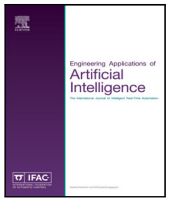
Engineering Applications of Artificial Intelligence 133 (2024) 108191

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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



Research paper

Spatial air quality prediction in urban areas via message passing

Sergio Calo [a,*], Filippo Bistaffa [b], Anders Jonsson [a], Vicenç Gómez [a], Mar Viana [c]

[a] Universitat Pompeu Fabra, Carrer de Roc Boronat 138, 08018, Barcelona, Spain
[b] IIIA-CSIC, Carrer de Can Planas (UAB Campus), 08193, Bellaterra, Spain
[c] IDAEA-CSIC, Carrer de Jordi Girona 18–26, 08034, Barcelona, Spain

ARTICLE INFO

Keywords:
Air quality
Graph neural networks
Graph signal reconstruction

ABSTRACT

Air pollution in urban areas poses a significant and pressing challenge for modern society. Unfortunately, the existing network of pollution detectors in many cities is limited in scope and fails to adequately cover the entire geographical area. Consequently, the implementation of spatial prediction algorithms becomes essential to generate high-resolution data. In this paper, we introduce two significant contributions: 1) We formalize the air pollution prediction problem as a Maximum A Posteriori (MAP) estimate within the framework of a Markov Random Field and 2) we propose a message-passing algorithm, which stands out as an efficient solution that surpasses the current state of the art. The experimental procedure has been carried out using the case study of the city of Barcelona, based on a dataset extracted from the BCN Open Data portal.

1. Introduction

Air quality is a major concern in urban areas, as poor air quality can lead to a range of negative health outcomes such as respiratory and cardiovascular diseases (Lelieveld et al., 2015). Accurate prediction of air quality is essential for mitigating these negative impacts and for developing effective strategies for improving air quality (Viana et al., 2020). However, predicting air quality in urban areas is a complex task, as it involves a range of factors such as meteorological conditions, traffic patterns, and emissions from various sources. Traditionally, air quality prediction has relied on models based on physical and chemical principles, which can be computationally expensive and may not accurately capture the complex interactions that occur in real-world urban environments.

In recent years, there has been a growing interest in the use of graph-based approaches for predicting air quality in urban areas. These approaches are based on the idea that air quality can be represented as a graph, with nodes representing different locations in the city and edges representing some kind of interaction between these locations. In this paper, we present a new approach to air quality prediction that utilizes message passing in graphs. Graph-based models are a powerful tool for representing complex systems, as they can capture the structural relationships between different elements of the system. This graph-based approach allows the use of Graph Neural Network (GNN) methods (Scarselli et al., 2008). GNNs are a class of deep learning models that operate on graph-structured inputs. In recent years, the field of GNN has experienced exponential growth in terms of published articles and general interest (Zhou et al., 2020). Its application has resulted in several breakthroughs in many different fields. To cite just a few examples, we have applications in Physics (Sanchez-Gonzalez et al., 2018), in combinatorial optimization problems (Khalil et al., 2017), in Computer Vision (Garcia and Bruna, 2017), in traffic forecasting (Rico et al., 2021; Peng and Zhang, 2023) and industry applications (Liu et al., 2023; Wang et al., 2023). We refer to Zhou et al. (2022) for a comprehensive survey. For this reason, we believe it is promising to explore the use of Graph Neural Networks in the proposed problem. This approach may allow the incorporation of structural information into the prediction process. By modeling the relationships between different variables, the graph neural network can capture more nuanced patterns and relationships that may not be captured by traditional models.

In this work, we propose a GNN-based approach that leverages message passing to model the interactions between different air quality sensors and predict the air quality at different locations in an urban area. To do so, the urban area of Barcelona is modeled as a graph, where the edges are its streets and the nodes are the intersections between them. The final approach consists of two main steps: regularization as graph signal smoothing and GNN refinement. The GNN training and validation are carried out using data from the city of Barcelona in different years. We show that with this mixed schema, we can achieve low prediction errors for these data.

* Corresponding author.
E-mail addresses: sergio.calo@upf.edu (S. Calo), filippo.bistaffa@iiia.csic.es (F. Bistaffa), anders.jonsson@upf.edu (A. Jonsson), vicen.gomez@upf.edu (V. Gómez), mar.viana@idaea.csic.es (M. Viana).

https://doi.org/10.1016/j.engappai.2024.108191
Received 29 May 2023; Received in revised form 5 January 2024; Accepted 29 February 2024
Available online 10 March 2024

2. Related work

Numerous studies in the literature have tackled the challenge of air pollutant prediction, often emphasizing either temporal or spatiotemporal perspectives. In the case of temporal prediction, the emphasis lies in forecasting air quality as a time series problem, centered on predicting the temporal evolution of monitored data (Reddy and Mohanty, 2017; Zeinalnezhad et al., 2020; Xayasouk et al., 2020). These studies commonly employ Deep Learning techniques like Long Short-Term Memory (LSTM) recurrent neural networks and sometimes integrate meteorological data to enhance predictive performance (Teng et al., 2018). Regarding spatiotemporal methodologies, the prevalent approach combines LSTM for temporal aspects with Convolutional Neural Networks (CNN) for spatial domains, aiming to analyze both temporal and spatial correlations in pollutant concentrations (Bekkar et al., 2021; Mao et al., 2021). However, alternative strategies exist, such as stacked autoencoder models extracting spatiotemporal features in a layer-wise manner (Li et al., 2016), or hierarchical Bayesian spatiotemporal approaches (Saez and Barceló, 2022). Zhang et al. (2022) deep learning model based on the Transformer architecture to learn long-term temporal dependencies for air quality PM2.5 prediction. Closer to our methodology, recent studies have explored Graph Neural Networks (GNNs) for capturing spatiotemporal correlations (Lu and te Li, 2020; Ouyang et al., 2021). These approaches represent spatial interdependencies between monitors using graph structures. Wang et al. (2020) use GNNs to model this type of spatiotemporal dependencies for the prediction of PM2.5 concentrations. Similarly, Han et al. (2022) employs GNNs to predict air quality in fine spatiotemporal granularity based on historical readings and various urban contextual factors such as weather conditions and traffic flows.

Unlike the aforementioned works, the peculiarities of the problem addressed mean that our method focuses only on spatial prediction. This lack of a temporal component makes our work differ from much of the existing literature. Another peculiarity of our setting is that we want to make predictions from a small number of known nodes to a large number of nodes without information. This means that the input we are working with is a large and very sparse signal. These peculiarities motivate the proposal of a method that is suitable for this setting.

Finally, it is also worth mentioning that the literature contains extensive analyses of the health impacts (Pierangeli et al., 2020; de Bont et al., 2019; Khomenko et al., 2021) and urban mobility implications (Rodríguez-Rey et al., 2020; Reche et al., 2022; Mueller et al., 2017) of air quality in Barcelona. These studies affirm the active and pertinent nature of air quality research in this city, with some delving into compositional analyses of air pollutants (Mota-Bertran et al., 2022).

3. Background

3.1. Graph signal processing

A graph is a data structure that captures different elements and the relationship between them. More formally, a graph G can be defined as (V, E, W), where V is a set of vertices or nodes, E is a set of connections, also called edges, between the vertices and W a weight matrix that represents the weights of those connections. We define a clique, C, as a subset of the vertices, $C \in V$, such that every two distinct vertices are adjacent. Each clique can be associated with a potential function ϕ. This can be any arbitrary positive function of the clique nodes.

The graph can be weighted, in case the edges have different weights, or unweighted if all these edges are equal. We can also distinguish between directed graphs, where the edges have a notion of direction (i.e., from node A to node B, but not vice versa), or undirected graphs, where this notion does not exist.

This structure admits different representations. One of them is the Adjacency matrix. This is an n by n square matrix, where n is the total number of nodes in the network. The elements of this matrix are zeros except in those positions (i, j) where a connection exists between nodes i and j. In that case, the value of the matrix element is 1 in the case of unweighted graphs or the value of the connection, $e_{i,j}$, in the case of weighted graphs. Furthermore, if the graph is undirected, the Adjacency matrix is symmetric.

Given the definition of a graph, we also can define the signal of a given node as a function f, such that

$$f : V \to \mathbb{R}$$

Given the node signal, we can then define a graph signal as the set of individual node signals in the following way

$$g = [f(v_1), f(v_2), \dots, f(v_N)] \mid v_1, v_2, \dots, v_N \in V$$

Graph signal processing (GSP) is an extension of the field of signal processing that deals with these signals in graphs, g. A specific family of problems within GSP is Graph Signal Reconstruction (GSR). The GSR process, generally understood, proposes the reconstruction of a complete graph signal from a series of samples extracted from it.

3.2. Graph signal processing as probabilistic graphical models

When it comes to GSP problems, numerous attempts to build a useful theoretical framework can be observed in recent literature. For instance, the spectral analysis of graph signals based on Fourier analysis, the same way as in one-dimensional signals or grids, is widely used in graphs for numerous applications. We can see this approach in Tseng and Lee (2021) and Srivastava et al. (2021), to name a few recent works. Among the different approaches present in the literature, we will build the theoretical framework of the present work from the point of view of Probabilistic Graphical Models (PGM). As a result, our method, which is based on a message-passing scheme, fits naturally into the framework. Another advantage is that we align our work with many recent papers in the field, which also take this approach. We advocate for a unified framework in the field to enhance coherence among diverse methodologies. Zhang et al. (2016) defend the idea of an approach to graph signal theory from the PGM point of view and its benefits. The authors also demonstrate the direct mapping between undirected weighted graphs and GMRF. According to Gadde and Ortega (2015), a probabilistic interpretation allows us to view graph signal sampling theory as a model-based method. Other works that solve GSP problems under this framework include methods based on the common graph smoothness assumption formulated in a Gaussian random field model (Ji and Han, 2012), and an approach to semi-supervised learning based on a Random fields (Zhu et al., 2003), among others.

3.3. Graph neural networks

Traditional neural network architectures and methods often struggle when handling complex graph-like data structures. This fact motivated new research to find models that work naturally with graphs as input. The work of Gilmer et al. (2017) unified various graph neural networks and Graph Convolutional Network approaches (Battaglia et al., 2018). This concept is known as Message Passing Neural Networks (MPNNs). This was initially conceived in the field of chemistry and constitutes a framework that systematizes and encompasses a large part of the current approaches. The most widely used graph neural network scheme is built on this same MPNN basis.

MPNNs are a particular family of algorithms within the Message Passing scheme, which follows an Aggregation and Combination process. The idea behind the algorithm is conceptually simple. Each node of the graph has a hidden state which can be a real value or a vector. For each node v, we aggregate the result of applying a function on the hidden states and (in certain cases) the edges of all neighboring nodes

with node v itself. We then update the hidden state of node v using the obtained message and the previous hidden state of that node.

More formally, according to the general approach of the Message Passing Neural Networks (Gilmer et al., 2017), hidden states h_v^t at each node in the graph are updated during the message passing phase based on messages m_v^{t+1} through the following Eqs. (1) and (2):

$$m_v^{t+1} = \frac{1}{n} \sum_{w \in N(v)} M_t(h_v^t, {}^t w, e_{vw}) \tag{1}$$

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1}) \tag{2}$$

where the message functions, M_t, and vertex update functions, U_t, are learned differentiable functions. We can repeat this MP algorithm for a given number of times in an iterative manner.

4. Method

4.1. Problem formulation

We model a graph as a Markov Random Field (MRF). An MRF is a model that belongs to the PGM family, where a set of random variables are described by an undirected graph following the Markov property. The joint probability of X given the evidence Y can be factorized over the cliques C of G as:

$$P(X = x|Y) = \prod_{c \in C} \phi_c(x_c, y_c) \tag{3}$$

where x_c and y_c are the variable and evidence nodes in clique c, respectively. ϕ_c is referred to as the potential of clique c, and it is a function that encodes direct statistical interactions between the nodes forming the clique.

The GSR problem is then translated under this framework into a Maximum a Posteriori (MAP) inference problem, where we seek to compute the assignment of X that maximizes the probability P given Y.

$$MAP_X = \max_X P(X|Y) = \max_X \prod_{c \in C} \phi_c(x_c, y_c) \tag{4}$$

Calculating this quantity is equivalent to maximizing the logarithm of the probability as follows:

$$MAP_X = \max_X log(P(X|Y)) = \max_X \sum_{c \in C} log(\phi_c(x_c, y_c)) \tag{5}$$

Assumption 1 (*Binary Cliques*)**.** Cliques of size 2 dominate in the graph.

$$C_2 \gg C_{3+}$$

Being C_2 the number of size 2 cliques and C_{3+} the number of cliques with size at least 3

Assumption 2 (*Homophily Assumption*)**.** Two nodes that are connected in the graph by an edge will tend to have similar node signals. This can also be understood as smoothness in the graph signal

At this point, we can define the clique potential ϕ_c. As previously noted, the potential function may be of any form, given that it is positive across all points. We will then define the potential function as stated in Eq. (6). Assumption 1 allows us to define the potential function on binary cliques.

$$\phi_c(x_c) = exp(-\frac{1}{2}(f(x_i) - f(x_j)))^2 \tag{6}$$

By selecting this potential, Eq. (4) transforms into a minimization problem, resembling a graph signal smoothing scenario. This function attains its minimum when minimizing the discrepancy between neighboring nodes. Emphasize that the trivial solution of having a constant signal is not possible because the signal in nodes where there are detectors remains fixed.

$$MAP_X = \max_X \sum_{i,j \in C} -\frac{1}{2}(f(x_i) - f(x_j))^2 = \min_X \sum_{i,j \in C} \frac{1}{2}(f(x_i) - f(x_j))^2 \tag{7}$$

4.2. Our method

We will now present the proposed method. This algorithm will consist of two distinct steps, which are executed serially. First, we will introduce a simple and novel method of interpolation between nodes, based on the message-passing scheme. This first step serves as a regularizing mechanism for a second step, where we train a GNN to obtain a more refined result. The general scheme of the method is shown in Fig. 1.

4.2.1. First step: Mean aggregation message passing

As mentioned above, as the first element of our method, we construct a non-parametric Message-passing algorithm that aggregates and combines the values of the hidden states, which are now simplified as the level of the contamination at the node. We will call this algorithm Mean Aggregation Message Passing (MAMP). This method will be based on the approach described in Section 3.3. For simplicity and clarity, we will keep the original notation h_v^t as the air quality value of node v at iteration t, which would correspond to our *variable* node values, X. The functions M_t and U_t that we will use are fundamentally arithmetic averages between the inputs. We choose these functions motivated by the particular MAP problem we are trying to solve, where we want to minimize the differences between the signals from nearby nodes. Based on the MPNN schema (Eqs. (1) and (2)) the resulting equations are therefore:

$$m_v^{t+1} = \frac{1}{|N(v)|} \sum_{w \in N(v)} h_w^t \tag{8}$$

$$h_v^{t+1} = \frac{1}{2}(h_v^t + m_v^{t+1}) \tag{9}$$

where $|N(v)|$ is the size of $N(v)$.

Due to the characteristics of our problem, we can exploit the information in those nodes that have a monitor. Therefore, through each iteration, we will keep constant the known values in those nodes.

Finally, it is necessary for the h_v^t values of the unknown nodes to be initialized to some value. We will study in detail the impact of this initial value in Section 5.6. As can be seen, no training data is needed for this step of the algorithm, since it does not have any trainable parameters.

Under the scheme just mentioned, it can be seen that our method is completely oblivious to the weights of the connections between nodes, all of them having the same weight in the aggregation of the hidden states of neighboring nodes. In our case, we do have edge information in the form of distances between road intersections. This information could be useful for the performance of the algorithm, we will compare the method already presented with a new version that will take these weights into account.

This variant will be as follows. The aggregation step will change its function from an arithmetic mean (8) to a weighted mean (10), where the weighting parameters depend on the weights of the edges. For example, in our case, the weights can be the inverse or square inverse of the distance. Different options will be compared.

$$m_v^{t+1} = \frac{\sum_{w \in N(v)} h_w^t \cdot e_{vw}}{\sum_{w \in N(v)} e_{vw}} \tag{10}$$

4.2.2. Second step: Introducing GNNs

In the second step of our algorithm, the output obtained by the MAMP method is used as input for a GNN. In this case, the GNN will have a series of trainable parameters, whose number will depend on its specific architecture. The objective of this GNN will be to refine the result obtained in the first step, the MAMP algorithm. The hypothesis behind this decision is that the GNN, thanks to its ability to adapt the weights to optimize the result, will be able to achieve higher precision than the MAMP algorithm alone. To prove that, we will train various GNN architectures and compare the best performance with the one achieved by other methods.

Fig. 1 shows the whole pipeline for the algorithm presented in this section.

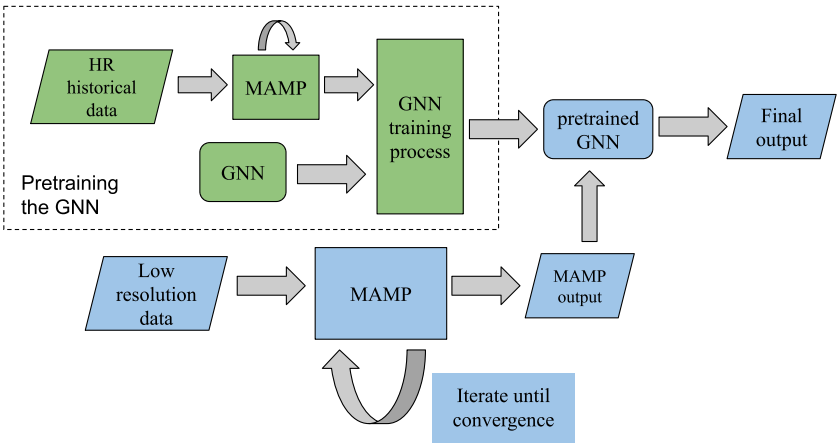


Fig. 1. Pipeline of the method presented in this section. *HR* stands for High resolution, meaning the historical data with street-level resolution that we will use to train the GNN. The area marked with a dotted line shows the GNN training process, as explained in Section 5.4. This training process employs the novel MAMP method proposed in this paper as a data preprocessing step. The other elements of the scheme show the inference process. This process is as follows: The proposed MAMP algorithm is applied to a sample graph containing a given percentage of unknown node values. The output of this step is refined by a GNN previously pre-trained on a different training dataset.

Algorithm 1 Mean aggregation message passing

Input: S_0, A, mask, ϵ, GNN ▷ S_0: input signal, A:Adj. matrix, mask: binary sampling vector, ϵ: convergence condition, GNN: Pretrained Graph Neural Network model (optional)
Output: S ▷ S: reconstructed signal
$S_0[mask] \leftarrow initial\_value$ ▷ Initialize the unknown node values
while $|S_t - S_{t-1}| > \epsilon$ **do**
  for $node$ in S_t **do**
    $N_{node} \leftarrow$ neighbors of $node$
    $m \leftarrow aggregate(N_{node}, A)$ ▷ Eq. 8 or 10
    $h \leftarrow combine(node, m)$ ▷ Eq. 9
    $node \leftarrow h$
  end for
  $S_{t+1}[mask] \leftarrow S_t[mask]$
end while
$S \leftarrow GNN(S)$ ▷ GNN for refinement. Optional step

5. Results

In this section, we will present first the metrics used for the evaluation of the methods. Subsequently, we will discuss in detail the experimental setup and procedure we followed to obtain the results. We then present the results of the experiments. In Section 5.3, we will compare the performance of the different methods proposed and other existing methods. In Section 5.4, we will present the results of the training process of the different GNN models. In Section 5.5 we will empirically study the convergence properties of our method. Finally, in Section 5.6 we will analyze the effect of different node initialization.

5.1. Dataset

The contamination level is not measured everywhere in the city during daily operation, and instead, there are only a few sensors dispersed around the city (Department of Sustainability and Environment of Barcelona, 2023), which creates the necessity to estimate the contamination level accurately across the entire area.

To perform the experiments, we will use data that does have this finer spatial resolution. The dataset has been obtained from the official database of the city council of Barcelona, Spain. This database, Open Data BCN (Ajuntament de Barcelona, 2023), is a project designed to provide open access to data in many different fields concerning the city of Barcelona. The limitation of these data is that they have an annual temporal resolution, and only one map is published per year. However, using different past years will allow us to train and validate our method. In addition, the data provided by the Open map is obtained by modeling based on emission factors, and thus the standard limitations of this kind of approach should be taken into account (Thunis et al., 2021). They are publicly accessible at https://opendata-ajuntament.barcelona.cat/data/ca/dataset/mapes-immissio-qualitat-aire.

In this dataset, we can find the annual average values of different pollutants for the years 2018 and 2019. Each year consists of a different network, both of them with a size of 9466 nodes and 14 684 edges. Specifically, the contaminants present in the dataset are the following: PM2.5, PM10, and NO_2. In our case, we will analyze only the pollutant PM2.5 for simplicity.

It is also necessary to mention that the concentration of pollutants is given by road section in the entire city of Barcelona. To build the graph from this point, we will take the data of road sections, intersections, and positions also present in the Open Data BCN platform, accessible at https://opendata-ajuntament.barcelona.cat/data/ca/dataset/mapa-graf-viari-carrers-wms. With this information, we can now construct the graph and assign each road section its contaminant concentration value. To distribute the graph signal on the nodes instead of on the edges of the graph, we will take each value of each node as an arithmetic mean of the concentration values of the road sections that connect to that node. A sample of the final result can be seen in Fig. 2.

We remark that an important point of our method is Assumption 1, in which we assume binary cliques. The dataset used presents a percentage of 98% of binary clicks. Therefore, we consider the assumption to be fulfilled.

5.2. Metrics

The evaluation of the models will proceed as follows. Each time we run each algorithm for a given number of known nodes, we will calculate a series of metrics that will measure the difference between the ground truth (Y) with the result predicted by the algorithm ($\hat{Y}$) at each node of the graph. These metrics will be in our case the Mean Squared Error (MSE) (11) and the Mean Absolute Percentage Error (MAPE) (12).

Mean Square Error (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_n - \hat{Y}_n)^2 \tag{11}$$

Mean Absolute Percentage Error (MAPE):

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{Y_n - \hat{Y}_n}{Y_n}\right| \tag{12}$$

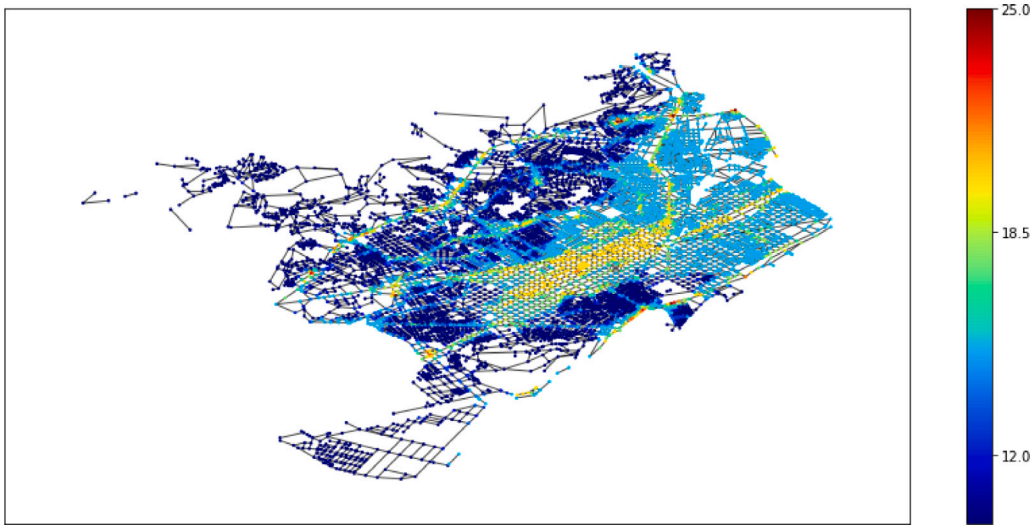


Fig. 2. Sample of one of the graphs used in our dataset. Specifically for the PM2.5 pollutant in the year 2018.

For the validation of models based on Graph Neural Networks, the procedure is the same as mentioned above. However, validating a model that has been trained with certain data, using the same data, is a clear bias. Therefore, a different training graph, belonging to the previous year's data set (2018), will be used for training the neural models. This allows these models to be evaluated under the same conditions and with the same data as the others. Therefore, it should be noted that none of the data used for the validation has been shown to the model during any previous phase.

5.3. Method comparison

In this section, we conduct an empirical comparison between the proposed method and different alternatives available in the literature. In addition to our proposed method, which combines MAMP with GNNs, we also evaluated the results using MAMP alone (no GNN refinement), and MAMPe (MAMP weighted by edge features). These results have also been compared with those obtained directly from a GNN, without using MAMP as a regularizing mechanism. Two architectures have been compared, Graph Convolutional Networks (GCN) (Kipf and Welling, 2016) and SAGE (Hamilton et al., 2017) for different model sizes. In the results of this section, we present results for the SAGE architecture, as it showed better overall results. See Section 5.4 for more details on the comparison between GNN architectures. Alongside our methods, we benchmark against several established methods prevalent in the literature, which are listed below.

The methods included in our comparison span diverse techniques. First, we integrate a Kernel method based on heat kernel computation, addressing the problem of graph signal interpolation through the resolution of the heat equation. Second, we incorporate the Gaussian Belief Propagation (GBP) (Bickson, 2008) method. GBP is a probabilistic inference algorithm that, similarly to a GNN, operates by passing messages between the nodes in a graph. Additionally, we integrate Interpolated Regularization (IR), a Laplacian Regularization method, drawing inspiration from the work detailed in Belkin et al. (2004). Finally, we also compared our method with IGNNK (Wu et al., 2021). This is a method based on GNN, which aims to attack the spatiotemporal prediction problem in cases of graph-structured data such as this one. This method is of particular interest to us, since it also employs GNN models. Recall that our setting differs from the spatiotemporal setting since the temporal dimension is missing.

Fig. 3 shows two graphs with the results obtained for each of these methods compared. While Fig. 3(a) shows the MSE (Eq. (11)), graph Fig. 3(b) shows the MAPE (Eq. (12)). We can see from this plot how our proposed method combining MAMP with GNNs achieves the best performance for both metrics in a large part of the studied range of the percentage of missing nodes.

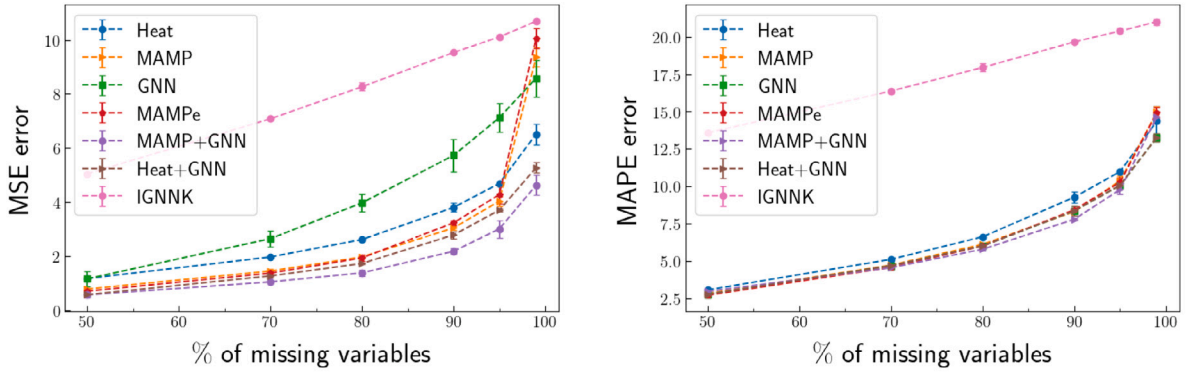
Table 1 presents the mean MSE values and their standard deviations after running the experiments repeatedly. We can see that there are two methods (GBP and IR) that did not produce reliable results for the performed experiments. This is because these two methods have intrinsic difficulties in handling the proposed problem. Firstly, GBP does not reach a convergence point. This is because the convergence of the algorithm is not assured when there are loops in the network, as is the case. Interpolated Regularization requires calculating an inverse of the Laplacian matrix of the subset of unknown nodes. Calculating the inverse of this matrix according to the proposed method involves first calculating its determinant. Thus, when the percentage of unknown nodes is 50 percent (the lowest considered in this work), the absolute number of unknown nodes is 4733. In this case, there is a numerical problem when dealing with such a large matrix, since the value of the determinant diverges towards infinity very quickly. We have verified experimentally that the maximum percentage of unknown nodes supported by this method for this problem is around 1% (about 100 nodes). It is important to mention that the graph we consider here is much larger than other scenarios where these methods have been tested. That is why some methods have encountered difficulties specific to their architecture. Concerning the IGNNK method, the outcome differs from those achieved by our method. We attribute this difference to the fact that, as we already mentioned, this method was originally intended for a spatiotemporal prediction problem. The temporal component is absent in our setting. We therefore believe that our method may be better suited to the spatial task considered here. It is also interesting to highlight the fact that the performance of GNNs becomes significantly worse when is not combined with MAMP. This is the case for both the GCN and SAGE architectures. We hypothesize that this is because a previous simple interpolation helps significantly to stabilize the training process.

In Table 1 we can also see other methods that have been able to find a convergence point. Specifically the HDK method and the three methods proposed in this paper. Of the four methods, the three proposed in this work outperform the kernel method, except for the case where 99 percent of the nodes are unknown. In this stretch, only the combination of the MAMP algorithm and a GNN clearly outperformed HDK. With respect to the standard deviations, these are not very large with respect to the absolute value obtained. These standard deviations move around the third significant figure, between 2 and 6% of the absolute value of the error. Therefore, we can conclude that both the proposed methods

Table 1
Mean squared error for the different methods. All quantities are expressed in $\mu g/m^3$. "–" denotes when the method has intrinsic problems that prevent it from returning a solution.

	% of missing nodes					
	50	70	80	90	95	99
GBP	–	–	–	–	–	–
IR	–	–	–	–	–	–
HDK	1.180 ± 0.042	1.974 ± 0.041	2.621 ± 0.037	3.81 ± 0.16	4.695 ± 0.084	6.52 ± 0.38
IGNNK	5.048 ± 0.062	7.085 ± 0.083	8.26 ± 0.15	9.535 ± 0.058	10.108 ± 0.090	10.69 ± 0.10
MAMP	0.791 ± 0.043	1.455 ± 0.083	1.977 ± 0.060	3.051 ± 0.080	4.04 ± 0.18	9.36 ± 0.35
GNN[a]	1.17 ± 0.29	2.65 ± 0.30	3.98 ± 0.33	5.73 ± 0.59	7.13 ± 0.52	8.58 ± 0.68
MAMP + edges	0.711 ± 0.070	1.377 ± 0.091	1.940 ± 0.083	3.238 ± 0.049	4.284 ± 0.080	10.05 ± 0.056
Heat + GNN	0.581 ± 0.015	1.274 ± 0.060	1.735 ± 0.029	2.79 ± 0.13	3.709 ± 0.052	5.28 ± 0.20
MAMP + GNN	**0.576 ± 0.015**	**1.054 ± 0.084**	**1.388 ± 0.037**	**2.186 ± 0.024**	**3.02 ± 0.14**	**4.627 ± 0.028**

[a] GNN denotes the results obtained by the best architecture among those compared.



(a) Method comparison. The plot shows MSE vs the *% of missing nodes*

(b) Method comparison. The plot shows MAPE vs the *% of missing nodes*

Fig. 3. Method comparison. The plot shows MSE and MAPE vs. the *% of missing nodes*, i.e., with unknown node signal.

and the HDK comparison method are quite stable in terms of accuracy. As can be seen, the method formed by the combination of the MAMP algorithm and a refinement GNN obtains the best results for all the missing node ranges.

5.4. Graph neural network training

To facilitate training, we partitioned the 2018 graph into separate training and validation sets while preserving its overall structure. Specifically, 80% of the nodes were allocated for training purposes, and the remaining 20% were designated for validation, with node selection carried out randomly. We consider the 2019 graph as the test set by itself, and the final scores once we have chosen the best model will be obtained using this test set. This data contain the contamination value at all nodes. For training, given the desired percentage of unknown nodes, we proceed to mask a certain number of nodes, making this information inaccessible to the algorithm. The next step will be to pass this masked graph through our first algorithm, MAMP, which will perform an interpolation of the graph signal with the available data. The result of this first interpolation will be the input to a Graph Neural Network.

We define the loss function as the MSE (11), a widely used loss function in regression problems like this one. The MSE between the predicted signal by the GNN and the actual node signal will be the objective function to minimize by the GNN. The optimizer used is ADAM (Kingma and Ba, 2015), with a learning rate parameter, lr, set to 0.001.

In Fig. 4 we show the results obtained for three different percentages of missing nodes for a given GNN architecture. More specifically, the figure shows the validation loss versus the training epoch. As we can see, the initial behavior of the loss in every case is very noisy. However, the performance of the GNN converges to a given value, which is higher

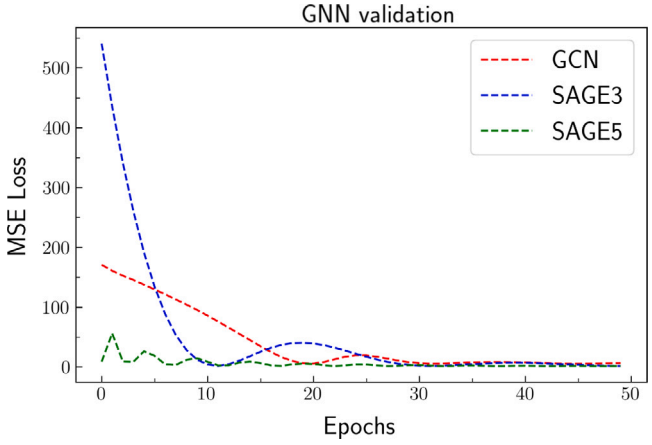


Fig. 4. Validation MSE loss after training for three of the proposed architectures: GCN, SAGE3, and SAGE5, all of them in their Medium version. The training was carried out for a fixed mask size of 0.8.

when the number of missing nodes is also higher. This result is in line with the intuition of the problem.

In Table 2 we show the results obtained for different GNN architectures. Due to the infinite variety of architectures that can be built, we have decided to systematize the experiments as much as possible. For this purpose, we have selected two well-known architectures, Graph Convolutional Networks (GCN) and GraphSAGE (or SAGE). Under the SAGE architecture, we have built two different models, SAGE3 and SAGE5, with 3 and 5 layers respectively. On the other hand, GCN has 7 layers. The letters S, M, and L (Small, Medium, and Large) correspond

Table 2
Comparison of different GNN architectures. Performance for 80 and 99 percent of missing nodes, average training time, and number of trainable parameters are shown. All quantities are expressed in $\mu g/m^3$.

	GNN architectures								
	GCN-S	GCN-M	GCN-L	SAGE_3-S	SAGE_3-M	SAGE_3-L	SAGE_5-S	SAGE_5-M	SAGE_5-L
80% MSE	5.349	5.347	5.411	1.628	1.534	**1.253**	1.982	1.38	1.373
99% MSE	8.823	8.654	8.367	**4.428**	4.636	4.888	7.648	5.132	5.478
80% MAPE	13.047	12.996	12.942	6.252	5.861	5.2645	4.9679	5.8403	**4.619**
99% MAPE	18.938	18.829	18.222	**14.279**	14.962	15.119	15.243	15.554	14.44
Training time (s)	4.736	9.691	26.74	**1.675**	5.392	5.893	2.456	5.3047	12.247
Parameters	1101	25 501	101 001	**261**	5301	20 601	681	15 401	60 801

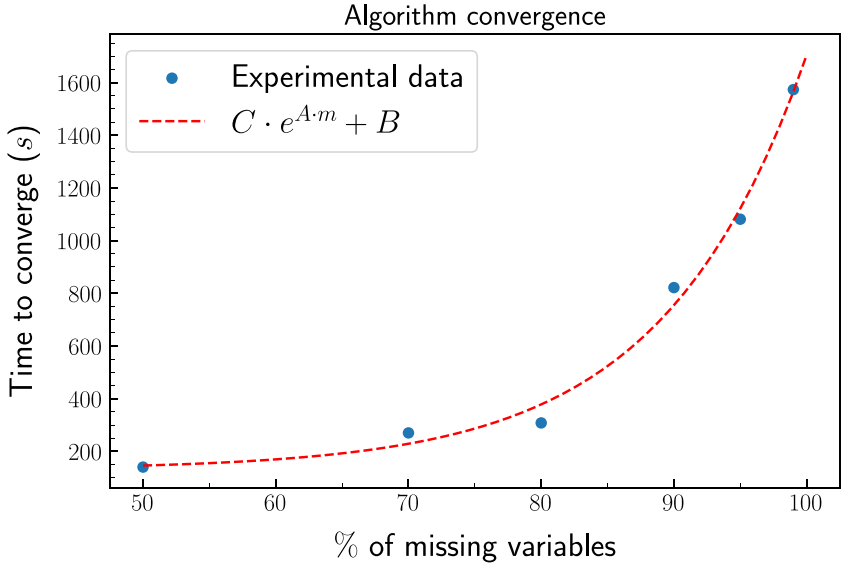


Fig. 5. Time of convergence in seconds given the % of missing nodes. The last point corresponds to 99% of missing nodes.

to the number of parameters in the hidden layers of the models. This number corresponds to 10, 50, and 100 respectively. As can be seen, the SAGE architectures achieve better performances for all the cases studied with ease. The lowest MSE obtained is found in the SAGE3 architecture, SAGE3-S the best for 99 percent of missing nodes, and SAGE3-L when the percentage is 80.

5.5. Convergence

As mentioned above, the method consists of sending and receiving messages iteratively. Yet, we have not theoretically determined the optimal number of iterations required for peak performance. A high dependency of the error on the iteration count could pose challenges when implementing the method in different scenarios. In this section, we experimentally analyze the behavior of the algorithm through its iterative process. These experiments aim to empirically study its convergence properties. In our approach, convergence is defined in accordance with the criteria outlined in Algorithm 1, employing a specific ϵ value set at $\epsilon = 10^{-3}$. This value determines the threshold for convergence (i.e., when the change or improvement in the model's performance falls below this threshold, the algorithm is considered to have converged). It is important to note that the selection of epsilon at 10^{-3} is not universally fixed, as it is subject to adjustment based on the nuances and specific requirements of each scenario.

Fig. 5 depicts a curve showing the convergence time in relation to the percentage of missing nodes. This relation appears to increase at a rate faster than linear. We performed a fitting of the graph to an exponential curve of the form $C \cdot e^{A \cdot m} + B$, with parameters $A = 0.09252$, $B = 131.05$ and $C = 0.1508$. This fit results in a R^2 value of 0.992.

5.6. Node initialization

As mentioned in Section 4.2, our method requires an initialization of the Graph Signal values at the unknown nodes. We wonder, is then our method dependent on this initialization value in terms of its performance? At the same time, how does it affect its convergence?

We have tried to answer the following questions by comparing different initialization values for the same graph as input. The result obtained can be seen in Fig. 6. As can be seen, the initialization value does not affect the convergence point. However, it does affect the velocity at which this point is reached. From the algorithm performance, it is clear that an initialization close to the actual value causes a faster convergence since it needs fewer messages to reach a stable value. The value of 12.8628 corresponds to the mean value of the Graph Signal. The other values are chosen arbitrarily to perform the experiment.

5.7. Experiment setup

To study the performance of the presented method, we performed a series of experiments. For this purpose, the methods have been implemented and tested using the Dataset previously shown. The language used was Python. We have also made use of the following libraries. Deep graph library (DGL) (Wang et al., 2019) was used for the construction of the dataset, as well as the implementation of the Graph Neural Networks. The `NetworkX` library (Hagberg et al., 2008) was used to visualize the graphs. Finally, the Heat kernel diffusion method was implemented with the help of the `pygsp` library (Defferrard et al.). The implementations of the IR method have been carried out thanks to the code of the work (Ferrer-Cid et al., 2022) provided by the authors, to whom we are grateful for their contribution.

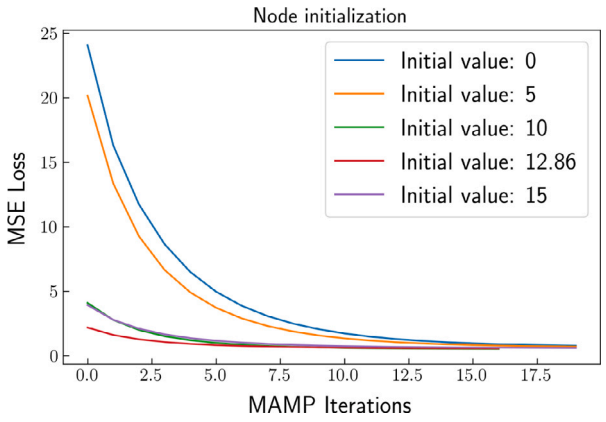


Fig. 6. Convergence comparison for different initialization values of the nodes, i.e., the value assigned to the unknown nodes at the beginning of the algorithm.

The experiments will be carried out as follows: in Section 5.3, the methods to be compared are run independently for different percentages of unknown nodes. We call this parameter *% of missing nodes*, and it indicates the percentage of nodes whose values we do not know out of the total number of nodes available in the full signal. It should be noted that, once the percentage of missing nodes is defined, the available nodes within the network are randomly selected with a uniform distribution until this percentage is reached. This may create a slight variability in the results. For this reason, each experiment is performed more than once to compare the standard deviation. This standard deviation will be present both in the graphs in the form of error bars and in the tables. In Section 5.5, we fix a given *% of missing nodes* and run the MAPE algorithm once. The aim is to analyze the behavior of the metrics throughout the iterations when running our method, to demonstrate experimentally that a point of convergence exists. Lastly, in Section 5.6, our goal is to study the effect of node value initialization on the performance of the method. To do this, we run the algorithm for different initial values, and we display the error as a function of the iterations for those different values. We will thus try to prove our hypothesis that the initial value of the nodes accelerates convergence when it is closer to the average value of the graph signal.

All the experiments, and therefore the computational time measurements, have been carried out on a machine running a processor Intel® Core™ i7-10750H CPU @ 2.60 GHz × 12 and an NVIDIA GeForce RTX 2060 GPU with 6 GB of memory.

6. Conclusions and future work

In this paper, we formalize the problem of air quality prediction from monitors using a Markov Random Field. Through this formalization, the problem is reduced minimizing a proposed energy function given the observations. We propose a method for this, based on message passing, which achieves a graph signal smoothing. Subsequently, we use graph neural networks trained in a supervised way to refine the result.

In conclusion, the proposed combination of a new message-passing algorithm for graph signal smoothing and the use of graph neural networks achieves the best performance among the other compared methods. This is a promising approach that has the potential to significantly improve the accuracy and efficiency of air quality spatial prediction models. We showed that the regularization step of graph signal smoothing effectively improved the robustness of the GNN model, leading to more accurate predictions. Additionally, the use of graph neural networks allowed for the incorporation of complex spatial dependencies in the data which the signal smoothing itself cannot capture.

While our study shows promising results, it is important to note the limitations of this approach. First, Assumption 2 may not be accurate enough in scenarios of multiple and extreme pollution hotspots. These areas often have significant gradients, and their absence in our data challenges accurate predictions. Future work should address these extremes, a considerable challenge for our approach. Second, exploring different graph neural network architectures and sizes could enhance our understanding. Additionally, testing our method on a wider range of datasets, including larger urban ones or new rural ones, is crucial to gauge its generalizability.

Finally, further research on the theoretical properties of the approach is needed, to obtain a better understanding of its generalization and limitations.

CRediT authorship contribution statement

Sergio Calo: Conceptualization, Data curation, Methodology, Software, Writing – original draft, Writing – review & editing. **Filippo Bistaffa:** Conceptualization, Writing – original draft, Writing – review & editing, Methodology, Software. **Anders Jonsson:** Conceptualization, Methodology, Writing – review & editing. **Vicenç Gómez:** Methodology, Writing – review & editing. **Mar Viana:** Writing – review & editing.

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.

Data availability

All the code is available in the link shared in the manuscript. All the data used in this work is free to access at the Open Data BCN portal.

Acknowledgments

The work presented in this article is the result of a collaboration carried out as part of the CSIC Interdisciplinary Thematic Platform (PTI) Mobility 2030 (PTI-MOBILITY 2030). The authors gratefully acknowledge the Open Data BCN portal provided by the Barcelona city council. Bistaffa was supported by the research projects ACISUD (PID2022-136787NB-I00) and Yoma OR (OPE02570).

Appendix A. Supplementary material

All the code used in the experiments, as well as the dataset, are available in a GitHub repository, accessible via this link: https://github.com/SergioCalo/MAMP-bcn-airquality.

References

Ajuntament de Barcelona, 2023. Open Data BCN | Servicio de datos abierto. URL https://opendata-ajuntament.barcelona.cat/es.

Battaglia, P.W., Hamrick, J.B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V.F., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gülçehre, Ç., Song, H.F., Ballard, A.J., Gilmer, J., Dahl, G.E., Vaswani, A., Allen, K.R., Nash, C., Langston, V., Dyer, C., Heess, N.M.O., Wierstra, D., Kohli, P., Botvinick, M.M., Vinyals, O., Li, Y., Pascanu, R., 2018. Relational inductive biases, deep learning, and graph networks. ArXiv abs/1806.01261.

Bekkar, A., Hssina, B., Douzi, S., Douzi, K., 2021. Air-pollution prediction in smart city, deep learning approach. J. Big Data 8 (1), 1–21.

Belkin, M., Matveeva, I., Niyogi, P., 2004. Regularization and semi-supervised learning on large graphs. In: COLT.

Bickson, D., 2008. Gaussian belief propagation: Theory and aplication. arXiv preprint arXiv:0811.2518.

de Bont, J., Casas, M., Barrera-Gómez, J., Cirach, M., Rivas, I., Valvi, D., Álvarez, M., Dadvand, P., Sunyer, J., Vrijheid, M., 2019. Ambient air pollution and overweight and obesity in school-aged children in Barcelona, Spain. Environ. Int. 125, 58–64.

Defferrard, M., Martin, L., Pena, R., Perraudin, N., PyGSP: Graph Signal Processing in Python, http://dx.doi.org/10.5281/zenodo.1003157, URL https://github.com/epfl-lts2/pygsp/.

Department of Sustainability and Environment of Barcelona, 2023. Xarxa de vigilància I previsió de la contaminació atmosfèrica (XVPCA). URL https://mediambient.gencat.cat/ca/05_ambits_dactuacio/atmosfera/qualitat_de_laire/avaluacio/xarxa_de_vigilancia_i_previsio_de_la_contaminacio_atmosferica_xvpca.

Ferrer-Cid, P., Barceló-Ordinas, J.M., García-Vidal, J., 2022. Graph signal reconstruction techniques for IoT air pollution monitoring platforms. ArXiv abs/2201.00378.

Gadde, A., Ortega, A., 2015. A probabilistic interpretation of sampling theory of graph signals. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, IEEE, pp. 3257–3261.

Garcia, V., Bruna, J., 2017. Few-shot learning with graph neural networks. arXiv preprint arXiv:1711.04043.

Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E., 2017. Neural message passing for quantum chemistry. In: International Conference on Machine Learning. PMLR, pp. 1263–1272.

Hagberg, A., Swart, P., S. Chult, D., 2008. Exploring Network Structure, Dynamics, and Function Using Networkx. Technical Report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

Hamilton, W., Ying, Z., Leskovec, J., 2017. Inductive representation learning on large graphs. Adv. Neural Inf. Process. Syst. 30.

Han, J., Liu, H., Xiong, H., Yang, J., 2022. Semi-supervised air quality forecasting via self-supervised hierarchical graph neural network. IEEE Trans. Knowl. Data Eng. 35 (5), 5230–5243.

Ji, M., Han, J., 2012. A variance minimization criterion to active learning on graphs. In: AISTATS.

Khalil, E., Dai, H., Zhang, Y., Dilkina, B., Song, L., 2017. Learning combinatorial optimization algorithms over graphs. Adv. Neural Inf. Process. Syst. 30.

Khomenko, S., Cirach, M., Pereira-Barboza, E., Mueller, N., Barrera-Gómez, J., Rojas-Rueda, D., de Hoogh, K., Hoek, G., Nieuwenhuijsen, M., 2021. Premature mortality due to air pollution in European cities: a health impact assessment. Lancet Planet. Health 5 (3), e121–e134.

Kingma, D.P., Ba, J., 2015. Adam: A method for stochastic optimization. CoRR abs/1412.6980.

Kipf, T.N., Welling, M., 2016. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

Lelieveld, J., Evans, J.S., Fnais, M., Giannadaki, D., Pozzer, A., 2015. The contribution of outdoor air pollution sources to premature mortality on a global scale. Nature 525 (7569), 367–371.

Li, X., Peng, L., Hu, Y., Shao, J., Chi, T., 2016. Deep learning architecture for air quality predictions. Environ. Sci. Pollut. Res. 23, 22408–22417.

Liu, J., Wang, X., Xie, F., Wu, S., Li, D., 2023. Condition monitoring of wind turbines with the implementation of spatio-temporal graph neural network. Eng. Appl. Artif. Intell. 121, 106000.

Lu, Y.-J., te Li, C., 2020. AGSTN: Learning attention-adjusted graph spatio-temporal networks for short-term urban sensor value forecasting. In: 2020 IEEE International Conference on Data Mining (ICDM). pp. 1148–1153.

Mao, W., Jiao, L., Wang, W., Wang, J., Tong, X., Zhao, S., 2021. A hybrid integrated deep learning model for predicting various air pollutants. GISci. Remote Sens. 58 (8), 1395–1412.

Mota-Bertran, A., Saez, M., Coenders, G., 2022. Compositional and Bayesian inference analysis of the concentrations of air pollutants in Catalonia, Spain. Environ. Res. 204, 112388.

Mueller, N., Rojas-Rueda, D., Basagaña, X., Cirach, M., Cole-Hunter, T., Dadvand, P., Donaire-Gonzalez, D., Foraster, M., Gascon, M., Martinez, D., et al., 2017. Health impacts related to urban and transport planning: a burden of disease assessment. Environ. Int. 107, 243–257.

Ouyang, X., Yang, Y., Zhang, Y., Zhou, W., 2021. Spatial-temporal dynamic graph convolution neural network for air quality prediction. In: 2021 International Joint Conference on Neural Networks (IJCNN). pp. 1–8.

Peng, D., Zhang, Y., 2023. MA-GCN: A memory augmented graph convolutional network for traffic prediction. Eng. Appl. Artif. Intell. 121, 106046.

Pierangeli, I., Nieuwenhuijsen, M.J., Cirach, M., Rojas-Rueda, D., 2020. Health equity and burden of childhood asthma - related to air pollution in Barcelona. Environ. Res. 109067.

Reche, C., Tobias, A., Viana, M., 2022. Vehicular traffic in urban areas: Health burden and influence of sustainable urban planning and mobility. Atmosphere 13 (4), 598.

Reddy, V.N., Mohanty, S., 2017. Deep air : Forecasting air pollution in Beijing , China.

Rico, J., Barateiro, J., Oliveira, A.L., 2021. Graph neural networks for traffic forecasting. ArXiv abs/2104.13096.

Rodríguez-Rey, D., Guevara, M., Garcia, J.C., 2020. An integrated model system tool to evaluate the impact of urban mobility policies on air pollution: Barcelona case study.

Saez, M., Barceló, M.A., 2022. Spatial prediction of air pollution levels using a hierarchical Bayesian spatiotemporal model in Catalonia, Spain. Environ. Model. Softw. 151, 105369.

Sanchez-Gonzalez, A., Heess, N., Springenberg, J.T., Merel, J., Riedmiller, M., Hadsell, R., Battaglia, P., 2018. Graph networks as learnable physics engines for inference and control. In: International Conference on Machine Learning. PMLR, pp. 4470–4479.

Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G., 2008. The graph neural network model. IEEE Trans. Neural Netw. 20 (1), 61–80.

Srivastava, D., Bagler, G., Kumar, V., 2021. Graph signal processing on protein residue networks helps in studying its biophysical properties. bioRxiv.

Teng, Y., Huang, X., Ye, S., Li, Y., 2018. Prediction of particulate matter concentration in Chengdu based on improved differential evolution algorithm and BP neural network model. In: 2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA). pp. 100–106.

Thunis, P., Crippa, M., Cuvelier, C., Guizzardi, D., de Meij, A., Oreggioni, G., Pisoni, E., 2021. Sensitivity of air quality modelling to different emission inventories: A case study over europe. Atmosp. Environ.: X 10, 100111.

Tseng, C.-C., Lee, S.-L., 2021. Frequency selective filtering of graph signal in directed graph Fourier transform domain. In: 2021 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW). IEEE, pp. 1–2.

Viana, M., de Leeuw, F., Bartonova, A., Castell, N., Ozturk, E., Ortiz, A.G., 2020. Air quality mitigation in European cities: Status and challenges ahead. Environ. Int. 143, 105907.

Wang, S., Li, Y., Zhang, J., Meng, Q., Meng, L., Gao, F., 2020. PM2. 5-GNN: A domain knowledge enhanced graph neural network for PM2. 5 forecasting. In: Proceedings of the 28th International Conference on Advances in Geographic Information Systems. pp. 163–166.

Wang, Y., Peng, H., Wang, G., Tang, X., Wang, X., Liu, C., 2023. Monitoring industrial control systems via spatio-temporal graph neural networks. Eng. Appl. Artif. Intell. 122, 106144.

Wang, M., Zheng, D., Ye, Z., Gan, Q., Li, M., Song, X., Zhou, J., Ma, C., Yu, L., Gai, Y., et al., 2019. Deep graph library: A graph-centric, highly-performant package for graph neural networks. arXiv preprint arXiv:1909.01315.

Wu, Y., Zhuang, D., Labbe, A., Sun, L., 2021. Inductive graph neural networks for spatiotemporal kriging. In: Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 35, pp. 4478–4485.

Xayasouk, T., Lee, H., Lee, G., 2020. Air pollution prediction using long short-term memory (LSTM) and deep autoencoder (DAE) models. Sustainability 12, 2570.

Zeinalnezhad, M., Chofreh, A.G., Goni, F.A., Klemeš, J.J., 2020. Air pollution prediction using semi-experimental regression model and adaptive neuro-fuzzy inference system. J. Clean. Prod. 261, 121218.

Zhang, C., Florêncio, D.A.F., Chou, P.A., 2016. Graph signal processing – A probabilistic framework.

Zhang, Z., Zhang, S., Zhao, X., Chen, L., Yao, J., 2022. Temporal difference-based graph transformer networks for air quality PM2. 5 prediction: a case study in China. Front. Environ. Sci. 10, 924986.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M., 2020. Graph neural networks: A review of methods and applications. AI Open 1, 57–81.

Zhou, Y., Zheng, H., Huang, X., Hao, S., Li, D., Zhao, J., 2022. Graph neural networks: Taxonomy, advances, and trends. ACM Trans. Intell. Syst. Technol. 13 (1), 1–54.

Zhu, X., Ghahramani, Z., Lafferty, J.D., 2003. Semi-supervised learning using Gaussian fields and harmonic functions. In: ICML.