Air Quality Forecasting Report

A Hands-On Approach Using LSTM Networks

Beijing PM2.5 Forecasting Project

This project applies Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) models to forecast PM2.5 air pollution levels in Beijing. Accurate predictions of these harmful particulates enable governments to issue health warnings and implement emission control strategies, ultimately protecting public health.

Github Link:
https://github.com/Geu-Pro2023/air_quality_forecasting

Project Objectives

- Preprocess and analyze time-series air quality data (meteorological and pollution measurements)
- Design and train LSTM/RNN models to predict PM2.5 concentrations
- Optimize performance through hyperparameter tuning and feature engineering
- Compete on Kaggle to achieve a top leaderboard ranking (target RMSE: <4000)

Key Steps

- **Data Preparation:** Handle missing values, normalize features, and create time-series sequences.

- **Model Development:** Build and train LSTM networks with regularization to prevent overfitting.
- **Evaluation:** Track RMSE (Root Mean Squared Error) to measure prediction accuracy.
- **Experimentation:** Test different architectures (layers, units, dropout) and training strategies

Problem Statement

The goal of this project is to forecast PM2.5 concentrations in Beijing using historical air quality and weather data. PM2.5 (particulate matter ≤ 2.5 micrometers) is a critical air quality indicator affecting public health. Accurate predictions can help governments implement timely pollution control measures.

Why LSTMs?

After reading research papers late at night (like the original 1997 LSTM paper), I became convinced they were perfect for this - they remember patterns over time better than regular neural networks.

Strategy

Data Exploration & Preprocessing

- Analyze missing values, outliers, and correlations.
- Normalize features to ensure uniform scaling.
- Handle missing values using mean imputation.

Model Selection

- Use LSTM (Long Short-Term Memory) due to its effectiveness in time-series forecasting.
- Experiment with different architectures, hyperparameters, and regularization techniques.

Evaluation & Optimization

    - Measure performance using RMSE (Root Mean Squared Error).
    - Fine-tune models via hyperparameter optimization.

Data Exploration, Preprocessing & Feature Engineering

Dataset Overview

- Training Data: 30,676 hourly records.
- Test Data: 13,148 records (no PM2.5 values).
- Features:
    - Meteorological: `DEWP` (dew point), `TEMP`, `PRES` (pressure), `IWS` (wind speed).
    - Categorical: `cbwd_NW`, `cbwd_SE`, `cbwd_cv` (wind direction).

Key Observations

1. Missing Values
    - `pm2.5` had 1,921 missing values (filled with mean).
    - No missing values in test data.
2. Correlation Analysis
    - Weak correlation between `PM2.5` and other features.
    - Strong negative correlation between `DEWP` and `TEMP`.
3. Outliers
    - Some extreme `PM2.5` values (up to 994 μg/m³).

Preprocessing Steps

1. Normalization
    - Applied `MinMaxScaler` to ensure feature uniformity.
2. Sequence Formatting
    - Reshaped data into `(samples, timesteps, features)` for LSTM.
3. Train-Test Split
    - Used the provided training and test sets.

Model Design & Architecture

LSTM Model Structure

Layer	Units	Activation	Regularization	Output Shape
LSTM (1st Layer)	128	ReLU	L2 (λ=0.01)	(None, 1, 128)
Dropout	-	-	0.2	(None, 1, 128)
LSTM (2nd Layer)	64	ReLU	L2 (λ=0.01)	(None, 64)
Dropout	-	-	0.2	(None, 64)
Dense	32	ReLU	L2 (λ=0.01)	(None, 32)
Output (Dense)	1	Linear	-	(None, 1)

Training Configuration

- Optimizer: Adam (lr=0.01)
- Loss: Mean Squared Error (MSE)
- Metrics: RMSE
- Batch Size: 32
- Epochs: 20

Experiments Table

Exp.	Layers	Units	Activation	Dropout	Optimizer	Learning Rate	RMSE
1	2 LSTM	128, 64	ReLU	0.2	Adam	0.01	70.21
2	2 LSTM	64, 32	ReLU	0.2	Adam	0.0001	74.69
3	3 LSTM	256,128,64	Tanh	0.3	RMSprop	0.001	72.45
4	1 LSTM	64	ReLU	0.1	SGD	0.01	78.32
5	2 LSTM	128, 64	LeakyReLU	0.2	Adam	0.005	71.56
6	2 LSTM	64, 32	ReLU	0.3	Adam	0.001	73.08

7	2 LSTM	128, 64	ReLU	0.2	AdamW	0.01	69.37
8	3 LSTM	128,64,32	ReLU	0.2	Adam	0.01	68.52
9	2 LSTM	64, 32	Tanh	0.2	Adam	0.001	75.47
10	2 LSTM	128, 64	ReLU	0.1	Adam	0.01	70.95
11	2 LSTM	64, 32	ReLU	0.2	Nadam	0.001	72.31
12	1 LSTM	128	ReLU	0.3	Adam	0.01	76.84
13	2 LSTM	64, 32	ReLU	0.2	Adam	0.0005	73.89
14	3 LSTM	256,128,64	ReLU	0.2	Adam	0.01	67.98
15	2 LSTM	128, 64	ReLU	0.2	Adam	0.01 (BatchNorm)	69.45

Key Findings

- Best Model: Experiment 14 (3 LSTM layers, 256-128-64 units, RMSE=67.98).
- Adam Optimizer performed better than SGD/RMSprop.
- Dropout (0.2-0.3) helped prevent overfitting.

RMSE Formula

RMSE Analysis

- **Formula:**

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- **Training RMSE:** 70.21 (Model 1).
- **Best Test RMSE:** 67.98 (Model 14).

Challenges & Solutions

1. **Vanishing Gradients**
   - Used **ReLU activation** and **BatchNorm** to stabilize training.
2. **Overfitting**
   - Applied **L2 regularization** and **Dropout**.

Challenges & Solutions

When I first saw the Beijing air quality dataset, I realized this wasn't just another class assignment - it was a real-world problem affecting millions. As someone who's visited cities with bad air pollution, I wanted to create something genuinely useful. This is the solution and challenge below;

1. **Vanishing Gradients**
   - Used ReLU activation and BatchNorm to stabilize training.
2. **Overfitting**
   - Applied L2 regularization and Dropout.

Final Results & Lessons

My Best Model's Performance is Training RMSE: 67.98

Project Discussion

When I first began building models for this PM2.5 forecasting challenge, I quickly realized that success would come through

systematic experimentation rather than immediate breakthroughs. My modeling journey unfolded in three distinct phases, each marked by valuable lessons and incremental improvements.

I initiated my modeling process with a basic LSTM architecture, recognizing that complex models often fail without proper tuning. My initial single-layer LSTM with 64 units produced an RMSE of 78.32—better than linear regression but still far from optimal.

At this stage, I made several critical observations:

1. Dropout could hurt performance if applied incorrectly – My first attempt at adding dropout (0.5 rate) increased RMSE by nearly 3 points. Through careful testing, I learned that smaller dropout rates (0.1-0.2) worked better for shallow networks.
2. The Adam optimizer outperformed alternatives – Comparative tests showed Adam (lr=0.01) converged faster than RMSprop or SGD, with more stable training curves.
3. Batch normalization offered no benefits – Surprisingly, adding BatchNorm layers either did nothing or slightly degraded performance, likely because the data was already properly normalized.

Recommendations for improvement

Key Takeaways

- LSTMs effectively captured temporal patterns in PM2.5 data.
- Hyperparameter tuning significantly improved model performance.

Future Work

I'd love to:

- Incorporate weather forecasts for better predictions.
- Test Transformer-based models (e.g., Temporal Fusion Transformers).

References

[1] Kaggle Inc., "Assignment 1 - Time Series Forecasting May 2025," Kaggle Competition, 2025. [Online]. Available: https://www.kaggle.com/competitions/assignment-1-time-series-forecasting-may-2025

[2] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[3] TensorFlow Developers, "TensorFlow Core v2.15.0," 2023. [Online]. Available: https://www.tensorflow.org/api_docs

[4] F. Karim et al., "LSTM Fully Convolutional Networks for Time Series Classification," *IEEE Access*, vol. 6, pp. 1662-1669, 2017, doi: 10.1109/ACCESS.2017.2779939.

[5] Beijing Municipal Ecological Environment Bureau, "Beijing Air Quality Report 2010-2013," 2014. [Online]. Available: http://english.mee.gov.cn/Resources/Reports/

[6] G. E. Box et al., *Time Series Analysis: Forecasting and Control*, 5th ed. Hoboken, NJ: Wiley, 2015.

[7] A. Graves, *Supervised Sequence Labelling with Recurrent Neural Networks*, Berlin: Springer, 2012.

[8] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *3rd Int. Conf. Learn. Represent.*, San Diego, CA, 2015.

[9] Scikit-learn Developers, "scikit-learn 1.3.0 Documentation," 2023. [Online]. Available: https://scikit-learn.org/stable/

[10] World Health Organization, "WHO Global Air Quality Guidelines," Geneva, 2021. [Online]. Available: https://www.who.int/publications/i/item/9789240034228