*sustainability*

MDPI

*Article*

# Attention-Based Distributed Deep Learning Model for Air Quality Forecasting

**Axel Gedeon Mengara Mengara, Eunyoung Park, Jinho Jang and Younghwan Yoo ***

School of Computer Science and Engineering, Pusan National University, Busan 46241, Korea; mengara@pusan.ac.kr (A.G.M.M.); ey5321@pusan.ac.kr (E.P.); jjh101101@pusan.ac.kr (J.J.)
* Correspondence: ymomo@pusan.ac.kr

**Abstract:** Air quality forecasting has become an essential factor in facilitating sustainable development worldwide. Several countries have implemented monitoring stations to collect air pollution particle data and meteorological information using parameters such as hourly timespans. This research focuses on unravelling a new framework for air quality prediction worldwide and features Busan, South Korea as its model city. The paper proposes the application of an attention-based convolutional BiLSTM autoencoder model. The proposed deep learning model has been trained on a distributed framework, referred to data parallelism, to forecast the intensity of particle pollution ($PM_{2.5}$ and $PM_{10}$). The algorithm automatically learns the intrinsic correlation among the particle pollution in different locations. Each location's meteorological and traffic data is extensively exploited to improve the model's performance. The model has been trained using air quality particle data and car traffic information. The traffic information is obtained by a device which counts cars passing a specific area through the YOLO algorithm, and then sends the data to a stacked deep autoencoder to be encoded alongside the meteorological data before the final prediction. In addition, multiple one-dimensional CNN layers are used to obtain the local spatial features jointly with a stacked attention-based BiLSTM layer to figure out how air quality particles are correlated in space and time. The evaluation of the new attention-based convolutional BiLSTM autoencoder model was derived from data collected and retrieved from comprehensive experiments conducted in South Korea. The results not only show that the framework outperforms the previous models both on short- and long-term predictions but also indicate that traffic information can improve the accuracy of air quality forecasting. For instance, during $PM_{2.5}$ prediction, the proposed attention-based model obtained the lowest MAE (5.02 and 22.59, respectively, for short-term and long-term prediction), RMSE (7.48 and 28.02) and SMAPE (17.98 and 39.81) among all the models, which indicates strong accuracy between observed and predicted values. It was also found that the newly proposed model had the lowest average training time compared to the baseline algorithms. Furthermore, the proposed framework was successfully deployed in a cloud server in order to provide future air quality information in real time and when needed.

**Keywords:** air quality forecasting; deep learning models; particle pollution; Busan metropolitan city; data parallelism architecture



**Citation:** Mengara Mengara, A.G.; Park, E.; Jang, J.; Yoo, Y. Attention-Based Distributed Deep Learning Model for Air Quality Forecasting. *Sustainability* **2022**, *14*, 3269. https://doi.org/10.3390/su14063269

Academic Editors: Filipe Araujo and Catarina Silva

Received: 10 January 2022
Accepted: 3 March 2022
Published: 10 March 2022



## 1. Introduction

Industrialization has brought us luxuries in life but, at the same time, many challenges too. Air pollution is one of the biggest challenges we are facing in the world today. The biggest contributors to air pollution each year are industries excreting dangerous gases, rapid urbanization, exponential fuel vehicles, and fossil fuel consumption. Air quality is getting worse. Several research reports have shown that higher atmosphere contamination levels and harmful particles impact the population's health, resulting in respiratory infections, cardiovascular diseases, and lung cancer. Therefore, air pollution research is critical and is consistently viewed as an important topic in environmental

protection. There is a growing need to evaluate and forecast the quality of air because, if people already know about air quality, they can take more precautions to prevent disease. Predicting air quality is also critical for emergency management in any administration. Projections aid the government in taking suitable contingency measures to minimize air pollution, such as limiting motor vehicles and lowering emissions from highly polluting companies. Unfortunately, predicting air quality is a difficult process, and improving forecast accuracy and reducing training time are urgent and difficult problems that need to be addressed in air pollution prevention.

Air quality prediction is a challenging task, but a lot of research has been done over the last decade to predict and address air quality challenges. This study uses an experimental setup to focus on two different forms, or models, of air quality forecasting. A knowledge-based model is one type, whereas a data-driven model is another. Knowledge-based paradigms focus on physical and chemical theories to demonstrate the transmission and deflection of air pollution particles. Several proposed knowledge-based methods have been studied in the literature survey to evaluate their impact. As expected, the successful implementation of knowledge-based methods involves intensive knowledge and research on environmental and atmospheric sciences. Using a knowledge-based model under varied conditions can also cause the chemistry and transfer rules to alter, resulting in inaccurate findings. Some scholars in the literature have employed statistical prediction approaches, including Hidden Markov Models [1], ARIMA models [2], and the LASSO model [3], to overcome this problem. The statistical prediction technique that uses mathematical reasoning and regression analysis, on the other hand, has two fundamental drawbacks: (1) a lack of precision and (2) time and energy consumption that is introduced from the examination of long-term historical monitoring data. In conclusion, statistical forecasting approaches [4] may accurately predict air quality, but a variety of air pollution factors make accurate forecasting challenging.

The other approach for air quality prediction involves evaluating data and is known as the data-driven model. Thanks to the big data analysis and artificial intelligence boom, this approach has recently been examined and utilized in various forecasting frameworks. The machine learning models used in this air quality prediction have been helpful to overcome the statistical and numerical drawbacks of previous approaches. These algorithms have established themselves as the backbone of air quality forecasting research. Machine learning methods for estimating air quality have so far yielded positive results. Wang et al. [5] proposed an online SVM model to forecast air pollution concentrations in downtown Hong Kong. They ran a comparative experiment between standard SVM and online SVM. Results showed that the online SVM proposed by Wang et al. [5] outperformed the standard SVM. In research performed in 2018, the air quality prediction in Murcia was conducted to predict ozone levels [6]. Researchers used different traditional machine learning models, out of which the random forest model performed the best [6]. Four regression techniques were used to predict air quality using machine learning algorithms, namely regression ANN, MLP regression, decision trees, and random forest regression [7]. This comparative study determined the best approach benchmark as processing time and data size. Random forest regression outperformed other algorithms as it optimally predicted air quality with various data sets, sizes, locations, and features.

Deep learning (DL) is commonly used in large data analysis to tackle a variety of issues, such as object recognition [8,9], speech recognition [10], classification [11], and prediction based on time series data [12]. The forecasted accuracy and efficiency of air quality is enhanced with the use of deep learning approaches. With the help of deep learning approaches [13], different air quality features can be obtained using the data-driven method. A wide range of articles in the literature have generated positive outcomes when employing deep learning approaches for air pollution prediction [14]. Zhao et al. [15] introduced a deep learning model named LSTM-Fully Connected Neural Network to predict the concentration level of particle pollution among specific monitoring stations over 48 h. Researchers used previous air quality data as well as meteorological and weather

forecast data. Finally, a data collection of records from 36 air quality monitoring stations was used to test the suggested approach. They made a comparative study between ANN and LSTM models. Qi et al. [16] used the deep learning approach to propose a hybrid model incorporating both graph CNN and LSTM networks (GC-LSTM) to model and predict the spatial-temporal fluctuation of particles concentrations. The authors created historical observations as spatiotemporal graph series, in which previous meteorological variables and air quality attributes were defined as graph signals. To evaluate their proposed system, they compared it with the existing methodologies and noticed that their proposed model outperformed the conventional approaches. To forecast the air pollution levels, Wen et al. [17] proposed a convolutional LSTM model. They combined a CNN model with an LSTM network to extract the spatiotemporal features. Model performance was also improved by incorporating meteorological and aerosol data. Authors in [18] conducted another study in which they used Convolutional Neural Network and LSTM to predict the air quality. The evaluation criteria included RMSE, MAE, agreement index, and Pearson correlation coefficient.

An hourly concentration of air quality forecasting system was proposed by Bai et al. [19]. The authors used a deep-stacked autoencoding algorithm. It combined the seasonal air quality analysis and feature learning to predict optimal air quality pollution. Their approach was evaluated based on the real-time dataset from three monitoring stations located at Beijing. A. Heydari et al. [20] proposed a hybrid model based on LSTM and multi-verse optimization algorithm to predict the concentration level of pollutants $SO_2$ and $NO_2$. The authors used a dataset collected from May to September 2019. The dataset included four features: wind speed, temperature, $SO_2$, and $NO_2$. The result showed the superiority of their approach against four other optimization-based algorithms. A multi-task learning approach based on GRU was proposed in [21] to predict air pollution particles, namely $PM_{2.5}$, $PM_{10}$, and $NO_2$. The authors formalized the particles pollution forecasting into a multitask architecture. They showed the superiority of their framework by comparing it with seven different baselines. Xiao et al. [22] proposed a weighted LSTM model to predict the concentration level of $PM_{2.5}$. The authors first generated the historical $PM_{2.5}$ time series data by using a sites distances feature, wind condition, and pollution concentration as input to a proposed MLP model. The experiment's results demonstrated that the proposed weighted LSTM model can improve the prediction of the $PM_{2.5}$ particle.

Yue-Shan Chang et al. [23] proposed an aggregated LSTM model to predict the $PM_{2.5}$ particle. Their research used a dataset based on 17 attributes which was collected from 2012 to 2015 by the Taiwan Environmental Protection Agency. In the experiments, the authors predicted $PM_{2.5}$ for the next 1–8 h. The results showed the superiority of the proposed approach against SVR, GBT, and a standard LSTM model. Another RNN based model was proposed in [24] to forecast the concentration level of $PM_{10}$ at different future time steps (6, 12, and 24 h). On the other hand, Guo et al. [25] proposed a feature engineering pipeline as well as a deep ensemble network algorithm which combines RNN, LSTM, and GRU networks to predict the $PM_{2.5}$ concentration of the next hour. During the data analysis step, the authors used the Pearson correlation coefficient to evaluate the correlation of $PM_{2.5}$ with meteorological data, season data, and time stamp data. The experiments showed that the proposed model surpassed the performance of the benchmark algorithms.

Attention mechanism is a learning concept which is based on mimicking human behavior by directing attention toward particular events. This notion of imitating human attention started a few years ago with computer vision studies [26,27]. The main goal was to decrease the computational complexity of image processing while enhancing performance by introducing an approach that would only focus on particular regions of images instead of the entire input frame. However, the attention mechanism had been utilized more in depth in the natural language processing field by Bahdanau et al. [28]. The authors in this article proposed an attention-based approach for a machine translation model. Today, attention mechanism has become an important part of deep learning models which is widely used in object detection [29], medical image segmentation [30], time series forecasting [31], natural

language processing [32], question-answering [33], and more [34–36]. Recently researchers have started to use attention mechanism for air quality forecasting as well. For instance, in [37], the authors used a variational autoencoder with multiple directed attention to predict the concentration of ambient pollutants, namely $NO_2$, $O_3$, $SO_2$, and $CO$. They evaluated their model by using six different evaluation metrics and outperformed the baseline models. Xiangyu et al. [38] proposed an LSTM-based attention algorithm for air quality prediction. They evaluated their model on a Beijing dataset and then compared it against six baseline algorithms: ARIMA, MFSVR, DeepST, LSTM, GC-LSTM, and ADAIN. The presented results showed how their models outperformed the baselines based on RMSE and $R^2$ evaluation. In [39], the authors proposed a seq2seq attention-based model for air quality forecasting. They replaced the RNN encoder with a pure attention mechanism with position embeddings. The proposed model was implemented at two different stations in Beijing city: the Dongsi and Olympic center monitoring stations. The experiments results showed that the attention-based GRU seq2seq model was more effective at predicting the particle $PM_{2.5}$ for the next 24 h. In the same vein, in [40], a forecasting approach based on dual LSTM was proposed. The authors first obtained each component by using a seq2seq method. Furthermore, they implemented an attention-based LSTM network as a multi-factor forecasting model which was afterwards connected to the seq2seq technology by using an Xgboosting tree in order to make the final prediction. On the other hand, Chen et al. [41] proposed an extreme value attention model based on an autoencoder network for air pollution forecasting. In order to capture long-term dependencies within the feature set, the authors used a temporal attention mechanism at the decoder level.

There have been numerous studies to address air quality forecasting with AI-based models, especially with machine learning and deep learning algorithms. Table 1 provides an overview of some previous approaches to solve the problem of air pollution forecasting.

**Table 1.** Literature review on air quality forecasting.

| Reference | Method | Dataset | Result |
|---|---|---|---|
| [1] | Hidden Semi-Markov models | EPA Air Quality System in Cook County, Illinois ($PM_{2.5}$ and meteorological data, 2000–2001 period) | Prediction accuracy of 100% with solar radiation, cloudiness, temperature, pressure, humidity, wind speed, dewpoint as input parameters |
| [2] | ARIMA, ARFIMA and HW smoothing | AQI from Chandigarh including RSPM, SPM, $SO_2$, $NO_2$ | ARIMA (RMSE: 18.20; MAE: 15.69; MAPE: 26.86) HW (RMSE: 30.12; MAE: 25.52; MAPE: 37.00) |
| [3] | LASSO regression combined to a nonlinear autoregressive model with exogenous inputs (NARX) | 1-AQI data from a national monitoring station in Nanjing (from 14 November 2018 to 11 June 2019) 2-Self data collection containing 234,717 raws. The data includes $PM_{2.5}$, $PM_{10}$, $CO$, $NO_2$, $SO_2$, $O_3$. | R-square: $PM_{2.5}$ (0.933); $PM_{10}$ (0.918); $CO$ (0.899); $NO_2$ (0.90); $SO_2$ (0.941); $O_3$ (0.936) RMSE: $PM_{2.5}$ (8.687); $PM_{10}$ (13.208); $CO$ (0.156); $NO_2$ (7.715); $SO_2$ (4.874); $O_3$ (12.190) MAE: $PM_{2.5}$ (5.951); $PM_{10}$ (8.981); $CO$ (0.098); $NO_2$ (4.806); $SO_2$ (2.464); $O_3$ (7.788) MAPE: $PM_{2.5}$ (0.146); $PM_{10}$ (0.146); $CO$ (0.095); $NO_2$ (0.177); $SO_2$ (0.131); $O_3$ (0.397) |
| [4] | Multiple Linear Regression Model (MLRM) Quantile Regression Model (QRM) Generalized Additive Model (GAM) Boosted Regression Trees 1way and 2way | Air quality pollutants data from the city of Makkah ($PM_{10}$, $CO$, $SO_2$, $NO_2$, humidity, temperature, wind speed) | Mean Bias Error (MBE): GAM (−39.9); MLRM (−29.3); QRM (−1.4); BRT1 (−43.9); BRT2 (−41.1) MAE: GAM (74.3); MLRM (80.0); QRM (61.0); BRT1 (75.6); BRT2 (80.4) MAPE: GAM (33.1); MLRM (35.7); QRM (27.2); BRT1 (33.7); BRT2 (35.8) RMSE: GAM (120.1); MLRM (123.8); QRM (95.6); BRT1 (121.1); BRT2 (125.6) |
| [5] | Online SVM | Air pollutant data in Hong Kong | Testing set MAE: 19.2902 RMSE: 25.8993 WIA: 0.7880 |

**Table 1.** *Cont.*

| Reference | Method | Dataset | Result |
| --- | --- | --- | --- |
| [6] | Bagging algorithm Random Committee Random Forest Random Forest KNN | Air quality data in the region of Murcia (NO, $NO_2$, $SO_2$, NOX, $PM_{10}$, Benzeno, Toluene, and Xileno) | Alcantarilla city results: Random Forest Year 2013: MAE (7.65); RMSE (10.20) Year 2014: MAE (7.33); RMSE (9.77) |
| [7] | Decision tree regression Random forest regression Gradient boosting regression ANN multi-layer perceptron regression | The dataset consists of five cities of China which include Guangzhou, Chengdu, Beijing, Shanghai, and Shenyang | MLP Results: Shanghai: MAE (13.84); RMSE (0.03) Guangzhou: MAE (12.2); RMSE (0.045) Chengdu: MAE (9.8); RMSE (0.108) Shenyang: MAE (13.65); RMSE (0.062) Beijing: MAE (21.79); RMSE (0.0806) |
| [14] | Convolutional Bi-Directional LSTM autoencoder model | Air quality data from South Korea ($PM_{2.5}$, $PM_{10}$, $NO_2$, $CO$, $O_3$, $SO_2$, temperature, humidity, and wind speed) | $PM_{2.5}$ MAE: 50.7 RMSE: 6.93 SMAPE:18.27 $PM_{10}$ MAE: 5.83 RMSE: 7.22 SMAPE: 17.27 |
| [15] | LSTM-Fully connected neural network | AQI dataset with 36 monitoring stations in Beijing (from 1 May 2014 to 3 April 2015) | 1–6 h Prediction: MAE: 23.97 RMSE: 35.82 7–12 h: MAE: 38.34 RMSE: 56.03 13–24 h: MAE: 47.13 RMSE: 65.60 25–48 h: MAE: 50.13 RMSE: 69.84 |
| [16] | Hybrid learning framework based on a Graph Convolutional network and a LSTM model | Hourly scaled dataset of pollutants ($PM_{2.5}$, $PM_{10}$ $NO_2$, $CO$, $O_3$, $SO_2$), from 76 stations over Beijing, Tianjin and Hebei. | +1 h prediction: IA: 0.98 MAE: 13.72 RMSE: 22.41 +72 h prediction IA: 0.92 MAE: 24.21 RMSE: 38.83 |
| [17] | A spatiotemporal convolutional LSTM extended model | Hourly $PM_{2.5}$ concentration data collected at 1233 air quality monitoring stations in Beijing and the whole China from 1 January 2016 to 31 December 2017. | RMSE: 12.08 MAE: 5.82 MAPE: 17.09 |
| [18] | Deep neural network model that integrates the CNN and LSTM | $PM_{2.5}$ dataset of Beijing | MAE: 14.63446 RMSE: 24.22874 Pearson Correlation: 0.959986 IA: 0.97831 |
| [19] | Stacked autoencoder model | The data was collection from three monitoring stations in Beijing (Station I, Wangshouxigong; Station II, Nongzhanguan; Station III, Shunyixincheng) | Station 1 (Spring): MAE: 8.01 RMSE: 10.28 R-square: 0.880 |
| [20] | LSTM model combined with multi-verse optimization algorithm | AQI data from Iran. It composed of wind speed, air temperature, $NO_2$, and $SO_2$ for five months. The data was collected from May–September 2019 with a time step of 3 h | With data type (1): Month of September for $NO_2$ prediction RMSE: 0.0545 MAE: 0.0465 MAPE: 17.4011 |
| [21] | Deep multi-task learning framework based on residual GRU | KDD CUP of Fresh Air | RMSE: 1.85 MAE: 1.15 |

**Table 1.** *Cont.*

| Reference | Method | Dataset | Result |
| --- | --- | --- | --- |
| [22] | Weighted LSTM extended model | Daily pollutants concentration and meteorological data from Beijing–Tianjin–Hebei | RMSE: 40.67<br>MAE: 26.10<br>Total accuracy index: 0.59<br>Spatial anomaly correlation: 0.9524<br>Temporal correlation coefficient: 0.9930 |
| [23] | Aggregated LSTM neural network | The data was collected in Taiwan from 2012 to 2017. It contains 17 attributes based on air pollutants and meteorological information. | RMSE: 0.44<br>MAE: 0.91<br>MAPE: 16.3 |
| [24] | RNN and LSTM models | Air quality data collected at Skopje | 6 h prediction SimpleRNN + Dense with ReLU:<br>MSE: 0.0007<br>RMSE: 0.0273 |
| [25] | Deep ensemble model which combines RNN, LSTM, and GRU networks | $PM_{2.5}$ concentration and meteorological data collected at 3 stations in Shanghai (From 1 January 2010 to 31 December 2015) | Group 1:<br>MAE: 6.72<br>MAPE: 19.60% |
| [37] | VAE with multiple directed attention mechanism | Data from the United States Environmental Protection Agency which includes $NO_2$, $SO_2$, $CO$, and $O_3$ | Case of $NO_2$ prediction in Pennsylvania:<br>RMSE: 12.373<br>MAE: 10.370<br>R-square: 0.831<br>Explained variance: 0.955<br>MAPE: 13<br>Mean bias error (MBE): −2.47718<br>Relative MBE: −3.21039 |
| [38] | LSTM model based on a spatiotemporal attention mechanism | Beijing air quality dataset (from 1 January 2018 to 31 December 2018) | 1 h-Prediction:<br>RMSE: 12.23<br>R-square with different set of features:<br>R-square: 0.78 |
| [39] | Seq2Seq model with attention mechanism | Beijing air quality data from April 2017 to March 2018 | Olympic Center:<br>RMSE: 38.119<br>R-square: 0.493<br>Dongsi:<br>RMSE: 59.508<br>R-square: 0.337 |
| [40] | Integrated dual LSTM with attention mechanism | Beijing air quality data from 2013 to 2018 | RMSE: 14.36<br>MAE: 8.39<br>MAPE: 35.78<br>R-square: 0.89<br>IA: 0.93 |
| [41] | Extreme value attention network based on encoder and decoder framework | Air quality data from Fuzhou (2 November 2017 to 2 October 2018) and Beijing (1 January 2018 to 31 December 2018) | Fuzhou dataset:<br>RMSE: 17.9914<br>MAE: 9.3818<br>Beijing dataset<br>RMSE: 3.2606<br>MAE: 2.1867 |

Recently, the literature has showed that the hybrid model and the attention-based approach are very effective in predicting the air quality; however, they face some challenges:

- Slow training speed: This issue is caused by the huge amount of data that needs to be trained on a centralized deep learning architecture. In previous research, the authors used a centralized deep learning approach for training. This problem with slow training speed can be bothersome when the data changes. Therefore, models need to be retrained for more accurate predictions.
- Noisy data: The second disadvantage of these methods is that they dismiss noise effect. Indeed, noise effect is another potential challenge to evaluating air quality and meteorological data. Noise affects the accuracy and performance of the forecasts because the algorithms do not extract the optimal feature and information from pollutant and meteorological data.

- No real-time traffic data: The third challenge is that these models do not count traffic information in their set of features during the training phase. It is widely recognized that traffic flow is a significant source of air pollutants that could damage air quality. Not considering traffic information while predicting air quality can lead to nonrealistic forecasting.

These challenges show a need to have a more optimal deep learning model to evaluate air quality. We first conducted a prior study for air quality forecasting in Busan City [14]. Compared to our previous work, in this research we propose a deep learning model focusing on the attention-based convolutional BiLSTM autoencoder framework to forecast air quality. A distributed architecture (data parallelization) has been implemented during the training. The dataset used in this research is based on three specific feature sets: particle features, meteorological features, and car traffic-related features. The major contribution of this research includes:

- The review of new deep learning and machine learning approaches for air quality prediction alongside attention mechanism.
- The implementation of YOLOv5 model with DeepSort algorithm to collect traffic information data on the road.
- An extensive feature engineering to find the most important features while predicting both particles $PM_{2.5}$ and $PM_{10}$. Knowing a feature's importance is an essential step in prediction tasks. Machine learning models have been widely used in the literature to find the most relevant features in a given dataset. In this research, we made a comparative study with four different machine learning algorithms to perform this task.
- The development of an innovative attention-based convolutional BiLSTM autoencoder model to predict air quality. The 1D-CNN layers extract deep spatial correlation and local patterns features from the air pollutants. The autoencoder model encodes the meteorological and the traffic data. The Bi-LTSM layer interprets the obtained features, then passes them to several attention layers to make the final optimal prediction.
- The evaluation of the proposed framework which is based on two specific steps. Firstly, we trained the deep learning framework in centralized architecture using a single training server. This trained model has been compared with seven state-of-the-art algorithms, including two attention-based deep learning approaches. Secondly, we trained the proposed model using a parallelization training approach. We evaluated the improvement in its accuracy and the reduction of training time during this phase.
- The creation of a deployment pipeline to run online inference based on the proposed deep learning model. We noticed that most of the previous studies only focused on offline learning and inference. In our research, we are not only considering offline prediction but also online prediction based on real-time data.

The structure of the rest of this research paper is as follows. In Section 2, we present the data collection and methods. In Section 3, we introduce our deep learning framework. We explain the data pre-processing step, the feature correlation and selection, as well as the proposed algorithm. The experiments are described in Section 4. The conclusion and future work are discussed in Section 5.

## 2. Data Collection and Methods

### *2.1. Pollution Particles and Meteorological Data Collection*

This research has been conducted based on three main data sources, which are: (1) air quality particle data from Air Korea (Seoul, South Korea) [42], (2) meteorological information from the Korean Meteorological Agency (KMA, Seoul, South Korea) [43], and (3) real-time traffic data. Both (1) and (2) data sources were collected from Korean government websites and contain ten specific features, which are: $PM_{2.5}$, $PM_{10}$, $NO_2$ (Nitrogen dioxide), $CO$ (Carbon monoxide), and $SO_2$ (Sulfur dioxide) for air pollution-related features, and temperature, rain precipitation, dew point, wind speed, and humidity for meteorological

features. The traffic-based data were collected on the road using a computer vision-based approach explained in the following section.

### *2.2. Traffic Data Collection—Car Counting*

One of the contributions of this research is the utilization of traffic data to improve the prediction of air quality. We collected the traffic data near the monitoring stations. Figure 1 shows a road near Namsan Station in Busan city where we collected the traffic data for the air quality station in Namsan-dong. We used a computer server with an NVIDIA Geforce RTX 2060 GPU and a camera with a resolution of 1080p 30 fps. The traffic count algorithm used the object detection model YOLOv5 [44] and the 'DeepSort' algorithm to track objects (cars) to prevent multiple counting for one object. Algorithm 1 presents the process of car counting using DeepSort.

**Figure 1.** Traffic collection setup in Namsan Station.

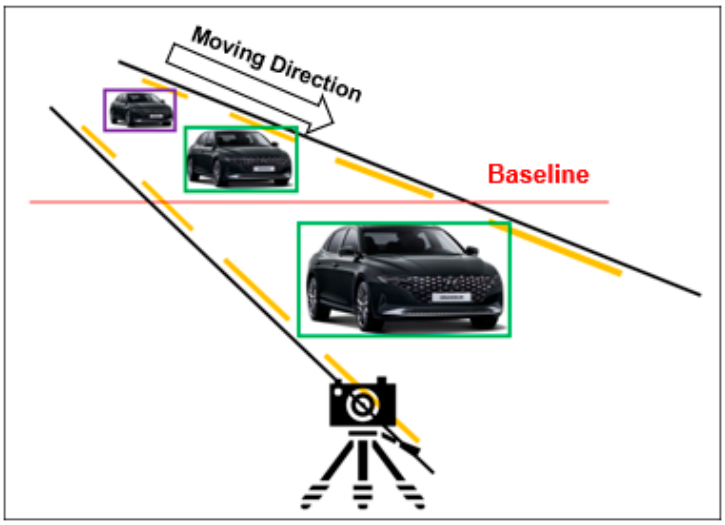
**Algorithm 1:** Input Data verification

**Input**: Car /*Object that is tracked by Deep Sort Algorithm. */
1. **IF** Car > diff /*diff: difference reference line and object's center point. */
2. **FOR** each entry tl ∈ TL do
3. **IF** Car in TL /*TL: Track List */
4. THEN pass
5. **ELSE**
6. THEN Add Car to TL
7. **END**
8. **RETURN** length(TL)
**END** Car Counting

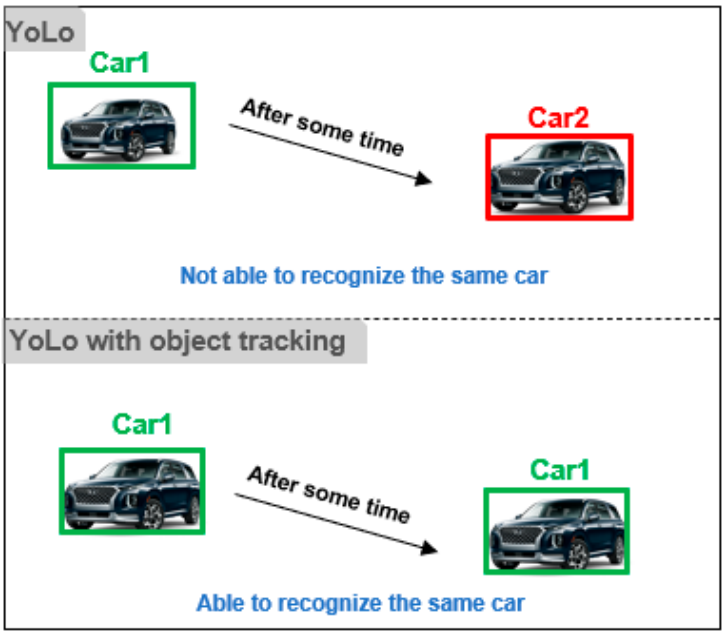
The YOLOv5 is the fifth model of You Only Look Once (YOLO). Compared with the previous version of YOLO and other object detection models, YOLOv5 considerably improves the training speed while maintaining a good detection accuracy. The model divides the input images into grid cells and creates a bounding box for the objects. The score of each grid is calculated by multiplying the probability that an object exists in the bounding box on the grid, intersection over union (IOU), and the probability that the object is the corresponding class. By comparing the score with the threshold, the model can define from which class the bounding box belongs to. The DeepSort algorithm predicts the next location of the object by applying the location information of the object detected in the previous frame with the Kalman filter. The IOU distance of the currently detected object

and the object predicted in the previous frame is calculated in the next frame. By applying the calculated result to the Hungarian algorithm, we determined whether it is the same object or not.

The DeepSort algorithm cannot track the object properly if the boundary box is smaller than a specific size. Therefore, as shown in Figure 2a, a virtual baseline is set to apply the algorithm only to the size of the objects. If the center of the bounding box that indicates the vehicle is located below the reference line, the count increases. To prevent multiple counting of a single object, the algorithm keeps the bounding box's index number in a temporal array and compares it with the new frame's bounding boxes. The union of YOLOv5 and DeepSort algorithm is used to recognize an object that has been detected before. For example, if Object 1 was already detected in Frame 1, the algorithm will not consider it again if it comes in Frame 2 or 3, as shown in Figure 2b.




(**a**)




(**b**)

**Figure 2.** Traffic Data Collection. (**a**) Counting process. (**b**) Car detection process.

## 3. Proposed Framework

The proposed framework workflow is shown in Figure 3. Its major phases are comprised of the following processes: (i) pre-processing data; (ii) features correlation; (iii) deep learning models formulation; (iv) hyperparameter fine-tuning; and (v) model evaluation.

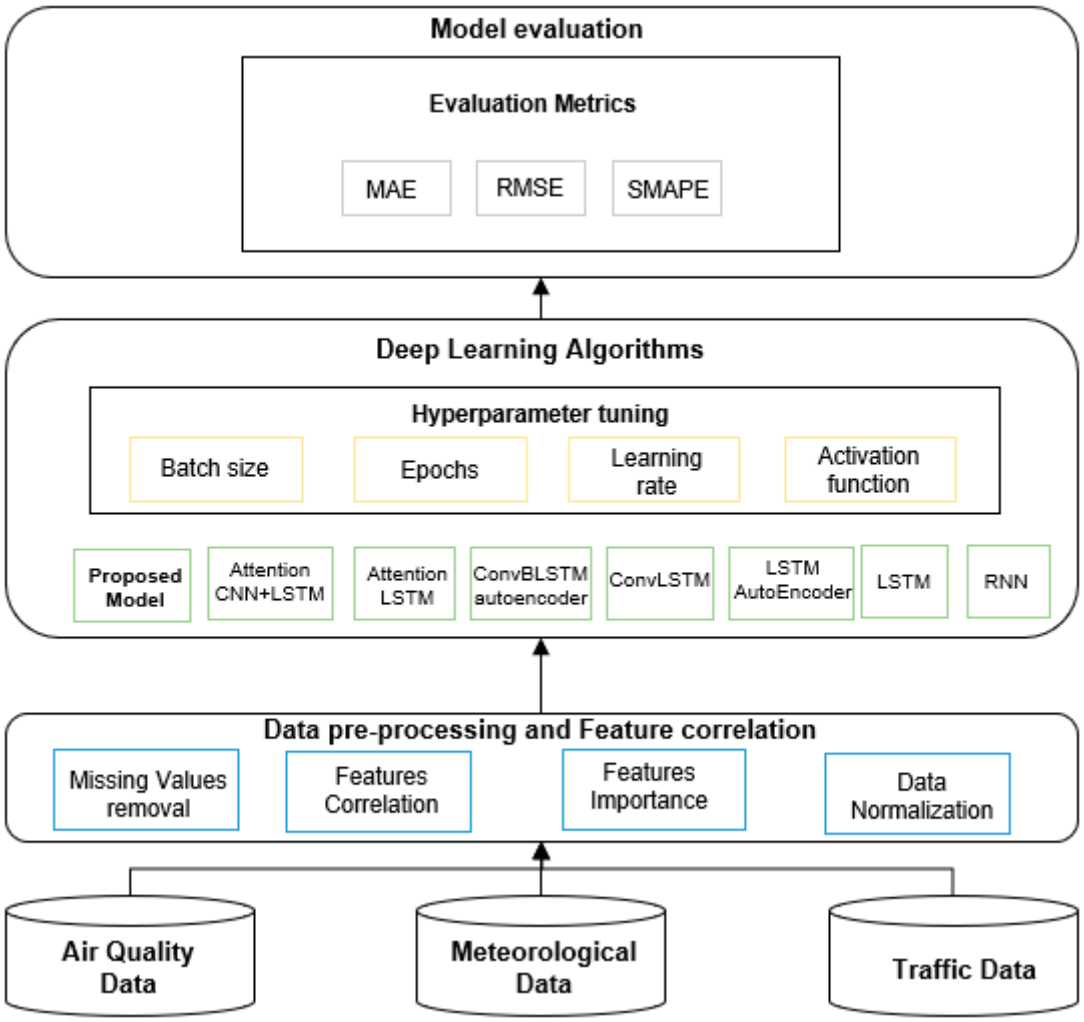



**Figure 3.** Proposed framework workflow.

### *3.1. Data Preprocessing*

There are over 490 air quality monitoring stations in South Korea. Data from several stations encounter the problem of missing values due to some errors with the sensors. To solve this problem, we used a linear interpolation technique. Linear interpolation imputed the missing values using the mean value of the last and first available in the dataset. It is based on the following formula [45]:

$$f(x) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0} \times (x - x_0) \quad (1)$$

In the presented equations, $x$ represents the independent value and $x_0$ and $x_1$ are the known values of the independent variable. The air quality is impacted by several factors, including meteorological and traffic data that have their physical properties and dimensions. With accurate analysis and seamless training, these factors can be helpful in identifying the air quality. We have combined all data sources into a single dataset in our framework

during the data pre-processing step. Each dataset is obtained by normalizing it using linear scaling based on the following equation [46], where $x^i$ represent the normalized data.

$$x^i = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{2}$$

### 3.2. Data Correlation and Analysis

Optimal air pollutants prediction can be achieved by a model that must quantify and evaluate various air quality indexes. The motivation of this research is to predict the intensity of air pollutant particles, namely $PM_{2.5}$ and $PM_{10}$. We extend our understanding by finding the relation of these particles with other features, especially meteorological and traffic data. Findings showed that the particles $PM_{2.5}$ and $PM_{10}$ can be impacted by many measurable factors. However, the challenging part was separating the data that is not beneficial for the prediction task, as irrelevant factors are burdensome for the model and training phase. It was important to find a correlation between different relevant and important particles with meteorological data. To overcome this, a correlation coefficient ($\delta$) was calculated to each feature and the target particle based on the following formula [47], supposing that we had an observation vector A = ($a_1, a_2, a_3, \ldots, a_n$) with another vector B = ($b_1, b_2, b_3, \ldots, b_n$):

$$\delta = \frac{n\sum_{i=1}^{n} a_i b_i - \sum_{i=1}^{n} a_i \sum_{i=1}^{n} b_i}{\sqrt{n\sum_{i=1}^{n} a_i^2 - \left(\sum_{i=1}^{n} a_i\right)^2}\sqrt{n\sum_{i=1}^{n} b_i^2 - \left(\sum_{i=1}^{n} b_i\right)^2}} \tag{3}$$

The prior formula presents a positive correlation when $0 < \delta < 1$ and a negative correlation when $-1 < \delta < 0$. Similarly, in a scenario where the value of $\delta$ is closer to 1, a small variation between A and B is observed, along with a higher correlation. Figure 4 shows the feature correlation heatmap of datasets used in this research analyzing particles such as $PM_{2.5}$ and $PM_{10}$ as targets. As shown in Figure 4, $PM_{2.5}$ and $PM_{10}$ particles are negatively correlated with temperature and wind speed, so in low-temperature conditions, these particles will be more concentrated in the air, resulting in worse air quality prediction. If we consider the wind speed factor, with faster or higher wind speed, the lower concentration of the pollution particles results in good air quality. With humidity factors, the concentration of pollution particles is higher, resulting in bad air quality. Other gases which impact the higher concentration levels of harmful particles relate to high levels of $NO_2$, $CO$, $SO_2$, and a low level of $O_3$. On the other hand, the traffic data is highly correlated with both particles, which makes us think that high car traffic on a specific location can produce air pollution. Based on this feature correlation heatmap, we can clearly see how the traffic information impacts levels of both particles, as they are highly correlated.

### 3.3. Features Importance Assessment

Features importance is a significant step in forecasting tasks. It assesses the relevance of a feature from the given feature set $F = \{f_1, f_2, f_3, \ldots, f_n\}$; the importance value of each feature is computed by using the permutation importance. In this research, a feature assessment was carried out to rule out the useless features that can increase algorithm complexity. Machine learning algorithms have been widely used in the literature to find the most important feature in a given dataset. In this study, we implemented four different machine learning models to perform this task: random forest, Xgboost, AdaBoost, and gradient boost.

As shown in Figure 5, all the algorithms except Xgboost algorithm selected the cars traffic as one of the most important features.

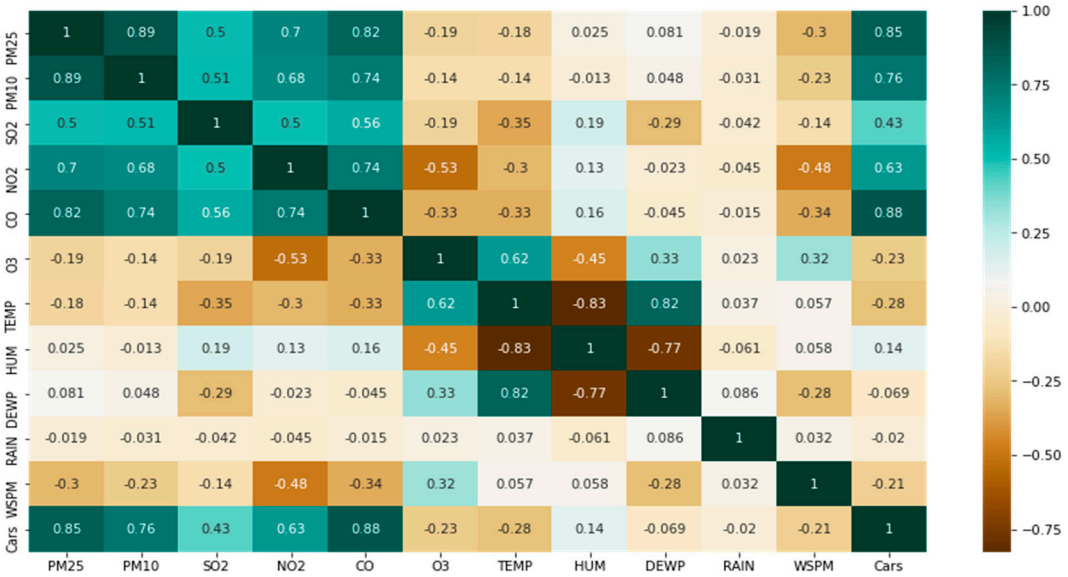



**Figure 4.** Data correlation.

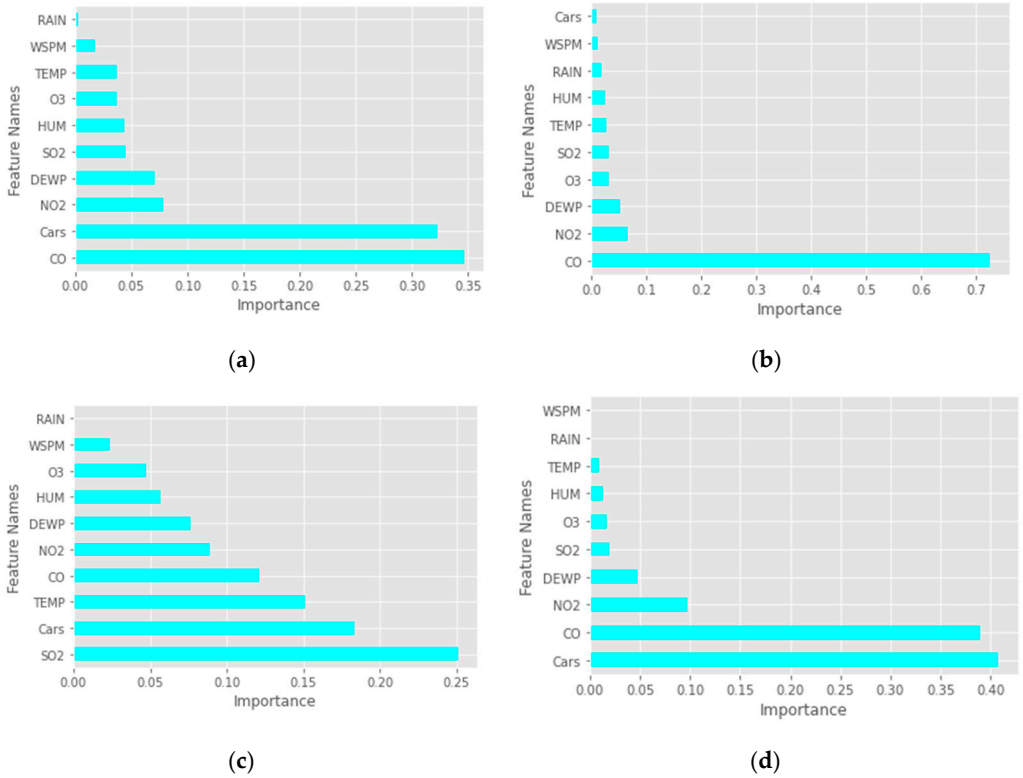



**Figure 5.** Feature importance representation. (**a**) Random Forest, (**b**) Xgboost, (**c**) AdaBoost, (**d**) Gradient Boost.

Pollution gas such as $CO$, $NO_2$, and $SO_2$ are also labelled as relevant features for the prediction of particle matter ($PM_{2.5}$ and $PM_{10}$). We then evaluated the error rate of each of these models while predicting $PM_{2.5}$ in order to select the best features that will be used to train the deep learning algorithms presented in the experiments. Based on the results presented in Figure 6, we decided to focus only on the feature importance of random forest, Xgboost, and gradient, since these algorithms had the lowest error rate (MAE, RMSE, and SMAPE); this led us to remove the rain feature and keep the remaining ones for the next training step. One reason that explains the poor performance of AdaBoost in this

experiment is the presence of noise in the air quality dataset. As discussed in previous studies [48,49], AdaBoost has been proven to be a very efficient ensemble learning model, which recurrently generates a set of diverse weak learners and combines their outputs using the weighted majority voting rule as the final decision. However, in some cases, AdaBoost leads to overfitting, especially when the training dataset is noisy, engaging in both its reduced generalization performance and non-robustness.

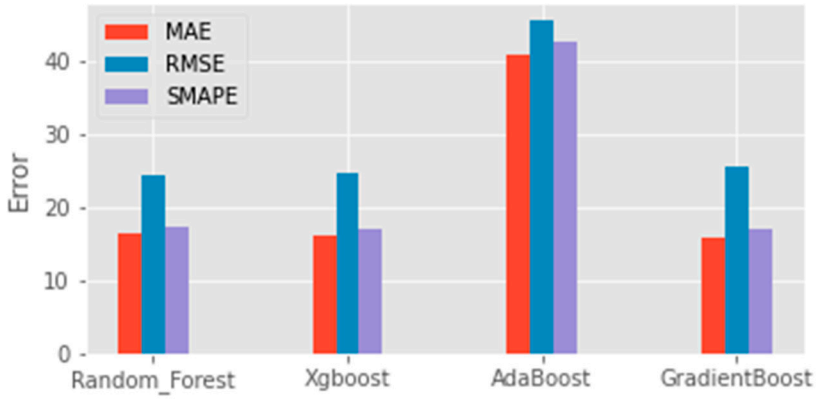



**Figure 6.** Feature importance models evaluation.

### *3.4. Deep Learning Model*

The standard deep learning model's effectiveness can be enhanced using different hybrid learning approaches. The proposed framework consists of a stacked deep autoencoder, a convolutional neural network, and an attention-based bi-directional LSTM layer. Together these layers of the framework calculate the concentration level of $PM_{2.5}$ and $PM_{10}$ based on data collected in Busan city. The deep features were extracted in two stages based on the proposed deep learning architecture. The first stage extracted all suitable features from both particles of $PM_{2.5}$ and $PM_{10}$ along with time series data, whereas a stacked autoencoder layer was used to encode meteorological, gas, and traffic-related patterns during the second stage. Figure 7 shows the architecture of the proposed model.

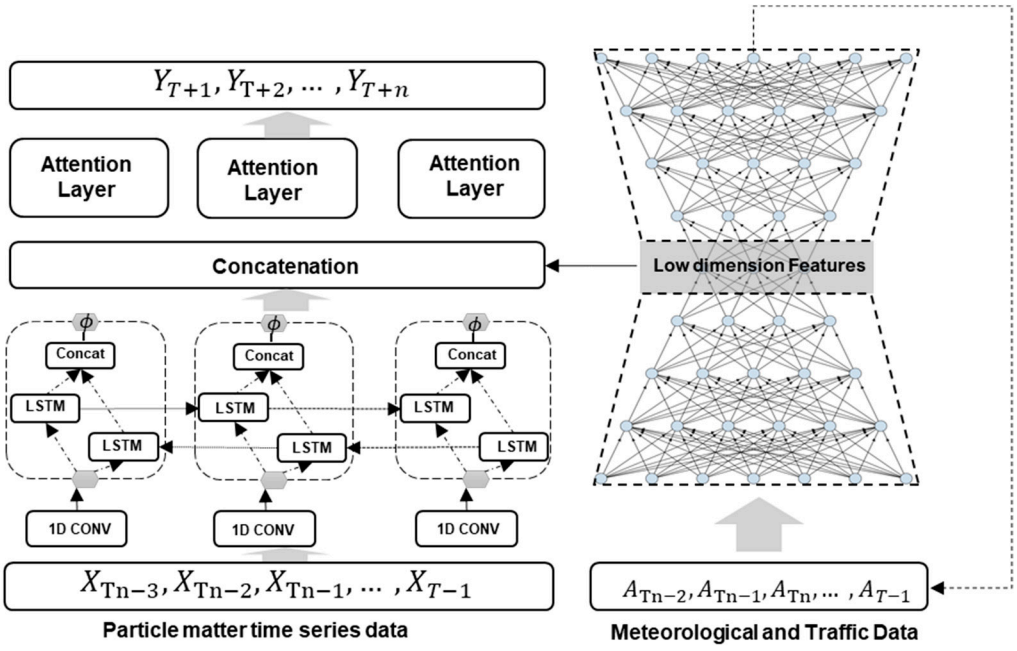



**Figure 7.** Proposed deep learning architecture.

3.4.1. One-Dimensional CNN for Local Features Mining

Deep convolutional networks are special neural networks that are based on a convolution operation. Convolution operation is frequently used for extracting the frequency or temporal features from specific signals in machine learning. The convolution of two functions $f(t)$ and $g(t)$ can be represented as follows [50]:

$$(f \times g)(t) = \int_{-\infty}^{+\infty} f(\vartheta) g(t - \vartheta) d\vartheta \tag{4}$$

In the above equation, $g(t)$ is called kernel and * represents the convolution operator. The time series data for air quality particles is fed into the 1D CNN network to extract local features of particles. Considering the time series of the $i^{th}$ time window as:

$$X_i = \left[x_i^1, x_i^2, \ldots, x_i^l\right], x_i \in R^d \tag{5}$$

where $l$ represents the window size, $x_i^t$ is the multivariate time series obtained at time step $t$ and d represents the dimension. The temporal convolution layer is the first layer of the model. It is able to automatically extract local features of multivariate time series. In this setup, the convolution operation is represented as follows:

$$C_\tau = ReLU\left(w_\tau \times x + b\right) \tag{6}$$

where $C_\tau$ is the result feature map of the $\tau^{th}$ kernel, $w_\tau$ and $b$ represent the weight and the deviation parameters, respectively. The 1D CNN model constantly moves the matching convolution step from the start of the time series to complete the convolution operation of the entire data. The features obtained from the convolution layer are represented as follows:

$$C_i = [C_1, C_2, C_3, \ldots, C_{l-m+1}] \tag{7}$$

In the above equation, $m$ denotes the size of the convolution kernel. As depicted in Figure 8, which represents the overall diagram of one-dimensional convolution computing, the input size is represented by $n \times l \times d$, in which $n$ denotes the number of input samples, the output feature map size is computed by $\frac{(l-m)}{(s+l)} \times \tau$ , with $\tau$ representing the number of kernels. In order to complete the convolution operation at each time step to obtain the local features, we set the kernel size of $m$ to 1.

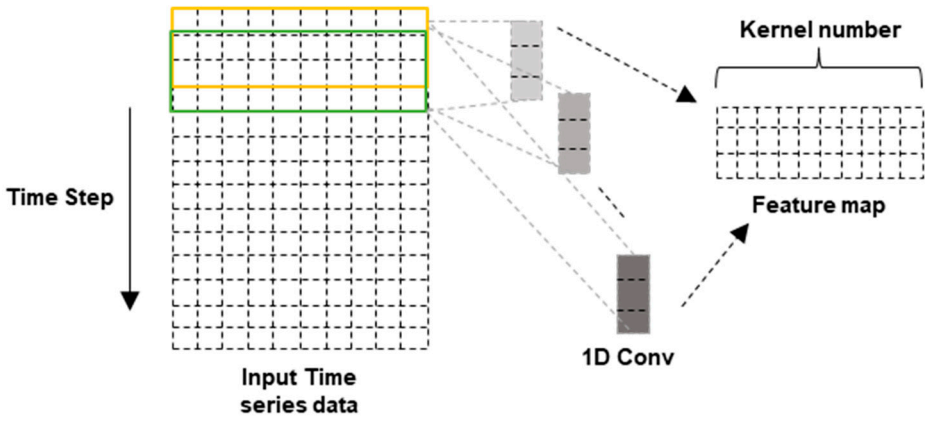



**Figure 8.** Representation of a convolution at time window $l$.

The detailed parameter settings of our 1D CNN layers are presented in Table 2.

**Table 2.** 1D CNN Parameters settings.

| 1D-Convolution | Settings |
|---|---|
| Convolution Layer | Kernel Size = (15, 1), Filter = 20, Stride = 1 |
| Max Pooling Layer | Pool-size = (2, 1), Stride = 2 |
| Dropout | 0.20 |
| Convolution Layer | Kernel Size = (10, 1), Filter = 40, Stride = 1 |
| Max Pooling Layer | Pool-size = (2, 1), Stride = 2 |
| Dropout | 0.20 |
| Convolution Layer | Kernel Size = (5, 1), Filter = 80, Stride = 1 |
| Max Pooling Layer | Pool-size = (2, 1), Stride = 2 |
| Dropout | 0.20 |

#### 3.4.2. Deep Autoencoders Model for Traffic and Meteorological Data Encoding

Air pollution forecasting requires several crucial indicators to be properly monitored. For example, to predict the particles $PM_{2.5}$ and $PM_{10}$, it is necessary to consider the traffic features and meteorological data, as they will help predict accurate air quality and precise decision-making. Failing to consider every relevant factor causes poor accuracy and bad decision-making. To improve the accuracy for predicting the concentration of $PM_{2.5}$ or $PM_{10}$ particles, we extract the relevant data from meteorological datasets and road traffic datasets. To achieve this, we implemented a deep autoencoder to encode information from such features. This autoencoder serves as a neural network with one hidden layer, which sets the output value equal to the input. This encoder compresses the input into the hidden layer, then reconstructs the output from that display. Autoencoder is an unsupervised model consisting of two steps: encoding and decoding. This keystroke enables this neural network to learn more abstract features unsupervised. To predict $PM_{2.5}$ or $PM_{10}$ particles, we present a vector representation for the traffic and meteorological data.

#### 3.4.3. BiLSTM Layers for Spatiotemporal Features Representation

Statistical-based approaches such as ARIMA and traditional learning models do poorly in predictive tasks because they dismiss the long-term dependence on time series data. The LSTM model is a specific RNN which was proposed by Hochreiter et al. [51]. This model has been found to be the best solution to address the problem of long-term information dependence on time series data. RNN networks are based on a chained network module. In a standard RNN, this repeating module has a very basic structure, such as a tanh layer. LSTM networks have the same structure, but the core of the structure differs from standard RNN. As shown in Figure 9, LSTM networks have four modules playing different roles. The core of an LSTM is composed of an input gate $i_t$, an output gate $o_t$, and a forgetting gate $f_t$.

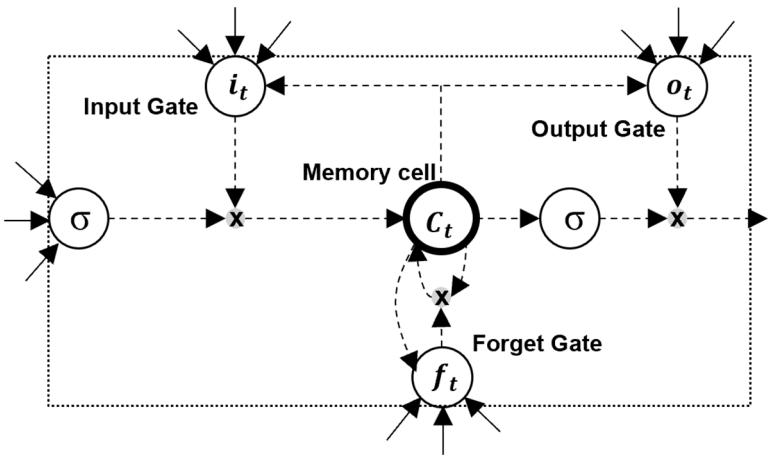



**Figure 9.** LSTM model representation.

The input gate decides the information that needs to be stored or updated in the cell. The input gate has a further two layers: the sigmoid layer responsible for deciding the value needed to be stored and the tanh layer responsible for building a new candidate value vector $\widetilde{c}_t$ and insert it in the state. The input gate is based on the following equation [50]:

$$i_t = \sigma(W_i \cdot [x_t\ h_{t-1}] + b_i) \tag{8}$$

The forgetting gate decides what information must be deleted from the cellular state at $t-1$ time with $h_{t-1}$ as the output value. It is based on the following equation [50]:

$$f_t = \sigma\left(W_f \cdot [x_t\ h_{t-1}] + b_f\right) \tag{9}$$

In this equation, $W_f$ is the weight matrix and $b_f$ is the bias matrix of the forget gate. $h_{t-1}$ is the historical data, σ represents the activation function, and $x_t$ represents the current input of new data and determines which previous information is discarded. After getting the coefficient's values of the input gate and the forget gate, the following equation [50] will be used to update current cell state:

$$c_t = f_t \circ c_{t-1} + i_t \circ \widetilde{c}_t \tag{10}$$

$$\widetilde{c}_t = tanh(W_c x_t + b_c h_{t-1}) \tag{11}$$

The output gate determines the current stage of output information. The sigmoid layer of output gates decides the section of cell state that will be output. The obtained value will be between $-1$ and 1 through the tanh layer and multiplied with the value of the first step output. The output gate is based on the following equations [50]:

$$o_t = \sigma(W_o + x_t h_{t-1} + b_o) \tag{12}$$

$$h_t = o_t \circ \tan \mathrm{h}(c_t) \tag{13}$$

A disadvantage of the traditional LSTM model is that it is only operating in one way and may discard important information when extracting deep significant features. BiLSTM, on the other hand, is able to process time series data in a bidirectional way simultaneously. This leads to deeper functions for better features extraction and prediction. BiLSTM is formed with a combination of forward and backwards operations for LSTM (as shown in Figure 10). The sequence of information processing is performed under the forward state. The forward state LSTM output is completely isolated with the input of backward LSTM and vice versa. The BiLSTM gathers hidden LSTM states of opposite directions to the same output represented as follows [13]:

$$y_t = \sigma\left(W \circ \left[\overrightarrow{h_t}, \overleftarrow{h_t}, x_t\right] + b\right) \tag{14}$$

The output layer receives information from both future and past states based on this architecture. The hidden states of the forward and backward layers are measured using the following equations:

$$h_f = o_f \circ \tan \mathrm{h}\left(c_f\right) \tag{15}$$

$$h_b = o_b \circ \tan \mathrm{h}(c_b) \tag{16}$$

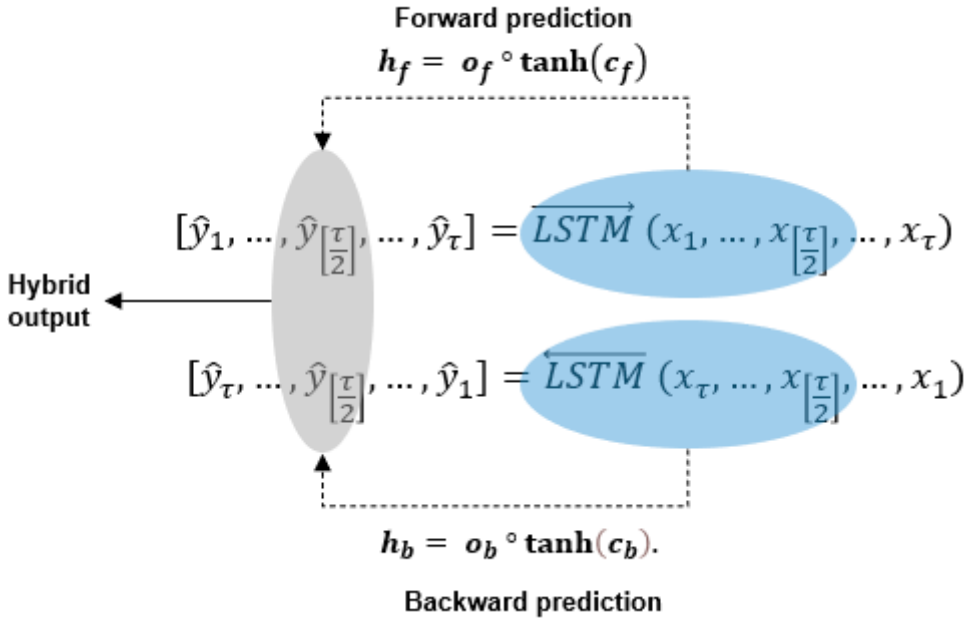



**Figure 10.** BiLSTM model representation.

3.4.4. Attention Layers

An attention layer is responsible for assigning weights to sample data and extracting essential information which can impact the features analysis and forecasting process. This process provides a better estimate of the final output and reduces the resource consumption of the computer. Moreover, it is also responsible for decreasing the redundant and noisy data while assigning them weights during the sampling and training phases. This mechanism is used to remember the long source of input. The attention layer creates the context vector, a weighted combination of input states to remember the important sequence while training. These weighted input states are different from the autoencoder model, which uses the context vector of the last hidden state. Therefore, the context vector can process the entire input sequence, thus eliminating the problem of forgetting. The Figure 11 shows the bidirectional LSTM architecture.

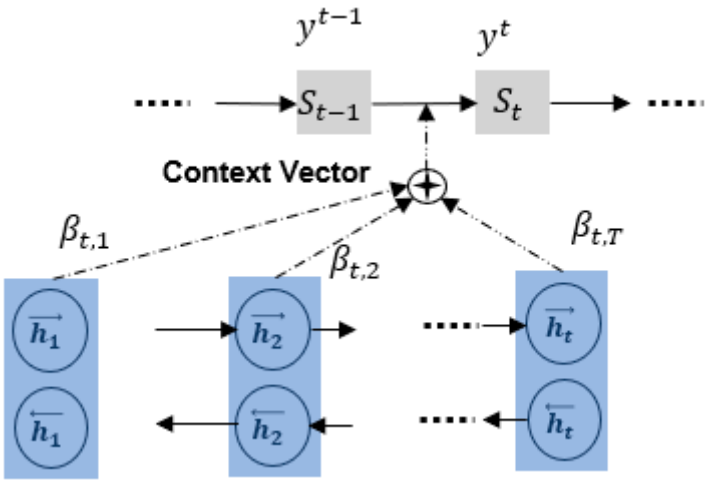



**Figure 11.** Attention-based BiLSTM with context vector.

Consider a scoring function f defined: $f : \mathbb{R}^m \times \mathbb{R}^m \mapsto \mathbb{R}$, which determines the assigned weight to its vector and indicates the important vectors. Therefore, the context

vector ($c_t$) is a weighted sum of hidden states H = {$h_1$, $h_2$, . . . , $h_{t-1}$} and represents the important information for the current time step as depicted in the following equations [52]:

$$c_t = \sum_{i=1}^{t-1} \beta_i h_i \tag{17}$$

$$\beta_i = \frac{\exp(f(h_i, h_t))}{\sum_{j=1}^{t-1} \exp(f(h_j, h_t))} \tag{18}$$

$$h_i = \left[ \overrightarrow{h_i^T} ; \overleftarrow{h_i^T} \right]^T, \; i = 1, \ldots, n \tag{19}$$

The context vector is then unified with the previous hidden state unit $S_{t-1}$ and previous target output $y_{t-1}$ to predict the current hidden state $S_t$ as depicted in the following equation [52]:

$$S_i = g(S_{i-1}, y_{i-1}, C_i) \tag{20}$$

#### 3.4.5. Distributed Training Approach

Research lacks evidence towards a deep distributed learning approach for air quality prediction, especially with training time and memory consumption investigations. We propose a novel approach by using distributed deep learning data parallelism to address these issues. It is achieved by distributing air quality traffic information and historical meteorological data across multiple training workers. Figure 12 shows the proposed training process.

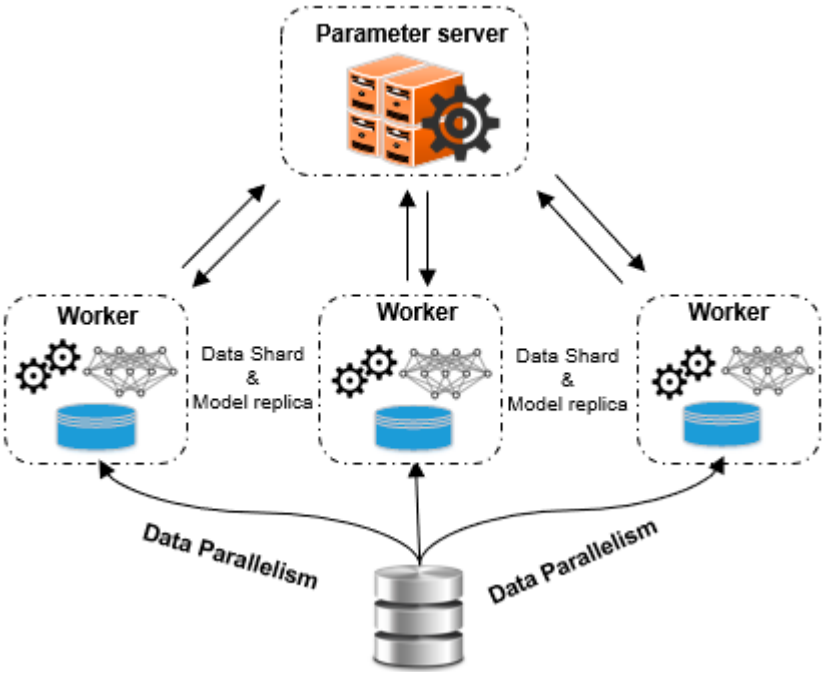



**Figure 12.** Proposed training process.

We implemented a distributed training process called data parallelization with $n$ worker machines to mitigate the training time and achieve significant computing performance. It helped us to achieve our training goals more quickly by processing n different partitions in the parallel data training approach. We distributed $n$ copies of the model over $n$ processing workers in this setup. The machines are trained locally under parallel mode to reduce the overall training time. A parameter server is used to collect the updates and request parameters from each training station.

## 4. Experimental Results and Discussion

In the following, we demonstrate the usefulness and the efficiency of the proposed deep learning framework by conducting extensive experiments.

### *4.1. Experiments Environment Setup*

This section unravels the hardware and software environment of the experiments. The detailed configuration is presented in Tables 3 and 4.

**Table 3.** Server's settings.

| Components | Specs |
|---|---|
| Server CPU | AMD Ryzen 7 2700X 3.7 GHz 8 Cores |
| Graphic Cards | NVIDIA GeForce RTX 2080 |
| Memory | 32 GB |

**Table 4.** Data distribution.

| Data Split | Distribution |
|---|---|
| Training | 70% |
| Validation | 20% |
| Testing | 10% |

The proposed model has been built using python programming language. TensorFlow, a deep learning library, was used to implement our learning model and the other deep learning state-of-the-art algorithms. The library PySpark was used to create a distributed computing system. We made a comparative study between our model and seven state-of-the-art algorithms presented in Table 5.

**Table 5.** Baseline Models.

| Algorithms | Description |
|---|---|
| RNN [53] | Recurrent neural networks are a type of neural network that used previous outputs as inputs while having hidden states. They are mostly used in times series forecasting, speech recognition, and natural language processing. |
| LSTM models [15] | LSTM models are a special type of RNN that overcome the long-term dependency problem faced by standard RNN. |
| Autoencoder [54] LSTM | The autoencoder LSTM is a hybrid model which is implemented with an LSTM encoder and decoder for sequence data. This model has the same structure setup as an autoencoder but is composed of several LSTM layers. |
| Convolutional LSTM [17] | This model is a hybrid framework based on CNN and LSTM models. |
| ConvBiLSTM autoencoder [14] | This model is based on a convolutional BiLSTM concatenated with an autoencoder model. |
| Attention LSTM [40] | This algorithm is a dual LSTM model with attention mechanism. |
| Attention CNN-LSTM [55] | This model combines a one-dimensional convolutional neural network, LSTM network, and attention-based network. |

In this experiment, the performance evaluations were carried out by calculating the Mean Absolute Error (*MAE*), Root Mean Square Error (*RMSE*), and Symmetric mean absolute percentage Error (*SMAPE*) [56].

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{21}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{22}$$

$$SMAPE = \frac{1}{2n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{(|y_i| + |\hat{y}_i|)} \times 100\% \tag{23}$$

However, our evaluations were divided into two steps. The first step was the training performed in a centralized deep learning environment. This approach was similar to other literature studies and evaluated by comparing it with other state-of-the-art models. The Algorithm 2 shows the steps of phase 1 experimentations.

**Algorithm 2:** Centralized training process

**Input**: Historical Data
**Output**: Error Rates
1. data ← MissingValuesFunc (Input)
2. data ← NormalizationFunc (data)
3. learning rate $\alpha$ = $\mathbf{10^{-3}}$
4. **Initialize** F(x) = Proposed model for N pollutants
5. **Split** data into train, test and validation sets
6. **For** epoch = 1 to N do
7. proposed_model ← fit_model (train, epoch, learningRate)
**8. For** t ← 1 . . . T **do**
9. Receive instance $\boldsymbol{x_t}$
10. $\hat{\boldsymbol{y}}_{\boldsymbol{t}}$ ← forecast_model (f_model, $\boldsymbol{x_t}$)
11. $\boldsymbol{y_t} \leftarrow \boldsymbol{test_y}$
12. Error_rate = evaluate_fit ($\boldsymbol{y_t}$, $\hat{\boldsymbol{y}}_{\boldsymbol{t}}$)
13. **RETURN** Error_rate
**END**

We used a distributed training environment called data parallelism to train our model in the second step. In this step, we employed ten worker machines to train ten different partitions of the air quality, meteorological, gas, and road traffic data in parallel. Each worker held a replica of the proposed algorithm. We then utilized a parameter server to aggregate the model's updates and parameter requests coming from each worker.

### *4.2. Experiment Results*

#### 4.2.1. Step 1

We employed four input encoding layers and four decoding layers with Relu as the activation function. The deep learning autoencoder and decoder have several hidden layers; for autoencoder, the hidden layers were {128, 100, 64,32} and, for the decoder, they were {32, 64,100, 128}. The traffic and meteorological data were used to pre-train the algorithm; the resulting data was a compressed hidden layer of the encoder. To make the final prediction, this encoded layer of data was concatenated with the output layer of BiLSTM.

Tables 6 and 7 summarize the proposed model's comparative results with the different baseline algorithms. It is based on the observed $PM_{2.5}$ and $PM_{10}$ particles with the corresponding predicted values for 1 h, 2 h, 4 h, 8 h, 10 h, 12 h, 24 h, and 48 h time spans.

**Table 6.** Models' evaluation results for $PM_{2.5}$ forecasting.

| Models | Metrics | +1 h | +2 h | +4 h | +8 h | +10 h | +12 h | +24 h | +48 h |
|---|---|---|---|---|---|---|---|---|---|
| RNN | MAE | 8.33 | 9.49 | 11.61 | 15.87 | 20.67 | 23.50 | 26.41 | 30.08 |
| | RMSE | 12.17 | 13.94 | 16.78 | 21.18 | 25.43 | 30.08 | 34.59 | 37.21 |
| | SMAPE | 25.11 | 28.54 | 29.91 | 36.11 | 39.09 | 41.17 | 44.96 | 45.78 |
| LSTM | MAE | 7.96 | 9.15 | 10.78 | 14.38 | 18.21 | 19.98 | 25.51 | 29.11 |
| | RMSE | 11.31 | 13.90 | 15.18 | 16.87 | 23.14 | 26.71 | 31.18 | 36.92 |
| | SMAPE | 24.87 | 26.47 | 28.39 | 34.17 | 35.29 | 39.49 | 43.42 | 46.03 |
| Autoencoder LSTM | MAE | 6.83 | 8.96 | 9.97 | 12.24 | 15.86 | 18.76 | 22.29 | 25.87 |
| | RMSE | 8.94 | 11.87 | 13.83 | 15.79 | 18.48 | 23.44 | 27.53 | 30.21 |
| | SMAPE | 22.39 | 25.57 | 28.09 | 32.93 | 34.28 | 35.96 | 39.49 | 44.27 |
| CNN+LSTM | MAE | 6.03 | 8.21 | 10.01 | 12.18 | 14.76 | 17.38 | 20.53 | 24.79 |
| | RMSE | 8.21 | 10.75 | 12.64 | 15.37 | 17.98 | 21.89 | 25.27 | 29.63 |
| | SMAPE | 21.08 | 24.12 | 27.72 | 30.33 | 33.09 | 35.09 | 37.89 | 40.99 |
| Attention LSTM | MAE | 5.83 | 8.10 | 9.91 | 12.05 | 14.53 | 17.30 | 20.45 | 24.14 |
| | RMSE | 8.14 | 10.43 | 12.37 | 15.14 | 17.51 | 21.38 | 24.80 | 29.10 |
| | SMAPE | 20.16 | 23.53 | 26.19 | 30.02 | 33.05 | 35.57 | 37.48 | 41.05 |
| ConvBiLSTM autoencoder | MAE | 5.34 | 7.97 | 9.87 | 11.62 | 14.21 | 16.99 | 19.84 | 23.96 |
| | RMSE | 7.92 | 9.90 | 12.18 | 15.11 | 16.89 | 20.34 | 24.12 | 28.17 |
| | SMAPE | 19.27 | 23.37 | 25.33 | 29.74 | 33.14 | 36.15 | 38.90 | 40.17 |
| Attention CNN-LSTM | MAE | 5.21 | 7.15 | 9.27 | 10.99 | 14.10 | 16.13 | 19.07 | 22.63 |
| | RMSE | 7.40 | 9.61 | 11.94 | 14.92 | 16.32 | 19.89 | 23.93 | 28.05 |
| | SMAPE | 18.15 | 21.24 | 25.09 | 28.13 | 30.24 | 33.98 | 38.22 | 39.77 |
| **Proposed Model** | **MAE** | **5.02** | **6.96** | **8.59** | **10.97** | **13.87** | **15.87** | **18.75** | **22.59** |
| | **RMSE** | **7.48** | **9.53** | **11.89** | **14.28** | **16.21** | **19.53** | **23.61** | **28.02** |
| | **SMAPE** | **17.98** | **20.91** | **24.57** | **27.91** | **30.28** | **34.49** | **37.18** | **39.81** |

**Table 7.** Models' evaluation results for $PM_{10}$ forecasting.

| Models | Metrics | +1 h | +2 h | +4 h | +8 h | +10 h | +12 h | +24 h | +48 h |
|---|---|---|---|---|---|---|---|---|---|
| RNN | MAE | 12.20 | 15.75 | 19.63 | 24.73 | 26.21 | 29.87 | 35.59 | 37.19 |
| | RMSE | 15.38 | 19.32 | 22.68 | 25.41 | 29.37 | 34.63 | 39.44 | 43.28 |
| | SMAPE | 29.74 | 33.52 | 36.37 | 40.08 | 42.19 | 45.60 | 46.32 | 48.17 |
| LSTM | MAE | 11.86 | 13.09 | 16.75 | 18.67 | 21.49 | 24.84 | 27.18 | 31.49 |
| | RMSE | 14.56 | 17.56 | 19.36 | 22.49 | 25.68 | 29.09 | 33.18 | 35.85 |
| | SMAPE | 27.41 | 30.25 | 34.34 | 37.58 | 40.18 | 45.02 | 46.82 | 47.32 |
| Autoencoder LSTM | MAE | 11.21 | 13.25 | 16.54 | 19.05 | 21.53 | 23.31 | 26.54 | 30.97 |
| | RMSE | 13.67 | 16.53 | 18.11 | 21.25 | 24.57 | 27.33 | 31.85 | 34.37 |
| | SMAPE | 25.89 | 28.09 | 31.52 | 35.47 | 37.69 | 40.08 | 44.16 | 45.99 |
| CNN+LSTM | MAE | 9.95 | 12.38 | 15.49 | 18.96 | 21.78 | 23.07 | 25.46 | 29.60 |
| | RMSE | 11.53 | 13.56 | 17.18 | 20.31 | 23.45 | 27.27 | 30.79 | 32.40 |
| | SMAPE | 23.96 | 26.54 | 28.49 | 30.08 | 33.17 | 35.53 | 38.28 | 41.72 |
| Attention LSTM | MAE | 9.17 | 12.05 | 14.21 | 17.31 | 21.11 | 22.35 | 25.30 | 28.37 |
| | RMSE | 11.36 | 13.33 | 16.47 | 19.87 | 22.69 | 27.04 | 29.76 | 32.11 |
| | SMAPE | 22.84 | 26.27 | 27.81 | 30.93 | 32.92 | 35.29 | 38.05 | 40.84 |

**Table 7.** *Cont.*

| Models | Metrics | +1 h | +2 h | +4 h | +8 h | +10 h | +12 h | +24 h | +48 h |
|---|---|---|---|---|---|---|---|---|---|
| ConvBiLSTM autoencoder | MAE | 8.29 | 11.25 | 13.85 | 16.67 | 19.07 | 21.48 | 25.07 | 27.39 |
| | RMSE | 11.27 | 13.96 | 16.34 | 18.75 | 21.49 | 26.07 | 28.96 | 31.09 |
| | SMAPE | 22.79 | 25.38 | 27.75 | 31.96 | 32.51 | 34.09 | 37.69 | 39.83 |
| Attention CNN-LSTM | MAE | 8.15 | 10.19 | 11.69 | 15.23 | 17.43 | 19.39 | 23.42 | 26.39 |
| | RMSE | 9.81 | 12.93 | 14.70 | 17.82 | 20.28 | 22.71 | 24.80 | 25.93 |
| | SMAPE | 20.48 | 23.37 | 25.97 | 29.05 | 30.97 | 30.69 | 33.13 | 34.33 |
| **Proposed Model** | **MAE** | **7.38** | **9.57** | **11.19** | **13.96** | **14.37** | **18.56** | **20.74** | **25.89** |
| | **RMSE** | **9.71** | **12.27** | **14.68** | **16.74** | **17.82** | **20.33** | **23.66** | **25.09** |
| | **SMAPE** | **19.57** | **23.17** | **24.99** | **27.07** | **29.49** | **30.28** | **32.99** | **34.26** |

- ***$PM_{2.5}$* forecasting results analysis:** The experimental results show that our proposed framework outperformed the baseline models both on short- and long-term predictions. As can be seen, for the forecasting results based on both 1 h (short-term) and 48 h (long-term) time steps, the proposed attention-based model obtained the lowest MAE (5.02 and 22.59, respectively, for short-term and long-term predictions), RMSE (7.48 and 28.02) and SMAPE (17.98 and 39.81) among all the models. This indicates strong agreement between observed and predicted values. The recurrent neural network had the lowest prediction accuracy both on short- and long-term predictions. It had the highest MAE, RMSE, and SMAPE values, which were, respectively, 8.33, 12.17, and 25.11 for the short-term prediction and 30.08, 37.21, and 45.78 for the long-term prediction. All hybrid LSTM based models performed better than the standard LSTM, which recorded 7.96, 11.31, and 24.87, respectively, for MAE, RMSE, and SMAPE evaluation metrics for the short-term prediction, and 29.11, 36.92, and 46.03 for the long-term prediction. Among the hybrid LSTM baseline models, the attention-based CNN-LSTM recorded the second lowest error rates after the proposed approach; these second lowest error rates were 5.21, 7.40, and 18.15 for MAE, RMSE, and SMAPE for the short-term prediction and 22.63, 28.05, and 39.77 for the long-term prediction. The ConvBiLSTM autoencoder model came right after as it obtained 5.34, 7.92, and 19.27 for MAE, RMSE, and SMAPE for the short-term prediction and 23.96, 28.17, and 40.17 for the long-term prediction. The ConvBiLSTM autoencoder model was followed by the attention-based LSTM algorithm with 5.83, 8.14, and 20.16 for the short-term prediction and 24.14, 29.10, and 41.05 for the long-term prediction.
- ***$PM_{10}$* forecasting results analysis:** As depicted in Table 7, our proposed model still performs better than the baseline models both on short- and long-term forecasting. It achieved the minimal MAE (7.38 and 28.89, respectively, for 1 h and 48 h future time steps), RMSE (9.71, and 25.09, respectively, for 1-h and 48-h future time steps), and SMAPE (19.57, and 34.26, respectively, for 1-h and 48-h future time steps). On the other hand, the recurrent neural network captured the highest error rates, which were 12.20, 15.38, and 29.74, respectively, for the MAE, RMSE, and SMAPE metrics through 1h time step prediction and 37.19, 43.28, and 48.17 for the 48h time step prediction. The Attention CNN-LSTM model has the second-best result, as it recorded 8.15, 9.81, and 20.48, respectively, for the MAE, RMSE, and SMAPE metrics on the 1h time step prediction. For the 48 h time step, it achieved 26.39 for MAE, 25.93 for RMSE, and 34.33 for SMAPE. The ConvBiLSTM autoencoder recorded 8.29, 11.27, and 22.79 for MAE, RMSE, and SMAPE on 1h time step prediction, and on 48 h time step, it achieved 27.39, 31.09, and 39.83 for the same evaluation metrics. The other hybrid LSTM baseline models have similar results as they both outperformed the standard LSTM model on both 1 h (short-term) and 48 h (long-term) time steps.

The following images shown in Figure 13 illustrate how the proposed model outperformed all baseline algorithms through the eight different future time steps. In Figure 13,

we have graphed each evaluation metric (MAE, RMSE, and SMAPE) for each short- and long-term prediction of both particles $PM_{2.5}$ and $PM_{10}$. As shown in the figure, the performances of $PM_{2.5}$ and $PM_{10}$ long-term predictions are significantly lower than that of the short-term predictions. As the forward prediction size increases, the forecasting performances of these models gradually decrease. Nonetheless, even with long-term forecasting, the proposed model is still better than the baseline methods. This is mainly because the attention mechanism improves the long-term prediction performance of $PM_{2.5}$ and $PM_{10}$.

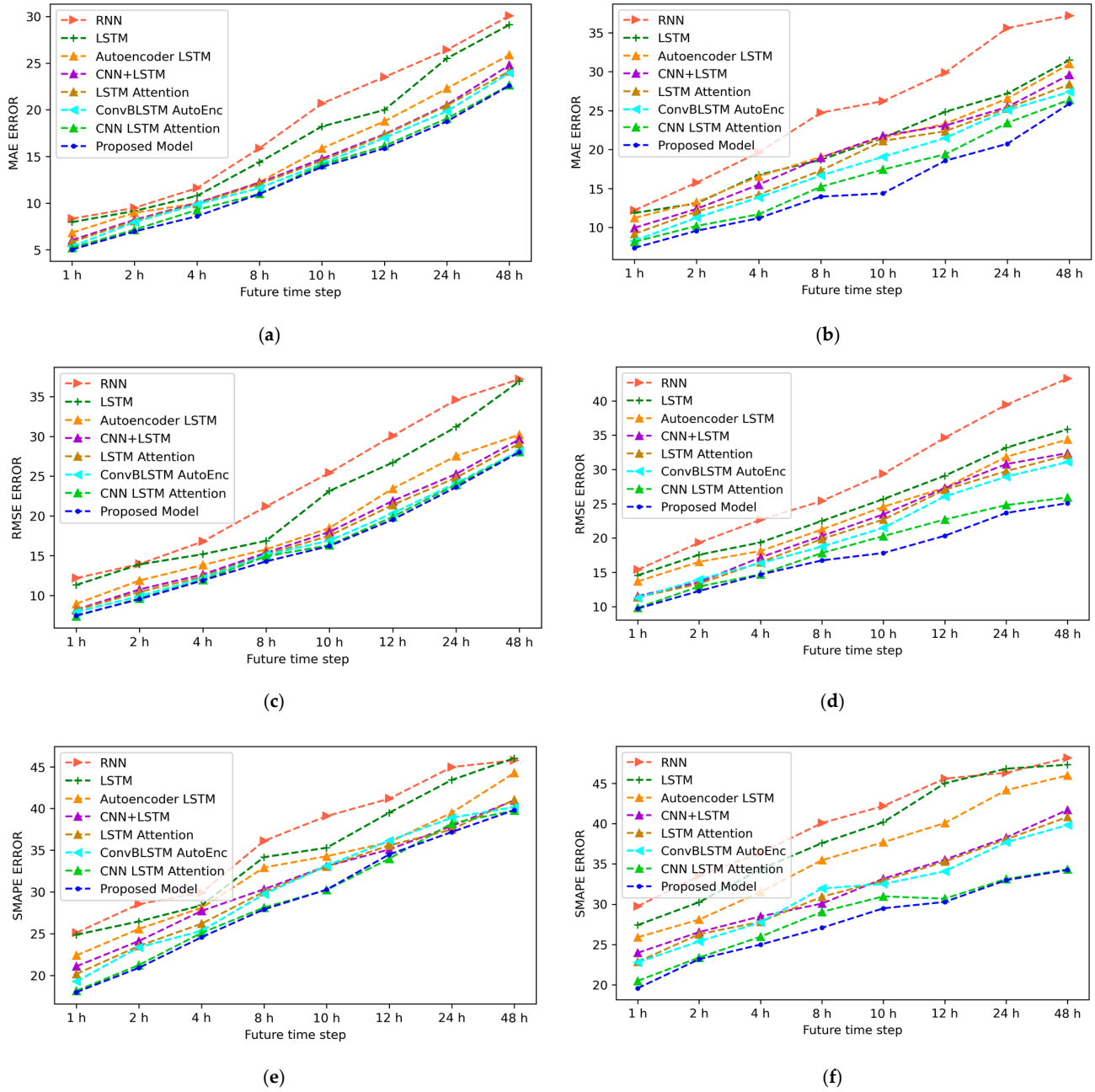



**Figure 13.** Comparison of evaluation metrics for different future time steps. (**a**) MAE comparison at different future time steps for $PM_{2.5}$ prediction. (**b**) MAE comparison at different future time steps for $PM_{10}$ prediction. (**c**) RMSE comparison at different future time steps for $PM_{2.5}$ prediction. (**d**) RMSE comparison at different future time steps for $PM_{10}$ prediction. (**e**) SMAPE comparison at different future time steps for $PM_{2.5}$ prediction. (**f**) SMAPE comparison at different future time steps for $PM_{10}$ prediction.

A detailed comparison between the proposed deep learning air quality model was conducted with baseline models of RNN, LSTM, LSTM autoencoder, CNN+LSTM, Attention-based LSTM, ConvBiLSTM autoencoder, and attention-based CNN LSTM to forecast air quality particles of $PM_{2.5}$ and $PM_{10}$. The factual results are presented in Figures 14 and 15 as the comparison between all models to predict the air quality values for $PM_{2.5}$ and $PM_{10}$. The x-coordinates represent observation time steps, and y-coordinates represent the $PM_{2.5}$ values in Figure 14 and $PM_{10}$ values in Figure 15. We decided to focus on 1-h short-term prediction only. As represented in these figures, the results show that the prediction pattern of the proposed deep learning air quality model is similar to the real pattern of time series data for both particles.

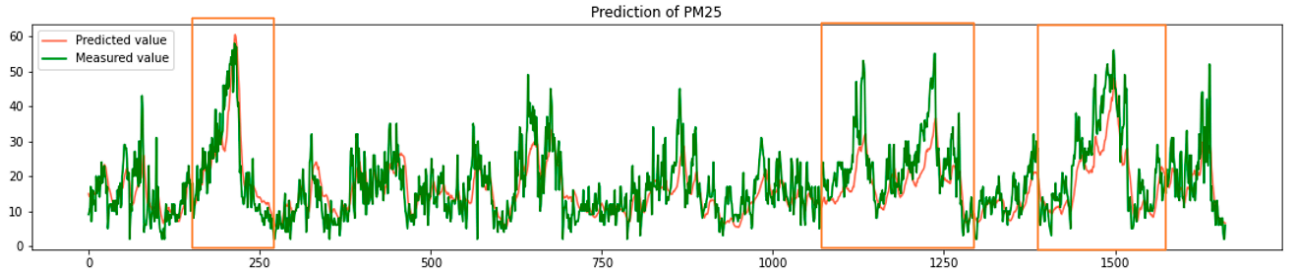
The proposed model performed better in forecasting as it predicted the pollution particles accurately and quickly, verifying the feasibility of our algorithm to capture the temporal variations, whereas other algorithms, especially RNN, were unable to track to the trends of $PM_{2.5}$ and $PM_{10}$. The attention-based models also produced good forecasting trends. In this experiment, the wave peak period highlighted the difference of prediction and ground truth pattern more accurately.

Moreover, we evaluated the training time of all models as shown in Figure 16, and we found that the proposed framework had not only the best accuracy compared to the baseline models but it also had the best training average times alongside the Attention LSTM and the ConvBiLTM autoencoder model, as they all averaged 310/s during the training process for the prediction of both particles $PM_{2.5}$ and $PM_{10}$.

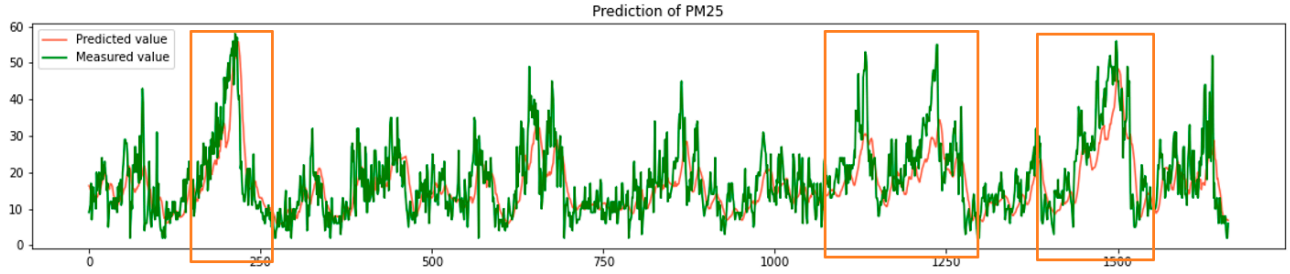
The RNN model took the longest training time with 482 s, followed by the autoencoder LSTM with 479 s. The standard LSTM and the convolutional LSTM models had almost the same average training time. In step 2 we will show how the distributed architecture can reduce the training time of the proposed model and be further optimized to perform better.

#### 4.2.2. Step 2

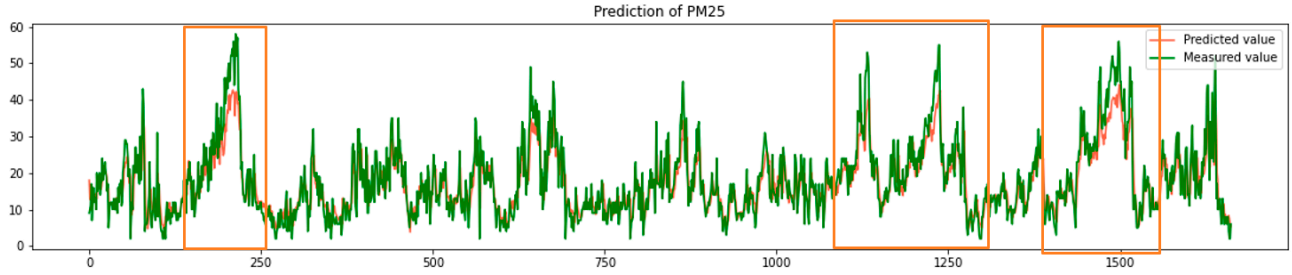
We distributed the architecture and employed ten specific training workers to perform the task to achieve better performance. Each training worker was provided with the same model parameters, and the algorithm was performing with the same capacity. We intended to predict the same particles using the same data as step 1. For this step, we only focused on the 1 h, 24 h, and 48 h future time step predictions to evaluate how the proposed model will perform both on short- and long-term predictions while being trained using ten worker nodes. For short-term prediction, as shown in Figure 17a,b, the proposed model performed the best for both particles $PM_{2.5}$ and $PM_{10}$ when it is using five and six workers, respectively, during the training process. As depicted in that figure, at five workers for $PM_{2.5}$ forecasting, we can see how the error rates decrease suddenly. The same observation can be seen at six workers during $PM_{10}$ forecasting. In Figure 17c,d, which correspond to t+24 h future time step forecasting, the proposed model achieved its best performance for both particles $PM_{2.5}$ and $PM_{10}$ while training it with seven workers. And finally, in Figure 17e,f, which represent to t + 48 h future time step prediction, the best performance results for both particle forecasting was recorded while using eight workers. The shape changes in the rate error are mostly due to the fact that the model had some difficulty generalizing well with a smaller number of training workers, and it was not stable at the beginning of the training. In further research, we are planning to investigate in depth all the potential causes of such changes, which may include the utilization of some optimization techniques. This experiment concluded that the right number of workers for the training of these two air quality particles should be between five and eight nodes.



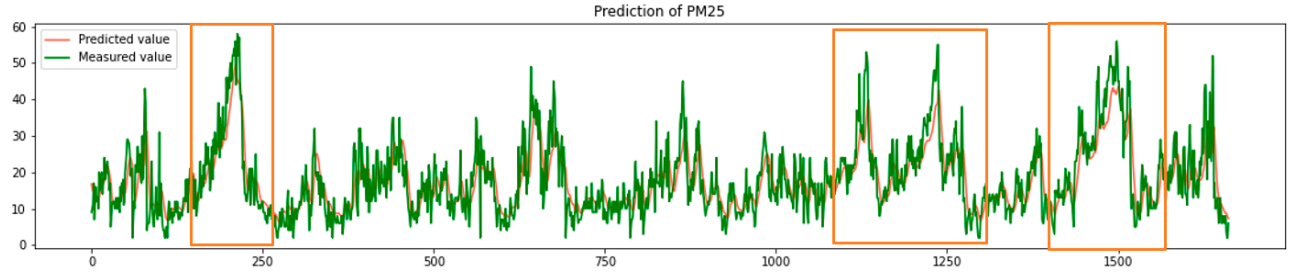

(**a**)



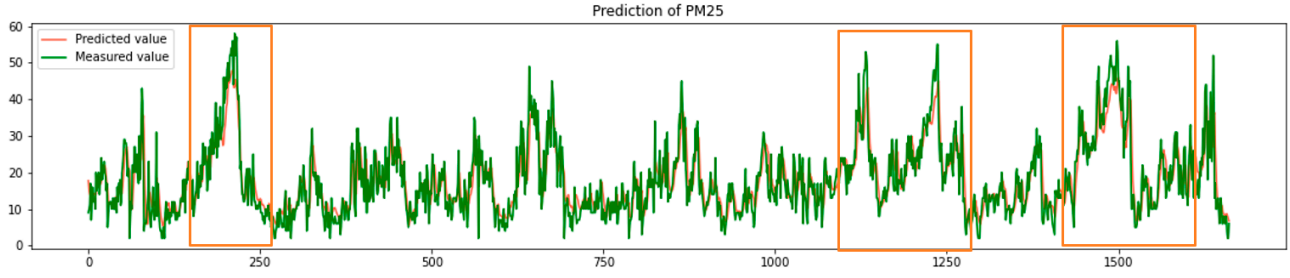

(**b**)




(**c**)

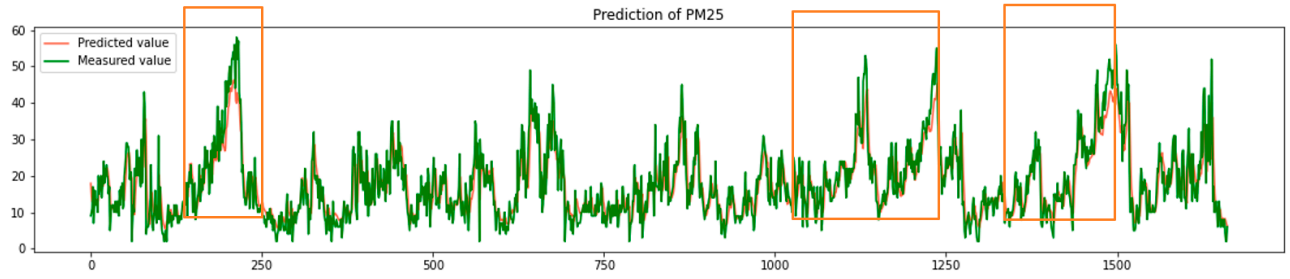



(**d**)

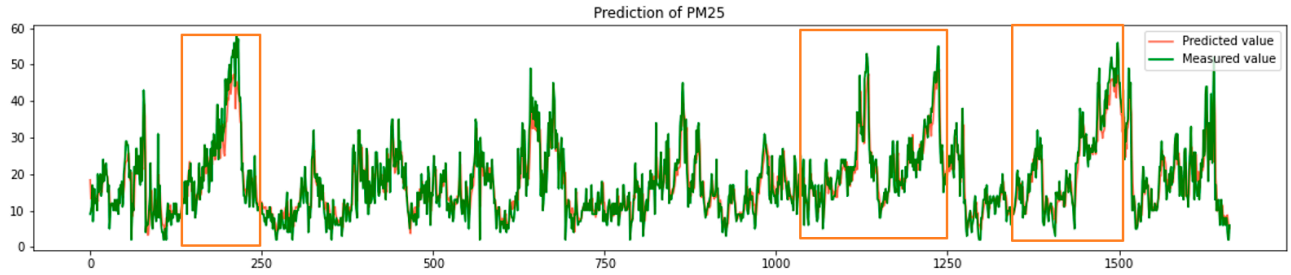



(**e**)

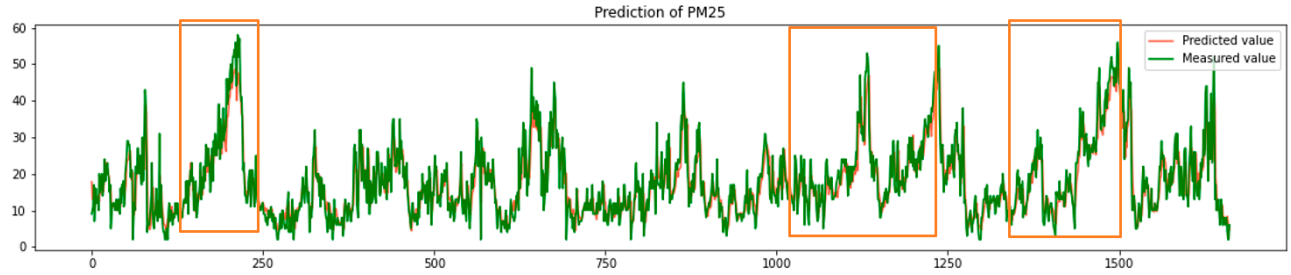
**Figure 14.** *Cont.*




(**f**)

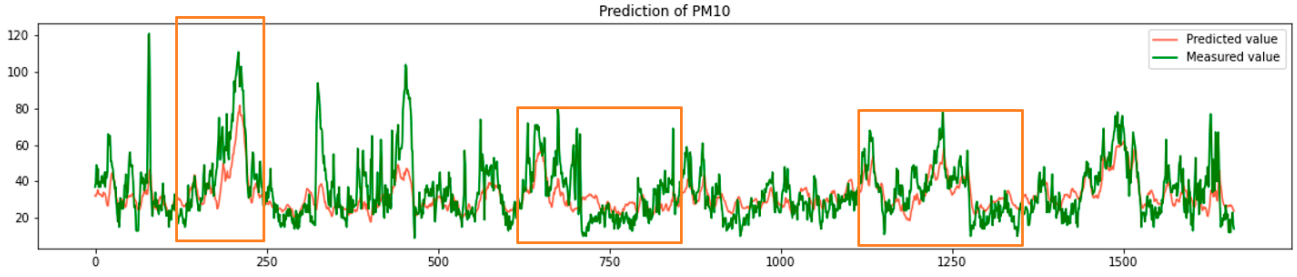



(**g**)



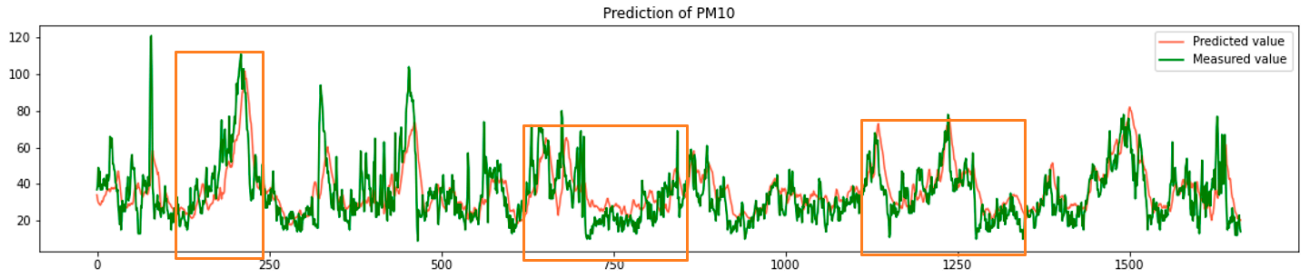

(**h**)

**Figure 14.** Prediction of $PM_{2.5}$. (**a**) RNN result. (**b**) LSTM result. (**c**) Autoencoder LSTM result. (**d**) ConvLSTM result. (**e**) Attention LSTM result. (**f**) ConvBiLSTM result. (**g**) Attention CNN LSTM result. (**h**) Proposed model result.

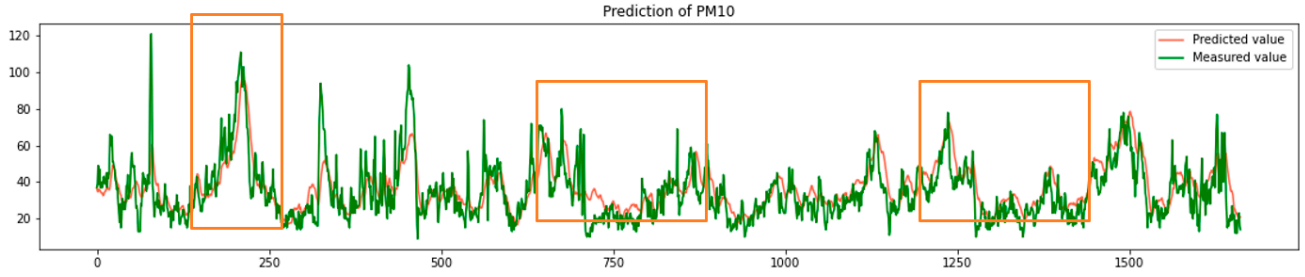



(**a**)

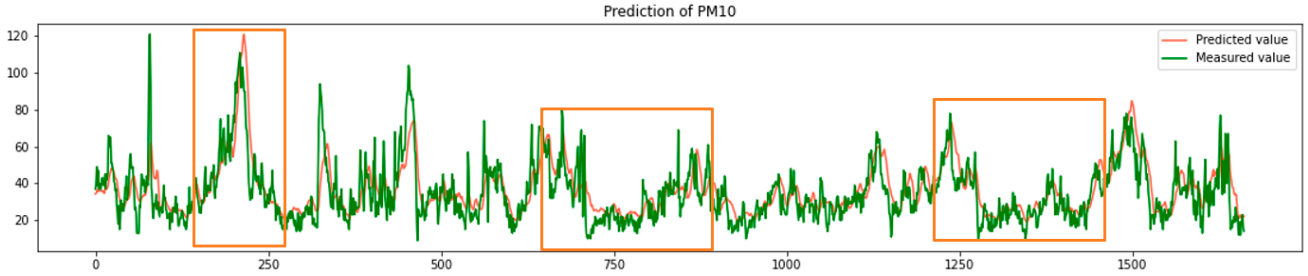
**Figure 15.** *Cont.*



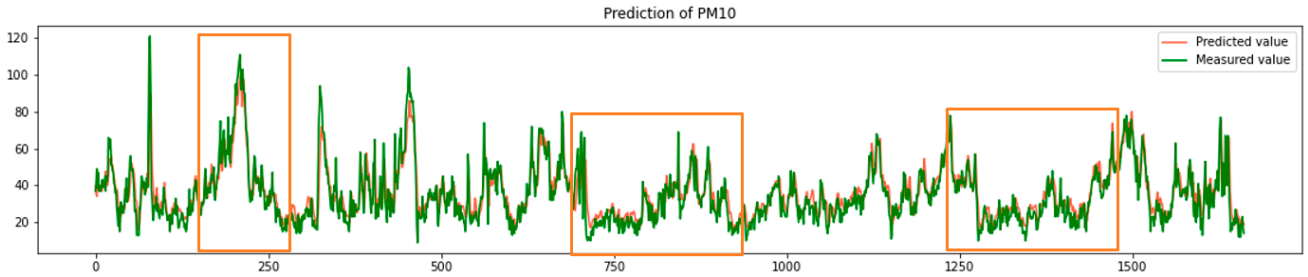

(**b**)



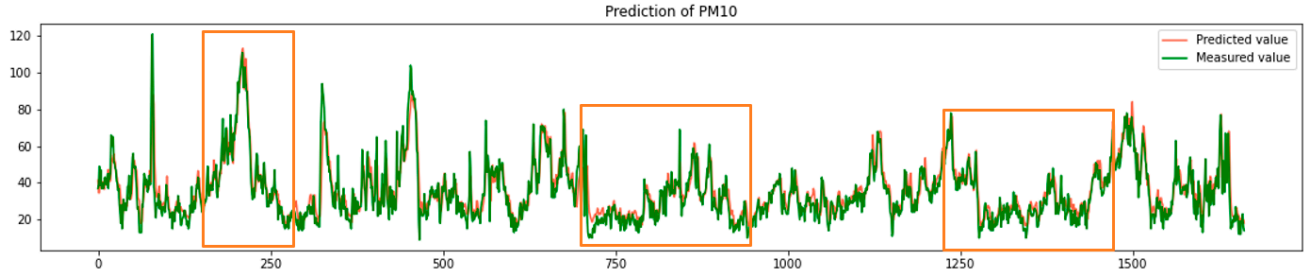

(**c**)




(**d**)

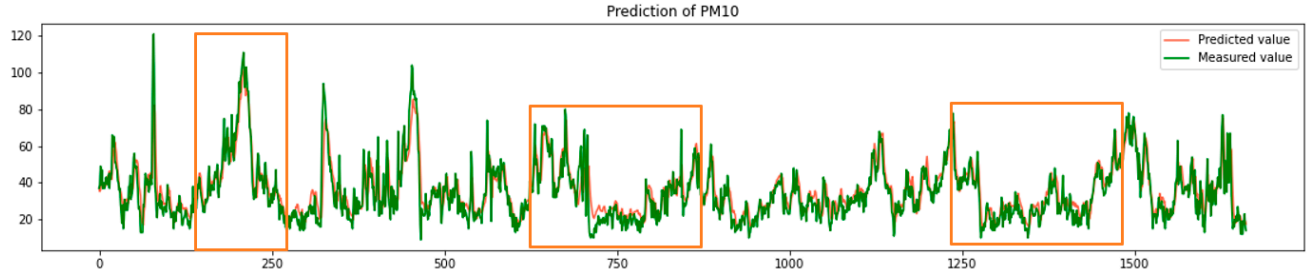



(**e**)

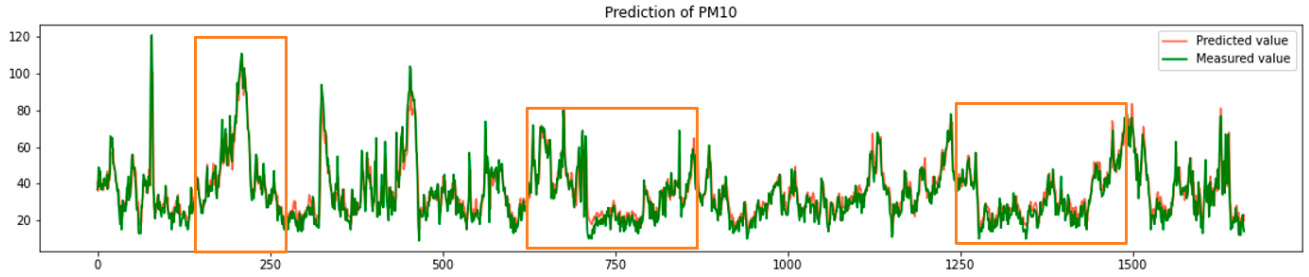



(**f**)

**Figure 15.** *Cont.*




(**g**)




(**h**)

**Figure 15.** Prediction of $PM_{10}$. (**a**) RNN result. (**b**) LSTM result. (**c**) Autoencoder LSTM result. (**d**) ConvLSTM result. (**e**) Attention LSTM result. (**f**) ConvBiLSTM result. (**g**) Attention CNN LSTM result. (**h**) Proposed model result.

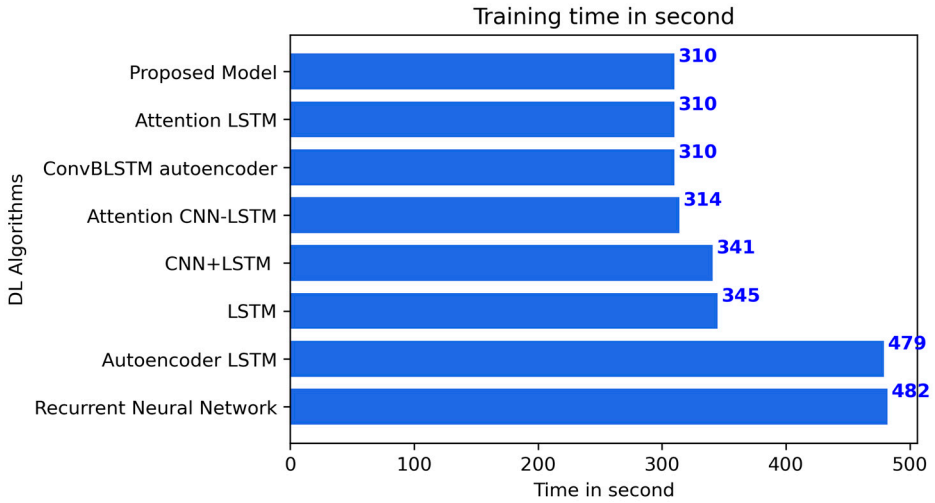



**Figure 16.** Training time evaluation.

We were interested in evaluating the reduction in training time with the introduction of distributed architecture and parallel training sessions. As shown in Figure 18, the training time of our framework decreased while the number of workers increased. At five training workers, we reached 250 s, and the training time increased slightly at six workers to 260 s, then decreased again to the lowest recorded time at eight workers, which was 200 s, and, from eight to 10, it was consistent. Compared to the centralized training in step 1, the training time in phase 2 almost decreased twice, making our method more effective and less time-consuming.

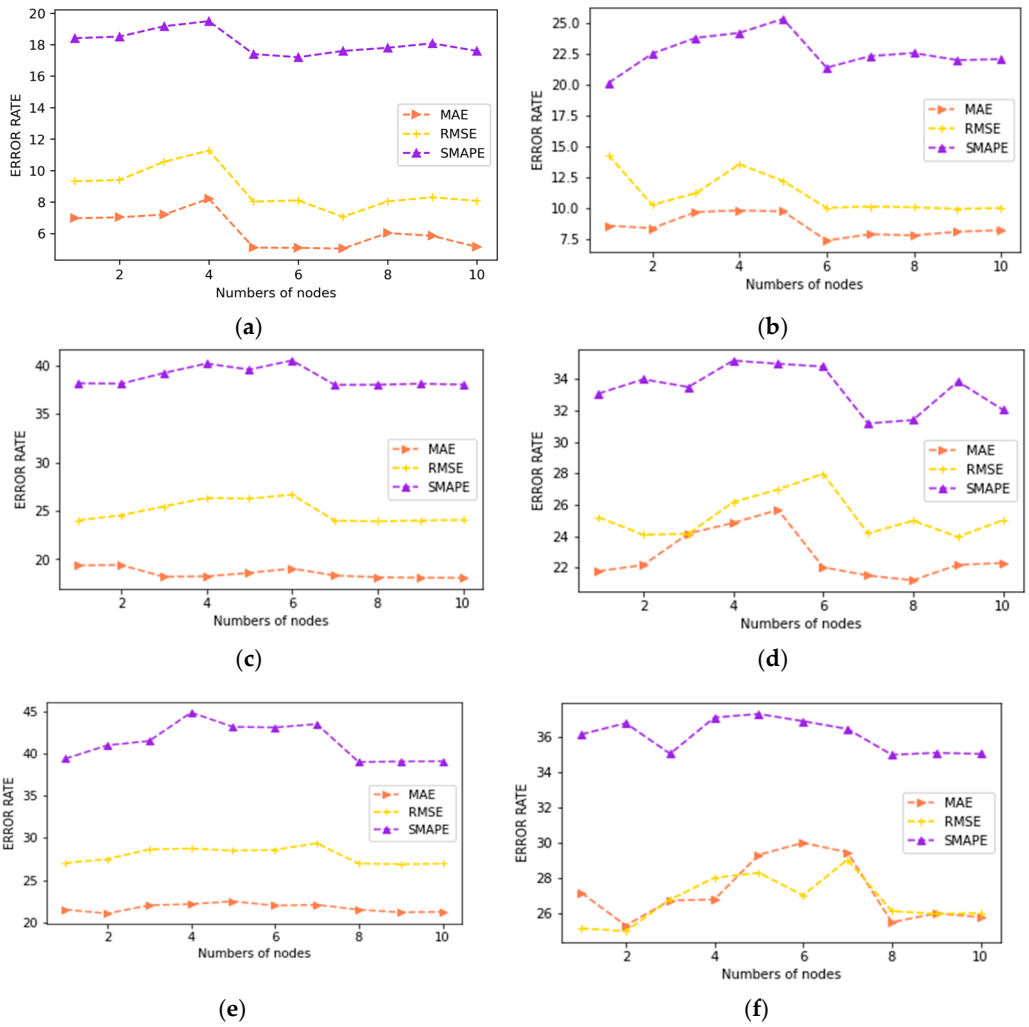



**Figure 17.** Evaluation of the proposed model during distributed training over 10 workers at t + 1 h, 24 h, and 48 h future step prediction. (**a**) Error rate at t + 1 h future time step ($PM_{2.5}$). (**b**) Error rate at t + 1 h future time step ($PM_{10}$ ). (**c**) Error rate at t + 24 h future time step ($PM_{2.5}$ ). (**d**) Error rate at t + 24 h future time step ($PM_{10}$ ). (**e**) Error rate at t + 48 h future time step ($PM_{2.5}$ ). (**f**) Error rate at t + 48 h future time step ($PM_{10}$ ).

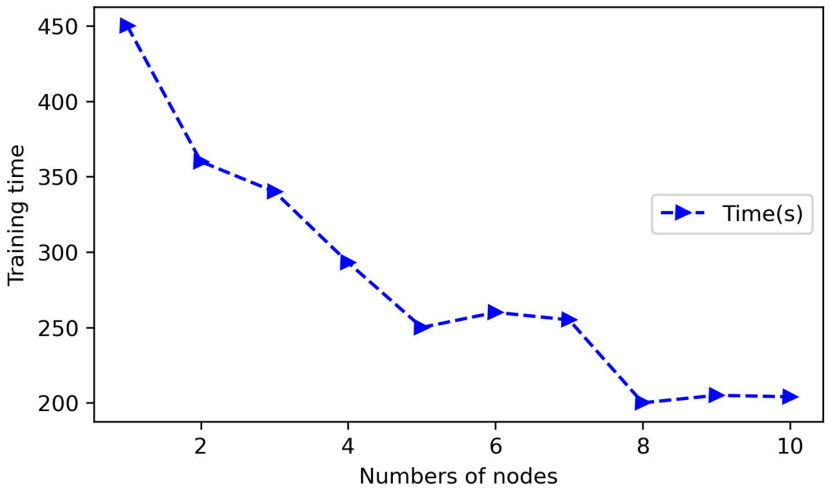



**Figure 18.** Training time of the proposed model through 10 workers.

### *4.3. Proposed Model Deployment and Online Inference*

With the goal of starting to use the proposed distributed deep learning model for practical decision-making, it needs to be effectively deployed into production. To do so, we created an online inference pipeline based on stream data, cloud server, and GPU instances, as depicted in Figure 19.

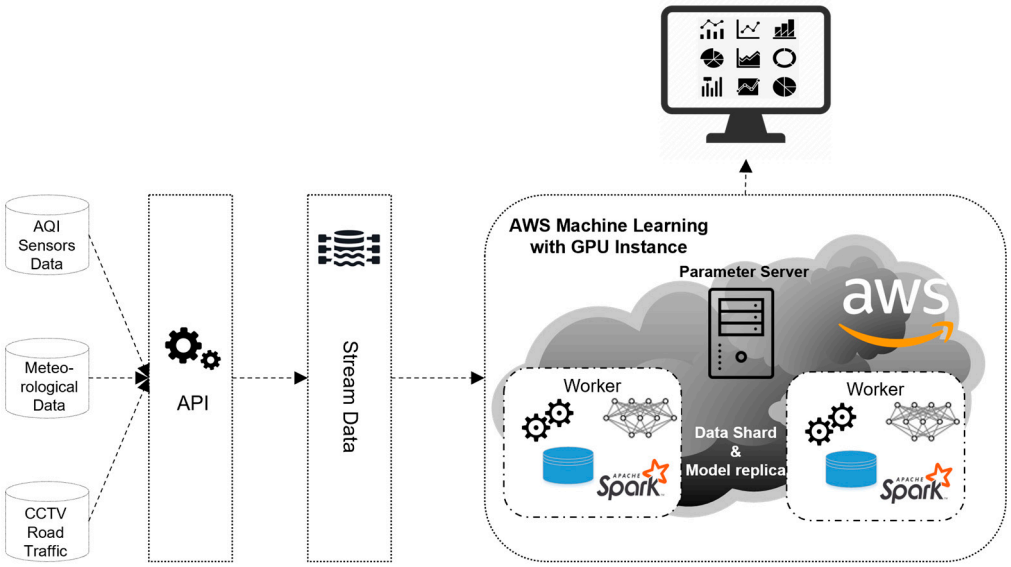



**Figure 19.** Proposed model deployment pipeline.

- **Input Stream Data**: The air quality, meteorological, and traffic data are collected every hour and then preprocessed and stored inside cloud servers.
- **Online Worker**: Since we decided to deploy our distributed model based on data parallelism, we have created several data pipelines based on Apache Spark that represent our workers. These training workers will have a model replica and a partition of the stream data. The workers will train their local replica by using the assigned data partition.
- **Parameter Server**: The parameter server is responsible for aggregating model updates and parameter requests coming from different online workers.
- **User Interface:** After the distributed training, the final model is then used to make the predictions every hour for the next 48 h and show it on the web application for the user to observe the predicted values as presented in Figure 20a. The web application also offers the option to upload a csv file containing the meteorological and road traffic features to predict air quality. This would help the government and the citizens to take all the necessary precautions to either control air pollution or protect themselves from the harmful effects of bad air quality.


(**a**) (**b**)

**Figure 20.** Online prediction results: (**a**) 18 h real-time prediction. (**b**) Comparison between predicted values and ground truth.

## 5. Conclusions

This study proposed a new attention-based convolutional BiLSTM autoencoder model for air quality prediction. The approach utilized historical and time series data for traffic information, as well as encoded meteorological information in order to perform predictions of air quality at various time points. Car traffic data was collected by implementing a computer vision-based model called YOLO with DeepSort algorithm. Since the DeepSort algorithm could not track the approaching objects (cars) on the road properly if the boundary box was smaller, a virtual baseline or reference line was applied to the tracking algorithm. If the center of the bounding box indicated the vehicle was located below the reference line, the count increased. Multiple counting is prevented by keeping the previous bounding box's index number in a temporal array and comparing it with the new frame's bounding boxes. The proposed deep learning approach performed optimally in different atmospheric conditions, i.e., stable and unstable. This was achieved by using a two-stage feature extraction method. The first feature extraction stage was achieved by using a stacked autoencoder to extract information on several features, including pollution, traffic, and various meteorological properties. During the second stage, an attention-based convolutional BiLSTM layer extracted deep spatial correlation features from air quality particles data and performed the final prediction after concatenation with the encoded features from stage 1. The attention mechanism was used to capture the dynamic correlation of both $PM_{2.5}$ and $PM_{10}$ particles between previous and future time points. A distributed training method, data parallelism, was adopted to train the proposed model in the second phase of the experiments. It used a coordinator server that successfully aggregated the updates on the model and the parameter requests of working nodes. The experimental results showed that the proposed deep learning model outperformed the classical state-of-the-art deep learning architectures. For $PM_{2.5}$ prediction, the model obtained the lowest MAE (5.02 and 22.59, respectively, for the 1-h short term and 48-h long-term predictions), RMSE (7.48 and 28.02) and SMAPE (17.98 and 39.81). The proposed model also achieved the best results for $PM_{10}$ prediction with the minimum MAE (7.38 and 28.89), RMSE (9.71 and 25.09), and SMAPE (19.57 and 34.26). This indicates a strong agreement between observed and predicted values. For both benchmarks of short-term and long-term predictions, our proposed

model performed better. In addition, the distributed training architecture significantly reduced the training time. It is a helpful step towards resources saving, more optimal training, smart training of algorithms, and more precise and accurate prediction. With improved training time, different organizations and governments can mitigate air pollution levels in a timely manner and take the proper steps to fix problems with air pollution.

**Author Contributions:** A.G.M.M. investigated the theoretical basis for this work, implemented the deep learning models for the experiments as well as the proposed approach, and wrote the manuscript. E.P. and J.J. collected the data used in the experiments section and made some adjustments in the data pre-processing part. Y.Y. supervised the project and helped to write the manuscript. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the Korea Agency for Infrastructure Technology Advancement (KAIA) grant funded by the Ministry of Land, Infrastructure and Transport (Grant 21CTAP-C152124-03), and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1A2C1095635).

**Informed Consent Statement:** Informed consent was obtained from all subjects involved in the study.

**Data Availability Statement:** Data is available at: https://www.airkorea.or.kr/index%20/ (accessed on 19 December 2021); https://www.weather.go.kr/w/index.do (accessed on 19 December 2021).

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Dong, M.; Yang, D.; Kuang, Y.; He, D.; Erdal, S.; Kenski, D. $PM_{2.5}$ concentration prediction using hidden semi-Markov model-based times series data mining. *Expert Syst. Appl.* **2009**, *36*, 9046–9055. [CrossRef]
2. Nimesh, R.; Arora, S.; Mahajan, K.K.; Gill, A.N. Predicting air quality using ARIMA, ARFIMA and HW smoothing. *Model Assist. Stat. Appl.* **2014**, *9*, 137–149. [CrossRef]
3. Liu, B.; Jin, Y.; Xu, D.; Wang, Y.; Li, C. A data calibration method for micro air quality detectors based on a LASSO regression and NARX neural network combined model. *Sci. Rep.* **2021**, *11*, 21173. [CrossRef] [PubMed]
4. Sayegh, A.S.; Munir, S.; Habeebullah, T.M. Comparing the Performance of Statistical Models for Predicting $PM_{10}$ Concentrations. *Aerosol Air Qual. Res.* **2014**, *14*, 653–665. [CrossRef]
5. Wang, W.; Men, C.; Lu, W.-Z.J. Online prediction model based on support vector machine. *Neurocomputing* **2008**, *71*, 550–558. [CrossRef]
6. Martínez-España, R.; Bueno-Crespo, A.; Timón, I.; Soto, J.; Muñoz, A.; Cecilia, J.M. Air-pollution prediction in smart cities through machine learning methods: A case of study in Murcia, Spain. *J. Univers. Comput. Sci.* **2018**, *24*, 261–276.
7. Ameer, S.; Shah, M.A.; Khan, A.; Song, H.; Maple, C.; Islam, S.U.; Asghar, M.N. Comparative Analysis of Machine Learning Techniques for Predicting Air Quality in Smart Cities. *IEEE Access* **2019**, *7*, 128325–128338. [CrossRef]
8. Wang, X. Deep Learning in Object Recognition, Detection, and Segmentation Foundations and Trends R in Signal Processing. *Signal Process.* **2014**, *8*, 217–382. [CrossRef]
9. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 10778–10787. [CrossRef]
10. Zhang, Y.; Qin, J.; Park, D.S.; Han, W.; Chiu, C.C.; Pang, R.; Le, Q.V.; Wu, Y. Pushing the Limits of Semi-Supervised Learning for Automatic Speech Recognition. *arXiv* **2020**, arXiv:2010.10504. Available online: http://arxiv.org/abs/2010.10504 (accessed on 19 December 2021).
11. Li, Y.; Wu, C.Y.; Fan, H.; Mangalam, K.; Xiong, B.; Malik, J.; Feichtenhofer, C. Improved Multiscale Vision Transformers for Classification and Detection. *arXiv* **2021**, arXiv:2112.01526. Available online: http://arxiv.org/abs/2112.01526 (accessed on 22 December 2021).
12. Liu, M.; Zeng, A.; Xu, Z.; Lai, Q.; Xu, Q. Time Series is a Special Sequence: Forecasting with Sample Convolution and Interaction. *arXiv* **2021**, arXiv:2106.09305. Available online: http://arxiv.org/abs/2106.09305 (accessed on 22 December 2021).
13. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
14. Mengara, A.M.; Kim, Y.; Yoo, Y.; Ahn, J. Distributed Deep Features Extraction Model for Air Quality Forecasting. *Sustainability* **2020**, *12*, 8014. [CrossRef]
15. Zhao, J.; Deng, F.; Cai, Y.; Chen, J. Long short-term memory—Fully connected (LSTM-FC) neural network for $PM_{2.5}$ concentration prediction. *Chemosphere* **2019**, *220*, 486–492. [CrossRef] [PubMed]
16. Qi, Y.; Li, Q.; Karimian, H.; Liu, D. A hybrid model for spatiotemporal forecasting of $PM_{2.5}$ based on graph convolutional neural network and long short-term memory. *Sci. Total Environ.* **2019**, *664*, 1–10. [CrossRef] [PubMed]
17. Wen, C.; Liu, S.; Yao, X.; Peng, L.; Li, X.; Hu, Y.; Chi, T. A novel spatiotemporal convolutional long short-term neural network for air pollution prediction. *Sci. Total Environ.* **2018**, *654*, 1091–1099. [CrossRef] [PubMed]

18. Huang, C.-J.; Kuo, P.-H. A Deep CNN-LSTM Model for Particulate Matter ($PM_{2.5}$) Forecasting in Smart Cities. *Sensors* **2018**, *18*, 2220. [CrossRef]
19. Bai, Y.; Li, Y.; Zeng, B.; Li, C.; Zhang, J. Hourly $PM_{2.5}$ concentration forecast using stacked autoencoder model with emphasis on seasonality. *J. Clean. Prod.* **2019**, *224*, 739–750. [CrossRef]
20. Heydari, A.; Majidi, M.; Davide, N.; Garcia, A.; Keynia, F.; de Santoli, L. Air pollution forecasting application based on deep learning model and optimization algorithm. *Clean Technol. Environ. Policy* **2022**, *24*, 607–621. [CrossRef]
21. Wang, B.; Yan, Z.; Lu, J.; Zhang, G.; Li, T. Deep Multi-task Learning for Air Quality Prediction. *Lect. Notes Comput. Sci. Incl. Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinform.* **2018**, *11305*, 93–103. [CrossRef]
22. Xiao, F.; Yang, M.; Fan, H.; Fan, G.; Al-Qaness, M.A.A. An improved deep learning model for predicting daily $PM_{2.5}$ concentration. *Sci. Rep.* **2020**, *10*, 20988. [CrossRef] [PubMed]
23. Chang, Y.-S.; Chiao, H.-T.; Abimannan, S.; Huang, Y.-P.; Tsai, Y.-T.; Lin, K.-M. An LSTM-based aggregated model for air pollution forecasting. *Atmos. Pollut. Res.* **2020**, *11*, 1451–1463. [CrossRef]
24. Arsov, M.; Zdravevski, E.; Lameski, P.; Corizzo, R.; Koteli, N.; Gramatikov, S.; Mitreski, K.; Trajkovik, V. Multi-Horizon Air Pollution Forecasting with Deep Neural Networks. *Sensors* **2021**, *21*, 1235. [CrossRef] [PubMed]
25. Guo, C.; Liu, G.; Chen, C.-H. Air Pollution Concentration Forecast Method Based on the Deep Ensemble Neural Network. *Wirel. Commun. Mob. Comput.* **2020**, *2020*, 8854649. [CrossRef]
26. Heess, N.; Graves, A. Recurrent Models of Visual Attention. *Adv. Neural Inf. Processing Syst.* **2014**, *27*, 1–12.
27. Larochelle, H.; Hinton, G. Learning to combine foveal glimpses with a third-order Boltzmann machine. In Proceedings of the Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010, Vancouver, BC, Canada, 6–9 December 2010.
28. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015; pp. 1–15.
29. Cao, J.; Chen, Q.; Guo, J.; Shi, R. Attention-guided Context Feature Pyramid Network for Object Detection. *arXiv* **2020**, arXiv:2005.11475. Available online: https://arxiv.org/abs/2005.11475 (accessed on 17 December 2021).
30. Sinha, A.; Dolz, J. Multi-scale guided attention for medical image segmentation. *arXiv* **2019**, arXiv:1906.02849, 1–10. Available online: https://arxiv.org/abs/1906.02849 (accessed on 17 December 2021).
31. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference, 2–9 February 2021; pp. 1–15.
32. Lin, J.; Su, Q.; Yang, P.; Ma, S.; Sun, X. Semantic-Unit-Based Dilated Convolution for Multi-Label Text Classification. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018. [CrossRef]
33. Dynamic Memory Networks for Visual and Textual Question Answering. *PMLR* **2015**, *48*, 2397–2406.
34. Chaudhari, S.; Mithal, V.; Polatkan, G.; Ramanath, R. An Attentive Survey of Attention Models. *ACM Trans. Intell. Syst. Technol.* **2021**, *12*, 1–32. [CrossRef]
35. Lee, J.B. Attention Models in Graphs: A Survey. *ACM Trans. Knowl. Discov. Data* **2019**, *13*, 1–25. [CrossRef]
36. Wang, F. Survey on the Attention Based RNN Model and Its Applications in Computer Vision, Computer Vision and Pattern Recognition. Available online: https://doi.org/10.48550/arXiv.1601.06823 (accessed on 16 December 2021).
37. Dairi, A.; Harrou, F.; Khadraoui, S.; Sun, Y. Integrated Multiple Directed Attention-Based Deep Learning for Improved Air Pollution Forecasting. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–15. [CrossRef]
38. Zou, X.; Zhao, J.; Zhao, D.; Sun, B.; He, Y.; Fuentes, S. Air Quality Prediction Based on a Spatiotemporal Attention Mechanism. *Mob. Inf. Syst.* **2021**, *2021*, 6630944. [CrossRef]
39. Liu, B.; Yan, S.; Li, J.; Qu, G.; Li, Y.; Lang, J.; Gu, R. An Attention-Based Air Quality Forecasting Method. In Proceedings of the 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, USA, 17–20 December 2018; pp. 728–733. [CrossRef]
40. Chen, H.; Guan, M.; Li, H. Air Quality Prediction Based on Integrated Dual LSTM Model. *IEEE Access* **2021**, *9*, 93285–93297. [CrossRef]
41. Chen, Z.; Yu, H.; Geng, Y.-A.; Li, Q.; Zhang, Y. EvaNet: An Extreme Value Attention Network for Long-Term Air Quality Prediction. In Proceedings of the 2020 IEEE International Conference on Big Data (Big Data), Atlanta, GA, USA, 10–13 December 2020; pp. 4545–4552. [CrossRef]
42. AIR KOREA. Available online: https://www.airkorea.or.kr/web (accessed on 30 July 2020).
43. Korea Meteorological Administration. Available online: https://web.kma.go.kr/eng/index.jsp (accessed on 30 July 2020).
44. Jocher, G. "YOLOv5," YOLOv5 Is a Family of Object Detection Architectures and Models Pretrained on the COCO Dataset, and Represents Ultralytics Open-Source Research into Future Vision AI Methods. Available online: https://github.com/ultralytics/yolov5 (accessed on 30 November 2021).
45. Huang, G. Missing data filling method based on linear interpolation and lightgbm. *J. Phys. Conf. Ser.* **2021**, *1754*, 012187. [CrossRef]
46. Normalization | Data Preparation and Feature Engineering for Machine Learning | Google Developers. Available online: https://developers.google.com/machine-learning/data-prep/transform/normalization (accessed on 10 February 2022).

47. Coefficient, C.; Two, B.; Variables, R. 5 Pearson Correlation Coefficient. *Noise Reduct. Speech Process.* **2009**, *2*, 1–2. [CrossRef]
48. Banfield, R.E.; Hall, L.O.; Bowyer, K.; Kegelmeyer, W. A Comparison of Decision Tree Ensemble Creation Techniques. *IEEE Trans. Pattern Anal. Mach. Intell.* **2006**, *29*, 173–180. [CrossRef] [PubMed]
49. Dietterich, T.G. An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. *Mach. Learn.* **2000**, *40*, 139–157. [CrossRef]
50. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
51. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
52. Brauwers, G.; Frasincar, F. A General Survey on Attention Mechanisms in Deep Learning. *IEEE Trans. Knowl. Data Eng.* **2021**, *4347*, 1–20. [CrossRef]
53. Athira, V.; Geetha, P.; Vinayakumar, R.; Soman, K.P. DeepAirNet: Applying Recurrent Networks for Air Quality Prediction. *Procedia Comput. Sci.* **2018**, *132*, 1394–1403. [CrossRef]
54. Cai, J.; Dai, X.; Hong, L.; Gao, Z.; Qiu, Z. An Air Quality Prediction Model Based on a Noise Reduction Self-Coding Deep Network. *Math. Probl. Eng.* **2020**, *2020*, 3507197. [CrossRef]
55. Li, S.; Xie, G.; Ren, J.; Guo, L.; Yang, Y.; Xu, X. Urban $PM_{2.5}$ Concentration Prediction via Attention-Based CNN–LSTM. *Appl. Sci.* **2020**, *10*, 1953. [CrossRef]
56. Pan, Q.; Darabos, C.; Moore, J. *Performance Evaluation: Metrics, Models and Benchmarks*; Springer: Berlin/Heidelberg, Germany, 2008; Volume 7246.