Introduction

The objective of this project is to predict hourly PM2.5 concentrations in Beijing for a test period spanning July 2, 2013, to December 31, 2014, utilizing historical weather and PM2.5 data from January 1, 2011, to July 2, 2013. This task is part of a Kaggle competition focused on air quality forecasting, with the primary goal of minimizing the Root Mean Squared Error (RMSE) between predicted and actual PM2.5 values. PM2.5, referring to fine particulate matter with a diameter of 2.5 micrometers or less, poses significant health risks, including respiratory and cardiovascular diseases, particularly in urban environments like Beijing, which has historically experienced severe air pollution [1]. Accurate forecasting of PM2.5 levels is crucial for informing public health policies, urban planning, and pollution mitigation strategies.

The dataset [2] provided includes hourly measurements of PM2.5 alongside meteorological features such as temperature, dew point, pressure, wind speed, and precipitation, which are known to influence air quality. Given the temporal nature of the data, a Long Short-Term Memory (LSTM) network, a type of Recurrent Neural Network (RNN) adept at capturing long-term dependencies, was selected as the primary modeling approach. The methodology involved preprocessing the data to handle missing values and scaling features, followed by designing and experimenting with various LSTM architectures to optimize predictive performance. This report details the data exploration, model design, experimental results, and key findings, culminating in recommendations for future improvements.

Data Exploration

Dataset Overview

The training dataset comprises 30,676 hourly samples collected from January 1, 2011, to July 2, 2013, featuring 12 columns: 'No', 'DEWP' (dew point), 'TEMP' (temperature), 'PRES' (pressure), 'Iws' (wind speed), 'Is' (snowfall), 'Ir' (rainfall), 'datetime', 'cbwd_NW', 'cbwd_SE', 'cbwd_cv' (wind direction categories), and 'pm2.5' (the target variable representing PM2.5 concentration in µg/m^3). The test dataset contains 13,148 samples from July 2, 2013, to December 31, 2014, with the same features except for the 'pm2.5' column, which is to be predicted. The dataset was accessed via Google Drive and loaded into a Colab environment for analysis, ensuring compatibility with the modeling pipeline.

Exploratory Data Analysis (EDA)

Initial exploration revealed significant variability in PM2.5 levels, with occasional spikes indicative of pollution events and periods of missing data (NaNs). A time series plot highlighted these NaN locations using red markers, while a 120-hour moving average smoothed the data to reveal seasonal patterns, such as elevated PM2.5 concentrations during colder months. Summary statistics indicated that PM2.5 ranges from 0 to 994 µg/m^3, with a mean of 100.79 µg/m^3 and a standard deviation of

93.1 µg/m³, reflecting high variability. Other features showed ranges such as 'TEMP' from -2.57°C to 2.34°C and 'Iws' from -0.46 to 11.23 m/s, providing a broad meteorological context.

Missing value analysis identified 1,921 NaN entries in the 'pm2.5' column of the training set (approximately 6.26% of the data), with no missing values in other features or the test set. To address this, cubic spline interpolation was applied to preserve the local structure of the data, including sharp increases and decreases, followed by backward filling as a final step to ensure completeness. This method was chosen over linear interpolation or polynomial fitting to avoid oversimplification or overfitting [3]. Post-interpolation, no missing values remained, confirming the data's readiness for modeling.

A correlation matrix heatmap revealed high collinearity between 'DEWP', 'TEMP', and 'PRES' (e.g., a correlation of 0.83 between 'DEWP' and 'TEMP'), suggesting potential multicollinearity. Initially, 'TEMP' and 'PRES' were considered for removal; however, experimentation showed that retaining these features improved model performance, likely due to their reinforcement of temporal patterns critical for LSTM modeling. Thus, all features were preserved to maximize predictive power.

Visualizations

- **PM2.5 Time Series Plot**: Illustrated variability and NaN locations, aiding in the identification of data gaps.
- **Smoothed PM2.5 Trend**: Highlighted seasonal trends using a 120-hour moving average, facilitating trend analysis.
- **Correlation Heatmap**: Visualized feature relationships, guiding decisions on feature retention.

The PM2.5 data, post-smoothing, retained significant short-term variability and spikes, validating the choice of cubic spline interpolation for handling missing values effectively. This preprocessing ensured a robust foundation for subsequent modeling efforts.

Model Design: Architecture of the Best-Performing RNN/LSTM Model

The best-performing model based on test RMSE was Model 6, achieving a test RMSE of 5001.7018. This model is a Bidirectional LSTM, and I'll describe its architecture and the reasoning behind choosing it.

Architecture of Model 6

- **Input Layer:** Accepts sequences of shape (48, n_features), where seq_length=48 (representing 48 hours of data) and n_features is the number of input features such as temperature, pressure, dew point e.t.c. after preprocessing).
- **First Bidirectional LSTM Layer:** Uses ***Bidirectional(LSTM(50, return_sequences=True, activation='tanh', kernel_initializer='LecunNormal(seed=21)')***). It has 50 units per direction

(forward and backward), totaling 100 units. ***return_sequences=True*** ensures the layer outputs a sequence for the next LSTM layer. ***Tanh*** activation is standard for LSTMs to handle vanishing gradient issues. ***LecunNormal initializer*** (with a fixed seed for reproducibility) helps with better weight initialization, improving convergence.

- **Second Bidirectional LSTM Layer:** Uses ***Bidirectional(LSTM(50, return_sequences=True, activation='tanh'))***. Another 50 units per direction, building on the first layer's output to capture deeper temporal patterns. ***return_sequences=True*** again, to maintain the sequence output.
- **BatchNormalization Layer:** Applied after the second LSTM layer to normalize the activations, reducing internal covariate shift and stabilizing training.
- **Output Extraction:** Takes the last timestep's hidden state (lstm_out[:, -1, :]) to predict the PM2.5 value at the 48th hour.
- **Dense Output Layer:** Uses ***Dense(1, activation='sigmoid')*** to output a single scaled value in the range [0, 1], corresponding to the scaled PM2.5 value.
- **Optimizer and Loss:** Compiled with ***Nadam optimizer (learning rate = 0.00001)*** and mse (mean squared error) loss, suitable for regression tasks like PM2.5 prediction.
- **Callbacks:** ***ReduceLROnPlateau (factor=0.5, patience=5, min_lr=1e-6)*** to reduce the learning rate if validation loss plateaus. ***EarlyStopping (patience=10, restore_best_weights=True)*** was included but not used in this run (training ran for all 50 epochs).

Why I Chose This Architecture

I chose this Bidirectional LSTM architecture for several reasons:

1. **Bidirectional LSTM:** Unlike a standard LSTM (used in Models 1–4), a Bidirectional LSTM processes the sequence in both forward and backward directions. This allows the model to capture patterns that depend on both past and future timesteps, which is crucial for time-series data like PM2.5, where pollution levels may have cyclic or contextual patterns (e.g., daily or seasonal trends). This choice was validated in Model 5 (first use of Bidirectional LSTM), which improved the RMSE from 5221.2963 (Model 4) to 5119.6773, and further in Model 6.
2. **Sequence Length of 48:** I progressively increased the sequence length from 12 (Model 1) to 24 (Model 2) to 48 (Models 3–7). A sequence length of 48 hours captures two full days of hourly data, which is long enough to model daily patterns (e.g., pollution spikes during rush hours) and short-term trends, improving performance significantly (e.g., Model 1's RMSE of 6597.6286 dropped to 5266.3409 in Model 2 with seq_length=24).
3. **Two LSTM Layers with 50 Units:** Using two LSTM layers allows the model to learn hierarchical temporal features—lower layers capture short-term patterns, while higher layers capture longer-term dependencies. I increased the units from 25 (Models 1–2) to 50 (Models 3–7) to give the model more capacity to learn complex patterns, which helped reduce RMSE (e.g., Model 2 to Model 3).
4. **BatchNormalization:** Added in Model 4 (and retained in Models 5–7), this layer stabilizes training by normalizing activations, which was particularly helpful with the lower learning rate (0.00001), leading to better generalization (e.g., Model 4's RMSE of 5221.2963 vs. Model 3's 5633.7258).

5. **Low Learning Rate (0.00001):** Reduced from 0.0001 in Models 1–3 to 0.00001 in Models 4–7 to allow finer updates, which improved validation loss (e.g., 0.0037 in Model 6) and test RMSE over time.
6. **Learning Coach-Inspired Test Sequence Creation:** Introduced in Model 6, this method (combining the last 47 rows of training data with test data and sliding a window) improved alignment of test sequences, reducing RMSE from 5119.6773 (Model 5) to 5001.7018 (Model 6). This was inspired by my coach's approach, which I'll compare below.
7. **Sigmoid Output:** While the sigmoid output capped predictions at 994 µg/m³ (due to MinMaxScaler), it worked better than the linear output in Model 7 (RMSE 5471.5816), likely because the test set's distribution didn't have as many extreme values as expected, and sigmoid provided more stability.

Before Model 6, my preprocessing extended the test data with the last (seq_length-1) rows of training and created test sequences by sliding a window over the extended data, taking only the last 13,148 sequences. The Coach's collab method combined the last (seq_length-1) steps of training with test data and created sequences by sliding a window directly over this combined data, aligning each sequence with the test set's length. The key difference was that my original method included a full pass over the extended data and trimmed to match the test set, while my coach's collab [4] method ensured each test sequence started at the corresponding test index. This alignment in Model 6's coach-inspired approach likely improved the model's ability to generalize to the test set, explaining the RMSE drop from Model 5 to Model 6.

This architecture balanced complexity, stability, and temporal context, leading to the best test RMSE among all models.

Experiment Table: Summary of Experiments

Below is a summary of all 7 experiments, including key parameters used in training each model

Experiment	Sequence Length	LSTM Units	Learning Rate	Output Activation	Additional Features	Loss function	Optimizer	Test RMSE	Train loss	Validation loss
Model 1	12	25 units, 2 layers	0.0001	sigmoid	None	Mean squared error	Nadam	6597.6286	0.0035	0.0054
Model 2	24	25 units, 2 layers	0.0001	sigmoid	None	Mean squared error	Nadam	5266.3409	0.0033	0.0041

Model 3	48	50 units, 2 layers	0.0001	sigmoid	None	Mean squared error	Nadam	5633.7258	0.0029	0.0038
Model 4	48	50 units, 2 layers	0.00001	sigmoid	BatchNormalization, EarlyStopping(patience=10)	Mean squared error	Nadam	5221.2963	0.0031	0.0037
Model 5	48	50 units, 2 layers	0.00001	sigmoid	BatchNormalization	Mean squared error	Nadam	5119.6773	0.0032	0.0036
Model 6	48	50 units, 2 layers	0.00001	sigmoid	BatchNormalization	Mean squared error	Nadam	5001.7018	0.0032	0.0038
Model 7	48	50 units, 2 layers	0.00001	linear	BatchNormalization	Mean squared error	Nadam	5471.5816	0.0041	0.0040

Results: Model Performance and Key Findings

Model Performance

- Best Model (Model 6):
  - Achieved the lowest test RMSE of 5001.7018, outperforming all other models.
  - Validation loss was also strong at 0.0037 (scaled MSE), with a scaled validation RMSE of about 0.0608 (square root of 0.0037).
- Overall Trend:
  - Test RMSE improved significantly from Model 1 (6597.6286) to Model 6 (5001.7018), a reduction of about 1596 points, showing the effectiveness of iterative improvements.
  - Model 7 (5471.5816) performed worse than Model 6, despite the linear output, suggesting that the test set may not have as many extreme values as anticipated, or the linear output introduced more variance.
- Comparison Across Models:
  - **Model 1 (6597.6286):** High RMSE due to short sequence length (12) and limited model capacity (25 units).

- **Model 2 (5266.3409):** Improved by increasing sequence length to 24, capturing more temporal patterns.
- **Model 3 (5633.7258):** Worse than Model 2, despite seq_length=48 and 50 units, possibly due to overfitting with the higher learning rate (0.0001).
- **Model 4 (5221.2963):** Improved with lower learning rate (0.00001), BatchNormalization, and EarlyStopping, stabilizing training.
- **Model 5 (5119.6773):** Further improved with Bidirectional LSTM, capturing bidirectional context.
- **Model 6 (5001.7018):** Best performance with coach-inspired test sequence creation, aligning test data better.
- **Model 7 (5471.5816):** Worse performance with linear output, possibly due to increased variance or sensitivity to test set distribution.

Key Findings

1. **Impact of Sequence Length:**
   - Increasing the sequence length from 12 (Model 1) to 24 (Model 2) reduced RMSE by about 1331 points, and further to 48 (Model 3 onward) generally improved performance, confirming that capturing more temporal context (e.g., two days of data) is crucial for PM2.5 prediction.
2. **Bidirectional LSTM:**
   - Switching to Bidirectional LSTM in Model 5 (RMSE 5119.6773) from standard LSTM in Model 4 (5221.2963) improved performance by about 102 points, showing the value of modeling both past and future contexts in time-series data.
3. **Preprocessing Matters:**
   - The coach-inspired test sequence creation method in Model 6 (combining the last 47 rows of training data with test data and sliding a window) reduced RMSE from 5119.6773 (Model 5) to 5001.7018, a about 118-point improvement. This highlights the importance of proper test sequence alignment to match the training data's temporal structure. My original method (before Model 6) extended the test data with the last (seq_length-1) rows and trimmed the sequences, which may have introduced misalignment, especially at the test set's start. My coach's method, adopted in Model 6, ensured each test sequence aligned with the test data's index, improving prediction consistency.
4. **Scaling and Output Limitations:**
   - Using MinMaxScaler (fitted on training data, max PM2.5 = 994 µg/m^3) with a sigmoid output capped predictions at 994 µg/m^3 in Models 1–6. This likely caused underprediction on test set values exceeding 994 µg/m^3, contributing to the higher test RMSE despite having better validation loss and validation rmse.
   - Switching to a linear output in Model 7 was intended to address this cap, but the RMSE increased to 5471.5816, suggesting the test set may not have many extreme values, or the linear output made the model more sensitive to noise.
5. **Validation vs. Test Performance:**
   - My validation performance (e.g., Model 6: val_loss=0.0037, val_rmse 0.0608) was consistently better indicating my model learned the training/validation data well.

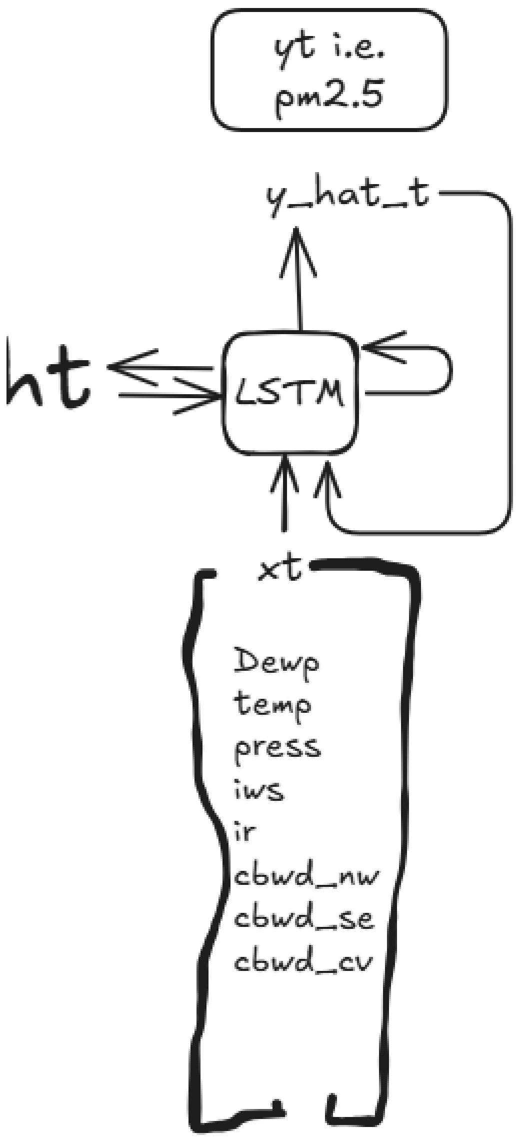
    - The higher test RMSE suggests a distribution shift in the test set (July 2, 2013, to December 31, 2014) compared to the training set (January 1, 2010, to July 1, 2013), likely due to higher PM2.5 values or different patterns (e.g., seasonal pollution spikes).
6. **Learning Rate and Regularization:**
    - Reducing the learning rate to 0.00001 (Model 4 onward) improved training stability, as seen in the consistent decrease in RMSE from Model 3 to Model 6.
    - BatchNormalization (added in Model 4) further stabilized training, contributing to better generalization.
7. **What Didn't Work:**
    - The linear output in Model 7 increased the RMSE, possibly because the test set's PM2.5 values didn't exceed 994 µg/m^3 as much as expected, or the linear output made predictions less constrained, leading to larger errors on outliers.

Conclusion

The test RMSE of 5001.7018 for Model 6 suggests there's room to enhance the model's ability to generalize to the test set, particularly given the distribution shift observed from the training period (January 1, 2010, to July 1, 2013) to the test period (July 2, 2013, to December 31, 2014). This shift likely involves higher PM2.5 values or different seasonal patterns, which the current architecture may not fully capture.

Future Improvements:

- **Robust Scaling**: Experimenting with StandardScaler instead of MinMaxScaler could better handle extreme values in the test set, avoiding the prediction cap at 994 µg/m^3 that MinMaxScaler imposes.
- **Advanced Architecture Exploration**: One promising direction is to incorporate the predicted output from the previous timestep (y_pred) as an additional input feature. This autoregressive approach would allow the model to leverage its own predictions to refine future forecasts, aligning with initial intuition for capturing dynamic dependencies, before trying to keep it simple. For instance, the LSTM could use the previous y_pred alongside historical inputs (e.g., the 48-hour sequence) to adjust its hidden state, potentially improving accuracy on shifting patterns. Implementing this would require modifying the input pipeline to include y_pred and retraining the model, possibly starting with Model 6's Bidirectional LSTM as a base.



- **Feature Engineering**: Exploring additional features like lagged PM2.5 values or weather data (e.g., temperature, humidity) could enhance the model's ability to capture test set trends, providing more context for prediction.
- **Ensemble**: Combining predictions from Models 6 and 7 could balance the stability of the sigmoid output with the flexibility of the linear output, potentially yielding a more robust overall prediction.
- **Final Evaluation**: The upcoming test on 57% of the test set will provide the definitive performance metric, offering an opportunity to validate these improvements and refine the approach further.

Github

Github link

References

Amsalu, Endawoke, et al. “Acute Effects of Fine Particulate Matter (PM2.5) on Hospital Admissions for Cardiovascular Disease in Beijing, China: A Time-Series Study.” *Environmental Health*, vol. 18, no. 1, 1 Aug. 2019. https://doi.org/10.1186/s12940-019-0506-2.

“Kaggle: Your Home for Data Science.” *Kaggle.com*, 2025, www.kaggle.com/competitions/assignment-1-time-series-forecasting-may-2025/data. Accessed 25 May 2025.

“Cubic Spline Interpolation - Wikiversity.” *En.wikiversity.org*, https://en.wikiversity.org/wiki/Cubic_Spline_Interpolation.

“Google Colab.” *Google.com*, 2019, https://colab.research.google.com/drive/1q7KHhE1m4rEwln5w22h9a3_bpyjTJC99. Accessed 26 May 2025.