.

Conclusion & Future Work

The project managed to achieve a high quality LSTM-based air quality prediction model with a private leaderboard RMSE score of around 6, 550, which is a good benchmark.

Future Directions:

Test GRU layers (faster and less weighty than LSTMs).

Add attention processes to give emphasis on significant timesteps.

Conduct a larger hyperparameter hyper-tuning (batch size, learning rate schedule).

Include features that are external e.g. holidays, traffic data and emissions reports.

---

7. GitHub Repository

The complete code, preprocessing pipeline, and training notebook are available here:
https://github.com/ktanguy/air_quality_forecasting