Inception-inspired LSTM-GRU Hybrid Network for PM2.5 Forecasting

Yixiao Wang

January 20, 2024

Abstract

Predicting PM2.5 is crucial for environmental management. In recent years, with the rapid development of machine learning and deep learning, numerous machine learning methods have been applied to time series forecasting, such as RNN, LSTM[3], and GRU[1]. However, different models have their own advantages and disadvantages when handling various data types. Recently, some novel architectures have begun integrating different types of neural networks to leverage their respective strengths. The Inception module, a typical example of this approach, was first proposed by GoogLeNet[4]. Inspired by the Inception algorithm, this essay introduces an LSTM-GRU hybrid network architecture for PM2.5 forecasting, combining RNN and LSTM. This architecture demonstrates significant performance improvement compared to models using only LSTM or GRU. Experiments were conducted based on meteorological data from Chengdu between 2013 and 2015, with the corresponding code and test results provided.

1 Introduction

PM2.5 pollution has become a major global environmental concern. PM2.5 poses a threat to human health and negatively impacts the ecosystem. Therefore, accurately predicting PM2.5 concentration is essential for formulating effective environmental governance policies and taking timely measures. However, PM2.5 concentration is influenced by various complex factors, including meteorological conditions, emission sources, and regional dif-

fusion, exhibiting significant temporal and spatial dynamics, which makes forecasting a highly challenging task.

With the rapid development of machine learning and deep learning technologies, more and more studies have applied these methods to time series forecasting tasks. Among these methods, Recurrent Neural Networks (RNNs) have garnered attention due to their ability to capture temporal dependencies in sequential data. Improved models of RNNs, such as Long Short-Term Memory (LSTM)[3] networks and Gated Recurrent Units (GRU)[1], are widely used in various fields, including air quality forecasting, for their superior performance in handling long-term dependencies.

Although LSTM and GRU perform well in time series forecasting, they each have strengths and weaknesses depending on the dataset and application scenarios. For instance, LSTM excels in handling long-term dependencies, while GRU is advantageous in terms of computational efficiency and simplicity. To further enhance predictive performance, researchers have started exploring methods that combine multiple neural network models to fully leverage their strengths in different tasks.

Inspired by GoogLeNet's Inception module[4], this essay proposes an LSTM-GRU hybrid network architecture for PM2.5 time series forecasting. This architecture processes input data in parallel through both LSTM and GRU, then integrates their outputs, thus combining the advantages of both while maintaining computational efficiency. Compared with traditional single network models, this architecture demonstrates significant improvements in prediction accuracy and model stability. The results of this study provide new insights and technical approaches for deep learning-based air quality forecasting and lay the groundwork for future research.

The structure of this essay is as follows: The second section reviews LSTM and GRU neural networks and describes the proposed LSTM-GRU hybrid network architecture in detail; the third section presents data processing and exploratory data analysis; the fourth section showcases the experimental design process and results analysis; the fifth section summarizes the contributions of this study and provides an outlook on future research directions.

2 Methodology

2.1 LSTM

The LSTM network is a form of RNN that addresses the issues of gradient explosion and gradient vanishing in RNNs, demonstrating better performance in handling long-term dependencies [**?**]. LSTM shares the same chain structure as RNNs, but its units include three additional gates: input gate, forget gate, and output gate. Figure **??** illustrates the structure of an LSTM block, where c_{t-1} and h_{t-1} represent the cell state and hidden state from the previous time step, x_t is the current input, c_t and h_t are the updated cell state and hidden state, z_i is the input gate, z_f is the forget gate, z_o is the output gate, tanh and σ denote the hyperbolic tangent function and sigmoid function, respectively. All three gates use the sigmoid function, so their outputs range between 0 and 1, reflecting the proportion of information to retain or discard. In the figure, $\otimes$ denotes element-wise multiplication, and $\oplus$ denotes element-wise addition. The cell update process is shown in equation (2.6).

$$z_f = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{2.1}$$

$$z_i = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{2.2}$$

$$z_o = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{2.3}$$

$$\tilde{c_t} = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{2.4}$$

$$c_t = z_f \otimes c_{t-1} \oplus z_i \otimes \tilde{c_t} \tag{2.5}$$

$$h_t = z_o \otimes \tanh(c_t) \tag{2.6}$$

Here, z_f, z_i, and z_o represent the forget gate, input gate, and output gate, respectively; W_{xi}, W_{xf}, W_{xo}, W_{xc} are weight matrices; b_i, b_f, b_o, b_c are the corresponding biases.

2.2 GRU Network

The Gated Recurrent Unit (GRU) is an improved version of RNN that simplifies the structure of LSTM while retaining its ability to handle long-term dependencies. Similar to LSTM, GRU uses gating mechanisms to control the flow of information, but it only employs two gates: the reset gate and the update gate.

In GRU, the reset gate r_t and update gate z_t control how the information from the previous time step influences the current state update and how the current state combines with the previous state. The update equations for GRU are as follows:

$$z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z) \tag{2.7}$$

$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r) \tag{2.8}$$

$$\tilde{h_t} = \tanh(W_{xh}x_t + r_t \otimes (W_{hh}h_{t-1}) + b_h) \tag{2.9}$$

$$h_t = (1 - z_t) \otimes h_{t-1} \oplus z_t \otimes \tilde{h_t} \tag{2.10}$$

In the above equations, z_t is the update gate, controlling the proportion of new information mixed with old information; r_t is the reset gate, controlling the influence of the previous state on the current state; $\tilde{h_t}$ is the candidate hidden state. GRU enhances computational efficiency by simplifying the gate structure while achieving similar performance to LSTM in many tasks.

2.3 LSTM-GRU Hybrid Network

The core structure of the LSTM-GRU hybrid network is shown in the code. We define an Inception-like module, which contains three parallel branches:

- **LSTM branch**: Applies an LSTM layer with 30 neurons.
- **GRU branch**: Applies a GRU layer with 30 neurons.
- **LSTM + GRU branch**: First applies an LSTM layer with 20 neurons and returns the full sequence output, then applies a GRU layer with 20 neurons.

The outputs of these three branches are fused using the `concatenate` operation to form a comprehensive feature representation. In this experiment, the data is processed through two connected Inception-like modules, and the final prediction results are generated through a fully connected layer. Figure 1 shows the structure of the model.

3 Data Processing

3.1 File Import

- **Path**: The dataset is imported from the file `../data/ChengduPM20100101_20151231.csv`.
- **Libraries Used**: `pandas` is utilized for data processing and manipulation.

3.2 Dataset

- **Dataset**: The dataset is sourced from the UCI Machine Learning Repository. This study focuses on the weather data of Chengdu and predicts the PM2.5 concentration in Chengdu.

3.3 Column Description

The dataset contains the following columns:

- **datetime**: The date and time of the data record.
- **season**: The season at the time of data recording.
- **PM**: PM2.5 concentration (measured in micrograms per cubic meter).
- **DEWP**: Dew point temperature (measured in degrees Celsius).
- **TEMP**: Temperature (measured in degrees Celsius).
- **HUMI**: Humidity (measured as a percentage).
- **PRES**: Atmospheric pressure (measured in hectopascals).
- **cbwd**: Combined wind direction.
- **Iws**: Cumulative wind speed (measured in meters per second).
- **precipitation**: Hourly precipitation (measured in millimeters).
- **Iprec**: Cumulative precipitation (measured in millimeters).

3.4 Data Preparation

1. **Merge Date and Time Columns**: Combine the `year`, `month`, `day`, and `hour` columns into a single `datetime` column and set it as the index.

2. **Remove Irrelevant Columns**: Remove the `No` column to avoid the influence of irrelevant data.

3. **Select Data Range**: Analysis starts from January 1, 2013, since PM2.5 measurements are only available after this date.

4. **One-Hot Encoding of Seasons**: Use `pandas.get_dummies` to apply one-hot encoding to the season column.

5. **Calculate PM2.5 Values**:

   - Create a new `PM` column to store the PM2.5 values averaged from the various stations (`PM_Caotangsi`, `PM_Shahepu`, `PM_US Post`).
   - Apply linear interpolation to handle missing values.

6. **Label PM Values**: Classify the `PM` values based on concentration and add a `PM_label` column. The air quality classification used in this study is based on the "Technical Regulation on Ambient Air Quality Index (AQI) (Trial)" (HJ 633—2012). Below are the specific classification standards and corresponding pollutant concentration limits:

Table 1: Air Quality Sub-Index and Corresponding Pollutant Concentration Limits

Air Quality Level	Air Quality Index	$PM_{2.5}$ ($\mu g/m^3$)	PM_{10} ($\mu g/m^3$)	SO_2 ($\mu g/m^3$)	NO_2 ($\mu g/m^3$)
Excellent	0-50	0-35	0-50	0-50	0-40
Good	51-100	36-75	51-150	51-150	41-80
Mild Pollution	101-150	76-115	151-250	151-475	81-180
Moderate Pollution	151-200	116-150	251-350	476-800	181-280
Severe Pollution	201-300	151-250	351-420	801-1600	281-565
Serious Pollution	301-500	>250	>420	>1600	>565

7. **Remove Redundant Columns**: Remove the `hour`, `year`, and `day` columns.

8. **Convert Time Series to Supervised Learning Data**: Since PM2.5 values are closely related to the previous time step's state and exhibit continuity, we employed

a time series conversion strategy. This method transforms time series data into a format suitable for supervised learning by introducing lag values for each time step, creating input-output pairs, and removing rows with missing values. This approach helps capture short-term dependencies in the time series, enhancing the model's predictive capability. For more details, refer to [2].

9. **Dataset Splitting**: The dataset in this study covers three years of air quality data from 2013 to 2015. The data from 2013 and 2014 are used for model training, while the data from 2015 serve as the test set. This split allows the model to learn from the first two years of data and validate its generalization ability on the third year (2015).

3.5 Exploratory Data Analysis

Before performing time series forecasting, we conducted an exploratory data analysis (EDA) to uncover patterns and potential issues in the data. Figure 2 shows the trends of various features over time.

Based on the feature variations observed in the figure, we drew the following conclusions and recommendations for processing:

1. **Handling Wind Direction Feature:** The distribution of wind direction features is highly random, and the relationship between wind speed and PM2.5 concentration is more significant. Therefore, in subsequent model construction, we will remove the wind direction feature `cbwd` to simplify the model and reduce unnecessary noise.

2. **Seasonal Variation:** The PM2.5 concentration exhibits clear seasonal fluctuations, with significantly higher concentrations in winter than in summer. This seasonal variation may be related to heating, meteorological conditions, and other factors. In subsequent model training, we recommend introducing seasonal features (e.g., using a seasonal indicator variable or time series decomposition methods) to better capture this pattern.

4 Experimental Results and Parameter Settings

4.1 Model Configurations

In our experiments, we employed various models for comparison and configured them with the following parameters:

- **ARIMA Model**:
  - Order parameters: $(p, d, q) = (5, 1, 0)$
  - Training data: Supervised learning data
  - Prediction steps: 1-step prediction
- **SVR Model**:
  - Kernel function: RBF
  - Regularization parameter C: 1.0
  - ϵ: 0.1
  - Training data: Supervised learning data
- **DNN Model**:
  - Input layer: 100 neurons, ReLU activation function
  - Hidden layer: 50 neurons, ReLU activation function
  - Output layer: 1 neuron, linear activation function
  - Learning rate: 0.001
  - Batch size: 72
  - Epochs: 1000
- **LSTM Model**:
  - Hidden layer neurons: 50
  - Optimizer: Adam
  - Learning rate: 0.001

  - Batch size: 24
  - Epochs: 10000 (using early stopping)

- **LSTM-GRU Hybrid Model**:
  - Model structure: Inception-like block, including LSTM branch, GRU branch, and LSTM-GRU hybrid branch
  - Optimizer: Adam
  - Learning rate: 0.001
  - Batch size: 24 (24 hours per day)
  - Epochs: 10000 (using early stopping, patience=200)

4.2 Experimental Results

To compare the performance of different models, we recorded the RMSE, MAE, MAPE, and prediction accuracy on the test set for each model. The table below summarizes the comparison results of these metrics:

Table 2: Comparison Results of Different Models

Model	RMSE	MAE	MAPE (%)	Accuracy (%)
ARIMA	94.752	88.515	256.847	-
SVR	25.292	22.186	62.329	48.24
DNN	30.504	28.693	79.258	-
LSTM	8.682	6.151	14.051	91.56
LSTM-GRU Hybrid Model (Proposed)	8.261	5.621	11.224	94.44

4.3 Result Analysis

As shown in the table, the LSTM-GRU hybrid model outperforms the standalone LSTM model and other models in terms of RMSE, MAE, MAPE, and accuracy. The classification accuracy of the LSTM-GRU hybrid model improves by 34% compared to the LSTM model. This indicates that by combining the strengths of LSTM and GRU, the LSTM-GRU hybrid model can more effectively capture the trends in PM2.5 concentration, providing

more accurate predictions. The DNN model also performs relatively well, with results close to those of the LSTM-GRU hybrid model.

The traditional ARIMA model performs poorly in predicting PM2.5, with a MAPE as high as 256.847%, demonstrating its limitations in handling complex nonlinear time series data. Although the SVR model captures the trend in the data to some extent, it still cannot compare with deep learning models.

4.4 Visualization of Results

To further illustrate the effectiveness of the models, we compared the predicted PM values with the actual PM values, as shown in Figure 3. Figure 4 shows the changes in the loss function during training.

5 References and Article Notes

The inspiration for the Inception module in this article is entirely derived from GoogLeNet[4]. The strategy for converting time series data into supervised learning format is based on another method for analyzing PM2.5 forecasting. For more details, refer to [2]. The neural network framework diagrams in this article were adapted using resources from `https://github.com/dair-ai/ml-visuals`. The data and code used in this study can be accessed on my GitHub repository: `https://github.com/Yixiao-Wang-Stats/LSTM-GRU-Hybrid-Network`. The dataset used in this article is sourced from the UCI Machine Learning Repository. This article is based on my senior year time series analysis course project. The original text was in Chinese, and the content has been directly translated by ChatGPT-4o. Please excuse any discrepancies that may arise from the translation.

References

[1] BAHDANAU D, CHO K, BENGIO Y, 2014. Neural machine translation by jointly learning to align and translate[A].

[2] BROWNLEE J, 2020. Multivariate time series forecasting with lstms in keras[EB/OL]. https://machinelearningmastery.com/multivariate-time-series-forecasting-lstms-keras/.

[3] GRAVES A, 2012. Long short-term memory[M]//Supervised Sequence Labelling with Recurrent Neural Networks. Springer: 37-45.

[4] SZEGEDY C, LIU W, JIA Y, et al., 2015. Going deeper with convolutions[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 1-9.

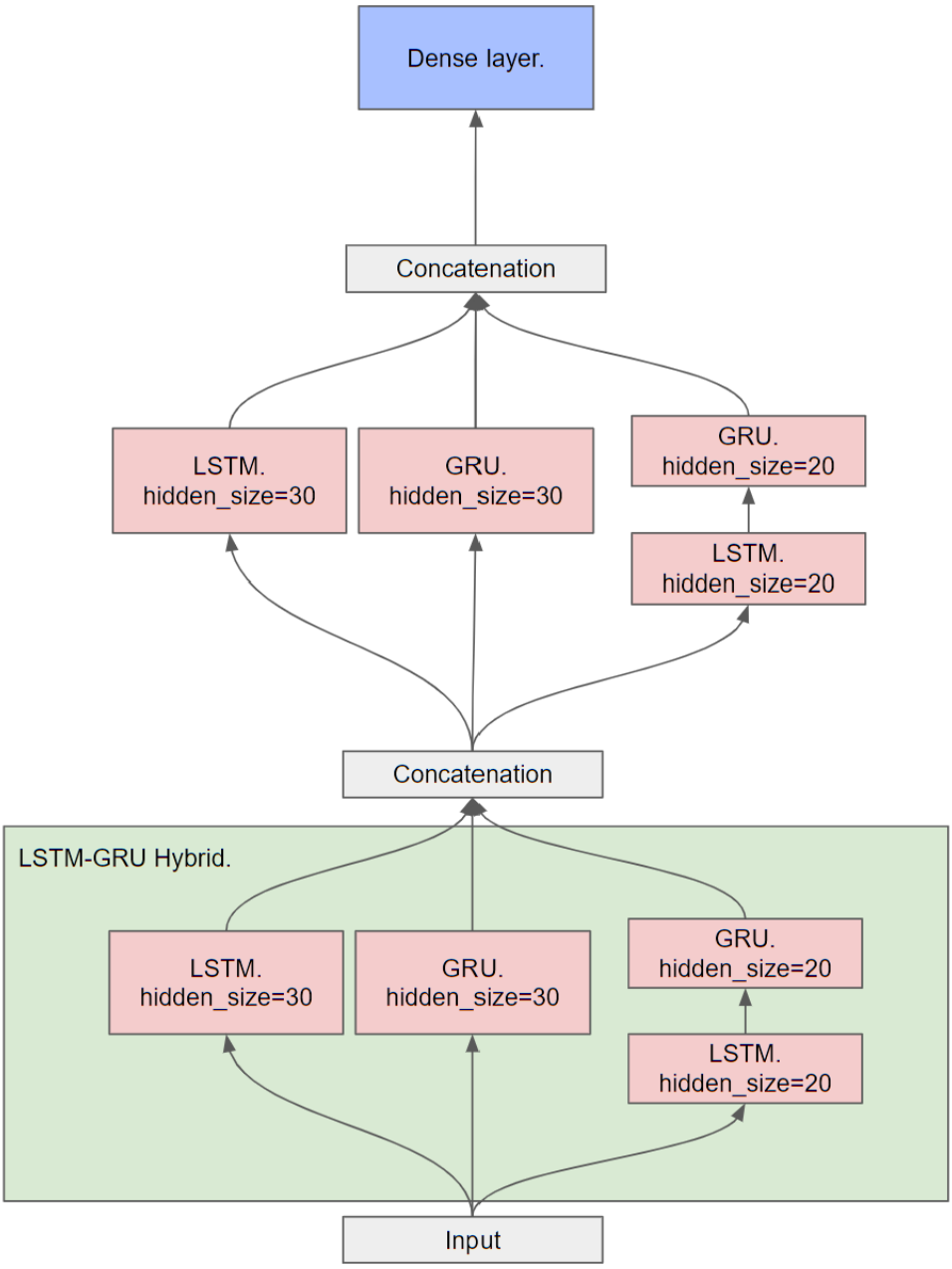

Figure 1: LSTM-GRU Hybrid Network Architecture

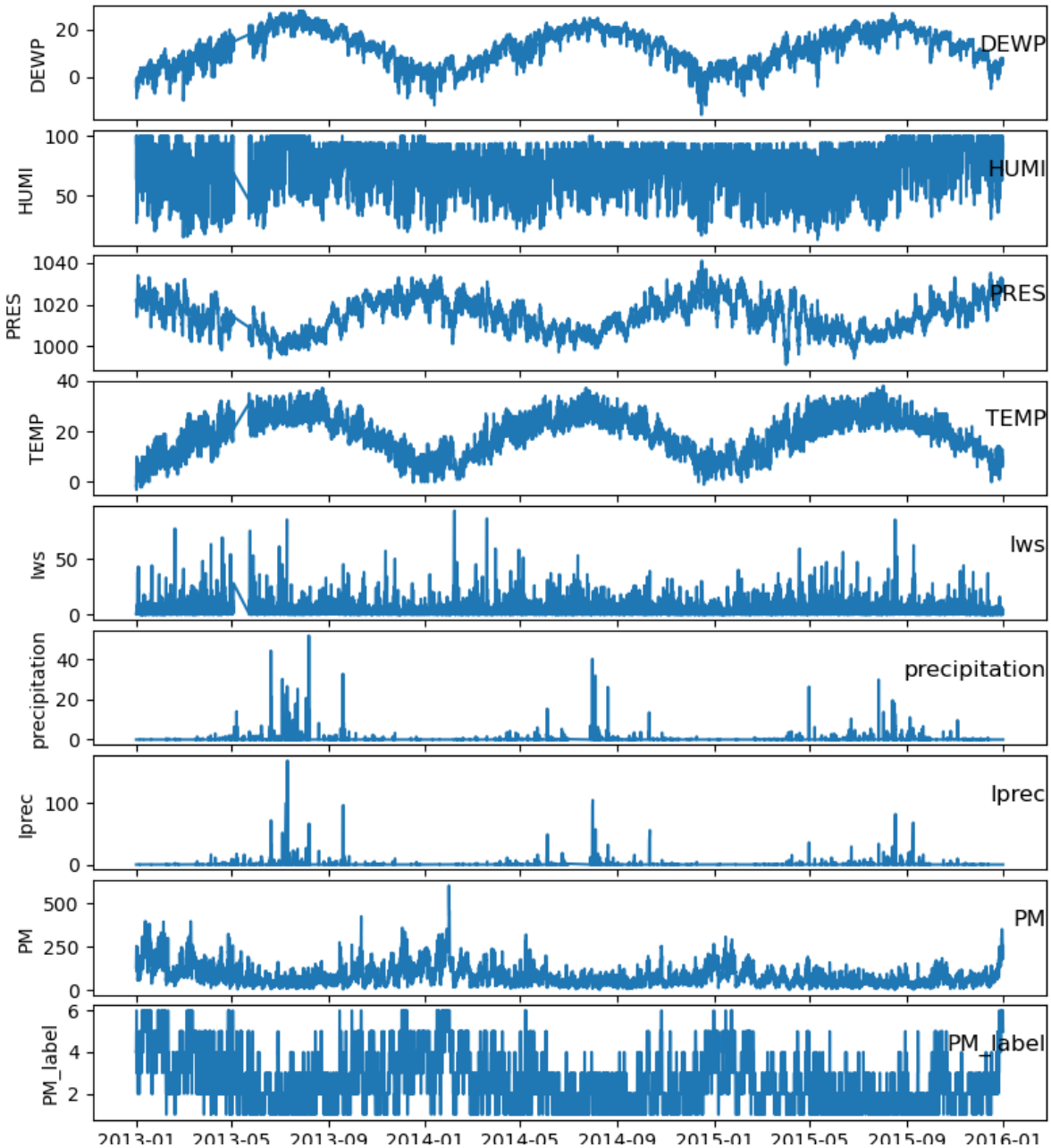

Figure 2: Trend of Various Features Over Time

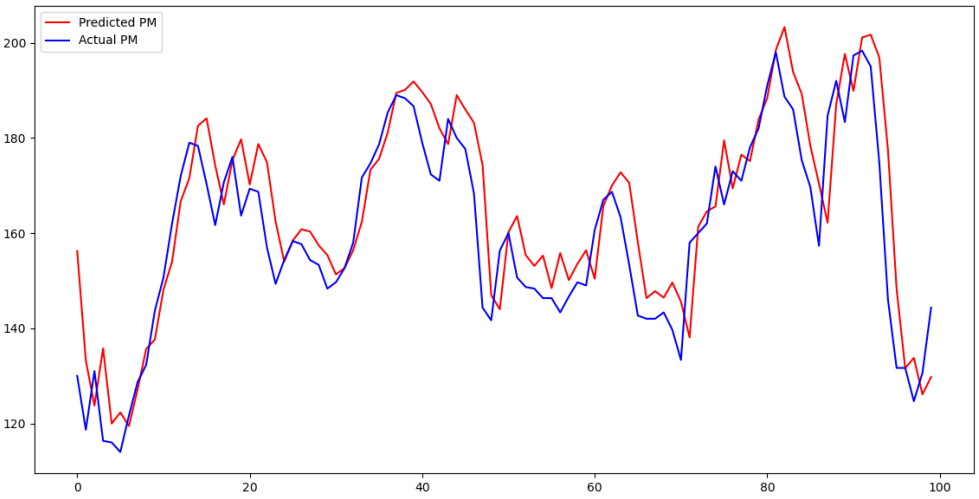

Figure 3: Comparison of Predicted PM Values and Actual PM Values

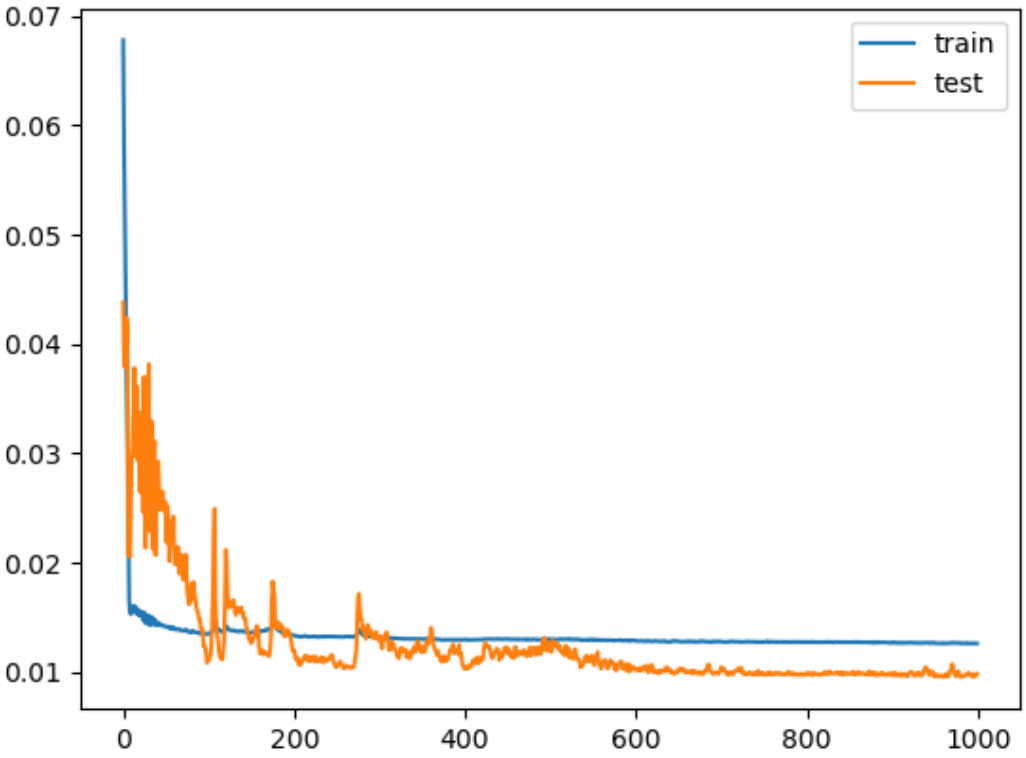

Figure 4: Loss Function Changes for Test and Training Sets