Applied Soft Computing 113 (2021) 107850

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A time series forecasting based multi-criteria methodology for air quality prediction

Raquel Espinosa [a], José Palma [a], Fernando Jiménez [a,*], Joanna Kamińska [b], Guido Sciavicco [c], Estrella Lucena-Sánchez [c,d]

[a] Department of Information and Communication Engineering, University of Murcia, Spain
[b] Dept. of Math., Wrocław University of Environmental and Life Sciences, Poland
[c] Dept. of Math. and Comp. Sci., University of Ferrara, Italy
[d] Dept. of Phy., Inf., and Math., University of Modena e Reggio Emilia, Italy

ARTICLE INFO

Article history:
Received 22 February 2021
Received in revised form 4 August 2021
Accepted 20 August 2021
Available online 7 September 2021

Keywords:
Air quality
Multivariate time series forecasting
Deep learning
Multi-criteria decision support systems

ABSTRACT

There is a very extensive literature on the design and test of models of environmental pollution, especially in the atmosphere. Current and recent models, however, are focused on explaining the causes and their temporal relationships, but do not explore, in full detail, the performances of pure forecasting models. We consider here three years of data that contain hourly nitrogen oxides concentrations in the air; exposure to high concentrations of these pollutants has been indicated as potential cause of numerous respiratory, circulatory, and even nervous diseases. Nitrogen oxides concentrations are paired with meteorological and vehicle traffic data for each measure. We propose a methodology based on exactness and robustness criteria to compare different pollutant forecasting models and their characteristics. 1DCNN, GRU and LSTM deep learning models, along with Random Forest, Lasso Regression and Support Vector Machines regression models, are analyzed with different window sizes. As a result, our best models offer a 24-hours ahead, very reliable prediction of the concentration of pollutants in the air in the considered area, which can be used to plan, and implement, different kinds of interventions and measures to mitigate the effects on the population.

1. Introduction

Contamination of the air, in particular in metropolitan areas, is a very well-known problem. The ever-growing population of cities and the increasing level of motorization contribute to the ever-increasing traffic volume, and consequently, the ever-increasing exhaust gases emissions. At the same time, the thickening of city buildings reduces ventilation and increases the porosity of surface, which ends up decreasing the effect of the wind on the evacuation of contamination. Wrocław (Poland) is a city founded in the 10th century, and it counts, currently, 641,000 residents. About 15 thousands of vehicles are estimated to move in city streets every day [1]. Among the main contaminants emitted by car engines are nitrogen oxides: NO_2 and $NO + NO_2$ (usually denoted by NO_x). The typical sources of air pollution are well-known, but difficult to eliminate, at least completely. Thus, most studies are focused on determining the impact of factors that may modify the concentrations of contaminants in the atmosphere such as transformation, retention or evacuation. To this end, models are created to describe the underlying phenomena with more or less detail. Recognition of factors having the highest effect on the concentration of contaminants in the air gives the opportunity to try to manipulate these factors in such a way as to ensure the most effective evacuation of such pollutants, thereby shortening the time of exposition to their effects and reducing the results of their action. The research on the impacts of car traffic and of meteorological factors on the concentrations of NO_2 and NO_x in the urban agglomeration air gives us the opportunity to try manipulating such factors and predicting the time and the conditions of maximal presence of contaminants in the air. Contamination models, and in particular early prediction models, can support designers for taking actions towards the improvements the quality of the air [2]. In central Europe, and in Poland in particular, this problem is of uttermost importance: it is estimated that the economic cost of air pollution in Poland is over 25 million euros per year, and that over 43 thousands people die prematurely in the country because of poor air quality. Several models for air contamination exist in the literature. The most basic approach is based on multidimensional regression models [3]. The most relevant advantage of simple regression models such as the linear one is having explainable models that can be used to assess the amount of the impact of

* Corresponding author.
E-mail address: fernan@um.es (F. Jiménez).

https://doi.org/10.1016/j.asoc.2021.107850

each predictor on the value of the explaining variable. Moreover, linear models can be extended to take into account polynomial effects of some variables [4], and the temporal aspects of the physical process can be also studied explicitly [5]. Highly non-linear models, on the other hand, have the advantage of being more accurate, especially for long-term forecasting. Typical black-box models of this kind are neural networks, in several variants (see, e.g. [6]). Also, alternative approaches to pollutants prediction and explanation in the atmosphere include, for data from the area of Wrocław [7], in which, in particular, the authors propose the use of lagged variables to enhance the accuracy of the prediction models, and proved the importance of their role.

We present a methodology to evaluate and compare deep learning models for multivariate time series forecasting, that includes lagged transformations, hyper-parameter tuning, statistical tests, multi-criteria decision making and *h*-step-ahead prediction. In particular, we compare three deep learning techniques and three conventional regression techniques that is, *random forest* (RF) [8], *least absolute shrinkage and selection operator* (Lasso) regression [9], and *support vector machine* (SVM) [10] against each other on the data containing traffic values, meteorological values, and pollution values in a highly trafficked street crossing in Wrocław, from 2015 to 2017. We focus on the quality and the performances of the prediction model, measured up to 24 hours-ahead prediction. Our data are collected from a very large street crossing in Wrocław, that encompasses several traffic lanes, located not far (about 30 m) from the measuring station, and monitored by traffic cameras. The station is located on the suburbs of the city, at 9.6 km from the airport. Contamination data are recorded by the Provincial Environment Protection Inspectorate, and includes the values of NO_2 and NO_x taken every hour. The traffic data are collected by the Traffic Public Transport Management Department of the Roads and City Maintenance Board in Wrocław, and encompasses the hourly count of all types of vehicles that pass through the intersection. The meteorological data belong to the Institute of Meteorology and Water Management, and include: air temperature, solar duration, wind speed, relative humidity, air pressure, and ozone. Four datasets are considered: those for NO_2 prediction, with and without ozone values, and those for NO_x prediction, with and without ozone values. The importance of these factors is largely discussed in the literature (see, e.g., [11,12]). The main contributions of our work are:

- Although other authors have proposed similar studies, most of them have focused on the prediction of *particulate matter* with a diameter of less than 2.5 micrometers ($PM_{2.5}$) and with a diameter of less than 10 micrometers (PM_{10}), and only a few of them on the prediction of NO_2 and NO_x. We focus our research on the prediction of NO_2 and NO_x.
- To the best of our knowledge, also, the role of O_3 was never systematically assessed in this context. In this study, we consider the presence of O_3 in the prediction of pollutants NO_2 and NO_x.
- Furthermore, the studies carried out so far do not propose specific methods for choosing the best prediction model among many possibilities. Our work, is not only a complete methodology to build deep learning and machine learning models for the forecast of contaminants NO_2 and NO_x with and without O_3, but also to compare these prediction models. The proposed methodology considers sliding window transformation with different window sizes chosen appropriately in the context of the problem.
- A decision process based on statistical tests and multi-criteria optimization is proposed for choosing the best model in a prediction horizon of *h*-step-ahead.
- We propose metrics based on *h*-step-ahead predictions to measure both exactness and robustness of forecasting models, and we propose a weighted additive function to aggregate the precision and robustness criteria into a single measure.
- It is the first time that a study of these characteristics has been carried out in the city of Wrocław.

As a result of our study, we are able to select very reliable models for 24-hour ahead prediction. Models such as the one we propose are important as they allow us to identify the factors having the greatest impact on pollution, and to quantify such impact. This information is useful for decision-making, e.g. about building construction and planning in an urban environment (leaving ventilation channels facilitates the evacuation of pollutants and reduces the exposure of residents to their unfavorable effects). But forecasting the concentrations of pollutant for the next few hours is also useful in order to warn residents against potential dangers due to too high levels, which can and should influence their behavior in order to reduce exposition by, for example, refraining from walking or cycling in a particular area at a particular time. In this sense, a 24-hours forecast horizon is probably the most suitable choice, that balances the accuracy of the prediction with the usefulness of an early warning to a small community. Moreover, forecast horizons with lengths of the order of hours are in fact in line with other, similar systems in the literature (see, e.g., [13]).

The rest of the article is organized as follows. Section 2 presents some related works published in the last 5 years; Section 3 describes the datasets used for this research; Section 4 presents the proposed methodology for the building and comparison of the models, as well as for the choice of the best model, and shows the results obtained; Section 5 analyzes and discusses the results; Section 6 draws conclusions and future work; finally, Appendixes A to E show charts, diagrams and tables that reinforce the exposition of the methodology and the results.

2. Related works

This section reviews the relevant work that has been published in the last 5 years (from 2017 to 2021) in the field of time series forecasting for air quality through deep learning or other machine learning techniques.

2017 Articles:

Patra [14] uses *multi-layer perceptron* (MLP), *support vector regression* (SVR) and *autoregressive integrated moving average* (ARIMA) models for one month ahead prediction of *CO* and NO_2 with the public AirQuality database obtained from the *UCI Machine Learning Repository* [15] with 390 instances of daily averaged responses from a collection of 5 metal oxide chemical sensors installed in an Air Quality Chemical Multisensor Device. The author concludes that the best results, presented in terms of *root mean square error* (RMSE), are obtained with MLP with an architecture $(4-8-1)$ for *CO* and with an architecture $(10-2-1)$ for NO_2.

Kok et al. [16] propose a deep learning model based on *Long Short Term Memory* (LSTM) [17] networks in order to make predictions for air pollution with the data from IoT smart city analysis. Network structure consists of an input layer, a hidden layer with 24 LSTM units, and an output layer, with batch size of 50 and 100 epoch. The experiments used a database of 17568 instances at five-minute intervals and the attributes of ozone, PM, carbon monoxide, sulfur dioxide, nitrogen dioxide, longitude, latitude and timestamp. Ozone and nitrogen dioxide are predicted and the models are evaluated using hold-out at 70% training and 30% testing, and the results are compared with SVM using RMSE and

mean absolute error (MAE) metrics. Finally, the data is labeled according to the daily values of the *air quality index* (AQI).

Fan et al. [18] develop a spatio-temporal framework incorporating *deep recurrent neural networks* (DRNN) along with interpolation algorithms to deal with missing values in time series data. Proposed DRNN consists of LSTM layers and fully connected layers. Data from northern China includes air quality of neighboring stations, local air quality properties, local meteorological properties and time and spatial properties. The model is able to predict the future 1 $\sim$ 8 h $PM_{2.5}$ concentration based on the historical records from the past two days. Data is divided into 60% for training, 20% for validation and the remaining 20% for testing. Proposed deep learning method is compared with *gradient boosting decision tree* and *deep feedforward neural networks* (DFNN). Model performance is measured by RMSE, MAE and index of agreement.

2018 Articles:

Lin et al. [19] propose the *geo-context based diffusion convolutional recurrent neural network* (GC-DCRNN) for forecasting the next 24-hour $PM_{2.5}$ concentrations. The authors show the spatial correlation of several monitoring stations through a graph, to diffuse convolution to the following step, based on the similarity of the relevant geographic features. GC-DCRNN apply the *sequence to sequence* architecture [20] to conduct the multi-step-ahead predictions. Two real-world databases are building (Los Angeles and Beijing) with air quality, meteorological and *OpenStreetMap* data. Authors evaluated the model by comparing the results with *linear regression* (LR), *vector autoregression* and *gradient boosting machines* by using the RMSE and MAE metrics.

Freeman et al. [21] use *recurrent neural networks* (RNN) with LSTM to predict local 8-hour averaged ozone accumulations based on hourly air monitoring station measurements. A pre-processing phase was carried out for missing data imputation, outlier detection and feature selection with decision trees. Hourly air quality and meteorological data were collected using OPSIS differential optical absorption spectroscopy analyzers placed near a local university in the State of Kuwait to train and forecast values up to three days. Feedforward neural network and ARIMA were compared with the proposed method using RMSE and MAE performance measures.

Bui et al. [22] propose an encoder–decoder model using RNN with LSTM units for prediction of $PM_{2.5}$ from time-series data of air quality and meteorological information in South Korea. Study reveals that using MAE loss function is more effective than *mean square error* (MSE).

Athira et al. [23] use RNN, LSTM, and *gated recurrent unit* (GRU) [24] for PM_{10} forecasting based on the time series from *AirNet* data. The authors compare RNN, LSTM and GRU with 1–4 layer architectures using MSE, RMSE and *mean absolute percentage error* (MAPE) metrics, and they conclude that the performance of the GRU network is slightly better than the RNN and LSTM networks.

Qi et al. [25] propose *deep air learning*, a model that integrates feature selection plus spatio-temporal semi-supervised neural network. Feature selection is performed in the input layer of a neural network, and middle layers and output layer implement spatio-temporal semi-supervised regression of labeled and unlabeled data. The authors use actual data sources obtained in Beijing in their experiments.

Sharma et al. [26] use a RNN-LSTM model for AQI estimation. Sub-indices for each of the pollutants are aggregated to reach the global AQI. Authors propose two diverse approaches to estimate the AQI: RMSE of all sub-indices and *Min/Max* aggregation.

2019 Articles:

Du et al. [27] propose a *deep air quality forecasting framework* (DAQFF) for $PM_{2.5}$ forecasting. DAQFF include *one-dimensional convolutional neural networks* (1DCNN) [28] and *bi-directional LSTM networks* (Bi-LSTM). The experiments are performed on the Beijing $PM_{2.5}$ and Urban Air Quality datasets from the *UCI Machine Learning Repository*, and the DAQFF results were compared with SVR with different kernel, ARIMA, LSTM, GRU and RNN.

Lin et al. [29] study PM_{10} concentrations during a dust storm. The proposed approach combines the data-driven machine learning (LSTM network) and physics-based model via data assimilation and production applying a physics-based simulation model.

Masih [30] writes a review paper where 38 of the most important studies in the area of environmental science and engineering, which have used machine learning techniques, are examined. The study reveals that when it comes to pollution estimation is generally achieved by adopting approaches based on ensemble learning and linear regression while forecasting tasks commonly rely on neural networks and SVM.

Karimian et al. [31] use *multiple additive regression trees* (MART), DFNN and LSTM for $PM_{2.5}$ estimation with data provided by *Tehran Air Quality Control Company*. RMSE, MAE and *coefficient of determination* (R^2) are employed to evaluate the models' performances. The best results were obtained with LSTM networks.

Tao et al. [32] propose a *convolutional-based bidirectional GRU* method that combines the ability of feature extraction from *convolutional neural networks* (CNN) and the capability of time series forecasting from RNN for $PM_{2.5}$ forecasting. In the comparisons, SVR, *gradient boosting regressor* (GBR), *decision tree regressor* (DTR), simple RNN, LSTM, GRU and *bidirectional* GRU were used, with performance metrics RMSE, MAE and *symmetric* MAPE on the Beijing $PM_{2.5}$ dataset from the *UCI Machine Learning Repository*.

Sun et al. [33] propose a GRU model to predict $PM_{2.5}$ concentrations using a dataset from Shenyang, China, with earth contamination control, industry emissions and surface climatology monitoring attributes along with monthly, daily and hourly dummy variables. *Multiple linear regression* (MLR), RF, SVR, *artificial neural network* (ANN), and LSTM were compared with the proposed GRU model using RMSE, MAE and MAPE performance metrics.

Ameer et al. [34] use Apache Spark to fit the hyper-parameters of four regression techniques. DTR, RF, GBR and ANN MLP were applied for the prediction of $PM_{2.5}$ in several large cities of China, using RMSE and MAE as evaluation criteria. RF performed best among the four regression algorithms.

2020 Articles:

Kaya and Gunduz [35] estimate PM_{10} with *deep flexible sequential* (DFS), a hybrid deep model including LSTM and CNN, with MAE and RMSE metrics for 4, 12 and 24 window size in four separate measurement stations of Istanbul, Turkey. DFS uses a dropout layer for generalization.

Li et al. [36] integrate 1DCNN, LSTM and *attention-based network*, for urban $PM_{2.5}$ concentration prediction. The attention-based layer weighs the prior feature states with the objective to increase prediction accuracy. The authors use data from Taiyuan, China, and the results are compared with the SVR, RF, MLP, simple RNN, LSTM, and CNN–LSTM methods, using the RMSE, MAE and R^2 metrics.

Surakhi et al. [37] highlight some of the most relevant works that propose ensemble models of different RNN versions, and they introduce a framework for air quality time series forecasting based on an ensemble of RNN.

Lin et al. [38] propose a neuro-fuzzy approach for air quality. A four-layer fuzzy neural network is created from fuzzy clusters selected automatically from training data. Then, a *particle swarm*

optimization and steepest descent backpropagation algorithms are used to optimize the parameters.

2021 Articles:

Lin et al. [39] exploit a deep learning network architecture GRU to design various predictive models for air quality forecasting that meet various spatial and temporal situations, and then propose a joint learning forecasting model, called Multiple Linear Regression based GRU, to integrate these predictive deep learning models. Data are collected from the years 2013–2019 using 67 monitoring stations in Taiwan. The results are compared to other ensemble methods using MAE, RMSE absolute error less than 3.

Nath et al. [40] compare statistical (*auto-regressive*, Holt-Winters, *seasonal* ARIMA and *Prophet*) and deep learning (LSTM, LSTM auto-encoder, Bi-LSTM, convolution LSTM) methods to predict $PM_{2.5}$ and PM_{10} concentrations in the upcoming months. Data was taken between 2016 and 2020 from a station at Victoria Memorial Hall in Kolkata, India. The Holt–Winters statistical model performed better for RMSE and MAE than the deep learning models.

Heydari et al. [41] propose a hybrid method based on LSTM and *multi-verse optimization* algorithm to predict and analyze NO_2 and SO_2 production by Combined Cycle Power Plants in Kerman, Iran. Data includes information taken during five months of 2019. The proposed model has been tested with RMSE, MAE and MAPE and obtains more stable results than other hybrid forecasting methods.

Du et al. [42] present a new hybrid deep learning architecture based on 1DCNN and Bi-LSTM that considers the spatio-temporal features of the time series for $PM_{2.5}$ prediction. Two different databases have been used, both with data from Beijing and with one-hour intervals. One of them taken from one monitoring station between the years 2010 and 2014 and the other belonging to 36 stations between 2014 and 2015. The RMSE and MAE metrics show that the proposed model performs better than other machine and deep learning techniques.

Tripathi and Pathak [43] studied different deep learning models and their advantages and disadvantages when applied to air quality prediction, both globally and in India. They also propose a framework for carrying out air pollution predictions and describe various public databases for conducting experiments. Finally, several metrics for the evaluation of the models created are highlighted.

Finally, Wang et al. [44] recently proposed a model for predicting $PM_{2.5}$ concentrations combining *Convolutional neural networks* and *Dense-based Bidirectional GRU* in order to obtain more accurate predictions.

3. Wrocław air quality database

The air quality measuring station considered in this paper is located within a wide street that features two lanes in each direction, at the GPS coordinates 51.086390 North and 17.012076 East (see Fig. 1). One of the most important street crossing in Wrocław, with 14 traffic lanes, is located approximately 30 meters from this station, and it is monitored by traffic cameras. The station is located in the suburbs of the city, at 9.6*kms* from the airport. Contamination data are recorded by the Provincial Environment Protection Inspectorate and includes the hourly NO_2 and NO_x concentration values during three years, specifically, from 2015 to 2017. The traffic data belong to the Traffic Public Transport Management Department of the Roads and City Maintenance Board in Wrocław, and encompass the hourly count of cars passing through the intersection. Public meteorological data belong to the Institute of Meteorology and Water Management, and they include: air temperature, solar duration, wind speed, relative humidity, air pressure, and ozone levels. For uniformity, solar duration values have been normalized in the real interval [0, 1]. Associated to the communication station, there are two O_3 sensors, located, respectively, at 9.5*kms* and at 4.97*kms* from the intersection.

In this paper, we are interested, among others, establish the statistical role of O_3 in NO_2 and NO_x concentration. Concentrations of NO_2, NO_x and O_3 in the air are strongly related to each other, especially in high concentrations. During daylight hours NO_2, NO_x and O_3 concentration are in a steady state, known as *photostationary* state. This state results from the simultaneous reactions: decomposition of NO_2 (under the influence of photons) into *NO* and atomic oxygen, and the oxidation reaction of O_3 + $NO \rightarrow NO_2 + O_2$. The time needed to reach the state varies from several to several dozen minutes depending on the level of pollution. At the same time, the oxidation reactions of NO_2 to NO_3 (with ozone) take place with NO_3 being quickly photolyzed back to NO_2. This means that the concentration of NO_x (being the sum of the concentration of all nitrogen oxides in the air) is less dependent on the ozone concentration, while the concentration of NO_2 is very much influenced by the ozone occurring simultaneously in the air. The full dataset contains 26304 observations. In the pre-processing phase, the instances with at least one missing value (617 samples, 2.3%) have been deleted. Some basic statistic indicators on the remaining 25687 instances are presented in Table 1, along with the symbol used in the tests for each variable.

4. Materials and methods

This section describes the proposed methodology for the construction of time series forecasting models for NO_2 (with and without O_3) and NO_x (with and without O_3) concentrations, the comparison of the models and the choice of the best model considering 24-steps-ahead predictions. The main objective of the proposed methodology is to compare different Deep Learning architectures with each other and with more conventional machine learning methods. Among the different Deep Learning architectures we choose:

- **1DCNN**. This type of convolutional neural network uses a one-dimensional convolution layer composed of 1×1 filters. This kind of filter only requires a single parameter for each input, reducing the complexity of the model. This 1×1 filter does not need any padding and the stride can be used to control the dimension of the output space. Due to the reduction of the number of parameters, 1D convolutional layers are preferable in problems dealing with 1D signals.
- **RNN**. A RNN is a neural network that takes into account previous states to predict future ones. Since its outputs are connected to its inputs, a RNN processes inputs sequences iterating over their elements and maintaining a state storing information on what the network has processed so far. Among the different RNN architectures, we choose the following:
  - **LSTM**. An LSTM is a type of RNN made of LSTM cells. An LSTM cell can control how much information from the current state is stored and how much information from previous states is used to process the current state. This is possible thanks to internal gates which control the information to be forwarded to the next state or to be forgotten. This characteristic makes it possible for an LSTM to learn long-term dependencies.
  - **GRU**. As an LSTM a GRU neural network is an RNN made of GRU cells. A GRU cell is similar to an LSTM cell but, instead of using both cell state and hidden state to transfer information, they only use the hidden state. Due to de reduction of the number of internal gates, GRU networks reduce the complexity of the model to be learned.

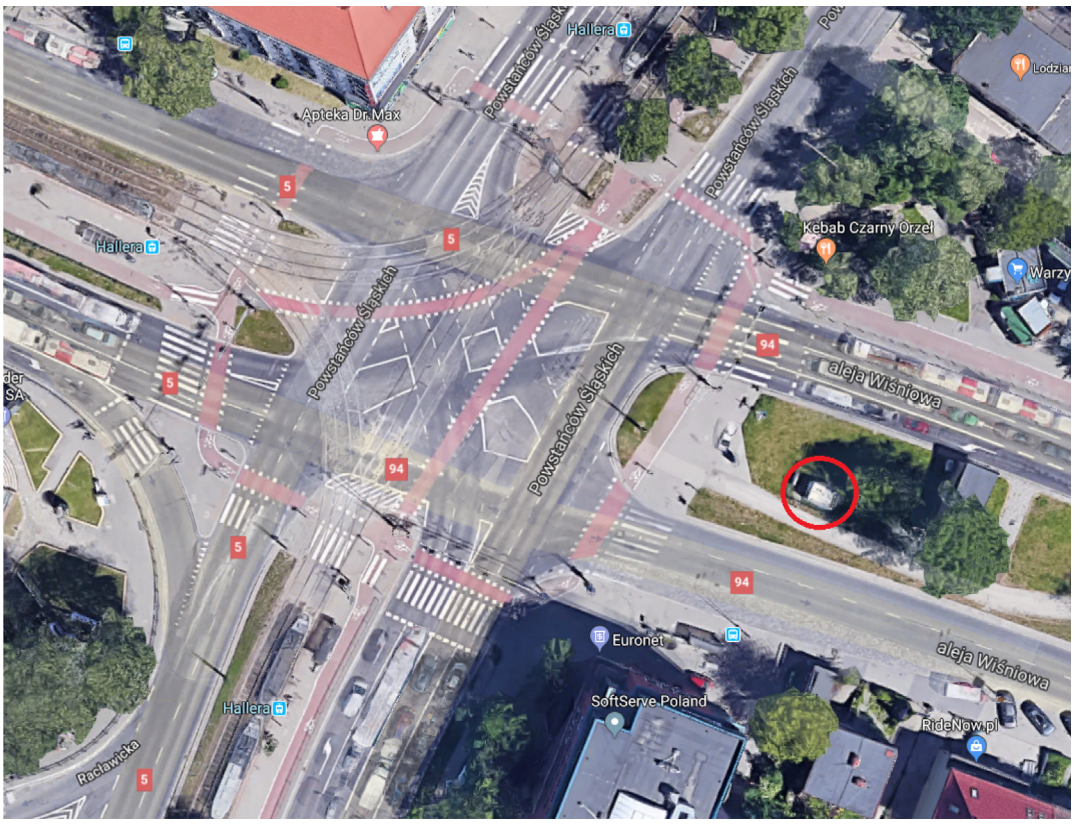


Fig. 1. An aerial view of the area of interest.

Table 1
Descriptive statistics. Data are collected hourly.

Variable	Unit	Mean	St.Dev.	Min	Median	Max
Air temperature	°C	10.9	8.4	−15.7	10.1	37.7
Solar duration	h	0.23	0.38	0	0	1
Wind speed	m s^{-1}	3.13	1.95	0	3.00	19
% Relative humidity	–	74.9	17.3	20	79.0	100
Air pressure	hPa	1003	8.5	906	1003	1028
Traffic	–	2771	1795.0	30	3178	6713
O_3	μm^{-3}	46.11	30.96	0	42.55	188
NO_2	$\mu g\ m^{-3}$	50.4	23.2	1.7	49.4	231.6
NO_x	$\mu g\ m^{-3}$	142.2	103.7	3.9	123.7	1728.0

As said before, in the proposed methodology, not only Deep Learning models are compared. We have included the following Machine Learning models:

- **RF**. RF is a technique that combines ensemble techniques with decision trees. Several trees are generated from different bootstrapped samples of the original data. During each tree building process, a random selection of features is used at each split. All trees contribute equally to the output of the model. It has been demonstrated that RF is more robust than other similar approaches.
- **SVM**. SVM is a computationally efficient technique of learning the separating hyperplane that optimizes the generalization bounds. Initially designed for linear separable problems, they can be easily adapted to non-linear separable problems using kernel method transformations.
- **Lasso**. Lasso Linear regression is a method for estimating a linear model that constrained the sum of the absolute values of the coefficients to be less than a certain constant. The inclusion of this constraint may force some coefficients to be exactly 0, producing more simple and interpretable models.

As can be seen, apart from including Deep Learning and conventional models, we have also included a linear model, Lasso Linear Regression, to compare its performance against other non-linear ones. Our methodology also includes: sliding window transformation, missing values imputation, hyper-parameter tuning, evaluation, statistical test, multi-criteria decision making to chose the best model, and finally, 1 to 24-steps-ahead predictions. Fig. 2 graphically shows the proposed methodology. We have applied the methodology independently for each of the NO_2, NO_x, NO_2 with O_3 and NO_x with O_3 prediction problems. Each of the steps of the proposed methodology is described separately below. We have used the *Python* packages *Scikit-Learn*, *Keras* and *TensorFlow* [45] to implement the proposed methodology.

4.1. Sliding window transformation and missing values imputation

Unlike the *autoregressive methods* [46], our methodology transforms the dataset to eliminate the temporal order of the individual instances by coding the time dependency through additional input variables, called *lagged variables*. Lagged variables allow the discovery of the possible relation existent between the past and

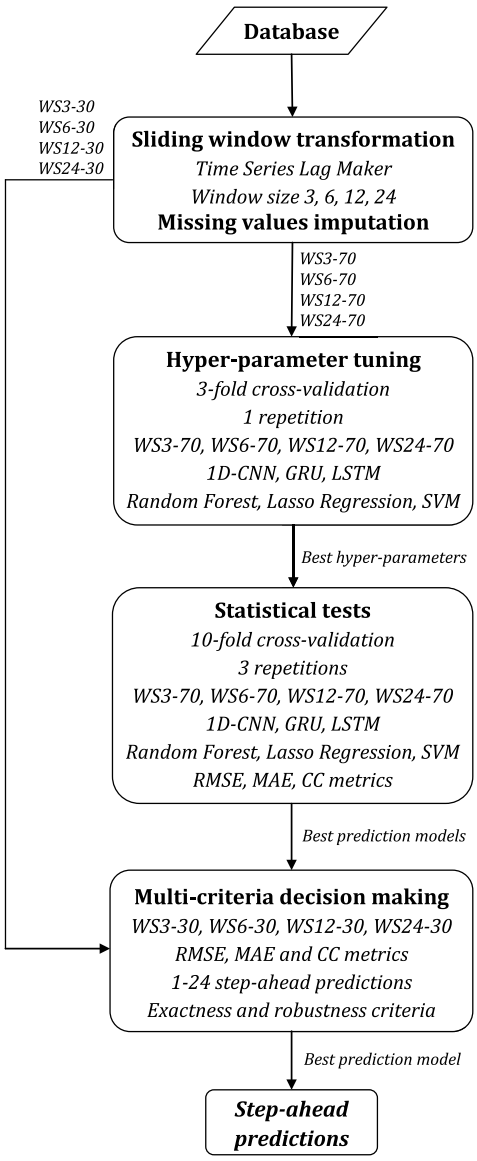


Fig. 2. Flow diagram of the proposed methodology for air quality time series forecasting.

present values of attributes in time series. To do this is necessary a transformation to create a window over a specific period. This transformation is called a *sliding window transformation* [47]. The number of lagged attributes is the *window size*. In this way, we can use any machine learning algorithm to model time series. A simple option is to work with MLR, but all techniques able to make predictions in regression problems can be used, from non-linear methods such as SVM or RF, to deep learning architectures based on neural networks.

Let $D = \{x_t^1, x_t^2, \ldots, x_t^m, y_t\}$, $t \in T = \{1, \ldots, n\}$ be a dataset representing m input time series $\{x_t^i : t \in T\}$, $i = 1, \ldots, m$, and one output time series $\{y_t : t \in T\}$ of n observations (a total of $(m+1) \cdot n$ values). Let l be the window size (number of lagged variables). The sliding window transformation process builds the following dataset D_l composed of $(l \cdot (m+l)+1) \cdot n$ values:

$$D_l = \Big\{\{x_{t-1}^1, x_{t-2}^1, \ldots, x_{t-l}^1\}, \{x_{t-1}^2, x_{t-2}^2, \ldots, x_{t-l}^2\}, \ldots, \{x_{t-1}^m, x_{t-2}^m, \ldots, x_{t-l}^m\}, \{y_{t-1}, y_{t-2}, \ldots, y_{t-l}\}, y_t\Big\}, \; t = 1, \ldots, n \tag{1}$$

Note that $x_t^i, i = 1, \ldots, m$, and y_t values with $t \leq 0$ do not exist and are therefore considered missing values. Since not all machine learning algorithms used in this paper can deal with missing values, in our methodology these values are imputed after the sliding window transformation process. We apply the *mean substitution* method which replaces all missing values for numeric attributes in a dataset with the means from the training data.

As can be seen from the flowchart in Fig. 2, the original database is transformed using 4 different window sizes: 3, 6, 12 and 24. These window sizes represent observation periods of 3 h, 6 h, half a day and one day, respectively. Each of these four databases is in turn divided into two databases, one with 70% of the data (6132 first instances) and the other with 30% of the data (2628 last instances). These databases have been called in this paper with the names WS3-70, WS6-70, WS12-70, WS24-70, WS3-30, WS6-30, WS12-30 and WS24-30. In this way, 70% of the data is used throughout the methodology for the construction and evaluation of the models, while 30% of the data remains unseen throughout this process. Finally, this 30% of the data is used to choose the best model and to make the 1 to 24-steps-ahead predictions. We use multi-criteria decision process described in Section 4.4 for this proposal.

4.2. Hyper-parameter tuning and deep learning architectures

As introduced in Section 4, the first step in the model training process consists in finding the best hyper-parameter combination for each model [48] and transformed training database. To this end, a hyper-parameter grid search has been performed using 3-fold cross-validation as a resampling strategy. Table 2 summarizes the search space for each parameter in RF, Lasso and SVM with *radial basis function* (SVMRadial).

For deep learning methods, hyper-parameters grid search process has been divided into two steps. Firstly, different values for *epochs* and *batch_size* has been tested. We have tried {500, 1000} for *epoch* and {32, 1533, 3066} for *batch_size*. Some works recommend using 32 as minimum for *batch_size* as it provides good results [49]. Afterwards, *batch_size* is increased first up to a quarter and, finally, up to half of the data in the training database. Once the best configuration of the previous hyper-parameters has been found, different configurations regarding deep learning architectures have been tested. Different hidden layers have been added, but none of these extended configurations have performed better than those achieved by single-layer configurations. Different optimization algorithms [50] have been tested: *Adam*, *RMSProp* and *Stochastic Gradient Descent with Momentum*, where *Adam* achieves the best performance. Additionally, the following activation functions [51] for the hidden layer have been tested: *rectified linear unit* (ReLU) activation [52], logistic function (Sigmoid) [53] and *hyperbolic tangent* [54]. Finally, ReLU has been selected, as it provided the best results and is able to cope with the vanishing gradient problem [55]. It has been decided to use the identity or linear activation function in the output layer to improve interpretability and also avoid the vanishing gradient problem [56]. We have added a dropout layer on all tested architectures. Dropout is a technique used to prevent overfitting [57]. Dropout works by randomly setting the neurons to be disabled in hidden layers. Ignoring these neurons, it is achieved that the neural network changes in each new training, thus preventing the overfitting of the model. Finally, MAE has been used as the loss function and MAE, RMSE and *correlation coefficient* (CC) as performance measures.

Table 2
Hyper-parameters grid search values for machine learning methods.

Method	Hyper-parameters
RF	n_estimators = {100,500,1000}, min_samples_split = {2,5,10,15}, min_samples_leaf = {1,5,10}
Lasso	alpha = {0.0001,0.001,0.01,0.1,1}
SVMRadial	C = 2^i, $i \in \mathbb{N}$, $i = [-5, 5]$, gamma = 2^i, $i \in \mathbb{N}$, $i = [-15, 0]$

Table 3
Performance and hyper-parameters of NO_2 models ordered by CC from highest to lowest.

Model	MAE	RMSE	CC	Hyper-parameters
RF-WS24–70	6.634445	9.075517	0.922591	min_samples_leaf: 1, min_samples_split: 2, n_estimators: 1000
Lasso-WS24–70	6.725042	9.087141	0.922383	alpha: 0.01
GRU-WS24–70	6.653827	9.130571	0.921587	batch_size: 1533, epochs: 1000
LSTM-WS3–70	7.059550	9.706929	0.918909	batch_size: 3066, epochs: 1000
LSTM-WS24–70	6.766519	9.311406	0.918149	batch_size: 1533, epochs: 1000
RF-WS3–70	7.160109	9.775655	0.917787	min_samples_leaf: 5, min_samples_split: 10, n_estimators: 1000
GRU-WS6–70	7.128003	9.812574	0.917109	batch_size: 1533, epochs: 1000
1DCNN-WS3–70	7.192001	9.809654	0.917058	batch_size: 1533, epochs: 1000
RF-WS6–70	7.186975	9.818650	0.917044	min_samples_leaf: 5, min_samples_split: 5, n_estimators: 100
RF-WS12–70	7.177771	9.829456	0.916882	min_samples_leaf: 5, min_samples_split: 2, n_estimators: 1000
1DCNN-WS6–70	7.344162	9.863462	0.915944	batch_size: 32, epochs: 1000
LSTM-WS6–70	7.205291	9.932934	0.914896	batch_size: 1533, epochs: 1000
GRU-WS3–70	7.304130	9.985874	0.913965	batch_size: 32, epochs: 500
GRU-WS12–70	7.284027	10.026551	0.913258	batch_size: 1533, epochs: 1000
Lasso-WS12–70	7.477842	10.035184	0.913153	alpha: 0.01
LSTM-WS12–70	7.376607	10.126055	0.911204	batch_size: 1533, epochs: 1000
Lasso-WS6–70	7.564295	10.172861	0.910643	alpha: 0.01
1DCNN-WS24–70	7.430167	9.758709	0.909769	batch_size: 32, epochs: 1000
1DCNN-WS12–70	7.572490	10.209787	0.909511	batch_size: 32, epochs: 500
Lasso-WS3–70	7.633988	10.267745	0.908874	alpha: 0.01
SVMRadial-WS3–70	10.484122	14.956933	0.794976	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS6–70	13.965616	19.356197	0.619709	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS12–70	18.227391	23.791666	0.263528	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS24–70	19.166062	24.706402	−0.003140	C: 0.01, gamma: 3.0517578125e−05

Table 4
Performance and hyper-parameters of NO_x models ordered by CC from highest to lowest.

Model	MAE	RMSE	CC	Hyper-parameters
GRU-WS3–70	21.859762	35.664969	0.913047	batch_size: 1533, epochs: 1000
GRU-WS6–70	22.048860	35.858617	0.912141	batch_size: 1533, epochs: 1000
LSTM-WS6–70	22.047705	35.972495	0.911462	batch_size: 1533, epochs: 1000
LSTM-WS3–70	22.105145	36.104703	0.910850	batch_size: 1533, epochs: 1000
1DCNN-WS3–70	22.761707	36.171371	0.910245	batch_size: 1533, epochs: 500
1DCNN-WS12–70	22.895998	36.275727	0.909818	batch_size: 1533, epochs: 1000
LSTM-WS12–70	22.434648	36.502489	0.908667	batch_size: 3066, epochs: 1000
GRU-WS24–70	22.716782	36.571653	0.908328	batch_size: 3066, epochs: 1000
GRU-WS12–70	23.201124	36.937790	0.906323	batch_size: 32, epochs: 1000
RF-WS12–70	22.995607	37.144906	0.905417	min_samples_leaf: 1, min_samples_split: 2, n_estimators: 500
RF-WS6–70	22.962943	37.300276	0.904414	min_samples_leaf: 1, min_samples_split: 2, n_estimators: 500
RF-WS3–70	23.135149	37.579219	0.902813	min_samples_leaf: 1, min_samples_split: 5, n_estimators: 1000
1DCNN-WS6–70	23.865844	37.624541	0.902441	batch_size: 32, epochs: 1000
RF-WS24–70	23.040165	37.758181	0.902157	min_samples_leaf: 5, min_samples_split: 10, n_estimators: 500
Lasso-WS24–70	24.869341	37.781958	0.901626	alpha: 1
LSTM-WS24–70	24.080405	37.802336	0.901575	batch_size: 32, epochs: 1000
1DCNN-WS24–70	24.880677	38.126577	0.899020	batch_size: 32, epochs: 500
Lasso-WS12–70	25.552197	38.517813	0.897582	alpha: 0.1
Lasso-WS6–70	25.956889	39.077441	0.894368	alpha: 0.1
Lasso-WS3–70	26.204132	39.458923	0.892119	alpha: 0.01
SVMRadial-WS3–70	32.895336	61.276628	0.715972	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS6–70	45.829883	74.740853	0.524520	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS12–70	61.248657	87.597625	0.087428	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS24–70	62.742814	88.776942	0.022558	C: 32, gamma: 3.0517578125e−05

4.3. Evaluation and statistical tests

Once best hyper-parameters have been found for every combination of databases and methods, new models have been obtained using stratified 10-fold cross-validation, which is repeated 3 times, as a sampling technique. Tables 3–6 shows the average MAE, RMSE, CC and hyper-parameters for each model, ordered by CC.

To check if there are statistically significant differences, a statistical pairwise paired t-test has been conducted for each problem and for each performance measure RMSE, MAE and CC, with a confidence level of 95%. Parametric test has been conducted since the size of samples (30 = 10 folds × 3 reps) let us assume normality. In order to summarize t-test results, method-database pairs have been ranked according to the difference between *wins* and *loses*. Every time one method-database pair tests statistically significantly better than another, it counts as a *win* and otherwise as a *loss*. Tables from Tables 7 to 10 depicted the ranking obtained in each problem.

Table 5
Performance and hyper-parameters of NO_2 with O_3 models ordered by CC from highest to lowest.

Model	MAE	RMSE	CC	Hyper-parameters
GRU-WS3–70	7.036541	9.716180	0.918623	batch_size: 1533, epochs: 1000
1DCNN-WS3–70	7.171476	9.834825	0.916594	batch_size: 1533, epochs: 1000
LSTM-WS3–70	7.154402	9.852496	0.916264	batch_size: 1533, epochs: 500
RF-WS3–70	7.200979	9.880605	0.915686	min_samples_leaf: 5, min_samples_split: 2, n_estimators: 500
RF-WS6–70	7.226511	9.926008	0.914867	min_samples_leaf: 5, min_samples_split: 5, n_estimators: 500
RF-WS12–70	7.247562	9.971308	0.914063	min_samples_leaf: 5, min_samples_split: 5, n_estimators: 500
Lasso-WS24–70	7.386578	9.988309	0.913734	alpha: 0.1
GRU-WS6–70	7.232512	9.996398	0.913685	batch_size: 3066, epochs: 1000
RF-WS24–70	7.329057	10.094719	0.911830	min_samples_leaf: 5, min_samples_split: 15, n_estimators: 800
1DCNN-WS6–70	7.481373	10.129247	0.910728	batch_size: 32, epochs: 1000
Lasso-WS12–70	7.546281	10.154159	0.910693	alpha: 0.01
Lasso-WS6–70	7.604603	10.260399	0.908741	alpha: 0.01
LSTM-WS6–70	7.534118	10.338057	0.907076	batch_size: 32, epochs: 500
Lasso-WS3–70	7.678376	10.356508	0.906922	alpha: 0.01
GRU-WS24–70	7.533502	10.407644	0.905766	batch_size: 1533, epochs: 1000
GRU-WS12–70	7.640449	10.483229	0.904696	batch_size: 32, epochs: 500
LSTM-WS12–70	7.770967	10.551377	0.903394	batch_size: 32, epochs: 500
1DCNN-WS12–70	7.856826	10.537669	0.902676	batch_size: 32, epochs: 500
1DCNN-WS24–70	8.132444	10.749313	0.897938	batch_size: 32, epochs: 1000
LSTM-WS24–70	7.896917	10.929484	0.895798	batch_size: 3066, epochs: 1000
SVMRadial-WS3–70	10.598137	15.139070	0.788993	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS6–70	14.328548	19.696575	0.601244	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS12–70	18.483168	24.038023	0.221525	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS24–70	19.181352	24.699811	0.049601	C: 0.0625, gamma: 3.0517578125e−05

Table 6
Performance and hyper-parameters of NO_x with O_3 models ordered by CC from highest to lowest.

Model	MAE	RMSE	CC	Hyper-parameters
GRU-WS3–70	22.051294	36.251769	0.912018	batch_size: 1533, epochs: 1000
GRU-WS12–70	22.339752	36.296900	0.911635	batch_size: 1533, epochs: 1000
LSTM-WS3–70	22.211966	36.547515	0.910597	batch_size: 1533, epochs: 1000
1DCNN-WS3–70	22.911719	36.680775	0.909800	batch_size: 1533, epochs: 500
LSTM-WS6–70	22.474928	36.824969	0.909151	batch_size: 3066, epochs: 1000
1DCNN-WS12–70	23.303860	36.825414	0.908964	batch_size: 1533, epochs: 1000
LSTM-WS24–70	23.094044	37.225815	0.906859	batch_size: 1533, epochs: 1000
LSTM-WS12–70	22.885755	37.446433	0.905742	batch_size: 1533, epochs: 500
1DCNN-WS6–70	23.706456	37.457742	0.905183	batch_size: 32, epochs: 500
GRU-WS6–70	23.315139	37.634744	0.904687	batch_size: 32, epochs: 500
RF-WS12–70	23.316390	38.091950	0.902386	min_samples_leaf: 5, min_samples_split: 10, n_estimators: 500
RF-WS6–70	23.306556	38.137500	0.902225	min_samples_leaf: 5, min_samples_split: 10, n_estimators: 500
1DCNN-WS24–70	24.441732	38.139667	0.902019	batch_size: 1533, epochs: 1000
RF-WS3–70	23.400119	38.337070	0.901114	min_samples_leaf: 5, min_samples_split: 10, n_estimators: 500
RF-WS24–70	23.514021	38.430899	0.900549	min_samples_leaf: 5, min_samples_split: 2, n_estimators: 500
Lasso-WS24–70	25.276650	38.443912	0.900263	alpha: 1
GRU-WS24–70	24.227884	38.656657	0.899029	batch_size: 32, epochs: 500
Lasso-WS12–70	25.916057	39.025479	0.897070	alpha: 0.1
Lasso-WS6–70	26.244573	39.485531	0.894500	alpha: 0.1
Lasso-WS3–70	26.454786	39.925581	0.892012	alpha: 0.1
SVMRadial-WS3–70	33.992117	62.940586	0.702594	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS6–70	47.810985	76.793643	0.496416	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS24–70	63.253389	89.499384	0.148149	C: 32, gamma: 3.0517578125e−05
SVMRadial-WS12–70	62.333781	88.728847	0.083796	C: 32, gamma: 3.0517578125e−05

4.4. Multi-criteria decision making

We have considered the 6 best models (a quarter of the total) separately identified in the wins − losses ranking tests for MAE, RMSE, and *CC* of the deep learning and machine learning models. The union of these sets of prediction models is a set $\Phi = \{\phi_1, \ldots, \phi_p\}$ of p models. The next step in our methodology is to compare the prediction models to choose the best. For this purpose, we propose the *multi-criteria decision-making* process described in Algorithm 1 and its workflow as shown in Fig. 3.

We consider two criteria to measure the *goodness* of the models: *exactness* and *robustness*. The exactness of a model is calculated by the sum of the normalized RMSE, MAE and 1− CC performance measures in the h-step-ahead. We have used in the experiments $h = 24$. In the proposed methodology, the *recursive strategy*, also known as *iterated* or *multi-stage strategy* [58], is used for obtaining the 1 to 24-steps-ahead predictions. In the recursive strategy, only a single model is built for one step-ahead. Once the model is trained, different step-ahead predictions are produced by using the model built using the predicted values as inputs for subsequent steps. For the robustness criterion, we consider that a prediction model is robust when it does not present large fluctuations in the forecast of successive steps-ahead. To measure this, the slopes of the lines (for the normalized RMSE, MAE, and 1− CC performance measures) between each pair of successive prediction points are added, and then the sum of these three values is calculated. Finally, the goodness of a prediction model is calculated as the weighted sum of the exactness values for each performance measure, with the weights being the robustness values. The exactness and robustness criteria are evaluated in hold-out using 30% of the data.

Table 11 shows the prediction models that compete in the multi-criteria decision-making process for each of the prediction problems of NO_2, NO_x, NO_2 with O_3 and NO_x with O_3. The winning prediction model in each problem have been marked in bold. Figures from A.4 to A.6 graphically show the performance

Table 7
Ranking of the NO_2 models for MAE, RMSE and CC (from top to bottom) with 10-fold cross-validation, 3 repetitions.

Model	Wins	Losses	Wins−Losses
RF-WS24–70	21	0	21
GRU-WS24–70	20	0	20
LSTM-WS24–70	19	0	19
Lasso-WS24–70	20	1	19
LSTM-WS3–70	12	3	9
GRU-WS6–70	9	4	5
RF-WS3–70	8	4	4
RF-WS12–70	7	4	3
1DCNN-WS3–70	7	4	3
LSTM-WS6–70	7	4	3
RF-WS6–70	7	5	2
GRU-WS12–70	6	4	2
GRU-WS3–70	6	5	1
1DCNN-WS6–70	4	4	0
LSTM-WS12–70	4	5	−1
1DCNN-WS24–70	4	6	−2
1DCNN-WS12–70	4	7	−3
Lasso-WS12–70	6	11	−5
Lasso-WS6–70	5	14	−9
Lasso-WS3–70	4	15	−11
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS12–70	1	22	−21
SVMRadial-WS24–70	0	23	−23
Lasso-WS24–70	20	0	20
RF-WS24–70	20	0	20
GRU-WS24–70	20	0	20
LSTM-WS24–70	14	0	14
LSTM-WS3–70	9	3	6
RF-WS3–70	7	3	4
RF-WS6–70	7	3	4
GRU-WS6–70	7	3	4
RF-WS12–70	7	4	3
1DCNN-WS3–70	6	3	3
LSTM-WS6–70	6	4	2
1DCNN-WS6–70	5	3	2
GRU-WS3–70	5	4	1
1DCNN-WS24–70	5	4	1
1DCNN-WS12–70	4	4	0
GRU-WS12–70	4	5	−1
LSTM-WS12–70	4	6	−2
Lasso-WS12–70	6	8	−2
Lasso-WS6–70	5	12	−7
Lasso-WS3–70	4	16	−12
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS12–70	1	22	−21
SVMRadial-WS24–70	0	23	−23
RF-WS24–70	10	0	10
LSTM-WS3–70	9	0	9
Lasso-WS24–70	8	0	8
RF-WS3–70	7	0	7
RF-WS6–70	7	0	7
RF-WS12–70	7	0	7
GRU-WS24–70	7	0	7
GRU-WS6–70	7	0	7
1DCNN-WS3–70	6	0	6
LSTM-WS6–70	6	0	6
GRU-WS3–70	5	0	5
LSTM-WS24–70	4	0	4
1DCNN-WS12–70	4	0	4
1DCNN-WS6–70	4	0	4
GRU-WS12–70	4	2	2
1DCNN-WS24–70	4	3	1
LSTM-WS12–70	4	3	1
Lasso-WS12–70	6	6	0
Lasso-WS6–70	5	11	−6
Lasso-WS3–70	4	13	−9
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS12–70	1	22	−21
SVMRadial-WS24–70	0	23	−23

Table 8
Ranking of the NO_x models for MAE, RMSE and CC (from top to bottom) with 10-fold cross-validation, 3 repetitions.

Model	Wins	Losses	Wins−Losses
LSTM-WS3–70	20	0	20
GRU-WS6–70	20	0	20
GRU-WS3–70	20	0	20
LSTM-WS6–70	20	0	20
LSTM-WS12–70	16	4	12
1DCNN-WS3–70	10	4	6
GRU-WS24–70	9	4	5
1DCNN-WS12–70	8	4	4
RF-WS12–70	9	5	4
RF-WS24–70	9	5	4
RF-WS3–70	8	5	3
RF-WS6–70	8	5	3
GRU-WS12–70	8	5	3
1DCNN-WS6–70	7	5	2
LSTM-WS24–70	7	9	−2
1DCNN-WS24–70	4	6	−2
Lasso-WS24–70	7	13	−6
Lasso-WS12–70	6	16	−10
Lasso-WS6–70	5	17	−12
Lasso-WS3–70	4	18	−14
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS12–70	1	22	−21
SVMRadial-WS24–70	0	23	−23
GRU-WS6–70	14	0	14
GRU-WS3–70	14	0	14
LSTM-WS6–70	12	0	12
1DCNN-WS3–70	12	0	12
LSTM-WS3–70	10	0	10
1DCNN-WS12–70	10	0	10
GRU-WS24–70	7	0	7
LSTM-WS12–70	8	1	7
1DCNN-WS6–70	4	0	4
GRU-WS12–70	7	3	4
1DCNN-WS24–70	4	0	4
RF-WS6–70	7	3	4
LSTM-WS24–70	4	1	3
RF-WS12–70	7	4	3
RF-WS24–70	6	6	0
RF-WS3–70	6	6	0
Lasso-WS24–70	7	7	0
Lasso-WS12–70	6	12	−6
Lasso-WS6–70	5	15	−10
Lasso-WS3–70	4	16	−12
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS12–70	1	22	−21
SVMRadial-WS24–70	0	23	−23
GRU-WS6–70	14	0	14
GRU-WS3–70	14	0	14
LSTM-WS6–70	12	0	12
1DCNN-WS3–70	12	0	12
LSTM-WS3–70	10	0	10
1DCNN-WS12–70	10	0	10
GRU-WS24–70	7	0	7
LSTM-WS12–70	8	1	7
1DCNN-WS6–70	4	0	4
GRU-WS12–70	7	3	4
1DCNN-WS24–70	4	0	4
RF-WS12–70	8	4	4
RF-WS6–70	7	3	4
LSTM-WS24–70	4	1	3
RF-WS3–70	6	6	0
Lasso-WS24–70	7	7	0
RF-WS24–70	6	7	−1
Lasso-WS12–70	6	12	−6
Lasso-WS6–70	5	15	−10
Lasso-WS3–70	4	16	−12
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS12–70	1	22	−21
SVMRadial-WS24–70	0	23	−23

measures in test set for each model considered in the multi-criteria decision-making process in each problem and each one of the 24-steps-ahead. Tables from Tables 12 to 15 show measures performances for each one of the winning models for all the

Table 9
Ranking of the NO_2 with O_3 models for MAE, RMSE and CC (from top to bottom) with 10-fold cross-validation, 3 repetitions.

Model	Wins	Losses	Wins−Losses
GRU-WS3–70	20	0	20
LSTM-WS3–70	16	0	16
1DCNN-WS3–70	16	0	16
RF-WS3–70	16	1	15
RF-WS6–70	16	1	15
RF-WS12–70	16	1	15
GRU-WS6–70	14	1	13
1DCNN-WS6–70	4	0	4
Lasso-WS24–70	9	6	3
RF-WS24–70	9	6	3
Lasso-WS12–70	8	9	−1
LSTM-WS6–70	5	7	−2
1DCNN-WS12–70	4	7	−3
GRU-WS24–70	4	7	−3
GRU-WS12–70	4	7	−3
1DCNN-WS24–70	4	7	−3
Lasso-WS6–70	5	10	−5
LSTM-WS12–70	4	10	−6
LSTM-WS24–70	4	11	−7
Lasso-WS3–70	4	11	−7
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS12–70	1	22	−21
SVMRadial-WS24–70	0	23	−23
GRU-WS3–70	18	0	18
RF-WS3–70	15	0	15
1DCNN-WS3–70	15	0	15
RF-WS6–70	14	0	14
LSTM-WS3–70	13	0	13
Lasso-WS24–70	11	1	10
RF-WS12–70	13	3	10
GRU-WS6–70	11	1	10
1DCNN-WS6–70	5	0	5
1DCNN-WS24–70	4	2	2
RF-WS24–70	7	5	2
Lasso-WS12–70	8	7	1
1DCNN-WS12–70	4	4	0
LSTM-WS6–70	5	7	−2
Lasso-WS6–70	6	9	−3
GRU-WS12–70	4	8	−4
GRU-WS24–70	4	8	−4
LSTM-WS12–70	4	10	−6
Lasso-WS3–70	5	11	−6
LSTM-WS24–70	4	14	−10
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS12–70	1	22	−21
SVMRadial-WS24–70	0	23	−23
GRU-WS3–70	16	0	16
RF-WS3–70	14	0	14
RF-WS6–70	14	0	14
1DCNN-WS3–70	14	0	14
LSTM-WS3–70	13	0	13
Lasso-WS24–70	11	1	10
RF-WS12–70	13	3	10
GRU-WS6–70	10	0	10
1DCNN-WS6–70	5	0	5
1DCNN-WS24–70	4	0	4
RF-WS24–70	7	5	2
Lasso-WS12–70	8	7	1
1DCNN-WS12–70	4	3	1
LSTM-WS6–70	5	6	−1
Lasso-WS6–70	6	9	−3
GRU-WS12–70	4	8	−4
GRU-WS24–70	4	8	−4
LSTM-WS12–70	4	10	−6
Lasso-WS3–70	5	11	−6
LSTM-WS24–70	4	14	−10
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS12–70	1	22	−21
SVMRadial-WS24–70	0	23	−23

Table 10
Ranking of the NO_x with O_3 models for MAE, RMSE and CC (from top to bottom) with 10-fold cross-validation, 3 repetitions.

Model	Wins	Losses	Wins−Losses
GRU-WS3–70	21	0	21
LSTM-WS3–70	20	0	20
GRU-WS12–70	20	0	20
LSTM-WS6–70	16	1	15
1DCNN-WS3–70	12	3	9
LSTM-WS12–70	10	3	7
RF-WS12–70	10	4	6
GRU-WS6–70	9	3	6
LSTM-WS24–70	9	4	5
1DCNN-WS12–70	9	4	5
1DCNN-WS6–70	8	3	5
RF-WS3–70	9	5	4
RF-WS6–70	9	5	4
GRU-WS24–70	7	4	3
RF-WS24–70	9	7	2
1DCNN-WS24–70	8	13	−5
Lasso-WS24–70	7	15	−8
Lasso-WS12–70	6	17	−11
Lasso-WS6–70	4	18	−14
Lasso-WS3–70	4	18	−14
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS12–70	1	22	−21
SVMRadial-WS24–70	0	23	−23
GRU-WS12–70	17	0	17
GRU-WS3–70	16	0	16
LSTM-WS3–70	14	0	14
1DCNN-WS3–70	14	0	14
LSTM-WS6–70	12	0	12
1DCNN-WS12–70	11	0	11
LSTM-WS24–70	10	1	9
1DCNN-WS6–70	6	0	6
LSTM-WS12–70	7	2	5
GRU-WS6–70	7	2	5
RF-WS3–70	5	4	1
RF-WS6–70	5	4	1
RF-WS12–70	6	5	1
Lasso-WS24–70	7	7	0
1DCNN-WS24–70	6	7	−1
RF-WS24–70	5	7	−2
GRU-WS24–70	4	7	−3
Lasso-WS12–70	6	10	−4
Lasso-WS6–70	5	13	−8
Lasso-WS3–70	4	18	−14
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS12–70	1	22	−21
SVMRadial-WS24–70	0	23	−23
GRU-WS12–70	17	0	17
GRU-WS3–70	16	0	16
LSTM-WS3–70	14	0	14
1DCNN-WS3–70	14	0	14
LSTM-WS6–70	12	0	12
1DCNN-WS12–70	11	0	11
LSTM-WS24–70	10	1	9
LSTM-WS12–70	7	2	5
1DCNN-WS6–70	5	0	5
GRU-WS6–70	7	2	5
RF-WS3–70	5	4	1
RF-WS6–70	5	4	1
RF-WS12–70	6	5	1
Lasso-WS24–70	7	7	0
1DCNN-WS24–70	6	7	−1
RF-WS24–70	5	7	−2
GRU-WS24–70	4	7	−3
Lasso-WS12–70	6	10	−4
Lasso-WS6–70	5	12	−7
Lasso-WS3–70	4	18	−14
SVMRadial-WS3–70	3	20	−17
SVMRadial-WS6–70	2	21	−19
SVMRadial-WS24–70	1	22	−21
SVMRadial-WS12–70	0	23	−23

Table 11
Set Φ of the competing models in the multi-criteria decision-making process and goodness obtained with the Algorithm 1.

NO_2	
Model	Goodness
LSTM-WS24-70	**1.277197**
Lasso-WS24–70	1.368710
RF-WS24–70	1.554364
LSTM-WS3–70	1.729293
GRU-WS6–70	1.804245
GRU-WS24–70	1.845464
RF-WS3–70	1.961961
RF-WS12–70	2.037450
RF-WS6–70	2.183755
NO_x	
Model	Goodness
GRU-WS3-70	**2.161577**
GRU-WS6–70	2.371751
1DCNN-WS3–70	2.442585
LSTM-WS3–70	2.530623
LSTM-WS6–70	2.619026
LSTM-WS12–70	2.742880
1DCNN-WS12–70	2.856967
NO_2 with O_3	
Model	Goodness
GRU-WS3-70	**1.478954**
LSTM-WS3–70	1.489350
Lasso-WS24–70	1.579511
1DCNN-WS3–70	1.634118
RF-WS3–70	2.138257
RF-WS12–70	2.222654
RF-WS6–70	2.288485
NO_x with O_3	
Model	Goodness
LSTM-WS3-70	**2.068695**
GRU-WS3–70	2.123610
LSTM-WS12–70	2.166866
1DCNN-WS3–70	2.212561
GRU-WS12–70	2.349709
1DCNN-WS12–70	2.550501
LSTM-WS6–70	2.863136

Table 12
Performance on training and test data, and differences test−training error of the LSTM-WS24 model for all 24-steps-ahead predictions in the NO_2 problem.

	Steps-ahead												
		1	2	3	4	5	6	7	8	9	10	11	12
Training	Instances	6129	6128	6127	6126	6125	6124	6123	6122	6121	6120	6119	6118
	MAE	6.730	8.344	9.360	9.972	10.332	10.568	10.723	10.840	10.927	11.008	11.092	11.155
	RMSE	9.341	11.694	13.064	13.843	14.314	14.608	14.806	14.955	15.078	15.196	15.307	15.394
	CC	0.922	0.877	0.848	0.832	0.823	0.818	0.815	0.813	0.811	0.810	0.808	0.807
Test	Instances	2628	2627	2626	2625	2624	2623	2622	2621	2620	2619	2618	2617
	MAE	6.267	7.666	8.445	8.902	9.167	9.320	9.409	9.470	9.510	9.536	9.569	9.598
	RMSE	8.523	10.464	11.528	12.114	12.443	12.637	12.743	12.810	12.855	12.896	12.944	12.992
	CC	0.923	0.883	0.857	0.843	0.835	0.830	0.828	0.826	0.825	0.824	0.823	0.822
Loss	MAE	−0.463	−0.678	−0.915	−1.07	−1.165	−1.248	−1.314	−1.37	−1.417	−1.472	−1.523	−1.557
	RMSE	−0.818	−1.23	−1.536	−1.729	−1.871	−1.971	−2.063	−2.145	−2.223	−2.3	−2.363	−2.402
	CC	−0.001	−0.006	−0.009	−0.011	−0.012	−0.012	−0.013	−0.013	−0.014	−0.014	−0.015	−0.015
	Steps-ahead												
		13	14	15	16	17	18	19	20	21	22	23	24
Training	Instances	6117	6116	6115	6114	6113	6112	6111	6110	6109	6108	6107	6106
	MAE	11.211	11.264	11.323	11.388	11.453	11.504	11.544	11.575	11.606	11.642	11.699	11.801
	RMSE	15.469	15.535	15.608	15.690	15.770	15.838	15.895	15.939	15.979	16.026	16.107	16.259
	CC	0.806	0.805	0.805	0.804	0.803	0.802	0.801	0.801	0.800	0.800	0.799	0.796
Test	Instances	2616	2615	2614	2613	2612	2611	2610	2609	2608	2607	2606	2605
	MAE	9.628	9.653	9.674	9.694	9.714	9.735	9.756	9.772	9.789	9.800	9.824	9.860
	RMSE	13.036	13.075	13.110	13.146	13.181	13.209	13.232	13.251	13.269	13.288	13.321	13.383
	CC	0.821	0.820	0.819	0.818	0.818	0.817	0.817	0.816	0.816	0.816	0.815	0.814
Loss	MAE	−1.583	−1.611	−1.649	−1.694	−1.739	−1.769	−1.788	−1.803	−1.817	−1.842	−1.875	−1.941
	RMSE	−2.433	−2.46	−2.498	−2.544	−2.589	−2.629	−2.663	−2.688	−2.71	−2.738	−2.786	−2.876
	CC	−0.015	−0.015	−0.014	−0.014	−0.015	−0.015	−0.016	−0.015	−0.016	−0.016	−0.016	−0.018

Table 13
Performance on training and test data, and differences test−training error of the GRU-WS3 model for all 24-steps-ahead predictions in the NO_2 with O_3 problem.

	Steps-ahead											
		1	2	3	4	5	6	7	8	9	10	11	12
Training	Instances	6129	6128	6127	6126	6125	6124	6123	6122	6121	6120	6119	6118
	MAE	6.807	8.506	9.437	10.073	10.432	10.680	10.828	10.933	11.017	11.093	11.175	11.229
	RMSE	9.445	11.738	12.924	13.569	13.951	14.156	14.279	14.360	14.438	14.511	14.583	14.630
	CC	0.924	0.880	0.852	0.836	0.826	0.820	0.817	0.815	0.812	0.810	0.808	0.807
Test	Instances	2628	2627	2626	2625	2624	2623	2622	2621	2620	2619	2618	2617
	MAE	6.088	7.587	8.513	9.109	9.469	9.659	9.789	9.852	9.912	9.967	10.006	10.027
	RMSE	8.594	10.725	12.010	12.837	13.331	13.619	13.794	13.920	14.016	14.097	14.137	14.153
	CC	0.931	0.891	0.861	0.840	0.826	0.818	0.814	0.810	0.807	0.805	0.803	0.802
Loss	MAE	−0.719	−0.919	−0.924	−0.964	−0.963	−1.021	−1.039	−1.081	−1.105	−1.126	−1.169	−1.202
	RMSE	−0.851	−1.013	−0.914	−0.732	−0.62	−0.537	−0.485	−0.44	−0.422	−0.414	−0.446	−0.477
	CC	−0.007	−0.011	−0.009	−0.004	0	0.002	0.003	0.005	0.005	0.005	0.005	0.005
	Steps-ahead												
		13	14	15	16	17	18	19	20	21	22	23	24
Training	Instances	6117	6116	6115	6114	6113	6112	6111	6110	6109	6108	6107	6106
	MAE	11.265	11.282	11.299	11.315	11.325	11.335	11.343	11.348	11.353	11.358	11.363	11.370
	RMSE	14.659	14.673	14.689	14.706	14.717	14.726	14.735	14.742	14.748	14.753	14.758	14.765
	CC	0.806	0.806	0.806	0.805	0.805	0.804	0.804	0.804	0.804	0.804	0.803	0.803
Test	Instances	2616	2615	2614	2613	2612	2611	2610	2609	2608	2607	2606	2605
	MAE	10.009	10.022	10.042	10.057	10.065	10.074	10.082	10.086	10.090	10.094	10.100	10.106
	RMSE	14.063	14.073	14.092	14.104	14.113	14.121	14.130	14.139	14.148	14.157	14.165	14.174
	CC	0.802	0.801	0.800	0.799	0.799	0.798	0.798	0.798	0.798	0.797	0.797	0.797
Loss	MAE	−1.256	−1.26	−1.257	−1.258	−1.26	−1.261	−1.261	−1.262	−1.263	−1.264	−1.263	−1.264
	RMSE	−0.596	−0.6	−0.597	−0.602	−0.604	−0.605	−0.605	−0.603	−0.6	−0.596	−0.593	−0.591
	CC	0.004	0.005	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.007	0.006	0.006

Table 14
Performance on training and test data, and differences test−training error of the GRU-WS3 model for all 24-steps-ahead predictions in the NO_x problem.

	Steps-ahead											
		1	2	3	4	5	6	7	8	9	10	11	12
Training	Instances	6129	6128	6127	6126	6125	6124	6123	6122	6121	6120	6119	6118
	MAE	6.730	8.344	9.360	9.972	10.332	10.568	10.723	10.840	10.927	11.008	11.092	11.155
	RMSE	9.341	11.694	13.064	13.843	14.314	14.608	14.806	14.955	15.078	15.196	15.307	15.394
	CC	0.922	0.877	0.848	0.832	0.823	0.818	0.815	0.813	0.811	0.810	0.808	0.807
Test	Instances	2628	2627	2626	2625	2624	2623	2622	2621	2620	2619	2618	2617
	MAE	37.271	52.544	61.902	67.533	71.362	73.699	75.104	75.924	76.564	77.201	77.646	78.134
	RMSE	75.834	104.220	119.080	126.997	131.627	134.625	136.233	136.704	137.022	137.538	137.993	138.520
	CC	0.895	0.813	0.766	0.735	0.716	0.704	0.699	0.702	0.706	0.707	0.707	0.707
Loss	MAE	30.541	44.2	52.542	57.561	61.03	63.131	64.381	65.084	65.637	66.193	66.554	66.979
	RMSE	66.493	92.526	106.016	113.154	117.313	120.017	121.427	121.749	121.944	122.342	122.686	123.126
	CC	0.027	0.064	0.082	0.097	0.107	0.114	0.116	0.111	0.105	0.103	0.101	0.1
	Steps-ahead												
		13	14	15	16	17	18	19	20	21	22	23	24
Training	Instances	6117	6116	6115	6114	6113	6112	6111	6110	6109	6108	6107	6106
	MAE	11.211	11.264	11.323	11.388	11.453	11.504	11.544	11.575	11.606	11.642	11.699	11.801
	RMSE	15.469	15.535	15.608	15.690	15.770	15.838	15.895	15.939	15.979	16.026	16.107	16.259
	CC	0.806	0.805	0.805	0.804	0.803	0.802	0.801	0.801	0.800	0.800	0.799	0.796
Test	Instances	2616	2615	2614	2613	2612	2611	2610	2609	2608	2607	2606	2605
	MAE	78.581	78.942	79.183	79.359	79.480	79.569	79.649	79.727	79.786	79.847	79.868	79.905
	RMSE	139.041	139.471	139.751	139.895	139.967	140.027	140.093	140.151	140.196	140.235	140.260	140.293
	CC	0.707	0.706	0.706	0.706	0.706	0.706	0.706	0.706	0.706	0.706	0.706	0.706
Loss	MAE	67.37	67.678	67.86	67.971	68.027	68.065	68.105	68.152	68.18	68.205	68.169	68.104
	RMSE	123.572	123.936	124.143	124.205	124.197	124.189	124.198	124.212	124.217	124.209	124.153	124.034
	CC	0.099	0.099	0.099	0.098	0.097	0.096	0.095	0.095	0.094	0.094	0.093	0.09

24-steps-ahead predictions on test data. In order to observe the overfitting of the models, the loss between the errors in the train and test evaluation has also been shown in these tables. The same information can be graphically seen in Figs. A.7 to A.9.

For a more thorough analysis, Figures from B.10 to B.13 show, for each problem, a portion of the actual test database values along with some of the step-ahead predictions. Finally, Figures from C.14 to C.17 show plots of the best deep learning architectures chosen for each problem. As can be seen in these graphs, all the deep learning models selected by the multi-criteria decision making process contain the following layers:

- An input layer that sends data to subsequent layers. The dimension of the input tensor is the tuple (timesteps, number of features). Timesteps is the memory of the neural network and it is always set to 1 in our deep learning models. The number of features is 168 for LSTM-WS24 NO_2 prediction model, 21 for GRU-WS3 NO_x prediction model, 27 for GRU-WS3 NO_2 with O_3 prediction model, and 27 for LSTM-WS3 NO_x with O_3 prediction model.
- A GRU or LSTM hidden layer, depending on the case, with 256 neurons and ReLU activation.

Table 15
Performance on training and test data, and differences test−training error of the LSTM-WS3 model for all 24-steps-ahead predictions in the NO_x with O_3 problem.

	Steps-ahead											
		1	2	3	4	5	6	7	8	9	10	11	12
Training	Instances	6129	6128	6127	6126	6125	6124	6123	6122	6121	6120	6119	6118
	MAE	20.836	26.720	29.872	31.650	32.865	33.596	34.132	34.514	34.852	35.095	35.253	35.340
	RMSE	33.848	44.750	50.158	52.859	54.320	55.039	55.553	55.928	56.340	56.673	56.850	56.953
	CC	0.924	0.865	0.828	0.807	0.794	0.787	0.782	0.779	0.775	0.772	0.770	0.769
Test	Instances	2628	2627	2626	2625	2624	2623	2622	2621	2620	2619	2618	2617
	MAE	34.426	47.866	55.525	59.927	62.553	63.785	64.390	64.931	65.409	65.788	66.021	66.083
	RMSE	67.919	95.964	111.086	118.836	123.167	125.173	126.276	126.918	127.581	128.168	128.532	128.630
	CC	0.908	0.821	0.768	0.738	0.722	0.719	0.718	0.719	0.718	0.717	0.715	0.714
Loss	MAE	13.59	21.146	25.653	28.277	29.688	30.189	30.258	30.417	30.557	30.693	30.768	30.743
	RMSE	34.071	51.214	60.928	65.977	68.847	70.134	70.723	70.99	71.241	71.495	71.682	71.677
	CC	0.016	0.044	0.06	0.069	0.072	0.068	0.064	0.06	0.057	0.055	0.055	0.055
	Steps-ahead												
		13	14	15	16	17	18	19	20	21	22	23	24
Training	Instances	6117	6116	6115	6114	6113	6112	6111	6110	6109	6108	6107	6106
	MAE	35.393	35.436	35.460	35.480	35.500	35.516	35.531	35.538	35.549	35.556	35.566	35.579
	RMSE	57.023	57.062	57.085	57.096	57.109	57.121	57.134	57.143	57.152	57.160	57.167	57.176
	CC	0.769	0.768	0.768	0.768	0.768	0.768	0.768	0.768	0.768	0.768	0.767	0.767
Test	Instances	2616	2615	2614	2613	2612	2611	2610	2609	2608	2607	2606	2605
	MAE	65.876	65.812	65.813	65.768	65.746	65.729	65.746	65.755	65.763	65.766	65.766	65.740
	RMSE	127.509	127.337	127.345	127.306	127.286	127.291	127.313	127.332	127.350	127.369	127.386	127.382
	CC	0.715	0.715	0.714	0.714	0.714	0.714	0.714	0.714	0.714	0.714	0.714	0.714
Loss	MAE	30.483	30.376	30.353	30.288	30.246	30.213	30.215	30.217	30.214	30.21	30.2	30.161
	RMSE	70.486	70.275	70.26	70.21	70.177	70.17	70.179	70.189	70.198	70.209	70.219	70.206
	CC	0.054	0.053	0.054	0.054	0.054	0.054	0.054	0.054	0.054	0.054	0.053	0.053

- A dropout hidden layer with a 0.2 probability of deactivating a neuron.
- An output layer with an output neuron and a linear activation function.

5. Analysis of results and discussion

Several considerations can be drawn upon analysis of our results. Let us focus, first on Tables 3 through 6, which allow us to asses how good the algorithms that we have used are able to fit the past pollution data. Three elements emerge very clearly: (*i*) the absolute correlations that we have obtained in the best cases are much better than those previously obtained on similar data (see, e.g., [4,59]) and better than those previously obtained on the same data [11]; (*ii*) the deep-learning technology (LSTM and GRU, in particular) revealed themselves as very interesting candidates for solving this problem, although, in same cases, more standard technologies, such as RF, showed notable performances, and (*iii*) the use of past values of independent variables, with windows varying from 3 to 24 h, has positive consequences on the CC. Moreover, counter-intuitively, our results show that adding O_3 in the prediction variables do not increase, in general, the CC. Finally, we can also observe that in predicting NO_2 one has a clear advantage in considering longer windows (up to 24 h), while better predictions of NO_x emerge with shorter windows (3 h).

Focusing on Tables 7 through 10, on the other hand, gives us the possibility of establishing if there are statistically significant differences among the several approaches and datasets. We can indeed observe that LSTM and GRU networks, as well as RF, have a clear advantage over all other approaches in all cases. Moreover, it seems that in predicting NO_2 without O_3 longer windows of past data are necessary, while in all other cases, even in predicting NO_2 with O_3, shorter windows show some advantage. One possible explanation include observing that longer windows imply more attributes, which, in turn, would require more training time to be dealt with, at least in the deep-learning approaches.

Algorithm 1 Multi-criteria decision-making.

Require: $\Phi = \{\phi_1, \ldots, \phi_p\}$ {Set of p prediction models}
Require: WS3-30, WS6-30, WS12-30, WS24-30 {Test datasets}
Require: h {Number of steps-ahead}
1: $RMSE_j^i \leftarrow$ Normalized $RMSE$ of ϕ_j on its corresponding test dataset in the i-step-ahead, $j = 1, \ldots, p$, $i = 1, \ldots, h$
2: $MAE_j^i \leftarrow$ Normalized MAE of ϕ_j on its corresponding test dataset in the i-step-ahead, $j = 1, \ldots, p$, $i = 1, \ldots, h$
3: $CC_j^i \leftarrow$ Normalized $1-CC$ of ϕ_j on its corresponding test dataset in the i-step-ahead, $j = 1, \ldots, p$, $i = 1, \ldots, h$
4: $eRMSE_j \leftarrow \sum_{i=1}^{h} RMSE_j^i, j = 1, \ldots, p$ {Exactness for $RMSE$}
5: $eMAE_j \leftarrow \sum_{i=1}^{h} MAE_j^i, j = 1, \ldots, p$ {Exactness for MAE}
6: $eCC_j \leftarrow \sum_{i=1}^{h} CC_j^i, j = 1, \ldots, p$ {Exactness for CC}
7: $rRMSE_j \leftarrow \sum_{i=1}^{h-1} \left|RMSE_j^{i+1} - RMSE_j^i\right|, j = 1, \ldots, p$ {Robustness for $RMSE$}
8: $rMAE_j \leftarrow \sum_{i=1}^{h-1} \left|MAE_j^{i+1} - MAE_j^i\right|, j = 1, \ldots, p$ {Robustness for MAE}
9: $rCC_j \leftarrow \sum_{i=1}^{h-1} \left|CC_j^{i+1} - CC_j^i\right|, j = 1, \ldots, p$ {Robustness for CC}
10: $G_j \leftarrow rRMSE_j \cdot eRMSE_j + rMAE_j \cdot eMAE_j + rCC_j \cdot eCC_j, j = 1, \ldots, n$ {Goodness}
11: $\phi^* \leftarrow \phi_{min} \mid G_{min} = \min_{j=1}^{p}\{G_j\}$
12: **return** ϕ^*

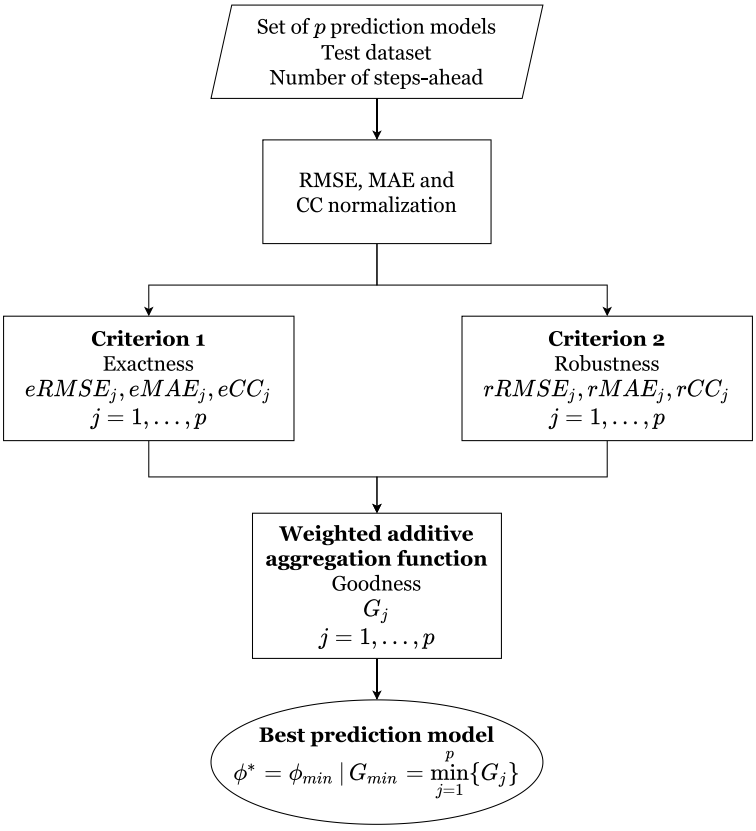


Fig. 3. Multi-criteria decision making flow chart.

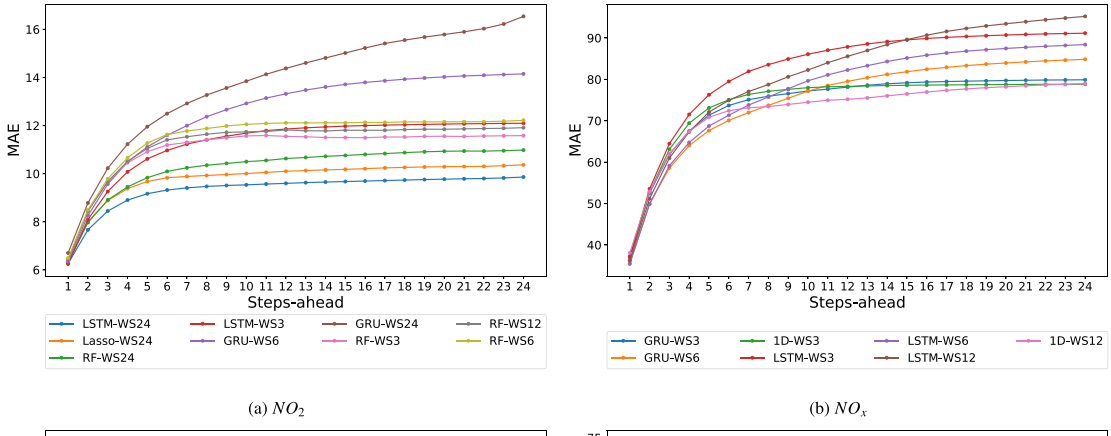
A different point of view comes from reading Table 11, in which we analyze the ability of the different approaches to predict future values. Observe that this is a different problem: while fitting a curve, we use past values of the predictors to predict the present value of the pollution concentration; here, instead, past values of the predictors are used to predict *future* values of the pollution concentration. Thus, as a line of principle, we prefer *good enough, robust* models over *very good, but not robust enough* ones. The goodness index, computed as explained above, formalizes this idea. Once again, the LSTM and the GRU approaches resulted to be the most stable ones, offering good enough predictions, with low enough errors, even with at a 24 h horizon in the future. Another surprising element is that models for NO_x predictions seem considerably more stable than models for NO_2 prediction, both with and without O_3. This is surprising as, both in this work and in past work with similar data, NO_x prediction has always been more difficult than NO_2 prediction. Looking at the Tables 12 to 15, LSTM and GRU have little overfitting for problems NO_2 and NO_2 with O_3, while they show higher overfitting for NO_x and NO_x with O_3 problems. Finally, one can observe that stability for higher numbers of steps-ahead is correlated with smaller windows, and a 3-hours window seem to be the one that works better.

Unlike previous work on similar data, here we focused on exploring different prediction techniques using past and present data, and instead of searching for explanatory models, we searched for usable prediction models. A deep-learning network such as a GRU or an LSTM, we proved, can be the ideal solution for an integrated system that, paired with a continuous monitoring of data, offers predictions up to 24 h ahead, and alerts in case of a too high contamination concentration prediction. Such a type of solutions seem to be quite common in the modern literature (see, e.g., [60]).

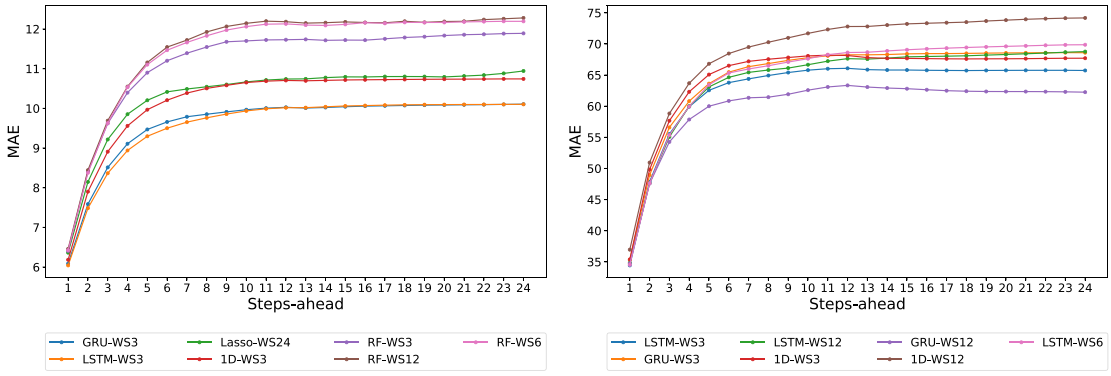
6. Conclusions and future work

Over the years, diverse factors such as the increase in the number of industries or vehicles have led to a high level of air pollution, becoming one of the most serious environmental problems in the world. Toxic gases can affect both the health of humans causing respiratory or cardiovascular problems and ecosystems with the appearance of acid rain. Therefore, it is of great interest to monitor these levels of air pollution. In this paper we proposed a methodology to evaluate and compare deep learning models for multivariate time series forecasting, that includes lagged transformations, hyper-parameter tuning, statistical tests, multi-criteria decision making and *h*-step-ahead prediction. We have designed an objective methodology to evaluate the goodness of a prediction technique, and applied to the ones we tested. We concluded that the deep-learning approach, and in particular LSTM and GRU, with windows between 3 and 24 h, allow for a very reliable 24-hours ahead prediction. Our results, that includes prediction with correlation indexes in many cases greater than 0.9, are far better than those previously obtained with similar data.

When comparing so many prediction models, as is the case in our research, it is difficult to discern which one is the best, as there are numerous performance measures to consider when making the decision. In these cases, a multi-criteria decision making process is required to obtain a single performance measure associated with each model and thus obtain a total ranking of all prediction models. The multi-criteria decision-making process proposed in this work has allowed us to clearly choose GRU-WS3-70, LSTM-WS3-70 and LSTM-WS24-70 deep learning models as the best according to criteria of exactness and robustness, depending on whether the problem target is NO_2 or NO_x (with and without O_3), improving the models 1DCNN.

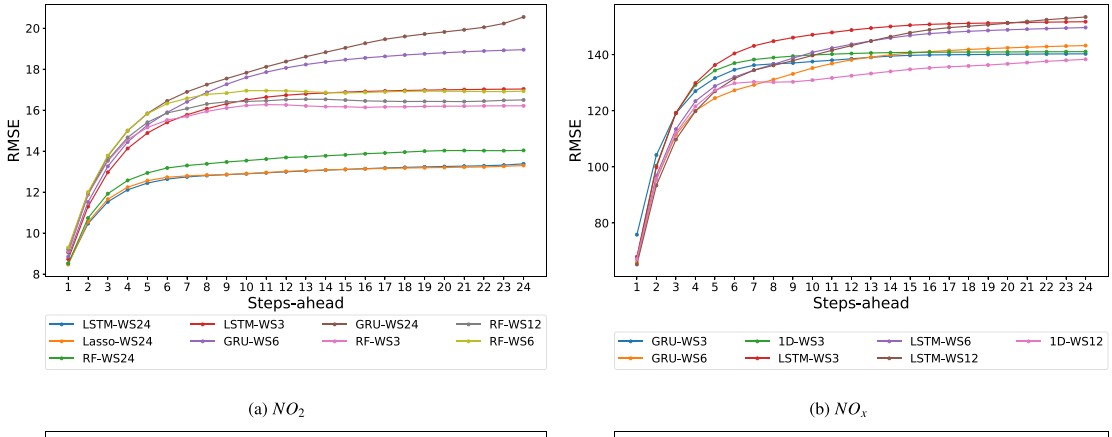


(a) NO_2 (b) NO_x

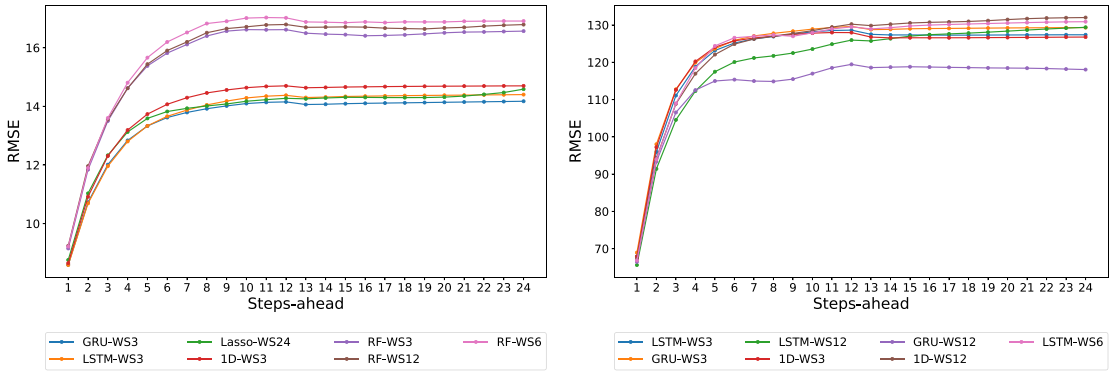


(c) NO_2 with O_3 (d) NO_x with O_3

Fig. A.4. MAE from 1 to 24-steps-ahead of the best models from each prediction problem.

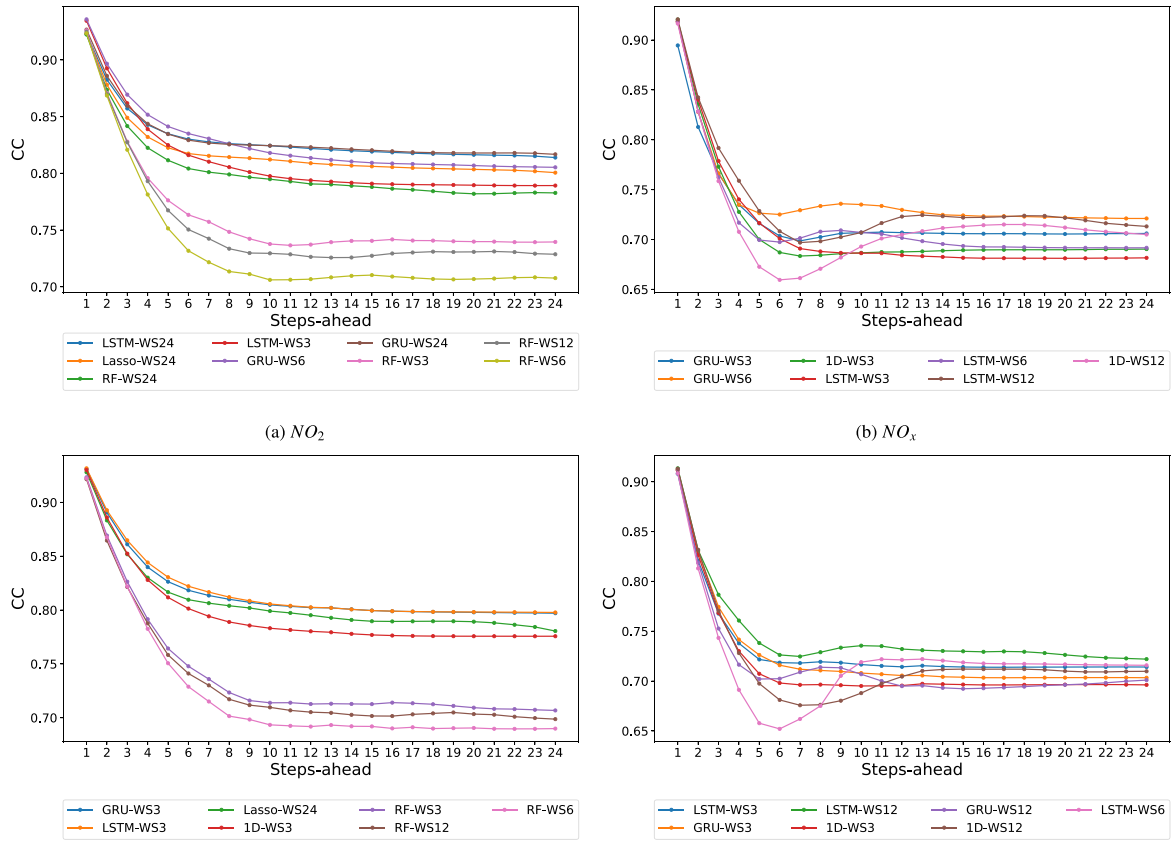


(a) NO_2 (b) NO_x



(c) NO_2 with O_3 (d) NO_x with O_3

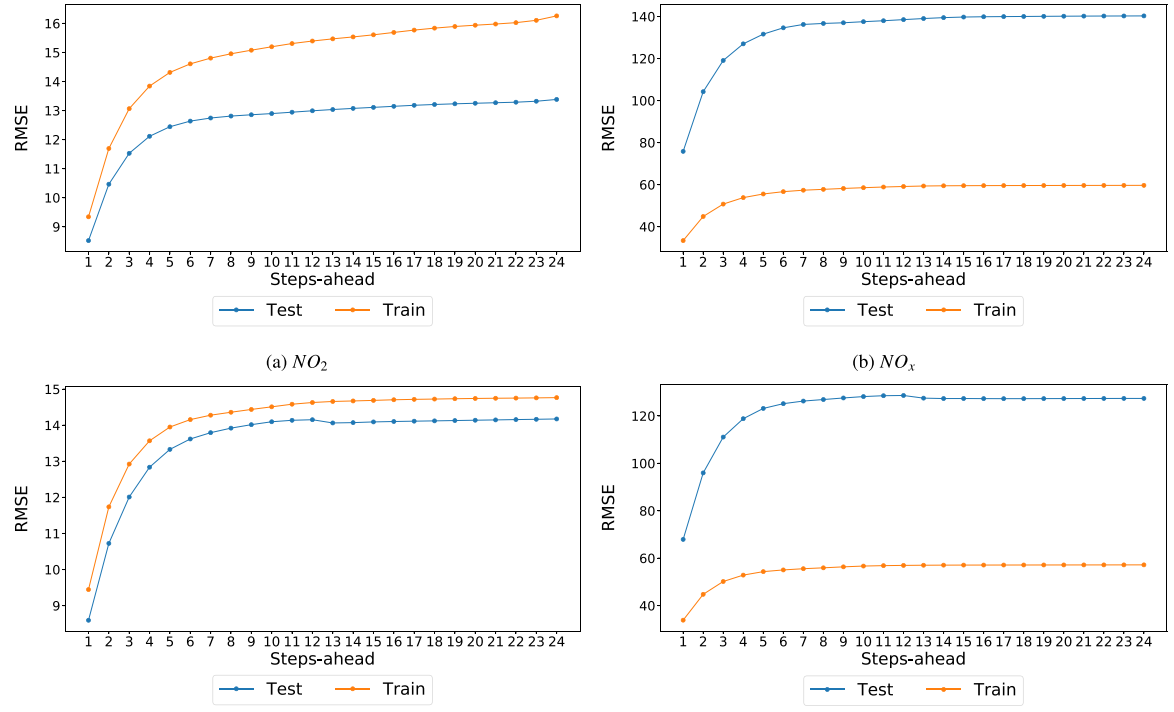
Fig. A.5. RMSE from 1 to 24-steps-ahead of the best models from each prediction problem.



(a) NO_2 (b) NO_x

(c) NO_2 with O_3 (d) NO_x with O_3

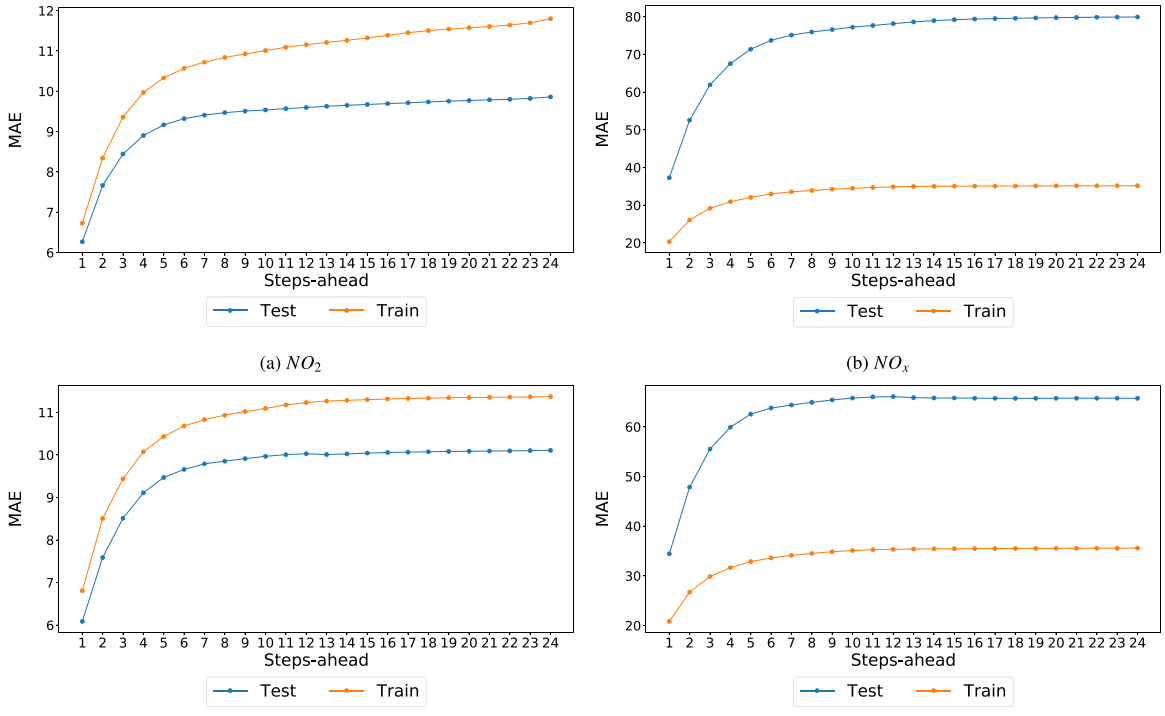
Fig. A.6. CC from 1 to 24-steps-ahead of the best models from each prediction problem.



(a) NO_2 (b) NO_x

(c) NO_2 with O_3 (d) NO_x with O_3

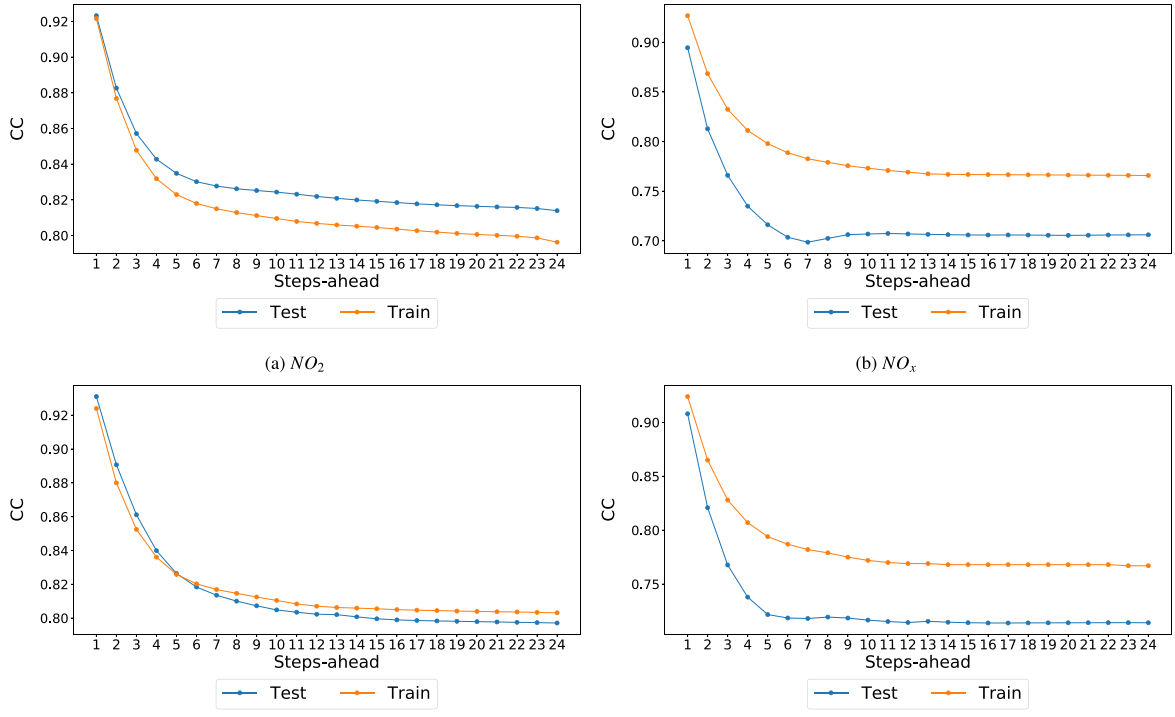
Fig. A.7. RMSE summary of the prediction models chosen with the proposed methodology.



(a) NO_2 (b) NO_x

(c) NO_2 with O_3 (d) NO_x with O_3

Fig. A.8. MAE summary of the prediction models chosen with the proposed methodology.



(a) NO_2 (b) NO_x

(c) NO_2 with O_3 (d) NO_x with O_3

Fig. A.9. CC summary of the prediction models chosen with the proposed methodology.

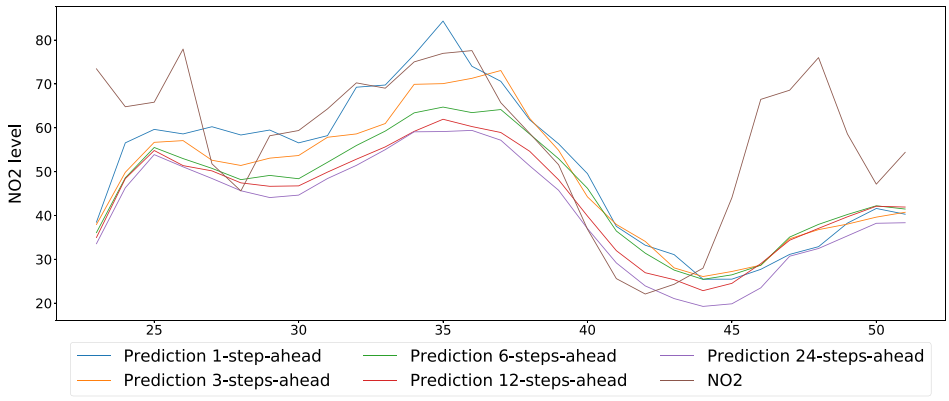


Fig. B.10. NO_2 prediction with LSTM-WS24.

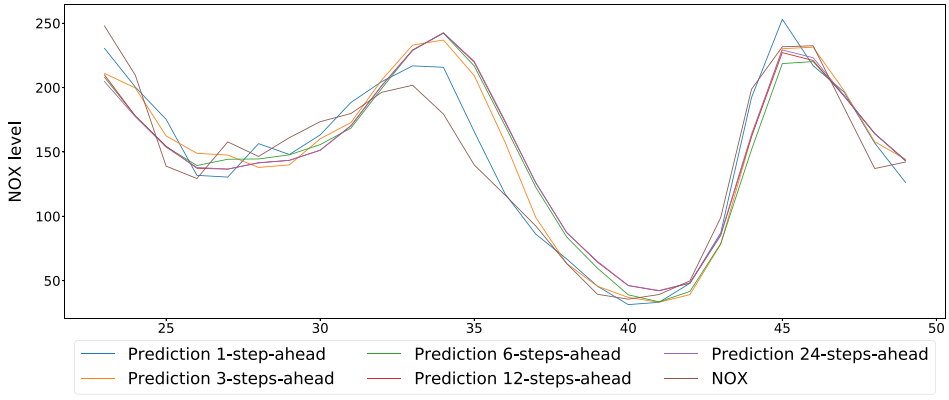


Fig. B.11. NO_x prediction with GRU-WS3.

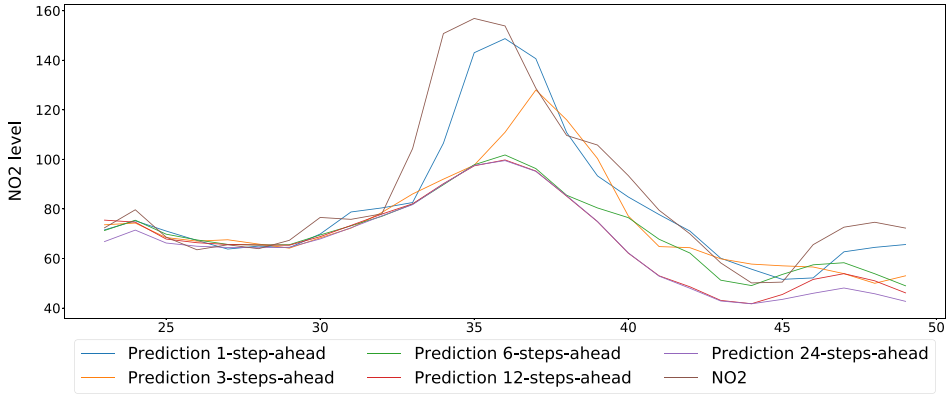


Fig. B.12. NO_2 with O_3 prediction with GRU-WS3.

Among future works, we want to include in the methodology different techniques to calculate step-ahead predictions. In this paper, we have used the recursive technique. Other extensions of the methodology will be focused on spatio-temporal models. Spatio-temporal models will make it possible to approach pollution forecasting using the information provided by a network of spatially distributed sensors. Furthermore, spatio-temporal models will allow us to capture patterns involving pollutants dynamics.

CRediT authorship contribution statement

Raquel Espinosa: Term, Conceptualization, Methodology, Software, Validation, Investigation, Resources, Data curation, Writing

– original draft, Writing – review & editing, Visualization, Project administration. **José Palma:** Term, Conceptualization, Methodology, Formal analysis, Investigation, Writing – original draft, Writing – review & editing, Supervision, Funding acquisition. **Fernando Jiménez:** Term, Conceptualization, Methodology, Formal analysis, Investigation, Writing – original draft, Writing – review & editing, Supervision, Funding acquisition. **Joanna Kamińska:** Validation, Resources, Data curation, Writing – original draft, Writing – review & editing. **Guido Sciavicco:** Validation, Data curation, Writing – original draft, Writing – review & editing. **Estrella Lucena-Sánchez:** Validation, Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was partially funded by the SITSUS project (Ref: RTI2018-094832-B-I00), given by the Spanish Ministry of Science, Innovation and Universities (MCIU), the Spanish Agency for Research (AEI) and by the European Fund for Regional Development (FEDER). This work was supported by the Science and Technology Agency, Séneca Foundation, Comunidad Autónoma Región de Murcia, Spain through the research projects 00004/COVI/20 and 00007/COVI/20.

Appendix A. Performance of prediction models

See Figs. A.4–A.9.

Appendix B. Step-ahead predictions

See Figs. B.10–B.13.

Appendix C. Deep learning architectures

See Figs. C.14–C.17.

Appendix D. Organizational chart of tables and figures

See Fig. D.18.

Appendix E. Abbreviations

See Table E.16.

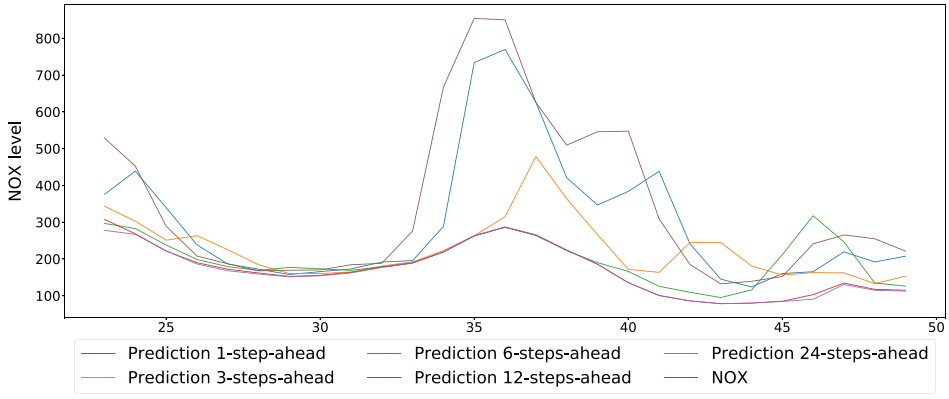


Fig. B.13. NO_x with O_3 prediction with LSTM-WS3.

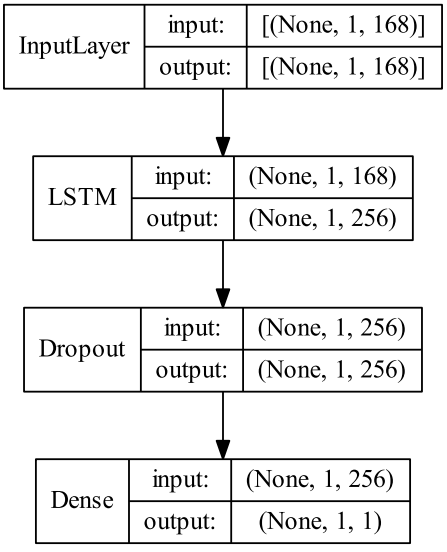


Fig. C.14. Architecture of the LSTM-WS24 deep learning model for NO_2 prediction.

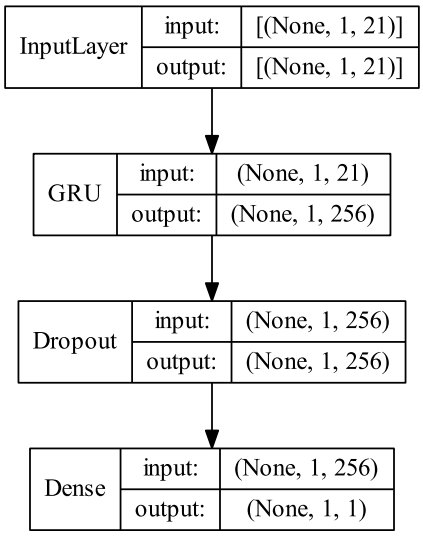


Fig. C.15. Architecture of the GRU-WS3 deep learning model for NO_x prediction.

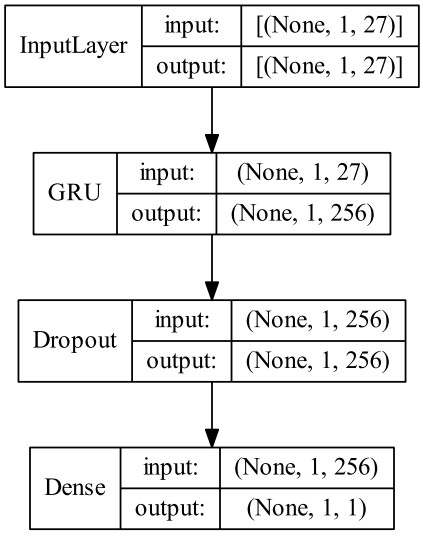


Fig. C.16. Architecture of the GRU-WS3 deep learning model for NO_2 with O_3 prediction.

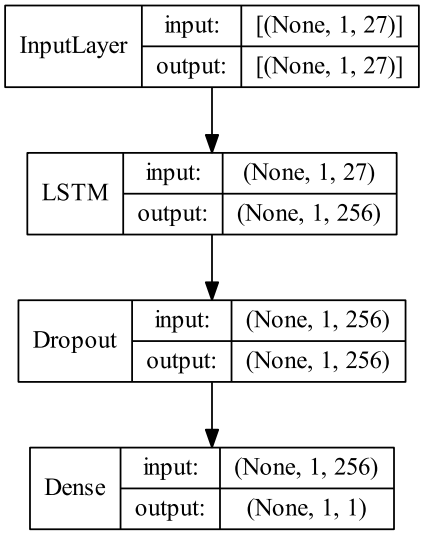


Fig. C.17. Architecture of the LSTM-WS3 deep learning model for NO_x with O_3 prediction.

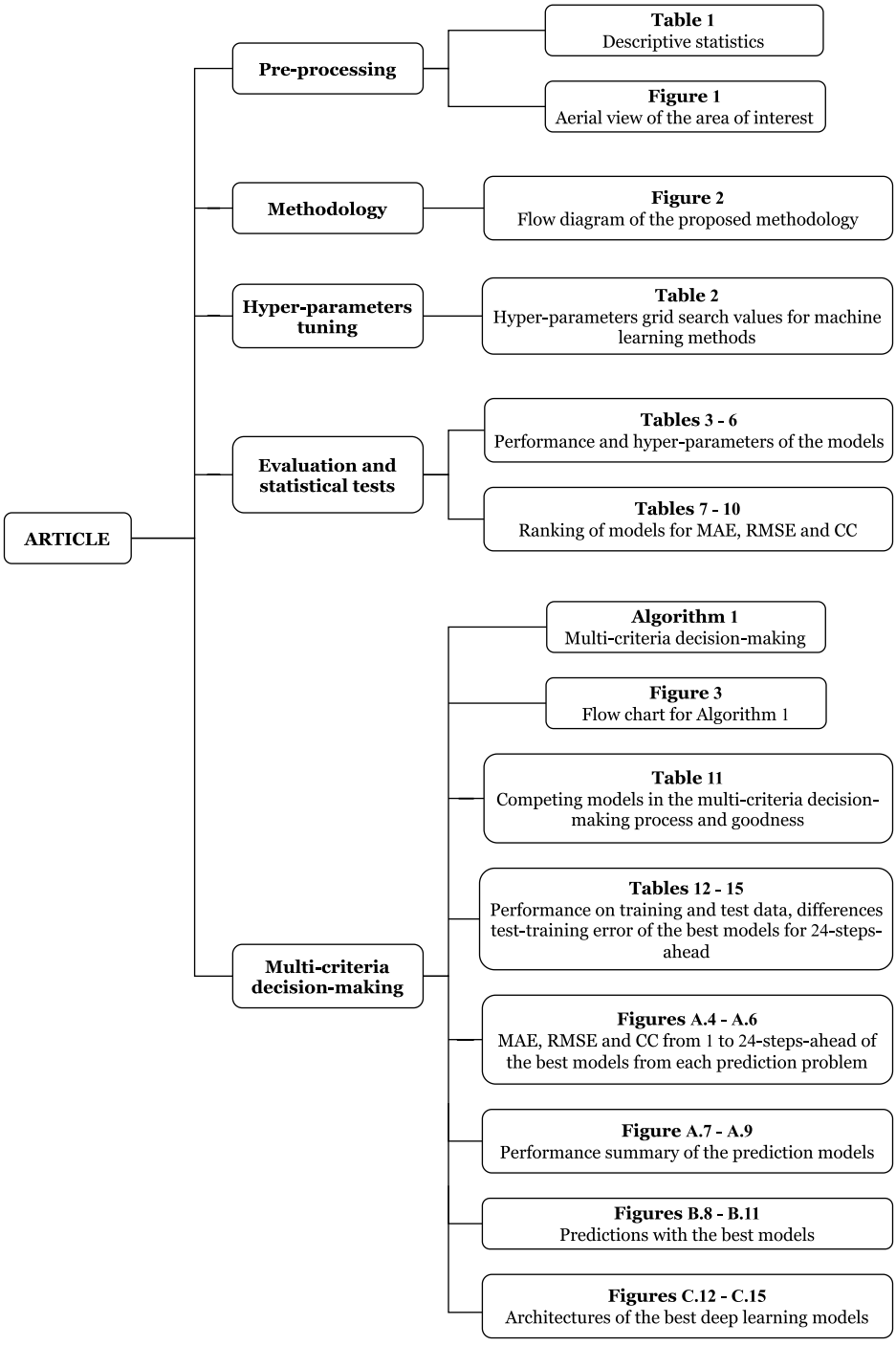


Fig. D.18. Organizational chart of links between the phases of the methodology and its results.

Table E.16
Abbreviations.

Abbreviation	Meaning
1DCNN	One-Dimensional Convolutional Neural Networks
ANN	Artificial Neural Network
AQI	Air Quality Index
ARIMA	Autoregressive Integrated Moving Average
Bi-LSTM	Bi-directional LSTM
CC	Correlation Coefficient
CNN	Convolutional Neural Networks
CO	Carbon monoxide
DAQFF	Deep Air Quality Forecasting Framework

(*continued on next page*)

Table E.16 (*continued*).

Abbreviation	Meaning
DFNN	Deep Feedforward Neural Network
DFS	Deep Flexible Sequential
DRNN	Deep Recurrent Neural Networks
DTR	Decision Tree Regressor
GBR	Gradient Boosting Regressor
GC-DCRNN	Geo-Context based Diffusion Convolutional Recurrent Neural Network
GRU	Gated Recurrent Unit
Lasso	Least Absolute Shrinkage and Selection Operator
LR	Linear Regression
LSTM	Long Short Term Memory
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MART	Multiple Additive Regression Trees
MLP	Multilayer Perceptron
MLR	Multiple Linear Regression
MSE	Mean Square Error
NO_2	Nitrogen dioxide
NO_x	Generic term for the nitrogen oxides (NO and NO_2)
O_3	Ozone
PM_{10}	Particulate Matter with a diameter of less than 10 micrometers
$PM_{2.5}$	Particulate Matter with a diameter of less than 2.5 micrometers
R^2	Coefficient of determination
ReLU	Rectified Linear Unit
RF	Random Forest
RMSE	Root Mean Square Error
RNN	Recurrent Neural Networks
SVM	Support Vector Machine
SVMRadial	Support Vector Machine with radial basis function
SVR	Support Vector Regression
WS	Window Size

References

[1] M. Chalfen, J. Kamińska, Identification of parameters and verification of an urban traffic flow model. a case study in wrocław, ITM Web Conf. 2018 23 (5) (2018).

[2] J.K. Kazak, D.G. Castro, M. Swiader, S. Szewranski, Decision support system in public transport planning for promoting urban adaptation to climate change, IOP Conf. Ser.: Mater. Sci. Eng. 471 (2019) 112007.

[3] J.P. Shi, R.M. Harrison, Regression modelling of hourly no_x and no_2 concentrations in urban air in London, Atmos. Environ. 31 (24) (1997) 4081–4094.

[4] K.P. Singh, S. Gupta, A. Kumar, S.P. Shukla, Linear and nonlinear modeling approaches for urban air quality prediction, Sci. Total Environ. 426 (2012) 244–255.

[5] E. Lucena-Sánchez, F. Jiménez, G. Sciavicco, J. Kaminska, Simple versus composed temporal lag regression with feature selection, with an application to air quality modeling, in: 2020 IEEE Conference on Evolving and Adaptive Intelligent Systems, EAIS 2020, Bari, Italy, May 27-29, 2020, IEEE, 2020, pp. 1–8.

[6] F. Nejadkoorki, S. Baroutian, Forecasting extreme PM_{10} concentrations using artificial neural networks, Int. J. Environ. Res. 6 (1) (2012) 277–284.

[7] A. Brunello, J. Kamińska, E. Marzano, A. Montanari, G. Sciavicco, T. Turek, Assessing the role of temporal information in modelling short-term air pollution effects based on traffic and meteorological conditions: A case study in wrocław, in: T. Welzer, J. Eder, V. Podgorelec, R. Wrembel, M. Ivanovic, J. Gamper, M. Morzy, T. Tzouramanis, J. Darmont, A.K. Latific (Eds.), New Trends in Databases and Information Systems, ADBIS 2019 Short Papers, Workshops BBIGAP, QAUCA, SemBDM, SIMPDA, M2P, MADEISD, and Doctoral Consortium, Bled, Slovenia, September 8-11, 2019, Proceedings, in: Communications in Computer and Information Science, 1064, Springer, 2019, pp. 463–474.

[8] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.

[9] R. Tibshirani, Regression shrinkage and selection via the Lasso, J. Roy. Statist. Soc. Ser. A 58 (1996) 267–288.

[10] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (3) (1995) 273–297.

[11] J. Kaminska, A random forest partition model for predicting no_2 concentrations from traffic flow and meteorological conditions, Sci. Total Environ. 651 (2019) 475–483.

[12] J. Kaminska, T. Turek, Explicit and implicit description of the factors impact on the no_2 concentration in the traffic corridor, Arch. Environ. Prot. (2020) 93–99.

[13] W. Sun, H. Zhang, A. Palazoglu, A. Singh, W. Zhang, S. Liu, Prediction of 24-hour-average PM2.5 concentrations using a hidden Markov model with different emission distributions in Northern California, Sci. Total. Environ. 443 (2013) 93–103.

[14] S. Patra, Time series forecasting of air pollutant concentration levels using machine learning, Adv. Comput. Sci. Inf. Technol. 4 (5) (2017) 280–284.

[15] D. Dua, C. Graff, UCI Machine Learning Repository, 2017, http://archive.ics.uci.edu/ml.

[16] I. Kok, M. Simsek, S. Ozdemir, A deep learning model for air quality prediction in smart cities, in: IEEE International Conference on Big Data, 2017, pp. 1983–1990.

[17] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.

[18] J. Fan, Q. Li, J. Hou, X. Feng, H. Karimian, S. Lin, A spatiotemporal prediction framework for air pollution based on deep RNN, ISPRS Annal. Photogramm. Remote Sens. Spatial Inf. Sci. 44W2 (2017) 15–22.

[19] Y. Lin, N. Mago, Y. Gao, Y. Li, Y.-Y. Chiang, C. Shahabi, J.L. Ambite, Exploiting spatiotemporal patterns for accurate air quality forecasting using deep learning, in: Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2018, pp. 359–368.

[20] I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, in: Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 27, Curran Associates, Inc., 2014, pp. 3104–3112.

[21] B.S. Freeman, G. Taylor, B. Gharabaghi, J. Thé, Forecasting air quality time series using deep learning, J. Air Waste Manage. Assoc. 68 (8) (2018) 866–886.

[22] T. Bui, V. Le, S. Cha, A deep learning approach for air pollution forecasting in South Korea using encoder-decoder networks & LSTM, 2018, CoRR, arXiv:1804.07891.

[23] A. V., G. P., V. R., S. K.P., DeepAirNet: Applying recurrent networks for air quality prediction, Procedia Comput. Sci. 132 (2018) 1394–1403, International Conference on Computational Intelligence and Data Science.

[24] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, 2014, CoRR, abs/1406.1078, arXiv:1406.1078.

[25] Z. Qi, T. Wang, G. Song, W. Hu, X. Li, Z. Zhang, Deep air learning: Interpolation, prediction, and feature analysis of fine-grained air quality, IEEE Trans. Knowl. Data Eng. 30 (12) (2018) 2285–2297.

[26] A. Sharma, A. Mitra, S. Sharma, S. Roy, Estimation of air quality index from seasonal trends using deep neural network, in: Artificial Neural Networks and Machine Learning. ICANN 2018. Lecture Notes in Computer Science, vol. 11141, Springer, 2018, pp. 511–521.

[27] S. Du, T. Li, Y. Yang, S. Horng, Deep air quality forecasting using hybrid deep learning framework, IEEE Trans. Knowl. Data Eng. (2019) 1.

[28] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, D.J. Inman, 1D Convolutional neural networks and applications: A survey, Mech. Syst. Signal Process. 151 (2021) 107398.

[29] H. Lin, J. Jin, J. Van Den Herik, Air quality forecast through integrated data assimilation and machine learning, in: J. van den Herik, L. Steels, A. Rocha (Eds.), Proceedings of the 11th International Conference on Agents and Artificial Intelligence, ICAART 2019, 2, SciTePress, 2019, pp. 787–793, ICAART 2019 : 11th International Conference on Agents and Artificial Intelligence ; Conference date: 19-02-2019 Through 21-02-2019.
[30] A. Masih, Machine learning algorithms in air quality modeling, Glob. J. Environ. Sci. Manag. 5 (4) (2019) 515–534.
[31] H. Karimian, Q. Li, C. Wu, Y. Qi, Y. Mo, G. Chen, X. Zhang, S. Sachdeva, Evaluation of different machine learning approaches to forecasting PM2.5 mass concentrations, Aerosol and Air Qual. Res. 19 (6) (2019) 1400–1410.
[32] Q. Tao, F. Liu, Y. Li, D. Sidorov, Air pollution forecasting using a deep learning model based on 1D convnets and bidirectional GRU, IEEE Access 7 (2019) 76690–76698.
[33] X. Sun, W. Xu, H. Jiang, Spatial-temporal prediction of air quality based on recurrent neural networks, in: Proceedings of the 52nd Hawaii International Conference on System Sciences, 2019, pp. 1265–1274.
[34] S. Ameer, M. Shah, A. Khan, H. Song, C. Maple, S. Islam, M. Asghar, Comparative analysis of machine learning techniques for predicting air quality in smart cities, IEEE Access PP (2019) 1.
[35] K. Kaya, S. Gunduz Oguducu, Deep Flexible Sequential (DFS) model for air pollution forecasting, Sci. Rep. 10 (2020) 1–12.
[36] S. Li, G. Xie, J. Ren, L. Guo, Y. Yang, X. Xu, Urban $PM_{2.5}$ concentration prediction via attention-based CNN-LSTM, Appl. Sci. 10 (2020) 1953.
[37] O. Surakhi, S. Serhan, I. Salah, On the ensemble of recurrent neural network for air pollution forecasting: Issues and challenges, Adv. Sci. Technol. Eng. Syst. J. 5 (2020) 512–526.
[38] Y.-C. Lin, S.-J. Lee, C.-S. Ouyang, C.-H. Wu, Air quality prediction by neuro-fuzzy modeling approach, Appl. Soft Comput. 86 (2020) 105898.
[39] C.-Y. Lin, Y.-S. Chang, S. Abimannan, Ensemble multifeatured deep learning models for air quality forecasting, Atmospheric Pollut. Res. 12 (5) (2021) 101045.
[40] P. Nath, P. Saha, A. Middya, S. Roy, Long-term time-series pollution forecast using statistical and deep learning methods, Neural Comput. Appl. (2021).
[41] A. Heydari, M. Majidi Nezhad, D. Astiaso Garcia, F. Keynia, L. De Santoli, Air pollution forecasting application based on deep learning model and optimization algorithm, Clean Technol. Environ. Policy (2021).
[42] S. Du, T. Li, Y. Yang, S. Horng, Deep air quality forecasting using hybrid deep learning framework, IEEE Trans. Knowl. Data Eng. 33 (2021) 2412–2424.
[43] K. Tripathi, P. Pathak, Deep Learning Techniques for Air Pollution, in: 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), 2021, pp. 1013–1020.
[44] B. Wang, W. Kong, P. Zhao, An air quality forecasting model based on improved convnet and RNN, Soft Comput. 25 (2021).
[45] A. Géron, Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, O'Reilly Media, 2019.
[46] R. Adhikari, R. Agrawal, An introductory study on time series modeling and forecasting, 2013, CoRR, arXiv:1302.6613.
[47] J. Brownlee, Time series forecasting as supervised learning, in: Machine Learning Mastery, 2020, https://machinelearningmastery.com/time-series-forecasting-supervised-learning/.
[48] S.S.R. Moustafa, M.S. Abdalzaher, M.H. Yassien, T. Wang, M. Elwekeil, H.E.A. Hafiez, Development of an optimized regression model to predict blast-driven ground vibrations, IEEE Access 9 (2021) 31826–31841.
[49] Y. Bengio, Practical recommendations for gradient-based training of deep architectures, 2012, CoRR, arXiv:1206.5533.
[50] D. Soydaner, A comparison of optimization algorithms for deep learning, Int. J. Pattern Recognit. Artif. Intell. 34 (2020).
[51] S. Hayou, A. Doucet, J. Rousseau, On the impact of the activation function on deep neural networks training, in: K. Chaudhuri, R. Salakhutdinov (Eds.), Proceedings of the 36th International Conference on Machine Learning, in: Proceedings of Machine Learning Research, 97, PMLR, 2019, pp. 2672–2680.
[52] V. Nair, G.E. Hinton, Rectified linear units improve restricted Boltzmann machines, in: Proceedings of the 27th International Conference on International Conference on Machine Learning, in: ICML'10, Omni Press, 2010, pp. 807–814.
[53] J. Han, C. Moraga, The influence of the sigmoid function parameters on the speed of backpropagation learning, in: from Natural To Artificial Neural Computation, 1995.
[54] A. Namin, K. Leboeuf, R. Muscedere, H. Wu, M. Ahmadi, Efficient hardware implementation of the hyperbolic tangent sigmoid function, in: Proceedings - IEEE International Symposium on Circuits and Systems, 2009, pp. 2117–2120.
[55] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: G. Gordon, D. Dunson, M. Dudík (Eds.), Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, in: Proceedings of Machine Learning Research, 15, PMLR, Fort Lauderdale, FL, USA, 2011, pp. 315–323.
[56] H.H. Aghdam, E.J. Heravi, Guide to Convolutional Neural Networks: A Practical Application to Traffic-Sign Detection and Classification, Springer Publishing Company, Incorporated, 2017.
[57] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[58] G.E. Box, G.M. Jenkins, G.C. Reinsel, G.M. Ljung, Time series analysis: forecasting and control, John Wiley & Sons, 2015.
[59] I. Laña, J.D. Ser, A. Padró, M. Vélez, C. Casanova-Mateo, The role of local urban traffic and meteorological conditions in air pollution: A data-based case study in Madrid, Spain, Atmos. Environ. 145 (2016) 424–438.
[60] M. Tastan, H. Gökozan, Real-time monitoring of indoor air quality with internet of things-based E-nose, Appl. Sci. 16 (9) (2019) 3435–3448.