Neural Computing and Applications
https://doi.org/10.1007/s00521-022-07224-2

ORIGINAL ARTICLE

Estimation of missing air pollutant data using a spatiotemporal convolutional autoencoder

I Nyoman Kusuma Wardana[1,3] · Julian W. Gardner[1] · Suhaib A. Fahmy[1,2]

Received: 10 January 2022 / Accepted: 29 March 2022

Abstract
A key challenge in building machine learning models for time series prediction is the incompleteness of the datasets. Missing data can arise for a variety of reasons, including sensor failure and network outages, resulting in datasets that can be missing significant periods of measurements. Models built using these datasets can therefore be biased. Although various methods have been proposed to handle missing data in many application areas, more air quality missing data prediction requires additional investigation. This study proposes an autoencoder model with spatiotemporal considerations to estimate missing values in air quality data. The model consists of one-dimensional convolution layers, making it flexible to cover spatial and temporal behaviours of air contaminants. This model exploits data from nearby stations to enhance predictions at the target station with missing data. This method does not require additional external features, such as weather and climate data. The results show that the proposed method effectively imputes missing data for discontinuous and long-interval interrupted datasets. Compared to univariate imputation techniques (most frequent, median and mean imputations), our model achieves up to 65% RMSE improvement and 20–40% against multivariate imputation techniques (decision tree, extra-trees, *k*-nearest neighbours and Bayesian ridge regressors). Imputation performance degrades when neighbouring stations are negatively correlated or weakly correlated.

Keywords Missing data · Air pollutant · Spatiotemporal · Autoencoder · Convolutional layer

1 Introduction

Rising population, urbanisation, economic growth and industrial expansion have increased air pollution worldwide [1]. The main causes of air pollution are vehicle exhaust, industrial emissions, agricultural and natural disasters, such as volcanic eruptions and wildfires. These air pollutant sources can produce particulate matter (PM), nitrogen dioxide (NO_2), carbon monoxide (CO), ozone (O_3), sulfur dioxide (SO_2), among other pollutants [2]. The effect of air contaminants on the human body differs, depending on the type of contaminants and the level and duration of any exposure. It causes negative impacts on human health and influences socio-economic activities [3, 4]. Concerning human health, air pollution is associated with lung cancer [5, 6], cardiovascular diseases [7–9], impaired cognitive function and human emotion [10, 11]. Premature mortality, negative social and educational outcomes, adverse market liquidity and catastrophic climate are the socio-economic aspects triggered by air pollution [12]. Moreover, around 4.9 million deaths were attributed to air pollution in 2017 [13].

Measuring air pollution with potential exposures and health impacts can be more challenging when missing data occurs. The existence of missing data can influence study interpretations and conclusions [14] and affect the functioning of air quality-related public services [15]. Missing

These authors contributed equally to this work.

✉ I Nyoman Kusuma Wardana
Kusuma.Wardana@warwick.ac.uk

Julian W. Gardner
J.W.Gardner@warwick.ac.uk

Suhaib A. Fahmy
suhaib.fahmy@kaust.edu.sa

[1] School of Engineering, University of Warwick, Coventry CV4 7AL, UK

[2] Present Address: Computer, Electrical and Mathematical Sciences and Engineering, King Abdullah University of Science and Technology (KAUST), Thuwal 23955, Saudi Arabia

[3] Department of Electrical Engineering, Politeknik Negeri Bali, Badung 80364, Bali, Indonesia

Published online: 04 May 2022

data are a common problem in air pollutant measurement and other fields such as clinical, energy and traffic [16–18]. The cause of missing data may vary, including sensor malfunction, sensor sensitivity, power outages, computer system failure, routine maintenance, human error and other reasons [19, 20]. Depending on the causes, air pollution data can be missing either in long-consecutive periods or short intervals [21]. While routine maintenance and temporary power outages can cause short intervals of missing data, sensor malfunction and other critical failures can cause longer gaps in data collection.

According to Rubin, incomplete data are classified based on their generating mechanisms, namely missing completely at random (MCAR), missing at random (MAR) and missing not at random (MNAR) [22]. MCAR occurs when data are genuinely missing as a result of random events [23]. MCAR assumes that missing values are a random sample of observed values, which is restrictive [24]. In MAR, the probability of missingness may depend on observed data values but not those that are missing. Under MAR conditions, there is a possibility to retrieve the missing values from other observed predictor variables [23, 25]. When the probability of an observation being missing is dependent on unobserved values, such as the values of the observation themselves, this condition is called MNAR [22, 23, 26]. MNAR is nonignorable missingness and is considered a condition that yields biased parameter estimates [27]. Missing data are most often neither MCAR nor MNAR [26]. The missingness is at least MAR for air quality data. Even though the air contaminant values are missed for unknown reasons (i.e. MCAR), most missing values are caused by explainable circumstances such as routine maintenance, sensor malfunction and power outages [23, 28]. Thus, we assume MAR conditions for the air quality data used in this study.

There are two common ways to handle missing data: delete the missing parts and impute (substitute) the missing values [29]. The deletion method can be further defined as pairwise deletion and listwise deletion. The pairwise deletion method discards the specific missing values, whereas the listwise method removes the entire record even if there is one missing value. The MCAR assumption allows for the exclusion of incomplete observations to yield unbiased results. However, a higher level of missing values may reduce the precision of the analysis [24]. Moreover, because the nature of pollutant measurement generates time-series data, the deletion method could break the data structure, and valuable information may be lost. Contrary to the deletion method, the imputation method reconstructs the missing data based on available information [30].

Reconstruction techniques inspired by machine learning have been used in recovering corrupted data, one of which is the denoising autoencoder (DAE) [31]. Standard DAE and its variants are implemented in many fields, such as image denoising [32–35], medical signal processing [36, 37] and fault diagnosis [38, 39]. Some works also utilised DAE for missing data imputation. Gondara et al. [40] tried to answer the challenge of multiple imputation by employing an overcomplete representation of DAEs. The proposed method does not need complete observations for initial training, making it suitable for a real-life scenario. Abiri et al. [41] demonstrated the robustness of DAE in recovering a wide range of missing data for different datasets. Abiri et al. proved that the proposed stacked DAE outperformed other established methods, such as K-nearest neighbour (KNN), multiple imputation by chained equations (MICE), random forest and mean imputations. Jiang et al. [42] utilised DAE for imputing the missing traffic flow data and compared three different architectures composing the DAE, namely standard ("vanilla"), convolutional neural network (CNN) and bidirectional long short-term memory (Bi-LSTM). Jiang et al. evaluated the proposed model's test sets with a general missing rate of 30%. Moreover, splitting traffic data into weekdays and weekends significantly improved the model performances.

The following discussions summarise the recent challenges and breakthroughs in methods for air quality missing data imputation. First, the problem of missing data repeatedly occurs in environmental research, and more studies are required to find effective imputation solutions. Although various methods have been proposed to handle missing data in many fields, more studies addressing air quality missing data prediction are needed [19]. The works mentioned earlier in this section mainly focus on clinical, energy, traffic, etc. Second, most of the related studies focused on a small amount of missing data. Ma et al. stated that the previous works are applicable for short-interval missing imputation or consecutive missing value with a level of missingness less than 30%. This issue was also mentioned by Alamoodi et al. [43]. Few works investigated missing data at large percentages (i.e. more than 80%), either using deletion or imputation. Third, the multiple imputation method can improve imputation performance [14]. We consider that implementing multiple imputation for air quality data is a deserving attempt. Fourth, many studies demonstrated the robustness of denoising autoencoder in recovering noisy data. However, few studies implemented the denoising autoencoder for missing air quality data imputation. Finally, even though air pollutants strongly relate to spatiotemporal characteristics, these factors are rarely included in predicting the missing values of air pollution data. The air quality data collected from air

monitoring stations can hold intensely stochastic spatiotemporal correlations among them [44].

Inspired by the capabilities of the denoising autoencoder to reconstruct corrupted data, we propose an imputation method based on the denoising autoencoder. We implement multiple plausible estimates for specific missing values. We propose a simple method suitable for both short-interval and long-interval consecutive missing imputation and simultaneously offer multiple imputations to obtain less biased results. We use a convolutional denoising autoencoder with spatiotemporal considerations to extract the air pollutant features. The proposed method takes advantage of data from nearby stations to predict the missing data in the targeted station. This method does not need external features like weather and climate data (air temperature, humidity, wind speed, wind direction, etc.). Thus, our proposed method involves only the intended pollutant data from neighbouring stations. We propose a simple yet promising way to estimate missing values in real-world applications.

2 Method

2.1 Research framework in general

Our proposed method exploits data from nearby sites to enhance predictions at the target station with missing data. When a target station fails to gather pollutant data from the environment, the neighbouring station data can help to estimate the current loss of the target site. As illustrated in Fig. 1, S^3 fails to collect data and acts as a target station. Neighbouring stations S^2, S^5 and S^6 send their data to S^3. The participating neighbouring stations eligible to send

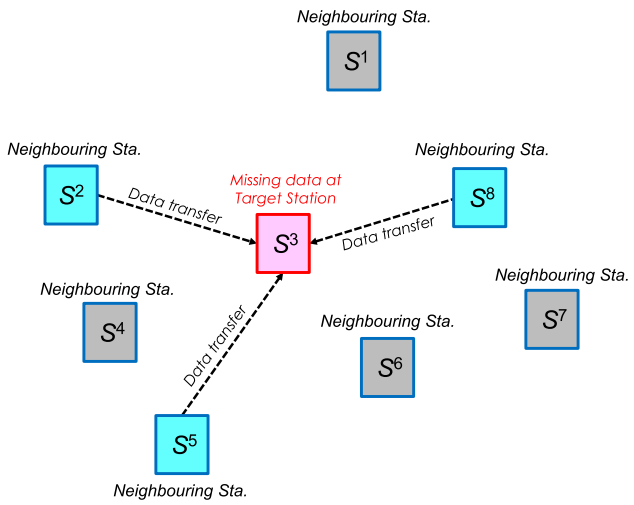

Fig. 1 Target station exploits data from neighbouring stations to impute the missing data

data are chosen based on their coefficient correlations with the target station. We implement a deep autoencoder model at S^3 and use a one-dimensional convolutional neural architecture to cover the spatiotemporal behaviour of pollutant data. Based on the collected spatiotemporal data at target and neighbouring stations, we predict the missing data at the target station.

Figure 2 shows the general research framework used in this study. There are seven main blocks, and each block consists of several tasks. The first block relates to the data sources used in this work. All data sources used in this study are available online, and they can be freely downloaded and used by adhering to the terms described in the given licences. A dataset contains different hourly air pollutant concentrations. Even though the dataset includes several air contaminants, we selected two attributes as the targeted pollutants. Ten monitoring stations are involved in the calculations to acquire the spatial characteristics of air pollutant data. Moreover, we verified our proposed method in three different air quality datasets to achieve less biased results. These are the monitoring of air quality in three major cities: London, Delhi and Beijing.

The data pre-processing in the second block is dedicated to examining the targeted air pollutant coefficient correlation among air monitoring stations. Calculating the coefficient correlation among pollutant concentrations is one of the main steps conducted in this study. For every target pollutant, we joined the same pollutant data taken from all locations into a single data frame and sorted them by the same hourly timestamp. We then calculated the correlation coefficient and selected the three highest correlations between the targeted and neighbouring monitoring stations. Based on these correlations, the data encompassing spatiotemporal characteristics are determined. The spatial behaviour is obtained using data from the targeted and three neighbouring stations (i.e. four monitoring stations in total). The temporal dependency is acquired by collecting the current value and its previously 7-hour values (i.e. 8-hour data in total).

The pre-processing procedure in the third block is carried out to make the spatiotemporal features suitable for the proposed deep learning model. All training and test features are normalised to values between 0 and 1, leading to the data variability reduction [45]. Additional pre-processing in this stage includes initialising missing data because the obtained datasets may contain some missing features. If missing data exist in the original dataset, only an unbroken series of data with a minimum of 1 week (168 hours) period is considered for the training set. We did not remove the remaining data but did not use them during training. Therefore, there are some chunks of unbroken data involved as inputs in the training phase. According to

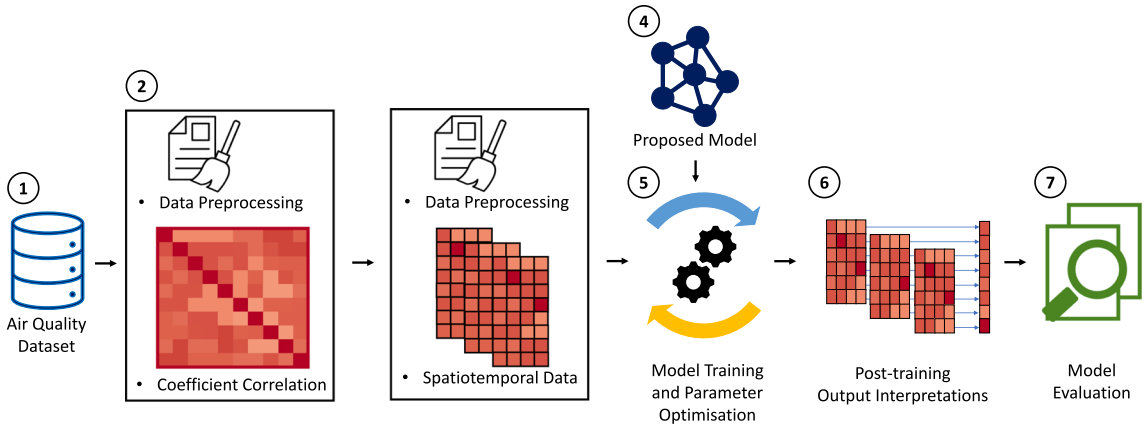

Fig. 2 General description of the research framework for data analysis

the number of data fragments, the training steps are done in multiple rounds. This step will maintain the temporal behaviour of the time-series data. This step maintains the temporal behaviour of the time-series data. Once we have a clean dataset, we artificially create random and consecutive missing data. The artificial missing values are filled with zeros. The final training and test sets are 3-dimensional matrices with the size of $(n \times 8 \times 4)$. The integer value of n indicates the number of training or test sets, 8 denotes the 8-hour observation period, and 4 denotes the number of features taken from four monitoring stations.

As indicated in the fourth block, we proposed a deep learning model to handle missing data. In this study, the proposed model architecture is a convolutional autoencoder, meaning that the autoencoder uses convolution layers as the encoding and decoding parts. The proposed convolutional autoencoder acts as a denoising model. By replacing some input features on purpose with zeros, the input sets can be seen as corrupted data, and the model learns to reconstruct these corrupted inputs by minimising the loss function. The training process is shown in the fifth block of the research framework.

The sixth and seventh blocks of the research framework are the post-training interpretation and evaluation steps. The model accepts and yields two-dimensional data, and thus post-training output interpretations are needed to find the intended prediction results. This process involves the aggregation procedure. Finally, some evaluation procedures are taken to examine the trained model, such as calculating error metrics, testing the model on different missing rates and locations, and implementing the proposed algorithm on other air quality datasets.

2.2 Description of the datasets

This study uses air quality datasets from three different cities. A total of 10 stations are selected for each city, and two pollutants per station are studied. We consider ten monitoring stations adequate for implementing our algorithm and evaluating its performance. We also vary the pollutant in each city to demonstrate that our proposed method can be applied to different pollutants. Some considerations are taken into account when selecting the stations. Availability of pollution data and measurement period for all stations are two of our major concerns. We included stations with at least three years data from the same period. Furthermore, since our method is based on the correlation coefficient between stations, we include stations with varying degrees of correlation.

The first dataset is air pollutant data of London city. The data were collected using the Openair tool [46]. Openair is an *R* package developed by Carslaw and Ropkins to analyse air quality data. For the London city dataset, we focus on two pollutants: nitrogen dioxide (NO_2) and particulate matter with a diameter of less than 10 µm (PM_{10}). We selected ten monitoring stations across London and used data from January 2018 to January 2021.

The second dataset is on India air quality. The dataset was compiled by Rohan Rao from the Central Pollution Control Board (CPCB) website and can be downloaded from Kaggle's collection [47]. Among many air quality monitoring stations, we selected ten monitoring stations across the city Delhi from February 2018 to July 2020. The chosen pollutants for the Delhi dataset are hourly measurements of NO_2 and PM with a diameter of less than 2.5 µm ($PM_{2.5}$).

The third dataset is Beijing multi-station air quality provided by Zhang et al. [48], which can be downloaded from the UCI Machine learning repository page [49]. The dataset contains hourly pollutant data from January 2013 to February 2017. We focused on carbon monoxide (CO) and ozone (O_3) data for the Beijing dataset. We selected ten monitoring stations, namely Aotizhongxin, Changping, Dingling, Dongsi, Guanyuan, Gucheng, Huairou,

Nongzhanguan, Shunyi and Tiantan. Table 1 summarises the air quality monitoring stations used in this study.

Tables 2, 3 and 4 show brief descriptive statistics of London, Delhi and Beijing air quality data. As shown in the tables, four statistics characteristics are shown, namely mean, standard deviation and two quartiles. The mean and standard deviation columns are calculated by excluding the missing values. Standard deviation measures how observed values spread from the mean. A low standard deviation in each station implies that the observed values tend to be close to the mean, whereas a high standard deviation indicates that the observed values are spread out over a broader range from the mean. The quartiles divide the ordered observed values (i.e. from smallest to largest) into four parts. The first quartile (25%) is the middle value between the minimum and the median, whereas the third quartile (75%) is defined as the middle value between the median and the maximum.

Table 2 Descriptive statistics of London monitoring stations

ID	NO_2				PM_{10}			
	Mean	Std.	25%	75%	Mean	Std.	25%	75%
1	29.13	0.03	16.01	39.39	18.63	11.39	11.40	23.00
2	38.80	0.39	22.21	52.14	20.53	18.58	12.49	24.86
3	50.47	0.22	30.01	67.12	22.70	14.24	13.34	28.47
4	39.13	0.33	23.62	51.64	19.64	10.88	12.65	23.93
5	23.72	0.01	11.98	31.72	18.74	12.51	11.28	22.55
6	45.73	0.25	28.35	60.18	39.30	28.06	19.32	51.94
7	41.50	0.00	25.03	55.29	19.03	12.54	10.83	23.60
8	28.82	1.16	15.51	38.14	17.34	10.64	10.33	21.61
9	20.94	0.01	8.70	28.91	17.89	12.05	10.60	22.20
10	22.40	0.10	9.25	30.49	17.28	11.56	9.70	21.30

2.3 Correlation of pollutant data

The same pollutant data from all monitoring stations are combined, and the coefficient correlation for each pollutant is calculated. For example, if the PM_{10} is decided as a target pollutant, then we collected all PM_{10} values from all monitoring stations. Pearson's correlation is used to find the relation of pollutant data among monitoring stations. Pearson's correlation measures the linear correlation between two sets of data and can capture the details between trends of two time-series data [19].

Assume that we have a temporal sequence of specific pollutant data in the targeted station as $\boldsymbol{S}^t = [s_1^t, s_2^t, s_3^t, \ldots, s_{n-1}^t, s_n^t]$ and a temporal sequence of the same pollutant data at a neighbouring station as $\boldsymbol{S}^s = [s_1^s, s_2^s, s_3^s, \ldots, s_{n-1}^s, s_n^s]$. Note that both $\boldsymbol{S}^t$ and $\boldsymbol{S}^s$ have the same time frame ranging from sample 1 to n. Then, the Pearson's correlation coefficient between these two series is described as follows:

$$r(\boldsymbol{S}^t, \boldsymbol{S}^s) = \frac{\sum((s_i^t - \mu_t)(s_i^s - \mu_s))}{\sqrt{\sum(s_i^t - \mu_t)^2 \sum(s_i^s - \mu_s)^2}} \quad (1)$$

where $r(\boldsymbol{S}^t, \boldsymbol{S}^s)$ denotes the Pearson's correlation coefficient between the time series $\boldsymbol{S}^t$ and $\boldsymbol{S}^s$, s_i^t and s_i^s represent the i-th samples of $\boldsymbol{S}^t$ and $\boldsymbol{S}^s$, respectively. Finally, $\mu_t = \frac{1}{n}\sum_{i=1}^{N} s_i^t$ and $\mu_s = \frac{1}{n}\sum_{i=1}^{N} s_i^s$ denote the mean values of time series $\boldsymbol{S}^t$ and $\boldsymbol{S}^s$, respectively.

In Eq. 1, the numerator is the covariance, a measurement about how series $\boldsymbol{S}^t$ and $\boldsymbol{S}^s$ vary together from their mean value. In the denominator, the equation expresses the variance of $\boldsymbol{S}^t$ and $\boldsymbol{S}^s$. Correlation is a normalised version of covariance, scaled between -1 to 1 [50]. When $r = 1$, it is said that $\boldsymbol{S}^t$ and $\boldsymbol{S}^s$ are completely positively correlated. When $r = -1$, $\boldsymbol{S}^t$ and $\boldsymbol{S}^s$ are completely negatively correlated. Finally, when $r = 0$, the linear correlation between $\boldsymbol{S}^t$ and $\boldsymbol{S}^s$ is not obvious [51].

Table 1 Dataset used in this study

	London		Delhi		Beijing	
ID	Code	Station	Code	Station	Code	Station
1	CT3	Aldgate school	DL02	Anand vihar	AOT	Aotizhongxin
2	GN5	Trafalgar road	DL03	Ashok vihar	CHA	Changping
3	GR8	Woolwich flyover	DL04	Aya nagar	DIN	Dingling
4	IS2	Holloway road	DL07	Mathura Rd.	DON	Dongsi
5	IS6	Arsenal	DL08	DTU Delhi	GUA	Guanyuan
6	LB5	Bondway intchg.	DL10	Dwarka-Sect. 8	GUC	Gucheng
7	LW4	Loampit vale	DL12	IGI airport	HUA	Huairou
8	SK6	Elephant & Castle	DL13	IHBAS, Delhi	NON	Nongzhanguan
9	TH001	Millwall park	DL14	ITO, Delhi	SHU	Shunyi
10	TH002	Victoria park	DL15	Jahangirpuri	TIA	Tiantan

Table 3 Descriptive statistics of Delhi monitoring stations

ID	NO_2				$PM_{2.5}$			
	Mean	Std.	25%	75%	Mean	Std.	25%	75%
1	87.69	61.82	43.95	114.17	131.50	120.90	53.00	168.25
2	41.72	33.26	18.30	56.82	113.49	112.17	40.50	144.42
3	23.78	17.62	13.59	28.12	80.20	77.19	31.88	104.48
4	43.84	44.46	21.51	46.32	105.10	99.32	39.83	133.56
5	37.64	29.88	20.47	46.27	112.83	104.78	42.24	149.25
6	39.68	30.89	19.42	50.60	103.52	96.29	39.00	138.25
7	35.35	36.98	15.68	42.21	82.36	79.66	32.25	104.87
8	43.23	30.40	20.95	58.20	100.63	85.59	45.85	128.57
9	50.24	40.81	23.45	66.17	110.73	94.02	48.75	140.25
10	65.94	49.27	31.68	80.77	128.85	116.22	48.00	172.00

Table 4 Descriptive statistics of Beijing monitoring stations

ID	CO				O_3			
	Mean	Std.	25%	75%	Mean	Std.	25%	75%
1	1264.45	1222.01	500	1500	56.45	58.16	8.00	82.00
2	1153.33	1105.00	500	1400	58.22	54.56	16.00	80.00
3	897.89	886.87	300	1100	68.87	54.19	31.00	91.00
4	1326.37	1176.91	600	1700	57.56	58.36	12.00	82.00
5	1265.01	1142.17	500	1600	56.05	57.80	7.00	82.00
6	1310.00	1181.11	600	1600	58.41	57.43	10.00	85.00
7	1013.71	878.00	400	1300	60.06	54.97	18.00	82.00
8	1312.83	1215.52	500	1600	59.07	58.86	10.00	85.00
9	1171.65	1121.10	400	1500	55.54	55.28	10.00	77.00
10	1287.12	1143.32	600	1600	56.44	59.53	8.00	82.00

2.4 Data pre-processing

2.4.1 Spatial characteristics

As depicted in Fig. 2, there are two kinds of pre-processing phases conducted in this study, i.e. block number 2 and number 3. The primary purpose of the first pre-processing phase is to find the pollutant correlations. The pollutant correlation among monitoring stations is utilised to capture the spatial characteristic of air contaminants. For each pollutant, we identified which neighbouring stations have the closest spatial relationship with the station under investigation. In other words, we tried to take advantage of the existing monitoring stations to fill the missing values in the targeted monitoring station. Choosing different kinds of air pollutants will vary the correlation coefficients. Thus, the selected monitoring stations might also differ.

Let

$$\boldsymbol{S}^t = \begin{bmatrix} s^t_{1,1} & s^t_{1,2} & \cdots & s^t_{1,n} \\ s^t_{2,1} & s^t_{2,2} & \cdots & s^t_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ s^t_{m,1} & s^t_{m,2} & \cdots & s^t_{m,n} \end{bmatrix} = (s^t_{i,j}) \in \mathbb{R}^{m \times n}$$

be a matrix containing m rows of measurement data and n different pollutants in monitoring station t, where t ranges from 1 to 10. Therefore, we have a pollutant data collection from all stations of $\boldsymbol{S}^1$, $\boldsymbol{S}^2$, $\boldsymbol{S}^3$,..., $\boldsymbol{S}^{10}$. In this case, each row in matrix $\boldsymbol{S}^t$ is hourly measurement data. Then, we create a matrix

$$\boldsymbol{J} = \begin{bmatrix} s^1_{1,p} & s^2_{1,p} & \cdots & s^{10}_{1,p} \\ s^t_{2,p} & s^2_{2,p} & \cdots & s^{10}_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ s^1_{m,p} & s^2_{m,p} & \cdots & s^{10}_{m,p} \end{bmatrix}$$

as a collection of the same pollutant p taken from all stations, where p is a single integer value chosen from 1 to n. The value of p represents the selected column in $\boldsymbol{S}^t$. In this scenario, we assume that all monitoring station data in the same city have the same column header. Then, we computed the pairwise correlation of columns in $\boldsymbol{J}$ using Eq. 1, excluding null/missing values. A graphical representation of this process is presented in Fig. 3.

As shown in Fig. 3, we collect the same pollutant for each monitoring station into a single data frame (or matrix) to achieve this goal. For example, when we calculate the

correlation of PM_{10} among stations in the London dataset, we collected PM_{10} data from CT3, GN5, GR8, IS2, IS6, LB5, LW4, SK6, TH001 and TH002 monitoring stations into a single data frame. Only the targeted pollutant (i.e. PM_{10}) is selected, and other pollutants are ignored. Before joining the data, we must ensure that targeted contaminants from all monitoring stations have the same time frame. We implemented these procedures using Python programming with the help of pandas library [52].

Once target pollutants have been collected, the Pearson's correlation calculation can be carried out using Eq. 1. For each station, we then sorted the correlation coefficients from the strongest to the weakest. The obtained coefficients indicate how strong the correlation of the same pollutant between two monitoring station is. Based on this result, the number of monitoring stations involved as input of the proposed model is evaluated. Based on the conducted experiments, we decided to take three neighbouring stations along with the target station. Thus, the input sets of our proposed model will have four columns. We will explain the process of deciding the number of monitoring sites in Sect. 3.3.

The second phase of the pre-processing blocks (i.e. the block number 3 in Fig. 2) is dedicated to capturing the temporal characteristic of the pollutants, conducting the perturbation procedure and creating input sets suitable for the proposed deep learning model.

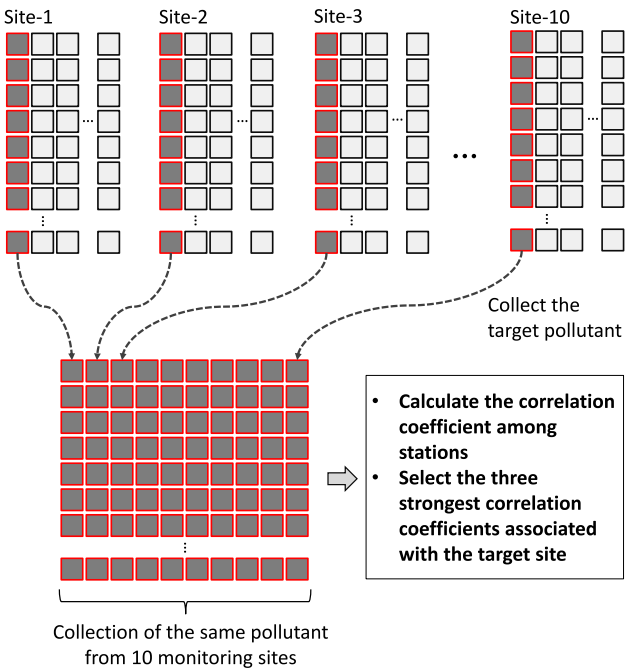

Fig. 3 The process of determining the correlation coefficient for target pollutant

2.4.2 Temporal characteristics

Besides involving spatial characteristics, this study also tries to capture the temporal behaviour of the pollutant data. The temporal behaviour describes the dependency among pollutants at different times [53]. In this study, we calculate the autocorrelation coefficient of the contaminant under investigation using Pearson's correlation. We computed this correlation between the series of targeted pollutants and its shifted self. Thus, instead of calculating the correlation between two different time series, the autocorrelation computes the relation between the same time series at current and lagged times. Given time-series of pollutant data at the target station $\boldsymbol{S}^t = [s_1^t, s_2^t, s_3^t, \ldots, s_{n-1}^t, s_n^t]$, we can rewrite equation 1 to find the lag-k autocorrelation function as:

$$r_k = \frac{\sum_{i=k+1}^{n}((s_i^t - \mu_t)(s_{i-k}^t - \mu_t))}{\sum_{i=1}^{n}(s_i^t - \mu_t)^2} \tag{2}$$

where r_k denotes the autocorrelation function, k is lag, s_i^t and s_{i-k}^t represent the i-th and lag-k samples of $\boldsymbol{S}^t$, and $\mu_t = \frac{1}{n}\sum_{i=1}^{N} s_i^t$ denote the mean values of time series $\boldsymbol{S}^t$.

In this study, we use 8-hour as the length of pollutant data. This length is obtained by computing the lag-k autocorrelation. The value of k will determine the size of input data. As discussed in Sect. 3.2.2, we determine $k = 7$. Please note that the value of time lag is started at 0 (or $k = 0$). The value of $k = 7$ means that we use 8 data in total. In other words, to find a single prediction, we use current and seven previous observed data as the input for our proposed model. To conclude, by involving the pollutant data from targeted and three other neighbouring stations (i.e. spatial consideration) and including current and seven previous data (temporal consideration), the final input for the proposed deep learning model will have a size of 8×4.

2.4.3 Missing data and perturbation procedure

Another pre-processing step carried out in this study is to handle the initial missing data in the original datasets. Missing values occur both in the form of discontinuous and consecutive missing patterns. As our proposed model is trained in a supervised manner, we have to provide input-target pairs. The model fits on the given training data consisting of input and target sets. While deleting missing data are a straightforward procedure, we avoid this method as this method can break the data structure, and valuable information may be lost. To minimise the defect of the original data structure, we carefully picked the series of data with a minimum period of one week (168 hours). As our input sets are comprised of pollutant data from multiple

monitoring stations, the minimum one-week selection is applied only for the target station. We let the other station periods comply with the target station period.

Figure 4 illustrates this idea. The shadowed areas indicate the period of the observed pollutant without missing values, whereas the white strips indicate the missing values. Based on the target station data, a minimum of 168 hours intervals without missing data were selected. The same selection periods were also applied to the neighbouring stations to maintain the consistency of the time frame between monitoring stations. After completing these steps, the target station will have no missing data. However, unlike the target station, there is a possibility that missing values exist in the neighbouring station parts. To overcome this issue, we filled the missing values with zeros.

To train the proposed model, we need pairs of input and target sets. Since the target station data contain no unknown values, the actual targets for all input sets can be provided. The perturbation procedure was carried out to reflect the missing values phenomena and train the proposed model. Some values in input sets were intentionally removed, and all deleted values were filled with zeros. In this scenario, the errors were generated in the correct dataset to evaluate the performance of the proposed imputation method [54]. Short-interval and long-interval consecutive missing patterns were applied to the input sets and let the model adjust its parameters to minimise the loss function.

For the short-interval perturbation procedure, different levels of missingness were applied to the input sets. Following the work conducted by Hadeed et al., four missing rates (i.e. 20%, 40%, 60% and 80% of missing rates) were set for the target station [25]. While the missing rate was varied for the target station, a fixed missing rate was applied to the neighbouring stations during the training and testing phases of the proposed model. The missing rate of 20% was considered as an error probability for the neighbouring stations [54]. Due to the initial zero imputation illustrated in Fig. 4, the neighbouring stations will have more than a 20% missing rate after the perturbation procedure.

For the long-interval perturbation procedure, a maximum of 500 hours of consecutive values was removed from some parts of the correct dataset. The successive missing periods were varied between 100 and 500 hours. This procedure was implemented only to the target station, and we let the neighbouring stations follow the short-interval process described previously. Figure 5 illustrates input set perturbation patterns for the input sets. It can be seen that both short- and long-interval missing patterns were generated only for the data in the target station, and a minimum of 20% missing rates was applied to all neighbouring stations.

2.4.4 Model input construction

Input sets resulting from the perturbation process are ready to be normalised. Once the normalisation step is completed, the model input construction can be performed. The current missing value is predicted using the current initial imputation (i.e. we filled the current missing value with zero) along with the last 7 hours data. As illustrated in Fig. 6, the dataset contains air pollutant data over a sample row from $t = 1, \ldots, T$, and the rolling window size is m. The input sets for the model are obtained by shifting the pre-processed dataset. We take 8 hours of data and shift the features by one hour to get the next input set. This process is similar to the rolling-window scheme. In our case, the

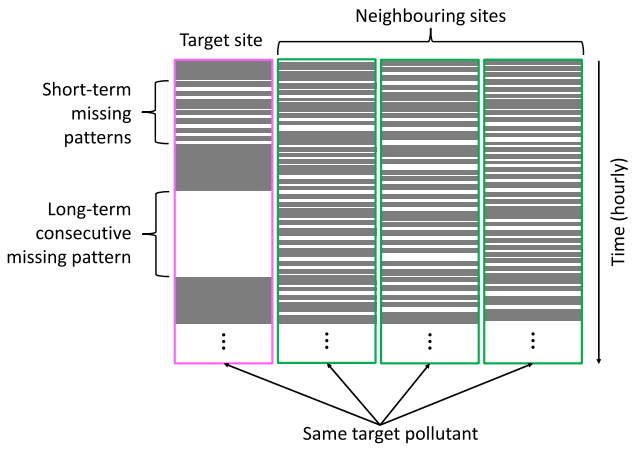

Fig. 4 Implemented method to handle the initial missing data in the original datasets. A minimum of 168 hours of observed data without missing values is carefully selected

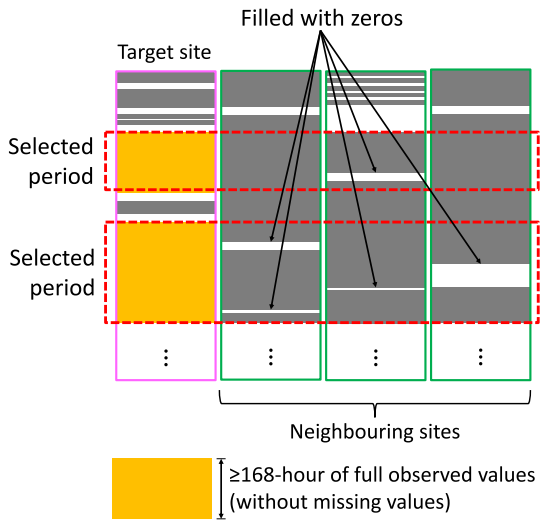

Fig. 5 Illustration of perturbation patterns applied to the dataset

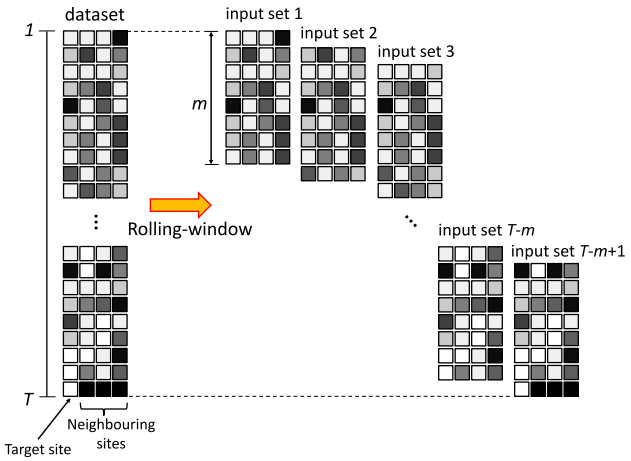

Fig. 6 Extracting input sets from the preprocessed dataset

increment between successive rolling windows is one period.

The proposed model acts as a denoising tool as the missing values are intentionally generated from the complete dataset, and the given target is the complete dataset itself. Thus, our proposed model can be called a denoising autoencoder [31]. Given the noisy inputs, the autoencoder model will reconstruct these inputs. Based on this concept, the imputation of missing values that exist in the given data is performed.

2.5 Proposed model

2.5.1 Convolutional autoencoder architecture

In this study, a convolutional autoencoder model is proposed to learn the missing patterns from the given corrupted input sets and the provided actual sets. The proposed model architecture is shown in Fig. 7. The autoencoder model accepts the collection of input sets in the form of 8×4 matrices. The individual input comprises four columns of pollutant data, a group of hourly targeted pollutant concentrations from four monitoring stations, and eight rows that indicate 8-hour of observed data. We purposely corrupted the input sets by deleting the actual values and filling them with zeros to train the model. The input columns represent spatial behaviour, and the rows capture temporal characteristics of air pollution features.

The autoencoder contains encoder and decoder parts, and both sections are based on one-dimensional convolution layers. The encoder is made up of convolution layers, while the decoder consists of transposed convolution layers. The proposed model receives only eight values as the feature's length, so we need to utilise a small kernel to obtain more detailed input features. In this case, the kernel size equal to two is applied to all layers. The kernel's size specifies the 1D convolution window operated in each layer, and it is used to extract the essential input features. In this model, no padding is implemented in each layer.

As illustrated in Fig. 7, the size of layers in the proposed model changes, both in height and width. The width of the following layers is controlled by the number of filters used in the previous layer. After various experiments, we determined the number of filters used in the proposed model, as presented in Table 5. The encoder has different output filters, from 80 in the first layer to 10 in the fifth layer. From the latent space, the number of the filter is expanded from 20 in the sixth layer to 80 in the ninth layer. Finally, we set the final layer with 8 output filters to get the equal size of reconstructed inputs (i.e. 8×4 matrices).

2.5.2 Model configuration and training

The proposed autoencoder model was built using Tensorflow CPU version [55] and written in Python. Some powerful Python libraries were also utilised, such as Keras [56], pandas [52], NumPy [57], scikit-learn[58], Matplotlib [59] and seaborn [60]. In this work, we used a local machine powered by Intel® Core™ i7-8565U CPU (4 core(s), @1.80GHz), 8 GB installed RAM and Windows 10 as the operating system.

After creating the model architecture, we configured the model for training. We selected Adam [61] as the optimiser with a learning rate of 0.001. The program computed the mean squared error (MSE) values between the given target and prediction during training. MSE was the selected loss

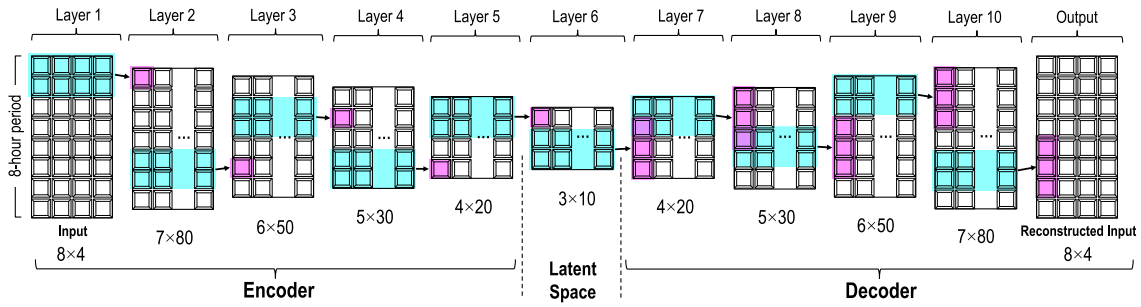

Fig. 7 The proposed convolutional autoencoder model architecture

Table 5 Layer properties of the proposed autoencoder model

No.	Type	Filter	Kernel	Stride	Activation	Padding	Output
0	Input Layer	–	–	–	–	–	(8,4)
1	1D Convolution	80	2	1	ReLU	No	(7,80)
2	1D Convolution	50	2	1	ReLU	No	(6,70)
3	1D Convolution	30	2	1	ReLU	No	(5,50)
4	1D Convolution	20	2	1	ReLU	No	(4,30)
5	1D Convolution	10	2	1	ReLU	No	(3,10)
6	1D Transposed conv.	20	2	1	ReLU	No	(4,30)
7	1D Transposed conv.	30	2	1	ReLU	No	(5,50)
8	1D Transposed conv.	50	2	1	ReLU	No	(6,70)
9	1D Transposed conv.	80	2	1	ReLU	No	(7,80)
10	1D Transposed conv.	4	2	1	ReLU	No	(8,4)

function to optimise the model during training and the metric to judge the model's performance. About 75% of data were used as training sets and the remaining data as testing sets. We used 32 as the batch size (i.e. number of samples per gradient update) and implemented an early stopping method to finish the training. The training process is terminated when there is no loss score improvement in three consecutive iterations.

Figure 8 illustrates the training process of the denoising autoencoder. Define the encoder function $f_\theta(\cdot)$ parameterised by $\theta = \{\mathbf{W}, \mathbf{b}\}$, and decoder function $g_\phi(\cdot)$ parameterised by $\phi = \{\mathbf{W}', \mathbf{b}'\}$, where $\mathbf{W}$, $\mathbf{W}'$, $\mathbf{b}$ and $\mathbf{b}'$ represent the weight and bias of the encoder and decoder, respectively. Thus, we define the encoder function as $\mathbf{h} = f_\theta(\mathbf{x})$ and the decoder function as $\mathbf{r} = g_\phi(\mathbf{h})$, where $\mathbf{x}$ is the input, $\mathbf{h}$ is the code representation learning, and $\mathbf{r}$ is the reconstructed input. The perfect condition for model learning is to set $g_\phi(f_\theta(\mathbf{x})) = \mathbf{x}$. However, the model cannot learn perfectly but instead tries to minimise the error between the actual input and the reconstructed input [62]. Then, for each training set $\mathbf{x}^{(i)}$, the parameters θ and ϕ are optimised to minimise the average reconstruction error [31]:

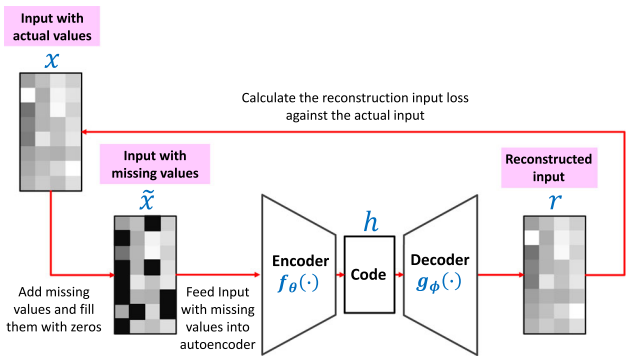

Fig. 8 A denoising convolutional autoencoder workflow

$$\theta^*, \phi^* = \arg\min_{\theta,\phi} \frac{1}{n} \sum_{i=1}^{n} L(\mathbf{x}^{(i)}, g_\phi(f_\theta(\mathbf{x}^{(i)}))) \tag{3}$$

where L is the model loss function. The typical loss function is squared error $L(\mathbf{x}, \mathbf{r}) = \|\mathbf{x} - \mathbf{r}\|^2$. For the denoising autoencoder, instead of $\mathbf{x}$, we define $\tilde{\mathbf{x}}$ as the noisy input of $\mathbf{x}$ [62]. Thus, the loss function of the denoising autoencoder is rewritten as:

$$L(\theta, \phi) = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}^{(i)} - g_\phi(f_\theta(\tilde{\mathbf{x}}^{(i)})))^2 \tag{4}$$

2.5.3 Post-training outputs interpretation

The test sets are fed to the model after the training process has been completed. The model accepts 8×4 input sets and yields outputs of the same size. As the trained values are scaled into [0,1], the output values must be transformed back to their original values.

After we undo the scaling of model output, we determined the single prediction for each hour. As illustrated in Fig. 9, the autoencoder produces overlapping outputs for a certain period of prediction. We aggregated the values by calculating the means of all overlapped output sets to give a single point estimate. As the targeted results are located in model outputs' first columns (target station), we can calculate the means only for the first columns of the output sets. These processes are systematically presented in Algorithm 1 in Appendix A.

2.6 Model evaluation metrics

There are several methods usually used to evaluate the model performance. In this study, root mean square error (*RMSE*) and mean absolute error (*MAE*) are used, following work done by [63]. Another broadly used method to evaluate model performance in machine learning studies is

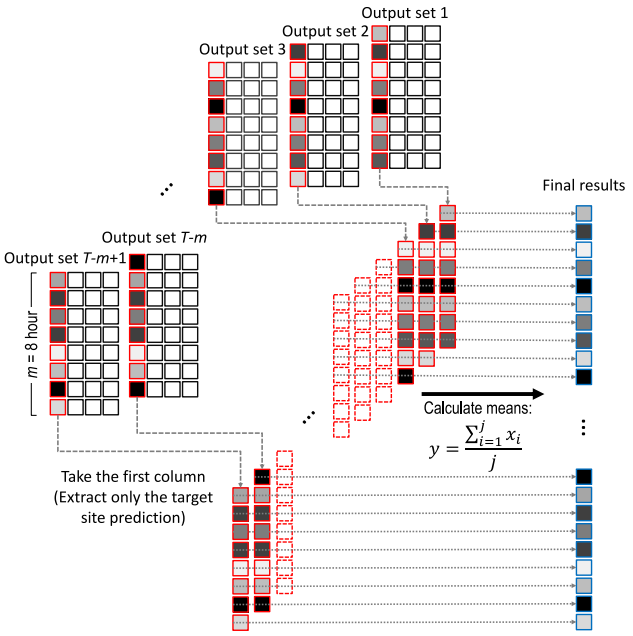

Fig. 9 Interpretation of model outputs

the coefficient of determination (R^2 or R-squared). Chicco et al. [64] suggested implementing R^2 for regression task evaluation as this method is more informative to qualify the regression results. However, the limitation of R^2 arises when the calculated score is negative. In this case, the model performance can be arbitrarily worse, but it is impossible to recognise how bad a machine learning model performed [65].

Following work conducted by Ma et al. [19], we implemented a rate of improvement on *RMSE* (*RIR*) to measure the performance of our methods in comparison with the existing imputation techniques. The *RIR* is calculated using the following equation:

$$RIR^{A,B} = \frac{RMSE^A - RMSE^B}{RMSE^A} \times 100\% \quad (5)$$

where $RMSE^A$ denotes the *RMSE* value of the benchmarked method and $RMSE^B$ is the *RMSE* value of our proposed method.

In addition to *RMSE*, *MAE*, R^2 and *RIR*, in this study, visual comparisons of actual and imputed data in the forms of line, bar or box plots were also presented to describe the model performance more intuitively.

3 Results and discussion

3.1 Distribution of missing periods

As we develop the proposed model for both short-interval and long-interval consecutive missing patterns, it is essential to know the nature of the lost patterns. We counted the duration of all missing patterns in the original dataset, with results shown in Fig. 10. The figure visualises the distribution of missing data durations using continuous probability density curves. The graphs give us an understanding of how the missing data durations are distributed. As shown in Fig. 10, we can conclude that most missing durations in the London dataset are less than 400 hours. The Delhi and Beijing datasets exhibit shorter missing durations of less than 200 hours. If we observe the peak of the probability density curve in all datasets, the highest points commonly occur within 100 hours. We can conclude that the missing data in all datasets are occupied mainly by short-interval missing patterns whose periods are less than approximately one week.

3.2 Evaluation of temporal characteristics

3.2.1 Autocorrelation coefficients of pollutant data

The temporal relationship is evaluated to determine the length of the input series to be fed to the model. Temporal behaviours for each monitoring station are assessed based on obtained Pearson's autocorrelation coefficient between the series. We computed the correlation coefficient

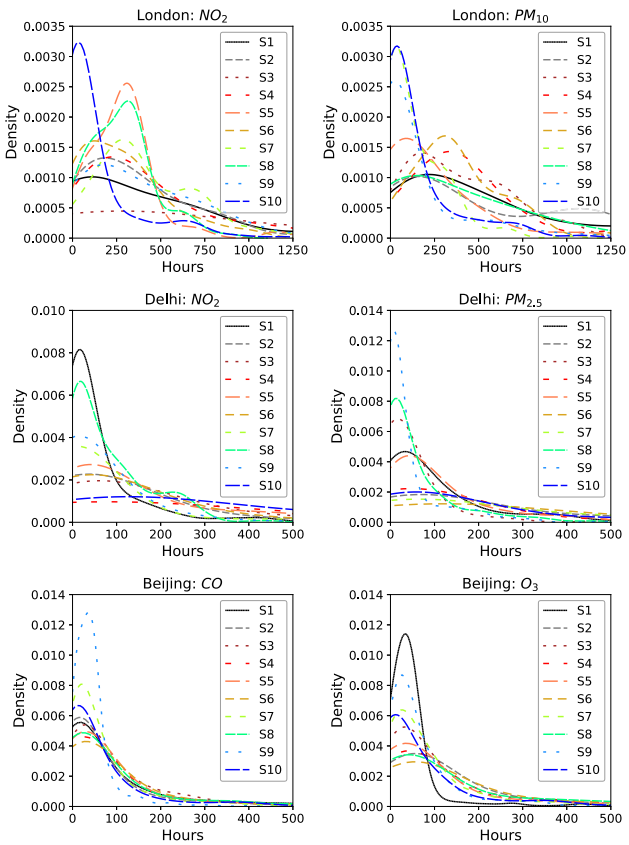

Fig. 10 Probability density function of missing data in all stations

between the actual time series data and their shifted lag-k hour data. The variable k is an integer number that varies between 1 and 11. For instance, the value of $k = 1$ means that the actual time-series data are shifted 1 hour backwards.

The autocorrelation coefficients for each city and pollutant are reported in Fig. 11. For $k \leq 11$ hours, the autocorrelation coefficients can vary between 1 (i.e. at $k = 0$) and 0. For the London dataset, all monitoring stations measuring NO_2 pollutants have similar autocorrelation coefficient slopes ranging from 1 to about 0.2. Monitoring station S^1 has the flattest slope, which indicates that S^1 has the strongest relationship among the lagged hours of NO_2 pollutants compared to other stations. For PM_{10} in the same air quality dataset, the stations S^2, S^3 and S^6 autocorrelation coefficients plunge to about 0.2 in the first six lagged hours, whereas other stations coefficients remained above 0.5. Among these, S^2 seems to have the weakest temporal dependency.

For the Delhi air quality dataset, the $PM_{2.5}$ autocorrelation coefficient slopes are relatively flattered for the same pollutant, ending between 0.55 and 0.65 at $k = 11$. Autocorrelation coefficients for NO_2 between monitoring stations in Delhi degrade more diversely, especially from $k = 3$ to $k = 11$. Station S^1 and S^8 have exceptional slopes, which the coefficients tend to increase after $k = 7$. Less varied autocorrelation coefficient slopes are shown in Beijing dataset for both CO and O_3 data. However, O_3 pollutant coefficients decrease more rapidly compared to CO coefficients.

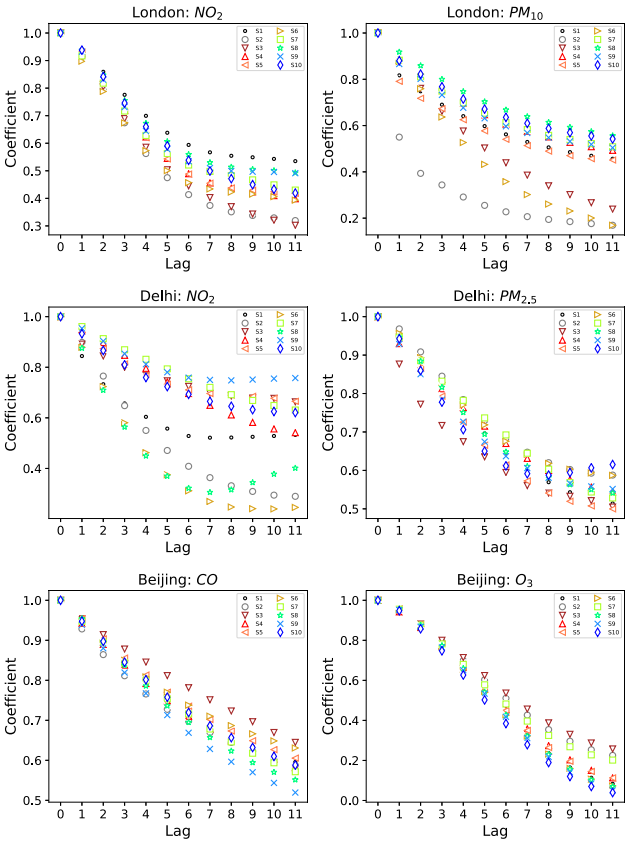

Fig. 11 Temporal characteristics of air quality datasets based on their autocorrelation coefficients

3.2.2 Temporal window size determination

We determine the number of pollutants (i.e. the length of the input set) for the autoencoder model based on the obtained coefficients shown in Fig. 11. A simple model is introduced as a base model. The base model is used to evaluate the temporal and spatial dependencies. Temporal evaluation determines the number of input set rows, whereas spatial evaluation defines the number of input columns. Some other model architectures are then derived from the base model until we finally decided to use the model whose properties are presented in Fig. 7 and Table 5.

Figure 12 shows the proposed base model. The model is based on autoencoder architecture, similar to the final model but has shallower hidden layers. For simplicity, the base architecture model in this study is written as $L^1 40 - L^2 30 - L^3 20 - L^4 30 - L^5 40 - L^6 x$. The integer value of x depends on the intended number of output columns. For the temporal evaluation, we set $x = 4$, which means that we use the target station together with three other neighbouring stations. The term $L^1 40$ means that the first layer has 40 output filters. The $L^1 40$ layer yields 40 columns placed in the second layer, as indicated in Fig. 12. The sixth layer (i.e. $L^6 x$, where $x = 4$) has four output filters and forms $n \times 4$ output sets, where n depends on the input length, kernel and filter size. We set kernel size equal to 2 and stride equal to 1 for all layers. In addition, no padding is applied for all layers.

In this experiment, 60% of the total observation are used as training sets, applied for each station and pollutant. In addition to training data, the test data are selected based on

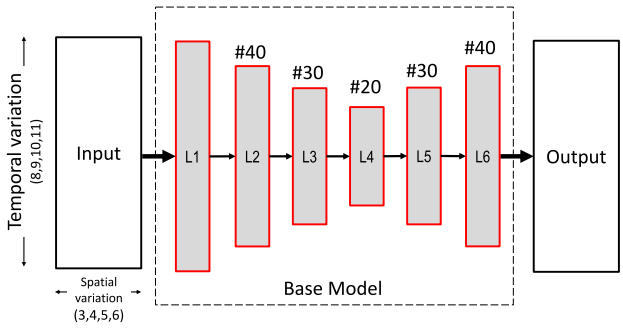

Fig. 12 The proposed base model for temporal and spatial characteristic evaluations

an unbroken time-series segment with a minimum of 400 hours of consecutive observed values. The target station is corrupted with a missing rate of 40%, whereas the neighbouring data are lightly corrupted with a missing rate of 20%. To obtain less biased results, we implemented five-fold cross-validation in the dataset. An example of temporal data evaluation is demonstrated in Table 6. Table 6 shows the results obtained from the London dataset for NO_2 as the target pollutant.

As shown in Table 6, most of the least values on the average *RMSE* are obtained from the input sets with lag-7 hours, indicated as bold texts. The lag-7 hours means that the model accepts eight temporal data as the number of rows (input length). Setting the number of $k = 7$, however, does not improve the *RMSE* values significantly. For example, the obtained *RMSE* at S^5 equals 6.28 $\mu g/m^3$ at $k = 7$, improving the *RMSE* to only about 4% from the worst result (at $k = 10$). In general, increasing the number of temporal data does not improve the model performance as the temporal correlations of measurement values are weaker. Weak temporal correlations contribute less essential features for the autoencoder model. To sum up, we settled on a window size of 8-time steps for our model.

3.3 Evaluation of spatial characteristics

3.3.1 Correlation coefficients of pollutant data

While autocorrelation defines the temporal relations, Pearson's correlation among stations determines spatial characteristics. Unlike the autocorrelation coefficient that calculates the correlation between the actual and its shifted self, Pearson's correlation coefficients for spatial evaluation are computed between two stations. Pairs of monitoring stations are created based on the city and pollutant, and the correlation coefficients are assessed between all the pairs. No lagged time is applied for each monitoring station data as we did in the temporal evaluation.

For example, we report the obtained correlation coefficients for NO_2 and PM_{10} in London air quality data in Tables 7 and 8, respectively. The same procedure mentioned below was also implemented for Delhi and Beijing datasets. The correlation coefficients reflect the linear relationship between station pairs and can be calculated using Eq. 1. As reported in Table 7, the correlation coefficients among monitoring stations measuring NO_2 fall between 0.49 and 1.00. The paired stations such as $S^1 - S^9$, $S^4 - S^6$, $S^8 - S^9$ and $S^5 - S^{10}$ have strong correlations for pollutant NO_2. In contrast, the paired stations $S^3 - S^5$, $S^3 - S^7$ and $S^3 - S^{10}$ have weaker correlations. The correlation coefficients between the paired stations measuring PM_{10} as presented in Table 8 are more diverse, ranging from 0.27 to 1.00. In all datasets, no negative coefficient was found.

We carefully selected the three neighbouring stations with the strongest correlation coefficients to the target station. Sorting from largest to smallest coefficients, we report the selected neighbouring stations for NO_2 and PM_{10} in Table 9.

3.3.2 Selecting the number of neighbouring stations

This section discusses the procedure for selecting the number of neighbouring stations involved to form the input sets for the autoencoder model. In this study, spatial evaluation determines the number of involved neighbouring stations. Varying the number of neighbouring stations affects the model input width (i.e. the number of columns). The columns of the input set consist of the target station data plus the neighbouring station data. As shown in Fig. 12, the width of the input set is evaluated from 3 to 6 monitoring stations. We also demonstrate the results of neighbourhood selection in Delhi and selected $PM_{2.5}$ as the

Table 6 The average of *RMSE* and standard deviation values after fivefold cross-validation for NO_2 of the London dataset

	Test period		Lag-k			
	Start	End	7	8	9	10
S^1	2020-12-02	2020-12-31	**8.56(0.35)**	8.93(0.43)	8.71(0.27)	8.87(0.28)
S^2	2021-01-12	2021-01-31	14.98(0.25)	15.07(0.34)	**14.25(0.43)**	14.26(0.64)
S^3	2020-12-07	2021-01-26	**15.86(0.6)**	16.92(0.88)	16.26(0.39)	16.43(0.88)
S^4	2021-01-12	2021-01-31	12.96(0.57)	**12.86(0.37)**	13.17(0.4)	13.02(0.35)
S^5	2021-01-12	2021-01-30	**6.28(0.19)**	6.42(0.39)	6.31(0.34)	6.56(0.11)
S^6	2020-12-07	2021-01-06	9.31(0.15)	9.58(0.17)	**9.19(0.24)**	9.39(0.19)
S^7	2021-01-11	2021-01-31	16.29(0.2)	16.32(0.72)	16.23(0.25)	**16.11(0.31)**
S^8	2021-01-08	2021-01-25	8.81(0.46)	**8.62(0.29)**	9.02(0.51)	8.85(0.24)
S^9	2020-12-30	2021-01-31	**7.43(0.19)**	7.70(0.17)	7.74(0.19)	7.45(0.23)
S^{10}	2020-10-29	2021-01-31	**7.38(0.14)**	7.54(0.23)	7.39(0.07)	7.4(0.14)

The best results are indicated in bold

Table 7 Coefficient of correlation for NO_2 in the London air quality data

	S^1	S^2	S^3	S^4	S^5	S^6	S^7	S^8	S^9	S^{10}
S^1	1.00									
S^2	0.77	1.00								
S^3	0.63	0.83	1.00							
S^4	0.73	0.78	0.82	1.00						
S^5	0.81	0.73	0.57	0.79	1.00					
S^6	0.78	0.79	0.79	0.84	0.71	1.00				
S^7	0.72	0.62	0.49	0.60	0.61	0.70	1.00			
S^8	0.84	0.67	0.50	0.68	0.84	0.74	0.75	1.00		
S^9	0.85	0.76	0.54	0.66	0.84	0.71	0.74	0.88	1.00	
S^{10}	0.80	0.68	0.49	0.69	0.88	0.66	0.66	0.85	0.85	1.00

Table 8 Coefficient of correlation for PM_{10} in the London air quality data

	S^1	S^2	S^3	S^4	S^5	S^6	S^7	S^8	S^9	S^{10}
S^1	1.00									
S^2	0.47	1.00								
S^3	0.57	0.46	1.00							
S^4	0.76	0.53	0.69	1.00						
S^5	0.70	0.47	0.55	0.82	1.00					
S^6	0.35	0.27	0.52	0.45	0.31	1.00				
S^7	0.75	0.48	0.55	0.76	0.69	0.33	1.00			
S^8	0.81	0.50	0.60	0.86	0.78	0.34	0.81	1.00		
S^9	0.77	0.53	0.58	0.82	0.75	0.36	0.80	0.85	1.00	
S^{10}	0.77	0.64	0.57	0.83	0.76	0.35	0.77	0.85	0.83	1.00

Table 9 The strongest correlation coefficient for neighbouring stations selection in London air quality data

Target station	Strongest corr. coeff. (NO_2)			Strongest corr. coeff. (PM_{10})		
	1st	2nd	3rd	1st	2nd	3rd
S^1	S^9	S^8	S^5	S^8	S^9	S^{10}
S^2	S^3	S^6	S^4	S^{10}	S^9	S^4
S^3	S^2	S^4	S^6	S^4	S^8	S^9
S^4	S^6	S^3	S^5	S^8	S^{10}	S^5
S^5	S^{10}	S^9	S^8	S^4	S^8	S^{10}
S^6	S^4	S^3	S^2	S^3	S^4	S^9
S^7	S^8	S^9	S^1	S^8	S^9	S^{10}
S^8	S^9	S^{10}	S^1	S^4	S^{10}	S^9
S^9	S^8	S^1	S^{10}	S^8	S^{10}	S^4
S^{10}	S^5	S^9	S^8	S^8	S^4	S^9

target pollutant. Fivefold cross-validation is also implemented in this step. We maintain the 8 step time window as determined in Sect. 3.2.2. Thus, we keep this result and adjust only the number of input columns. Table 10 shows the effect of involving different numbers of neighbouring stations on the final predictions.

Using three neighbouring stations along with the target station may improve the performance significantly. For example, the obtained average *RMSE* at S^1 equals 56.67 $\mu g/m^3$, improving the *RMSE* to about 3.8 $\mu g/m^3$ from the most degraded results (i.e. five neighbouring stations). In S^3, five neighbouring produces slightly better average *RMSE* than others. However, adding further neighbours does not improve performance and increases computation load. It can also be inferred that increasing the number of columns does not help the model learn the input sets' essential features. Similarly, decreasing the number of neighbours to two degrades performance. Thus, we use three neighbouring stations along with the target station in our model.

3.4 Model architecture evaluation

This section verifies the final model architecture proposed in this study. From the based model, some other alternative autoencoder architectures are derived. The proposed models are created by expanding the base model layers and adjusting the number of output filters. Kernel and stride are maintained to have the exact properties with the base model. Moreover, no padding is applied to all proposed models.

This section demonstrated the results obtained from air quality data in Beijing by selecting CO as the target pollutant. As presented in Table 11, we provided six different autoencoder architectures, labelled as *M*1, *M*2, ..., *M*6. We set kernel size equal to 2, stride equal to 1, and no padding is applied for all layers. The base model used to determine the spatiotemporal characteristics is identified as *M*1. This experiment applies 40% and 20% missing rates for target and neighbouring stations, respectively. Fivefold cross-validation is also performed. Based on the spatiotemporal evaluation, the input sets are fixed in the model selection step to have a size of 8×4.

Based on the final prediction results obtained from each model, *M*6 yields the most accurate imputation results, as shown in Table 12. Out of 10 monitoring stations, *M*6 yields the best prediction in six stations. For example, *M*6 predicts the missing data for S^8 with an *RMSE* value of 240.88 $\mu g/m^3$. It is about 30% better than the base model performance, which yields an *RMSE* value of 346.45 $\mu g/m^3$. In this study, deeper model architectures give better predictions. In most cases, the ten-layered models outperformed the six and eight-layered models. We avoid using deeper architecture as the length of latent space (code) will be very small.

3.5 Imputation performance

This study divides imputation performance into two categories: short-interval with missing rate variations and long-interval consecutive missing data. In this section, we cannot discuss imputations for all stations. However, we try to highlight the essential issues related to our proposed imputation method.

3.5.1 Short-interval imputation

This study defines the term "short-interval" as a missing period generated by removing some values in the actual data with a specific missing rate. The initial random state will determine which values are removed from the actual data. It can be set during the programming set. We intentionally deleted the actual data with four different missing rates (i.e. 20%, 40%, 60% and 80%). Figure 13 shows the example of a test set missing pattern variation at station S^3 in the London dataset. As we can see from the figure, there

Table 10 The average of *RMSE* (std. deviation) after fivefold cross-validation for selecting the number of involved neighbouring stations

	Test period		Number of neighbouring stations			
	Start	End	2	3	4	5
S^1	2019-11-03	2019-11-19	59.05(1.23)	**56.67(2.52)**	59.38(1.09)	60.49(0.83)
S^2	2020-03-28	2020-04-19	11.84(0.79)	**11.45(0.24)**	12.07(1.35)	12.51(1.52)
S^3	2020-05-31	2020-06-29	15.19(0.17)	14.98(0.30)	14.87(0.39)	**14.80(0.18)**
S^4	2020-03-07	2020-04-03	18.67(0.71)	**18.50(1.01)**	21.20(0.48)	20.68(0.36)
S^5	2020-04-11	2020-04-29	19.13(0.36)	**17.82(0.43)**	18.43(0.75)	18.01(0.31)
S^6	2020-03-23	2020-05-14	21.41(1.20)	21.74(1.05)	21.25(0.83)	**21.09(0.71)**
S^7	2020-04-22	2020-05-28	21.78(0.57)	**21.47(0.53)**	21.51(0.58)	21.57(0.78)
S^8	2019-03-16	2019-04-03	45.06(1.66)	**42.80(0.82)**	48.56(3.10)	50.32(2.03)
S^9	2019-05-23	2019-06-13	25.49(2.57)	25.68(1.67)	26.23(3.07)	**25.42(1.00)**
S^{10}	2020-03-29	2020-05-06	15.35(0.49)	**15.24(0.83)**	15.27(0.63)	14.60(0.46)

The best results are indicated in bold

Table 11 Proposed autoencoder architectures

	Assigned output filters
$M1$	$L^1 40 - L^2 30 - L^3 20 - L^4 30 - L^5 40 - L^6 4$
$M2$	$L^1 50 - L^2 40 - L^3 30 - L^4 20 - L^5 30 - L^6 40 - L^7 50 - L^8 4$
$M3$	$L^1 50 - L^2 40 - L^3 30 - L^4 20 - L^5 10 - L^6 20 - L^7 30 - L^8 40 - L^9 50 - L^{10} 4$
$M4$	$L^1 80 - L^2 70 - L^3 50 - L^4 30 - L^5 10 - L^6 30 - L^7 50 - L^8 70 - L^9 80 - L^{10} 4$
$M5$	$L^1 75 - L^2 60 - L^3 45 - L^4 30 - L^5 15 - L^6 30 - L^7 45 - L^8 60 - L^9 75 - L^{10} 4$
$M6$	$L^1 80 - L^2 50 - L^3 30 - L^4 20 - L^5 10 - L^6 20 - L^7 30 - L^8 50 - L^9 80 - L^{10} 4$

Table 12 The average *RMSE* for deep autoencoder architecture selection

	Test period		Autoencoder model					
	Start	End	$M1$	$M2$	$M3$	$M4$	$M5$	$M6$
S^1	2016-06-14	2016-07-26	207.27	210.33	189.25	191.37	189.56	**183.95**
S^2	2016-09-13	2016-10-19	432.21	431.53	435.73	**420.39**	437.41	435.33
S^3	2016-08-10	2016-09-06	204.86	205.77	**191.46**	201.56	202.59	199.66
S^4	2016-10-18	2016-11-25	395.77	388.99	**363.43**	369.30	372.26	373.27
S^5	2016-08-02	2016-08-29	312.01	305.28	311.99	309.76	298.78	**279.61**
S^6	2016-06-14	2016-07-26	207.27	210.33	189.25	191.37	189.56	**183.95**
S^7	2016-08-04	2016-08-25	216.54	213.53	204.10	199.69	203.75	**189.93**
S^8	2016-10-15	2016-11-08	346.45	329.10	248.19	246.75	241.69	**240.88**
S^9	2016-10-09	2016-10-26	349.76	338.14	**287.25**	315.04	312.94	302.92
S^{10}	2016-06-25	2016-07-27	187.51	180.91	164.21	171.75	169.32	**162.83**

The best results are indicated in bold

are 648 hourly samples of NO_2, collected from 20-Feb-2020 13:00:00 to 20-Mar-2020 00:00:00. The white stripes indicate the missing values, which we fill with zero. While the missing rate at the target station is varied, the missing rate at neighbouring stations is fixed at a level of 20%.

Table 13 shows the selected monitoring stations as representatives of short-interval imputation. As presented in the table, we selected two monitoring stations for each city and covered all pollutants in each dataset. Thus, there are 12 experiments in total. Table 14 shows the imputation results in correspondence to the experiment numbers presented in Table 13. The imputation performances are evaluated using three different error metrics, i.e. *RMSE*, *MAE* and R^2.

Among other monitoring stations, our method is less effective in imputing the missing values of NO_2 pollutants in Delhi. In this section, let us exclude the discussion of model performance for NO_2 pollutants in Delhi, as it will be discussed separately in Sect. 3.6. In general, lower missingness levels yield lower *RMSE*/*MAE* values and higher R^2 scores. Due to the physical nature of each pollutant, the *RMSE*/*MAE* values may significantly vary. For example, the values of *RMSE*/*MAE* are considerably higher

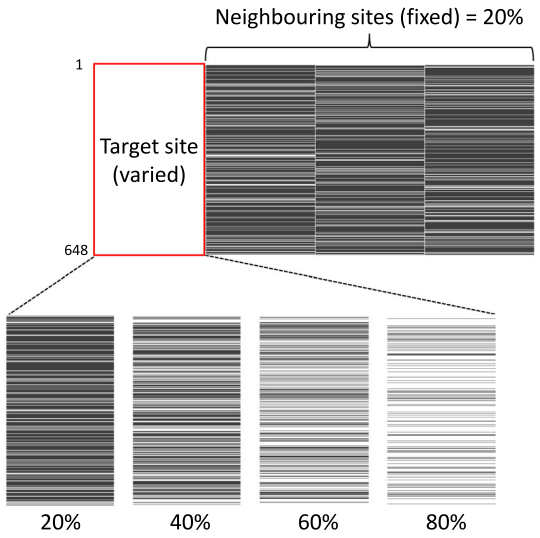

Fig. 13 Short-interval missing patterns in the test set obtained from station S^3 of London dataset

than PM_{10}. Thus, the R^2 score is introduced to see the performance more intuitively. As shown in Table 14, a 20% missing rate results in R^2 scores higher than 0.8 at all target stations, ranging from 0.80 to 0.95. Our proposed

Table 13 Properties of short-interval imputation experiment

No.	City	Station	Pollutant	Train period		Test period	
				Start	End	Start	End
1	London	S3	NO_2	2018-01-01	2019-10-21	2020-02-20	2020-03-20
2	London	S3	PM_{10}	2018-01-01	2019-11-18	2020-03-23	2020-04-20
3	London	S7	NO_2	2018-01-01	2019-09-29	2020-09-23	2020-10-13
4	London	S7	PM_{10}	2018-01-01	2019-11-23	2020-11-03	2020-12-01
5	Delhi	S2	NO_2	2018-02-05	2019-07-25	2020-05-31	2020-07-01
6	Delhi	S2	$PM_{2.5}$	2018-02-03	2019-07-17	2019-08-28	2019-10-27
7	Delhi	S7	NO_2	2018-02-05	2019-07-10	2019-11-21	2019-12-14
8	Delhi	S7	$PM_{2.5}$	2018-02-05	2019-07-16	2020-02-10	2020-04-22
9	Beijing	S1	CO	2013-03-03	2015-08-28	2016-11-11	2016-12-27
10	Beijing	S1	O_3	2013-03-03	2015-08-04	2016-12-10	2016-12-27
11	Beijing	S6	CO	2013-01-03	2015-09-13	2016-06-14	2016-07-26
12	Beijing	S6	O_3	2013-01-03	2015-08-16	2016-06-14	2016-07-26

Table 14 Performance metrics of short-interval imputation for all experiments described in Table 13

Missing rate	Err. metrics	Experiment no.											
		1	2	3	4	5	6	7	8	9	10	11	12
20%	*RMSE*	7.83	5.92	5.24	5.57	7.32	15.49	38.72	16.71	471.88	8.91	92.25	15.17
	MAE	5.99	3.97	3.87	3.90	4.07	9.14	22.94	10.81	296.36	3.93	71.97	10.78
	R^2	0.85	0.81	0.89	0.83	0.44	0.95	0.80	0.85	0.93	0.81	0.89	0.95
40%	*RMSE*	8.51	6.09	5.33	8.63	6.49	16.53	50.26	17.75	559.37	7.56	98.87	18.06
	MAE	6.62	4.08	4.05	4.69	4.11	9.36	28.63	11.10	354.90	3.85	77.51	13.13
	R^2	0.81	0.79	0.88	0.79	0.52	0.94	0.72	0.83	0.91	0.86	0.88	0.93
60%	*RMSE*	10.24	6.93	6.37	8.64	8.14	16.29	69.85	17.85	663.89	7.61	124.59	22.18
	MAE	7.74	4.67	4.99	5.17	4.68	9.55	40.04	11.42	421.72	4.19	93.15	16.66
	R^2	0.74	0.73	0.83	0.80	0.39	0.94	0.49	0.84	0.88	0.85	0.81	0.90
80%	*RMSE*	12.05	7.85	8.19	10.89	8.20	16.42	76.93	18.84	778.34	7.95	149.01	26.59
	MAE	9.21	5.53	6.47	5.77	4.78	9.98	45.75	12.31	514.88	4.76	111.02	20.43
	R^2	0.64	0.65	0.72	0.75	0.39	0.93	0.31	0.83	0.83	0.84	0.73	0.86

method yields satisfying results at this level of missingness. At 40% and 60% of missing levels, our proposed model can maintain its performance by giving R^2 scores between 0.72 and 0.94. At the level of 80%, more errors of imputation decrease the R^2 scores from 0.64 in experiment no. 1 to 0.93 in experiment no.2. For the selected test period, predicting the missing values of $PM_{2.5}$ at S^2 in Delhi (i.e. experiment no. 6) gives the best imputation with R^2 scores higher than 0.9 at all levels of missingness.

With the help of existing neighbouring stations, though prediction errors are inevitable, the proposed autoencoder can learn the input feature and fill the missing values effectively. Even when the target station data are severely corrupted, the proposed model and method can achieve the desired results, as shown in most monitoring stations in Table 14. To sum up, we conclude that our method can produce satisfactory accuracies for short-interval of missing imputation.

3.5.2 Long-interval consecutive imputation

Unlike the short-interval method that generates missing values based on a specific random state, the long-interval consecutive process removes all data at the target station for a specific period. To this end, we set the 400 hours as a minimum missing period. Figure 14 shows a test set pattern of long-interval missing values applied to S^8 (Nongzhanguan) of the Beijing dataset. The set consists of 514

hourly samples, taken from 23-Sep-2016 05:00:00 to 14-Oct-2016 14:00:00. As we can see from the figure, the values at the target set are entirely missing and filled with zeros. A 20% of missingness level is applied to all neighbouring stations. As no values can be obtained from the target station, the autoencoder predicts the missing values entirely based on the existing adjacent data.

There are six experiments conducted to represent long-interval consecutive imputation scenarios, as shown in Table 15. We select one station in each city, and each station covers all pollutant types. Station S^5, S^6 and S^8 represent the London, Delhi and Beijing datasets, respectively. Table 15 also shows the error metrics as the results of the long-interval imputation for specific missing periods. For a minimum of 400 hours (about 17 days) of missing data, our model can impute the missing values with very satisfying results, with some of them yield R^2 scores of 0.90 and higher. However, among the experiments, the S^6 station of Delhi measuring NO_2 produces the lowest R^2 score. The same results are shown in the short-interval imputation, where predicting NO_2 in Delhi consistently yielded the lowest performance. We observed that stations with low correlation coefficients might affect the imputation performance. We will discuss this issue separately in Sect. 3.6.

Figure 15 shows the plots between actual and imputed values for the experiments presented in Table 15. Following work done by [66, 67], the plots also show the 95% confidence interval. In this study, the interval is obtained by adding and subtracting two times of *RMSE* from the imputed values. Implementing the *RMSE* value to form the confidence interval gives a better summary than standard deviation and can be directly helpful in assessing the uncertainty of the imputed values [65]. From Figure 15, we can observe that the imputed values can track the dynamics of actual values. The current neighbour values can help the autoencoder model recognise the missing values at the target station effectively. We are confident that only 5% of imputed values fall outside the shaded interval areas.

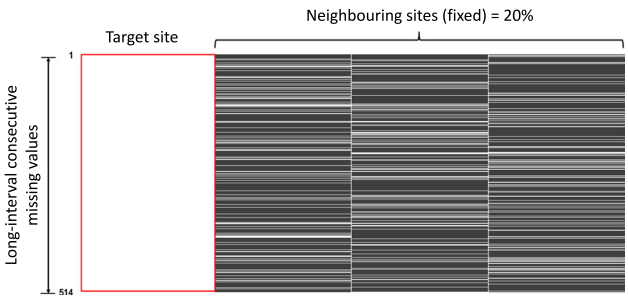

Fig. 14 Long-interval missing patterns in the test set obtained from station S^8 of Beijing dataset

3.6 Effect of correlation level

We observed the possibility of coefficient correlation levels among paired stations affecting the performance of our proposed method. We now focus on stations measuring NO_2 in Delhi, which contribute to poor estimations for both short- and long-interval imputations. The Delhi dataset's coefficient correlations of NO_2 and PM_{10} are shown in Tables 16 and 17. The minimum coefficient for NO_2 is 0.01, which is obtained from the pair of $S^3 - S^6$. The correlation between S^3 and S^{10} even contributes a negative value. Excluding the pair between stations themself, the maximum coefficient correlation of NO_2 is only 0.65, calculated from $S^2 - S^6$. In the same city, monitoring stations measuring PM_{10} yield much stronger correlation coefficients. Computed from the pairs of $S^3 - S^8$ and $S^3 - S^10$, the minimum coefficient is 0.67, whereas the pair of $S^1 - S^2$ contributes the maximum coefficient of 0.90.

Very low coefficient correlation among stations results in highly biased imputation values. We studied these phenomena after conducting various experiments, some of which are shown in Fig. 16. The figure depicts the scatter plots between the actual and imputed NO_2 and $PM_{2.5}$ at S^5 of the Delhi dataset. The experiments are set for a short-interval missing scenario with 20% and 40% missing rates. For NO_2, the test period started on 06-Apr-2020 at 04:00:00 and ended on 29-Apr-2020 at 23:00:00. The period from 22-Feb-2020 at 19:00:00 to 11-Mar-2020 at 14:00:00 is selected for $PM_{2.5}$. As shown in the figure, the imputation results of the two pollutants are considerably different, even the experiments are conducted at the same monitoring station. While the $PM_{2.5}$ imputation results are relatively close to the diagonal line, the missing estimations for NO_2 are more scattered.

Station S^5 and three neighbouring stations (S^7, S^8, and S^2) form the input set of NO_2 pollutants. The $S5 - S7$, $S5 - S8$ and $S5 - S7$ pairs have correlation coefficients of 0.38, 0.37 and 0.35, respectively. These values are low correlations. For $PM_{2.5}$ pollutants, the input sets are formed by the joint stations S^5, S^2, S^{10} and S^1. The computed correlation coefficients for $S^5 - S^2$, $S^5 - S^{10}$ and $S^5 - S^1$ are 0.90, 0.88 and 0.86, respectively. These coefficients are much stronger. Low correlations affect the input values by causing the sets to look more randomised. This results in neighbouring station data that does not contribute enough knowledge to the model. Figure 17 illustrates this issue more intuitively. The figure shows the first input set fed to the model with 40% missing rate for both NO_2 and $PM_{2.5}$. As we can see from the figure, the reconstructed input of NO_2 is less accurate than the reconstructed input of $PM_{2.5}$. It is obvious that the effect of weak correlation causes a

Table 15 Results of long-interval consecutive imputation

No.	City	Station	Pollutant	Start	End	*RMSE*	*MAE*	R^2
1	London	S^5	NO_2	2021-01-12	2021-01-30	5.840	4.371	0.845
2	London	S^5	PM_{10}	2020-11-10	2020-12-27	3.57	2.29	0.79
3	Delhi	S^6	NO_2	2020-05-14	2020-06-12	12.16	8.56	0.47
4	Delhi	S^6	$PM_{2.5}$	2019-09-29	2019-11-20	49.99	31.44	0.90
5	Beijing	S^8	CO	2016-03-25	2016-04-20	132.91	96.81	0.95
6	Beijing	S^8	O_3	2016-09-23	2016-10-14	13.91	8.29	0.92

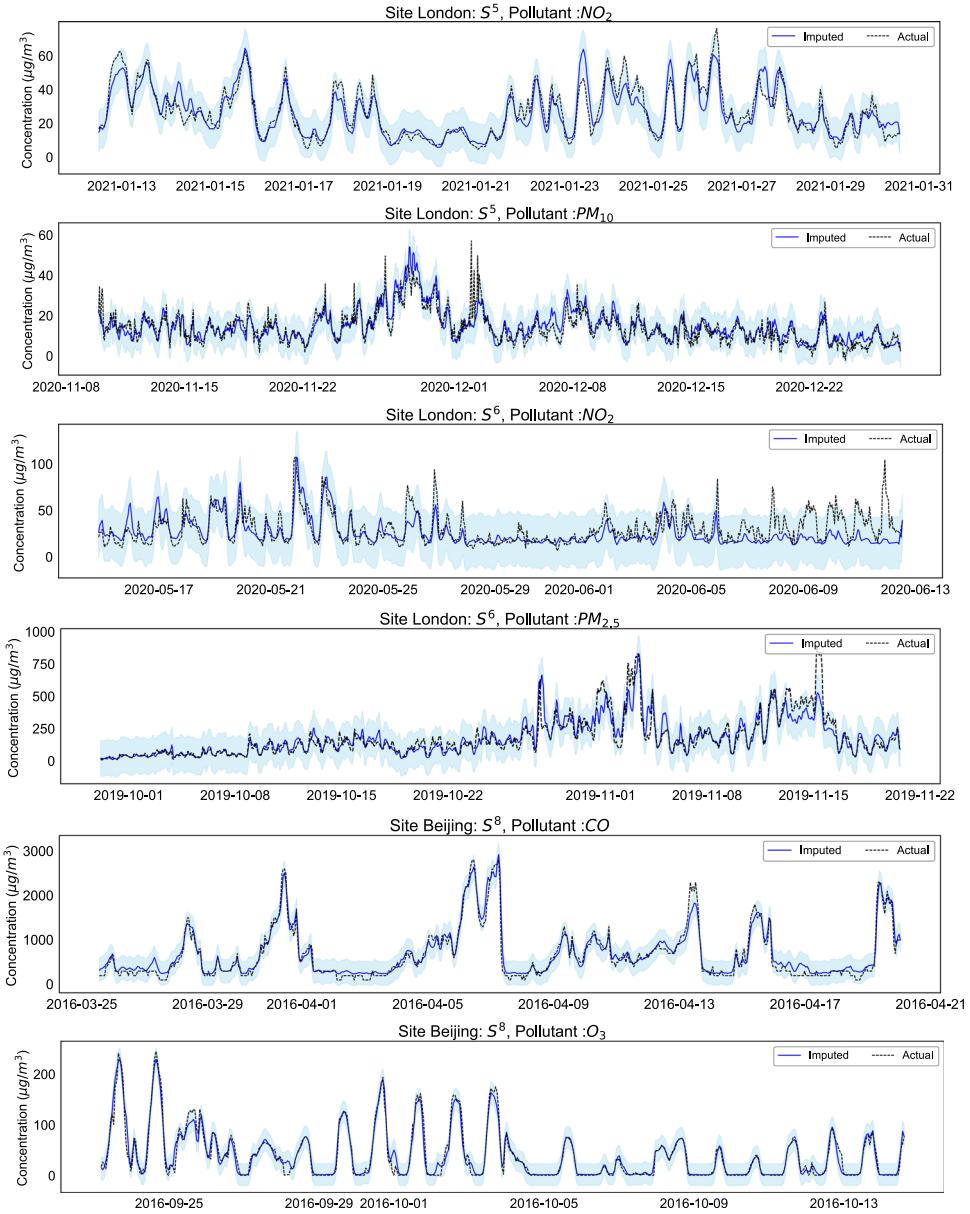

Fig. 15 Plot of long-interval missing imputation between actual and imputed values along with 95% confidence intervals. The properties of each figure are shown in Table 15

Table 16 Coefficient of correlation among stations measuring NO_2 in Delhi air quality data

	S^1	S^2	S^3	S^4	S^5	S^6	S^7	S^8	S^9	S^{10}
S^1	1.00									
S^2	0.41	1.00								
S^3	0.04	0.07	1.00							
S^4	0.14	0.32	− 0.03	1.00						
S^5	0.24	0.35	0.13	0.16	1.00					
S^6	0.35	0.65	0.01	0.27	0.30	1.00				
S^7	0.12	0.26	0.05	0.29	0.38	0.24	1.00			
S^8	0.50	0.58	0.06	0.33	0.37	0.55	0.26	1.00		
S^9	0.32	0.21	0.09	0.00	0.02	0.22	0.07	0.29	1.00	
S^{10}	0.33	0.49	− 0.24	0.34	0.27	0.55	0.30	0.48	0.12	1.00

Table 17 Coefficient of correlation among stations measuring $PM_{2.5}$ in Delhi air quality data

	S^1	S^2	S^3	S^4	S^5	S^6	S^7	S^8	S^9	S^{10}
S^1	1.00									
S^2	0.90	1.00								
S^3	0.71	0.72	1.00							
S^4	0.86	0.84	0.72	1.00						
S^5	0.86	0.90	0.69	0.81	1.00					
S^6	0.81	0.84	0.71	0.81	0.84	1.00				
S^7	0.82	0.84	0.76	0.82	0.83	0.87	1.00			
S^8	0.83	0.82	0.67	0.75	0.76	0.71	0.74	1.00		
S^9	0.85	0.85	0.73	0.82	0.81	0.79	0.80	0.79	1.00	
S^{10}	0.85	0.89	0.67	0.81	0.88	0.82	0.78	0.76	0.80	1.00

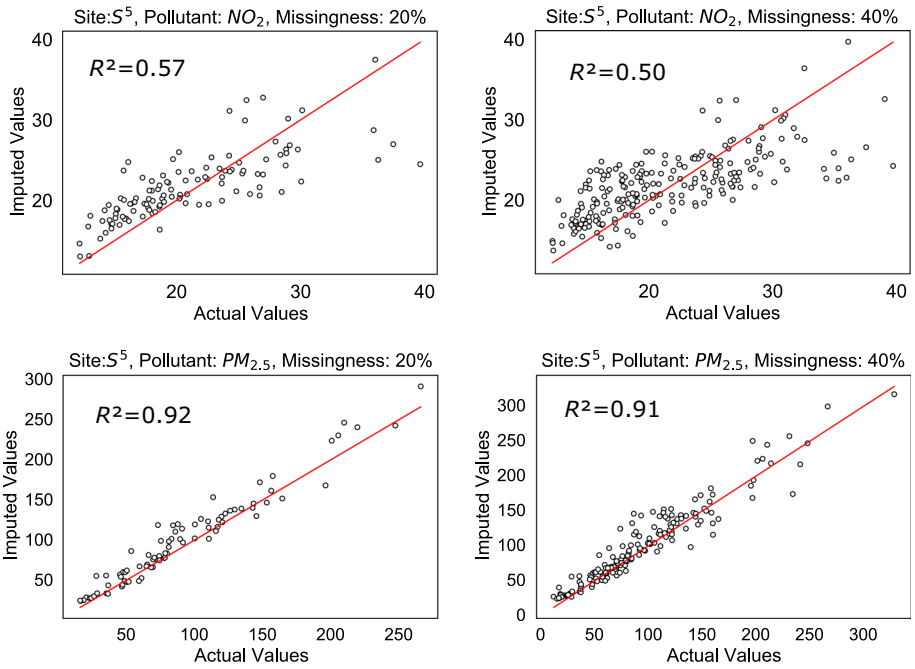

Fig. 16 Scatter plot of short-interval imputation at station S^5, with 20% and 40% of missingness levels

significant difference of contained values among columns in the input set, making it difficult for the model to estimate the missing parts.

3.7 Comparison with other methods

This section verifies the effectiveness of our proposed model in comparison with the existing methods. In this study, univariate and multivariate imputation methods are introduced. Imputing missing values using the univariate method is based only on the existing values in that feature dimension. In contrast, the multivariate imputation tries to utilise the non-missing data in the entire feature dimensions to estimate the missing values. The selected univariate imputations are most frequent, median and mean. Four estimators are used for multivariate imputation: Bayesian ridge, decision tree, extra-trees and *k*-nearest neighbours.

We demonstrated the effectiveness of our proposed model against other methods for all monitoring stations. In total, we conducted 60 experiments to cover different cities, stations and pollutants in our datasets. For the London dataset, the training data for NO_2 and PM_{10} are from January 2018 to around October 2019, whereas the test sets are taken from several unbroken segments from around November 2019 to January 2021. We combined short- and long-intervals perturbation procedures for the training and test sets. The perturbation step removes about 45% of the target training set and 50% of the test set. To obtain less biased results, we implemented fivefold cross-validation in the dataset.

For all pollutant data in the Delhi dataset (i.e. NO_2 and $PM_{2.5}$), the training period is from February 2018 to mid-July 2019, whereas the test period starts in July 2019 and ends in July 2020. The same perturbation procedures as the London dataset are applied for Delhi data, resulting in missing rates of about 45% and 60% for training and test sets in the target station. Pollutant CO and O_3 in Beijing monitoring stations are treated in the same way. The training data are selected from March 2013 to around September 2015, whereas the training data are chosen from September 2015 to February 2017. The missing values in the target station for training and test steps are maintained at the rate of 45% and 50%, respectively. The bar charts to visualise the proportion of the *RMSE* values obtained from each method are shown in Fig. 18.

Figure 18 presents the performance of our proposed model and seven commonly seen imputation methods and contains the following abbreviations: *Most* (most frequent imputation), *Med* (median imputation), *Mean* (mean imputation), *DecT* (decision tree regressor), *ExT* (extra-trees regressor), *KNN* (*k*-nearest neighbours regressor), *BaR* (Bayesian ridge regressor) and *Aut*∗ (proposed autoencoder). The proposed autoencoder charts are indicated with black-filled areas.

As we can see from the figure, the univariate imputation using statistic properties (most frequent, median and mean) yields the most inaccurate imputation results. Compared to the univariate, the multivariate imputation techniques return significantly lower imputation errors. Our proposed method outperforms other methods for all stations and pollutants except for Delhi monitoring stations measuring NO_2. Out of ten stations, other methods yield slightly better performance in three monitoring stations (i.e. S^3, S^5 and S^9). As discussed in the previous section, weak correlations among stations lower our proposed method performance.

Figure 19 shows the rate of improvement on *RMSE* (*RIR*). Positive *RIR* values indicate that our proposed model outperforms other methods. In contrast, negative *RIR* values imply other models have better performance than ours. Compared to the most frequent, median and mean imputations, our proposed autoencoder model can

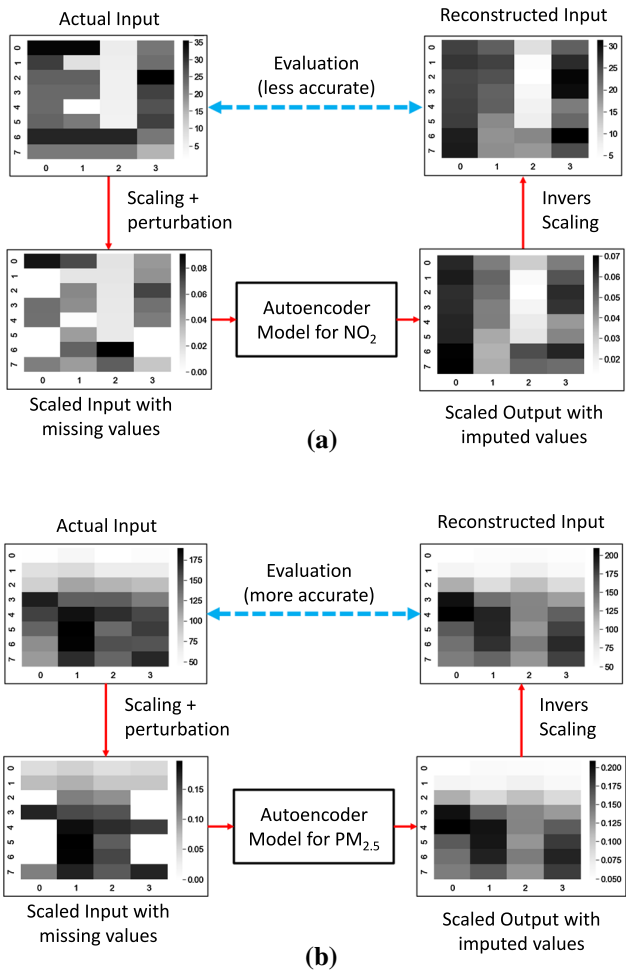

Fig. 17 Example of input and output sets retrieval before and after denoising process in Delhi station S^5: (a) retrieval of NO_2 and (b) retrieval of $PM_{2.5}$

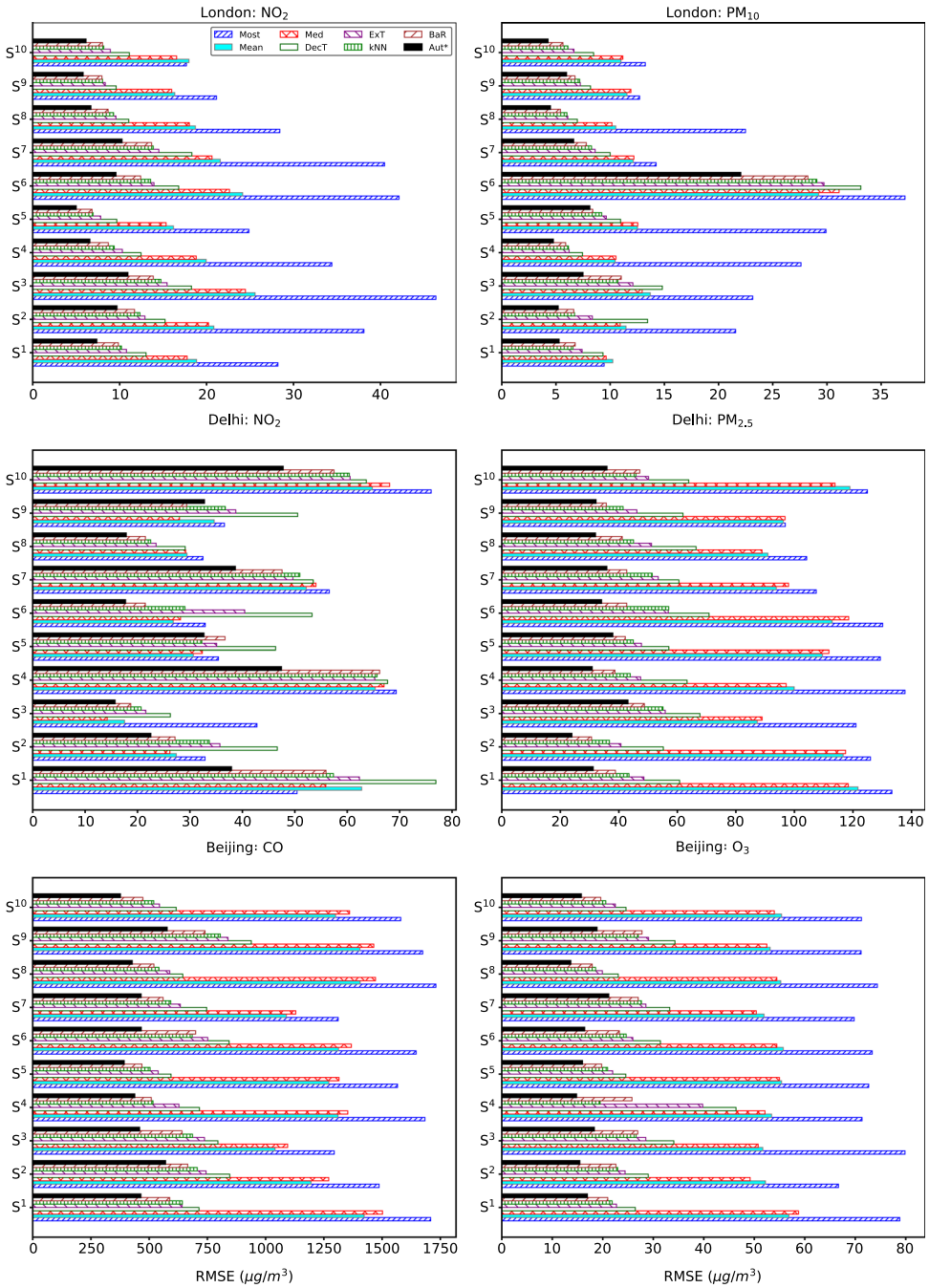

Fig. 18 Performance comparison of the proposed model against commonly used methods

significantly improve the *RMSE* values, ranging from 50 to 80 per cent in most cases. Our proposed method also contributes positive *RIR* values for Bayesian ridge, decision tree, extra-trees and k-nearest neighbours imputation methods, mainly improving between 10% and 50%. For Delhi measuring NO_2, our proposed method contributes six negative *RIR* values; half of them occur in station S^5. In monitoring S^5, mean, median and kNN imputations perform better than our proposed model, marginally improving 6.46%, 0.87% and 1.15% of *RIR* values, respectively. Out of six negative *RIR* values, half of them are caused by median imputation. Median imputation contributes the lowest *RMSE* for monitoring station S^9, about 17% better than our proposed model.

To acquire global knowledge, we calculated the average *RIR* values of each imputation method from all stations and pollutants. The results are summarised in Table 18. Our model outperforms the univariate imputations, improving the average *RIR* from around 50 to 65 per cent. For multivariate imputation, the average *RIR* improvement range is from about 20 to 40%.

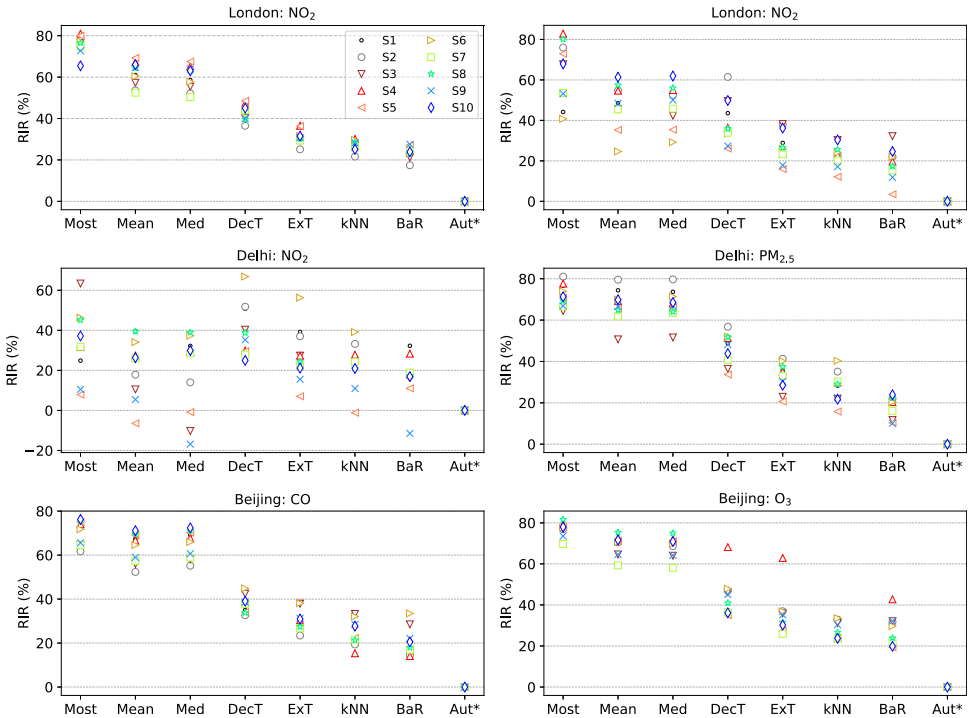

Fig. 19 Performance comparison of the proposed model against commonly used methods

4 Conclusions

Missing values are common real-world scenarios with collected data. Due to many factors, every measurement system can face this issue, and some of them may suffer from losing critical data. The existence of missing data can influence the study interpretations and affect the functioning of air quality-related public services. A strategy to overcome missing data needs to be addressed by proposing an imputation method. Moreover, understanding the spatiotemporal characteristics of air pollutant data can improve the robustness of air quality missing data imputation.

This study has tried to address the challenges of implementing a suitable method for air quality missing data imputation. Inspired by the capabilities of the denoising autoencoder to reconstruct the corrupted data, we proposed an imputation method that exploits both temporal and spatial data to improve imputation accuracy. We determined an ideal temporal window size of 8-time steps and a spatial combination of 3 neighbouring stations to provide an 8×4 input set to the model. The aggregation of the input sets is performed to obtain a single prediction at a specific time. In this study, two imputation scenarios are conducted, namely short-interval imputation and long-interval consecutive imputation. For the short-interval imputation, some levels of missingness are introduced (i.e. 20%, 40%, 60% and 80%). However, all data in a specific period are removed during the long-interval imputation steps.

Results show that our proposed method and model give satisfying imputation results with $R^2 \geq 0.6$, even when the data in the target station are entirely missing. Degraded imputation performances arises when among stations are weekly correlated. Low correlation coefficients compose more irregular input values, making our proposed autoencoder model unable to recover noisy inputs. Compared to univariate imputation techniques, our model improves up to 65% of average *RIR* and 20% - 40% against the multivariate imputation techniques.

Table 18 Average of *RIR* values calculated from all stations

Method	Average of $RIR^{(existing,proposed)}(\%)$
Most frequent	65.21
Mean	55.14
Median	54.33
Decision tree	41.69
Extra-trees	30.66
k-nearest neighbours	25.45
Bayesian ridge	20.82
Proposed autoencoder	0.00

Appendix A

Algorithm for post-training model outputs interpretation

Algorithm 1 Post-training model outputs interpretation

Input: Given test sets $\mathbf{X}$
Output: Series of imputed missing values $\mathbf{S}$

1: $\mathbf{Y} \Leftarrow \text{model\_predict}(\mathbf{X})$
2: $\mathbf{YY} \Leftarrow \text{invers\_transform}(\mathbf{Y})$
3: $\mathbf{YY} \Leftarrow \mathbf{YY}[:, :, 1]$
4: $r_y \Leftarrow \text{row\_size}(\mathbf{YY})$
5: $\mathbf{Z} \Leftarrow \text{zeros}(r_y, r_y - 1)$
6: $\mathbf{A} \Leftarrow \text{concatenate } ((\mathbf{YY}, \mathbf{Z}), axis = 1)$
7: $r_a \Leftarrow \text{row\_size}(\mathbf{A})$
8: **for** $i = 0, r_a$ **do**
9: $\quad \mathbf{A}[i, :] \Leftarrow \text{roll}(\mathbf{A}[i, :], i)$
10: **end for**
11: $\mathbf{S} \Leftarrow \text{sum}(\mathbf{A}, axis = 0)$
12: $l_s \Leftarrow \text{column\_size}(\mathbf{S})$
13: $l_p \Leftarrow 8$
14: **for** $i = 0, l_p - 1$ **do**
15: $\quad j \Leftarrow i + 1$
16: $\quad \mathbf{S}[i] \Leftarrow \mathbf{S}[i]/j$
17: $\quad \mathbf{S}[-j] \Leftarrow \mathbf{S}[-j]/j$
18: **end for**
19: **for** $i = l_p - 1, (l_s - l_p + 1)$ **do**
20: $\quad \mathbf{S}[i] \Leftarrow \mathbf{S}[i]/l_p$
21: **end for**

As shown in Algorithm 1, the test sets $\mathbf{X}$ are fed to the model (row 1), resulting in a 3-dimension prediction set $\mathbf{Y}$ with a size of (n, 8, 4), where n is the number of test sets fed to the model. The prediction set $\mathbf{Y}$ must be scaled back to their original values, resulting in a matrix $\mathbf{YY}$ (row 2). To minimise the computing process, we selected only the target station predictions. The target station prediction values are obtained by extracting the first column of each output set (row 3). This process results in a 2D matrix $\mathbf{YY}$ with a size of (n, 8). Next, the following row is right-shifted one step from the previous row (rows 8:10), provided that there is a sparse matrix $\mathbf{A}$ appropriate to handle this rolling scheme (rows 4:6). The sums of each column are computed to get a single row matrix $\mathbf{S}$ (row: 11). As the matrix $\mathbf{S}$ is obtained from different layers of overlapped values (see Fig. 6), the divisors of each element in $\mathbf{S}$ are varied (rows 14:21). For the first seven elements of $\mathbf{S}$, divisors are increased from 1 to 7, whereas for the last seven elements, divisors are decreased from 7 to 1. All values between the seven first and seven last elements of $\mathbf{S}$ are equally divided by 8.

Acknowledgements INK Wardana was supported with a studentship from Indonesia Endowment Fund for Education (LPDP).

Funding This work was supported in part by Indonesia Endowment Fund for Education (LPDP), Ministry of Finance, Republic of Indonesia under grant number Ref: S-1027/LPDP.4/2019.

Declarations

Conflict of interest The authors declare that they have no conflict of interest with respect to the research, authorship and/or publication of this article.

References

1. Ameer S et al (2019) Comparative analysis of machine learning techniques for predicting air quality in smart cities. IEEE Access 7:128325–128338. https://doi.org/10.1109/ACCESS.2019.2925082
2. Alsaber AR, Pan J, Al-Hurban A (2021) Handling complex missing data using random forest approach for an air quality monitoring dataset: A case study of kuwait environmental data (2012 to 2018). Int J Environ Res Public Health 18(3):1333. https://doi.org/10.3390/ijerph18031333
3. Ma J et al (2020) Air quality prediction at new stations using spatially transferred bi-directional long short-term memory network. Sci Total Environ 705:135771. https://doi.org/10.1016/j.scitotenv.2019.135771
4. Zhang Z, Zhang G, Su B (2021) The spatial impacts of air pollution and socio-economic status on public health: empirical evidence from china. Soc-Econom Plan Sci p. 101167. https://doi.org/10.1016/j.seps.2021.101167
5. Guo Y et al (2016) The association between lung cancer incidence and ambient air pollution in china: a spatiotemporal analysis. Environ Res 144:60–65. https://doi.org/10.1016/j.envres.2015.11.004
6. Hamra GB et al (2014) Outdoor particulate matter exposure and lung cancer: a systematic review and meta-analysis. Environ Health Perspect 122(9):906–911. https://doi.org/10.1289/ehp/1408092
7. Chen Q et al (2021) Air pollution and cardiovascular mortality in nanjing, china: evidence highlighting the roles of cumulative exposure and mortality displacement. Chemosphere 265. https://doi.org/10.1016/j.chemosphere.2020.129035
8. Saygin H, Mercan Y, Yorulmaz F (2021) The association between air pollution parameters and emergency department visits and hospitalizations due to cardiovascular and respiratory diseases: a time-series analysis. Int Arch Occup Environ Health. https://doi.org/10.1007/s00420-021-01769-w

9. Ma Y et al (2017) Short-term effects of air pollution on daily hospital admissions for cardiovascular diseases in western china. Environ Sci Pollut Res 24(16):14071–14079. https://doi.org/10.1007/s11356-017-8971-z
10. Delgado-Saborit JM et al (2021) A critical review of the epidemiological evidence of effects of air pollution on dementia, cognitive function and cognitive decline in adult population. Sci Total Environ 757:143734. https://doi.org/10.1016/j.scitotenv.2020.143734
11. Li C, Managi S (2022) Spatial variability of the relationship between air pollution and well-being. Sustain Cities Soc 76:103447. https://doi.org/10.1016/j.scs.2021.103447
12. Sivarethinamohan R et al. (2021) Impact of air pollution in health and socio-economic aspects: review on future approach. Mater. Today: Proceed 37: 2725–2729. https://doi.org/10.1016/j.matpr.2020.08.540, international Conference on Newer Trends and Innovation in Mechanical Engineering: Materials Science
13. Institute HE (2019) State of global air 2019 special report. Health Effects Institute
14. Zhou X-H (2020) Challenges and strategies in analysis of missing data. Biostatistics & Epidemiol 4(1):15–23. https://doi.org/10.1080/24709360.2018.1469810
15. Yu Y, Yu JJQ, Li VOK, Lam JCK (2020) A novel interpolation-svt approach for recovering missing low-rank air quality data. IEEE Access 8:74291–74305. https://doi.org/10.1109/ACCESS.2020.2988684
16. Austin PC, White IR, Lee DS, van Buuren S (2021) Missing data in clinical research: a tutorial on multiple imputation. Can J Cardiol 37(9):1322–1331. https://doi.org/10.1016/j.cjca.2020.11.010
17. Ma J et al (2020) A bi-directional missing data imputation scheme based on LSTM and transfer learning for building energy data. Energy and Build 216. https://doi.org/10.1016/j.enbuild.2020.109941
18. Laña I, Olabarrieta II, Vélez M, Ser JD (2018) On the imputation of missing data for road traffic forecasting: new insights and novel techniques. Trans Res Part C: Emerg Technol 90:18–33. https://doi.org/10.1016/j.trc.2018.02.021
19. Ma J et al (2020) Transfer learning for long-interval consecutive missing values imputation without external features in air pollution time series. Adv Eng Inform 44:101092. https://doi.org/10.1016/j.aei.2020.101092
20. Pena M, Ortega P, Orellana M (2019) A novel imputation method for missing values in air pollutant time series data. In: IEEE latin American conference on computational intelligence (LA-CCI). https://doi.org/10.1109/LA-CCI47412.2019.9037053
21. Moshenberg S, Lerner U, Fishbain B (2015) Spectral methods for imputation of missing air quality data. Environ Syst Res 4(1):26. https://doi.org/10.1186/s40068-015-0052-z
22. Rubin DB (1976) Inference and missing data. Biometrika 63(3):581–592. https://doi.org/10.1093/biomet/63.3.581
23. Gómez-Carracedo M, Andrade J, López-Mahía P, Muniategui S, Prada D (2014) A practical comparison of single and multiple imputation methods to handle complex missing data in air quality datasets. Chemom Intell Lab Syst 134:23–33. https://doi.org/10.1016/j.chemolab.2014.02.007
24. Junger W, Ponce de Leon A (2015) Imputation of missing data in time series for air pollutants. Atmos Environ 102:96–104. https://doi.org/10.1016/j.atmosenv.2014.11.049
25. Hadeed SJ, O'Rourke MK, Burgess JL, Harris RB, Canales RA (2020) Imputation methods for addressing missing data in short-term monitoring of air pollutants. Sci Total Environ 730:139140. https://doi.org/10.1016/j.scitotenv.2020.139140
26. Donders ART, van der Heijden GJ, Stijnen T, Moons KG (2006) Review: A gentle introduction to imputation of missing values. J Clin Epidemiol 59(10):1087–1091. https://doi.org/10.1016/j.jclinepi.2006.01.014
27. Graham JW (2009) Missing data analysis: Making it work in the real world. Annu Rev Psychol 60(1):549–576. https://doi.org/10.1146/annurev.psych.58.110405.085530
28. Plaia A, Bondì A (2006) Single imputation method of missing values in environmental pollution data sets. Atmos Environ 40(38):7316–7330. https://doi.org/10.1016/j.atmosenv.2006.06.040
29. Zhou X, Liu X, Lan G, Wu J (2021) Federated conditional generative adversarial nets imputation method for air quality missing data. Knowl-Based Syst 228:107261. https://doi.org/10.1016/j.knosys.2021.107261
30. Zhang Y-F, Thorburn PJ, Xiang W, Fitch P (2019) Ssim–a deep learning approach for recovering missing time series sensor data. IEEE Internet Things J 6(4):6618–6628. https://doi.org/10.1109/JIOT.2019.2909038
31. Vincent P, Larochelle H, Bengio Y, Manzagol P-A (2008) Extracting and composing robust features with denoising autoencoders. In: International Conference on Machine learning (ICML'08)https://doi.org/10.1145/1390156.1390294
32. Saleh Ahmed A, El-Behaidy WH, Youssif AA (2021) Medical image denoising system based on stacked convolutional autoencoder for enhancing 2-dimensional gel electrophoresis noise reduction. Biomed Signal Process Control 69:102842. https://doi.org/10.1016/j.bspc.2021.102842
33. Juneja M et al (2021) Denoising of magnetic resonance imaging using bayes shrinkage based fused wavelet transform and autoencoder based deep learning approach. Biomed Signal Process Control 69:102844. https://doi.org/10.1016/j.bspc.2021.102844
34. Fang Z et al (2018) Laser stripe image denoising using convolutional autoencoder. Results in Phys 11:96–104. https://doi.org/10.1016/j.rinp.2018.08.023
35. Bajaj K, Singh DK, Ansari MA (2020) Autoencoders based deep learner for image denoising. Procedia Comput Sci 171:1535–1541. https://doi.org/10.1016/j.procs.2020.04.164, third International Conference on Computing and Network Communications (CoCoNet'19)
36. Dasan E, Panneerselvam I (2021) A novel dimensionality reduction approach for ecg signal via convolutional denoising autoencoder with lstm. Biomed Signal Process Control 63:102225. https://doi.org/10.1016/j.bspc.2020.102225
37. Nagar S, Kumar A, Swamy M (2021) Orthogonal features-based eeg signal denoising using fractionally compressed autoencoder. Signal Process 188:108225. https://doi.org/10.1016/j.sigpro.2021.108225
38. Zhu H, Cheng J, Zhang C, Wu J, Shao X (2020) Stacked pruning sparse denoising autoencoder based intelligent fault diagnosis of rolling bearings. Appl Soft Comput 88:106060. https://doi.org/10.1016/j.asoc.2019.106060
39. Meng Z, Zhan X, Li J, Pan Z (2018) An enhancement denoising autoencoder for rolling bearing fault diagnosis. Measurement 130:448–454. https://doi.org/10.1016/j.measurement.2018.08.010
40. Gondara L, Wang K (2018) MIDA: Multiple imputation using denoising autoencoders. arXiv:1705.02737v3
41. Abiri N, Linse B, Edén P, Ohlsson M (2019) Establishing strong imputation performance of a denoising autoencoder in a wide range of missing data problems. Neurocomputing 365:137–146. https://doi.org/10.1016/j.neucom.2019.07.065
42. Jiang B, Siddiqi MD, Asadi R, Regan A (2021) Imputation of missing traffic flow data using denoising autoencoders. Procedia Comput Sci 184: 84–91. https://doi.org/10.1016/j.procs.2021.03.122, the 12th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 4th International

Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops
43. Alamoodi A et al (2021) Machine learning-based imputation soft computing approach for large missing scale and non-reference data imputation. Chaos, Solitons & Fractals 151:111236. https://doi.org/10.1016/j.chaos.2021.111236
44. Abirami S, Chitra P (2021) Regional air quality forecasting using spatiotemporal deep learning. J Clean Prod 283:125341. https://doi.org/10.1016/j.jclepro.2020.125341
45. Castelli M, Clemente FM, Popovič A, Silva S, Vanneschi L (2020) A machine learning approach to predict air quality in california. Complexity 2020:1–23. https://doi.org/10.1155/2020/8049504
46. Carslaw DC, Ropkins K (2012) openair — an r package for air quality data analysis. Environ Modell Softw 27–28:52–61. https://doi.org/10.1016/j.envsoft.2011.09.008
47. Rao R (2021) Air quality data in india (2015 - 2020). https://www.kaggle.com/rohanrao/air-quality-data-in-india
48. Zhang S et al (2017) Cautionary tales on air-quality improvement in beijing. Proceed Royal Soc A: Math Phys Eng Sci 473(2205):20170457. https://doi.org/10.1098/rspa.2017.0457
49. Dua D, Graff C (2017) UCI machine learning repository. http://archive.ics.uci.edu/ml
50. Carter N (ed.) (2020) *Data Science for Mathematicians* (Chapman and Hall/CRC)
51. Jebli I, Belouadha F-Z, Kabbaj MI, Tilioua A (2021) Prediction of solar energy guided by pearson correlation using machine learning. Energy 224:120109. https://doi.org/10.1016/j.energy.2021.120109
52. pandas development team T (2020) pandas-dev/pandas: Pandas, latest. https://doi.org/10.5281/zenodo.3509134
53. Qi Y, Li Q, Karimian H, Liu D (2019) A hybrid model for spatiotemporal forecasting of PM2.5 based on graph convolutional neural network and long short-term memory. Sci Total Environ 664:1–10. https://doi.org/10.1016/j.scitotenv.2019.01.333
54. Silva-Ramírez E-L, Cabrera-Sánchez J-F (2021) Co-active neuro-fuzzy inference system model as single imputation approach for non-monotone pattern of missing data. Neural Comput Appl 33(15):8981–9004. https://doi.org/10.1007/s00521-020-05661-5
55. Abadi M et al. (2015) TensorFlow: large-scale machine learning on heterogeneous systems. https://www.tensorflow.org/. Software available from tensorflow.org
56. Chollet F et al. (2015) Keras. https://keras.io
57. Harris CR et al (2020) Array programming with NumPy. Nature 585(7825):357–362. https://doi.org/10.1038/s41586-020-2649-2
58. Pedregosa F et al (2011) Scikit-learn: Machine learning in Python. J Mach Learn Res 12:2825–2830
59. Hunter JD (2007) Matplotlib: A 2d graphics environment. Comput Sci Eng 9(3):90–95. https://doi.org/10.1109/mcse.2007.55
60. Waskom M (2021) seaborn: statistical data visualization. J Open Source Soft 6(60):3021. https://doi.org/10.21105/joss.03021
61. Kingma DP (2014) & Ba, J. A method for stochastic optimization, Adam arXiv:1412.6980
62. Goodfellow I, Bengio Y, Courville A (2016) *Deep Learning* (MIT Press). http://www.deeplearningbook.org
63. Wardana INK, Gardner JW, Fahmy SA (2021) Optimising deep learning at the edge for accurate hourly air quality prediction. Sensors 21(4):1064. https://doi.org/10.3390/s21041064
64. Chicco D, Warrens MJ, Jurman G (2021) The coefficient of determination r-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. PeerJ Comput Sci 7. https://doi.org/10.7717/peerj-cs.623
65. Council NR (1991) Improving information for social policy decisions - the uses of microsimulation modeling. National Academies Press, Washington
66. Noori R, Hoshyaripour G, Ashrafi K, Araabi BN (2010) Uncertainty analysis of developed ANN and ANFIS models in prediction of carbon monoxide daily concentration. Atmos Environ 44(4):476–482
67. Moazami S et al (2016) Reliable prediction of carbon monoxide using developed support vector machine. Atmos Pollut Res 7(3):412–418