Estimating the state of epidemics spreading with graph neural networks

Tomy, Abhishek; Razzanelli, Matteo; Di Lauro, Francesco; Rus, Daniela; Della Santina, Cosimo

DOI
10.1007/s11071-021-07160-1

Publication date
2022

Document Version
Final published version

Published in
Nonlinear Dynamics

Citation (APA)
Tomy, A., Razzanelli, M., Di Lauro, F., Rus, D., & Della Santina, C. (2022). Estimating the state of epidemics spreading with graph neural networks. *Nonlinear Dynamics*, *109*(1), 249-263. https://doi.org/10.1007/s11071-021-07160-1

Nonlinear Dyn (2022) 109:249–263
https://doi.org/10.1007/s11071-021-07160-1

ORIGINAL PAPER

Estimating the state of epidemics spreading with graph neural networks

Abhishek Tomy · Matteo Razzanelli · Francesco Di Lauro · Daniela Rus · Cosimo Della Santina

Received: 16 February 2021 / Accepted: 31 October 2021 / Published online: 21 January 2022

Abstract When an epidemic spreads into a population, it is often impractical or impossible to continuously monitor all subjects involved. As an alternative, we propose using algorithmic solutions that can infer the state of the whole population from a limited number of measures. We analyze the capability of deep neural networks to solve this challenging task. We base our proposed architecture on Graph Convolutional Neural Networks. As such, it can reason on the effect of the underlying social network structure, which is recognized as the main component in spreading an epidemic. The proposed architecture can reconstruct the entire state with accuracy above 70%, as proven by two scenarios modeled on the CoVid-19 pandemic. The first is a generic homogeneous population, and the second is a toy model of the Boston metropolitan area. Note that no retraining of the architecture is necessary when changing the model.

Keywords Nonlinear inference · Network dynamics · State estimation · Epidemics · CoVid-19

1 Introduction

Models whose state assumes value on a graph rather than on a standard Euclidean space describe naturally many physical or artificial systems of considerable interest. Within this class of systems, the problem of estimating the entire state from partial measurements is a very relevant one. Standard techniques solve the challenge when the network follows linear and continuous dynamics. Yet, including non-ideal effects into the picture make the development of algorithmic solutions substantially more complicated. For example, [4] introduces constraints in communications bandwidth. State estimation for networks with distributed delays is discussed in [31]. A similar problem is dealt in [49] for the state estimation of a delayed neural network with known output, and in [51] for parameter uncertainty

Abhishek Tomy and Matteo Razzanelli contributed equally to this work.

A. Tomy
Centre of Innovation in Telecommunications and Integration of services, Inria Grenoble - Rhône-Alpes, Inovallée, France

M. Razzanelli
Proxima Robotics srl, Pisa, Italy

F. Di Lauro
Big Data Institute, University of Oxford, Oxford, UK

D. Rus
Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology (MIT), Cambridge, MA, United States

C. Della Santina (✉)
Cognitive Robotics Department, Faculty of Mechanical, Maritime and Materials Engineering, TU Delft, Delft, Netherlands

Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Oberpfaffenhofen, Germany
e-mail: c.dellasantina@tudelft.nl

and randomly occurring distributed delays. The case of switched networks with communication constraints is discussed in [54]. In this context, much attention has also been devoted to distributed estimation algorithms [13,44]. For example, [30] proposes a consensus-based Kalman filter for sensor networks subjected to random link failures, [12] introduces a distributed filter robust to malicious attacks, and [3] proposes a distributed extended kalman filter for sensor networks measuring a single nonlinear dynamics.

The spreading of an epidemic within a fixed population is among the network dynamics with especially relevant applications and dynamic behavior [25]. Here, the graph models the social network. Each node describes either a subject or a group of subjects, and the arcs the contacts. Simple local update rules can then already describe the spreading of the disease accurately. Recently, these models have been extensively used to describe the spreading of Covid-19. In [29] nodes represents European nations. The use of multi-level networks is discussed in [35]. A survey on the interplay of diseases, behaviors, and information spreading in epidemics is provided in [48]. Network models have been later extended to simplicial complexes in [20].

Estimating the state of epidemics from a reduced number of measurements has clear, practical implications. For example, estimating the number of infected subjects and who those infected subjects are can allow the implementation of effective isolation policies [2,5], feedback strategies [11,26], and possibly prevent the generation of clusters [43]. Nonetheless, we are not aware about previous works in epidemiology dealing with this challenge on the subject (i.e. node) level. Indeed, the works discussed above are not applicable since typical epidemic dynamics do not fulfill their assumptions. Several works [6,38] deal instead with the more common problem of extracting robust statistics on the total number of subjects in specific states - e.g., infected, recovered, hospitalized. This estimation can enable the forecast of the evolution of the epidemics [45,46]. Despite requiring to reason on the network dynamics, the task is still such that it can be attacked with model-based techniques since it is essentially a forward integration.

Instead, estimating the complete state of the epidemics is an essentially more difficult problem since it requires reasoning backward on the effects that the nodes of which we know the state could have had on the unknown states. This task is made even harder by the highly nonlinear, state-discrete, and stochastic dynamics which characterize these systems (see Sect. 2). Thus, making inferences at the level of the subjects is beyond the capabilities of current model-based techniques. To the authors' knowledge, there are no algorithms in the literature addressing this challenge.

Intending to offer an alternative route to solving this challenge, we propose using data-driven techniques in this work. We investigate the use of deep learning for creating a nonlinear inference system that can solve the discussed problem. Indeed, deep learning has proven to be able to successfully model complex dynamical behaviors [8,23,27,32]. Recently, many works have dealt with the generalization of deep learning to non Euclidean domains [7]. Particular interest have been given to deep learning on graphs [1,40,55], i.e. to the learning from data of the graph type. Many of these techniques have been categorized under the umbrella term Graph Neural Networks (GNNs). We are interested here in the use of GNNs as classifiers of nodes. The goal is to determine the labeling of nodes by integrating available information on them and on their neighborhood [24]. This is for example used as a recommendation engine - see Pinterest [53], and Uber Eats [21]. This task naturally generalizes to the case of state reconstruction, by considering as desired output the full state of the system. We apply this strategy to epidemics, by combining multiple GNN layers with a mechanism for codifying temporal information. The goal of this work is summarized in Fig. 1, where the state of two nodes out of sixteen is measured. Note that this is more testing than we are going to assume in our simulations. We test the results by using state of the art models of epidemics, with particular focus on CoVid-19 spreading in Italy and United States. Our results show that GNNs can be a viable solution to state reconstruction problem, even when the number of monitored subjects is as low as the 5% of the population.

Note that several works already applied GNNs to epidemics, specifically in the CoVid-19 context. Yet the focus has been different w.r.t. the present work. In [15,22] graph neural networks are used to forecast the pandemic evolution. An inverse problem is instead tackled in [10], where authors deal with the temporal reconstruction of the epidemics spreading. Similarly, in [42] these techniques are used to identify the patient

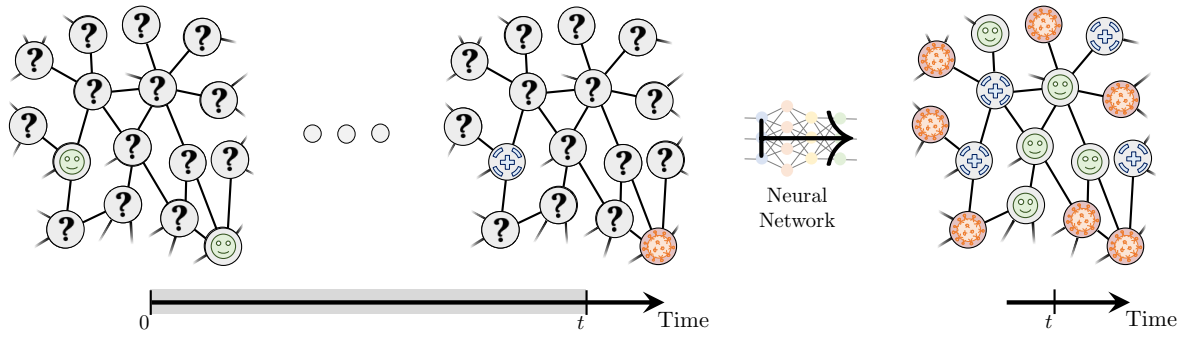


Fig. 1 This is a representation of the problem that we aim at solving in this paper. Consider an epidemic disease spreading on a social network. In practice, health operators cannot measure the condition of the whole population, but only a small number of subjects. On the other hand, subjects can be tested continuously over time. The left side of the picture describes this information. Each node is a subject. The question marks are subjects that we cannot measure. The state of only two nodes/subjects is measured here. At the initial time, the two subjects are healthy (green smile), while at time t, they are infected (red dots) or recovered (blue cross). Our goal is to reconstruct the entire configuration of the network by integrating this information. Therefore, the desired output is the knowledge of the health state of all (possible thousands) subjects. The graph on the right represents this. We propose the first algorithm dealing with this challenge. We use a neural network to achieve this goal

zero. GNNs have been used also for detecting COVID-19 from CT scans and X-rays of chest [39].

To conclude, with this work we provide the first algorithm capable of estimate the whole state of the spreading of a disease in a large social network, and we do that using deep learning. The paper is organized as follows: Sect. 2 describes the epidemic network that represents the disease transmission; Sect. 3 explains the graph fundamentals and the neural architecture for classification the GNN is based on; simulations are then described in Sect. 4; finally conclusions and discussions on results and possible future works are mentioned in Sect. 5.

2 Epidemics on networks

Deterministic models for epidemic spreading on a population are well-known because of their simplicity and their usefulness in terms of giving good prediction in terms of aggregate statistics (such as the total number of infected nodes) of the population. Yet, these models do not allow to describe the actual spreading of the epidemics on a population, thus preventing the implementation of targeted measures. For our objective, we need a model that can capture the fact that each individual is part of a social structure, and that the intrinsic hazard of getting infected depends not only on how many people they interact with, but also on how far they are from clusters of infections. A natural candidate is the framework of Network Epidemiology [25,36]. This framework allows to separate the topological properties of a contact network from the biological dynamics of the disease progression.

2.1 Network model

A network is described as a set $(V, \mathcal{E})$, where V is a set of N nodes (or vertices), and $\mathcal{E}$ is a set of edges (or links) connecting nodes, i.e. tuples $\{u, v\}$, where $u, v \in V$. In terms of modeling, individuals are associated with nodes, and contacts that are at risk of carrying the disease as links between nodes. For simplicity, we consider undirected networks, such that $\{u, v\} \in \mathcal{E} \iff \{v, u\} \in \mathcal{E}$. Fig. 1 and the right side Fig. 2 show pictorial representations of network models.

2.2 Epidemic model on network

We consider a model for disease transmission inspired by recent modeling of Covid-19 Yang et al. [52], He et al. [18]. Each individual is in one of the following states: S (susceptible), E (exposed), I (infected/infectious), R (recovered), or D (deceased). For this reason, this model is known as SEIRD. Fig. 2 illustrates the possible transitions of a susceptible node that is in contact with two infectious neighbors. Outbreaks are modeled as Markovian processes on the

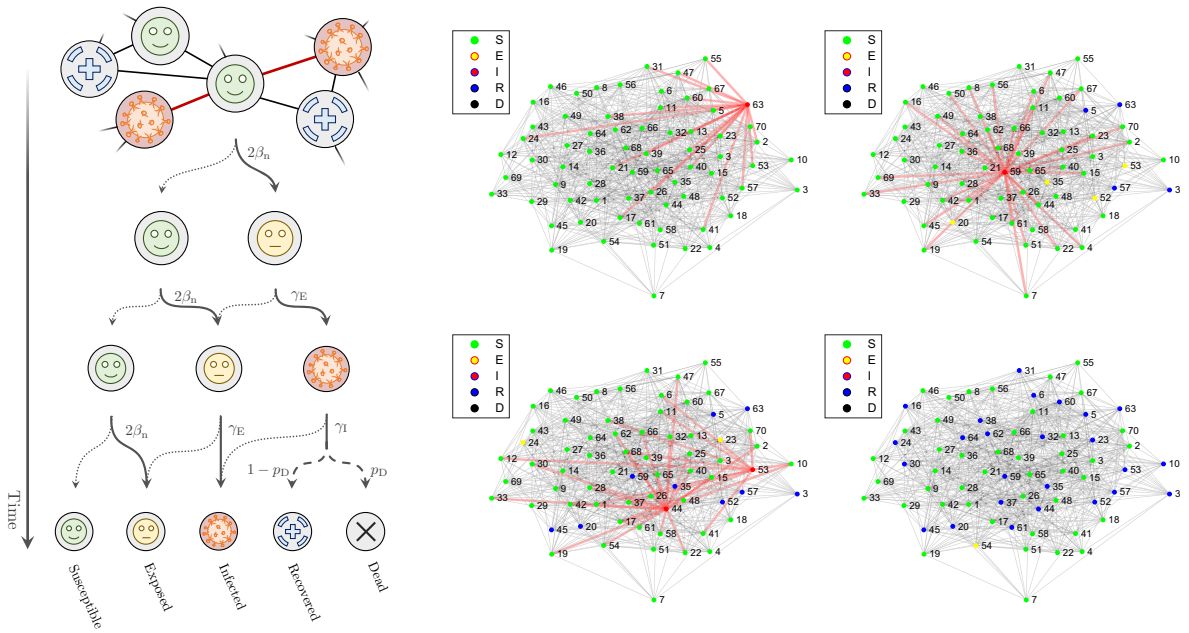


Fig. 2 Representations of the considered dynamic model of spreading an epidemic disease over a social network. The left panel represents the transitions a single node might undergo in time. We declare the meaning of the five symbols at the bottom of the picture. These are the five states a subject can find themselves in, and they go from not having contracted the virus to being recovered or died. Each arrow shows the probability of transition at any time. We assume for simplicity that the state of the neighbors does not change. Since the subject is in contact with two infected subjects, its probability of contracting the virus is twice β_n. The right side of the picture shows a few snapshots of the state of a small Erdős-Rényi network of size 70 with average connection degree 20. Links in red carry the infection from an infectious node to a susceptible neighbor. Note that this social network is orders of magnitude smaller than the ones considered in our validation

generated network, in which an infected node spreads the disease, via links, to its susceptible neighbors at a constant rate β, turning them into exposed. Exposed nodes represent people who are undergoing their latent people, and are about to become infectious. The next transitions that exposed nodes undergo are network-independent. An E node becomes I after a time exponentially distributed with rate γ_E. Once a node is infectious, he transmits the disease to its neighbors at a constant rate β. The node eventually stops being infectious after an exponentially distributed random time with rate γ_I. When this happens, with probability p_D the node becomes D - representing individuals that do not survive to the disease. The remaining nodes are instead recovered and play no further role in the epidemic. At time $t = 0$, $I(0) = N\imath(0) \ll N$ randomly chosen nodes are infected. The remaining ones are initialized as susceptible.

In more formal terms [25], the dynamics is a continuous-time Markov chain defined on a network with N nodes, with adjacency matrix g_{ij} defined as

$$g_{ij} = \begin{cases} 1, & \text{if a link exists between node} i \text{ and node } j; \\ 0, & \text{otherwise.} \end{cases}$$

In this paper, we do not allow for self-loops, i.e. $g_{ii} = 0, \quad \forall i \in \{1 \dots N\}$. The state space of the Markov chain is the set $\{\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_n\}$, where $\mathcal{S}_i$ is a particular derangement of the 5 states (S, E, I, R, D) among N nodes. At any given time, the only transitions allowed are among states that differ at most by the status of one node j. The probability of transition depend on the status of the node and of its neighbors. This transition mechanism is pictorially represented by the tree in the left side of Fig. 2. We denote with $n^j_{S/E/I/R/D}$ the number of neighbors of node j in states S, E, I, R, D, respectively. The rate of transition between states $\mathcal{S}_\alpha$ and $\mathcal{S}_\beta$ is given by

$$h(\mathcal{S}_\alpha, \mathcal{S}_\beta) = \begin{cases} 0, & \text{if } \mathcal{S}_\alpha \text{ and } \mathcal{S}_\beta \text{ differ for at least two nodes;} \\ f^j_{\mathrm{QT}}(Q, n^j_{S/E/I/R/D}), & \text{if } \mathcal{S}_\beta \text{ differs from } \mathcal{S}_\alpha \text{ in the state of node } j; \\ -\sum_{l\neq\alpha} h(\mathcal{S}_\alpha, \mathcal{S}_l) & \text{if } S_\alpha = S_\beta. \end{cases} \tag{1}$$

Here f^j_{QT} denotes the rate at which node j in state Q transitions into state $T \neq Q$. For our model, we have that the only $f^j_{\mathrm{QT}} \neq 0$ are

$$f^j_{S,E}(S, n^j_{S/E/I/R/D}) = \beta n^j_I,\ f^j_{E,I}, (E, n^j_{S/E/I/R/D}) = \gamma_E,$$
$$f^j_{I,R}(I, n^j_{S/E/I/R/D}) = \gamma_I p_R,\ f^j_{I,D}, (I, n^j_{S/E/I/R/D}) = \gamma_I p_D.$$

Let $X_\alpha(t)$ be the probability that the system is in state $\mathcal{S}_\alpha$ at time t, then the master equation of the system is

$$\dot{X}_\alpha(t) = \sum_{\beta=1}^{n} h(X_\beta, X_\alpha) X_\beta(t). \tag{2}$$

This system can be written as a system of the form

$$\dot{X}(t) = PX(t),$$

where P is the transpose of the transition rates, that is, $P_{\alpha\beta} = h(X_\beta, X_\alpha)$, and $X(t)$ is a vector of size n whose elements are $X_\alpha(t)$. We rely on a Gillespie algorithm [17] adapted to networks [25] to simulate this process. In Fig 2 we show a realization of an outbreak on a network of modest size, to highlight how the topology impacts the dynamics.

We describe the evolution of the state of the pandemic on the network as

$$x : \mathbb{R}^+ \to \{S, E, I, R, D\}^N. \tag{3}$$

Therefore at each time $t > 0$ the variable $x(t)$ provides a full picture of thee spreading of the disease. Without loss of generality, we consider t to be expressed in days. We refer to the state of the node $i \in V$ as $x_i \in \{S, E, I, R, D\}$.

3 State inference from incomplete data

3.1 Goal

Consider the graph $(V, \mathcal{E})$ describing the social network. We hypothesize to have full knowledge of the state of a subset of nodes $\mathcal{M} \subset V$ at the end of each day. We will populate $\mathcal{M}$ by selecting nodes from V according to an uniform random distribution. We therefore define the set of measurements as $y \in \{S, E, I, R, D\}^{\#\mathcal{M}}$. Finally, for the prediction purposes, classes are combined based on their usefulness in intervention into 3 classes. Our goal is thus to find an algorithm which implements the following mapping

$$\{V, \mathcal{M}, \mathcal{E}, y([0, t))\} \mapsto \tilde{x}(t) \tag{4}$$

where $\tilde{x} \in \{S, E + I, R + D\}^N$ is our reconstructed state, with $E+I$ representing nodes that either exposed or infected, and with $R+D$ We want (4) to be such that $\tilde{x}$ is as coherent as possible with the full state x. Indeed, in the practice we are only interested in knowing if the subject is healthy (S), has contracted the virus (E and I), or is no more infected (R and D). This challenge is summarized in Fig. 1.

Note that in this work we assume full knowledge of the social structure $(V, \mathcal{E})$ as input for the network. This is a strong assumption that we will relax in future work. Also, we will discuss the robustness of the algorithm to changes of topology.

3.2 First stages

We start by transforming $y([0, t))$ in a data structure that can be effectively put as input of our neural network. More specifically, we introduce the nodes label $l \in \mathbb{N}^{N\times 3}$. For all $i \in \mathcal{M}$, the vector l_i codifies the state of the nodes in the past k days. We do that in a bag-of-words fashion [50]. We sample y on a daily basis $y_i(\lfloor t \rfloor), y_i(\lfloor t \rfloor - 1) \ldots, y_i(\lfloor t \rfloor - k + 1)$. The value $k \in \mathbb{N}$ is an hyperparameter which will be later optimized. We then take $l_{i,1}$ equal to the number of times the state S appears in the sampling. Similarly, $l_{i,2}$ counts the occurrences of E and I, and $l_{i,3}$ of R and D. Therefore the sum of elements in l_i is always equal to k for $i \in \mathcal{M}$. The remaining nodes are labeled as *unknown* by taking $l_i = 0$ for all $i \notin \mathcal{M}$. These operations are graphically summarized in the left part of Fig. 3.

3.3 Neural architecture

Graph Neural Networks operate in the domain of the graph. In the graph, each node comes with its label. A common framework in the GNN is the classification problem setup where the goal is to predict the label of

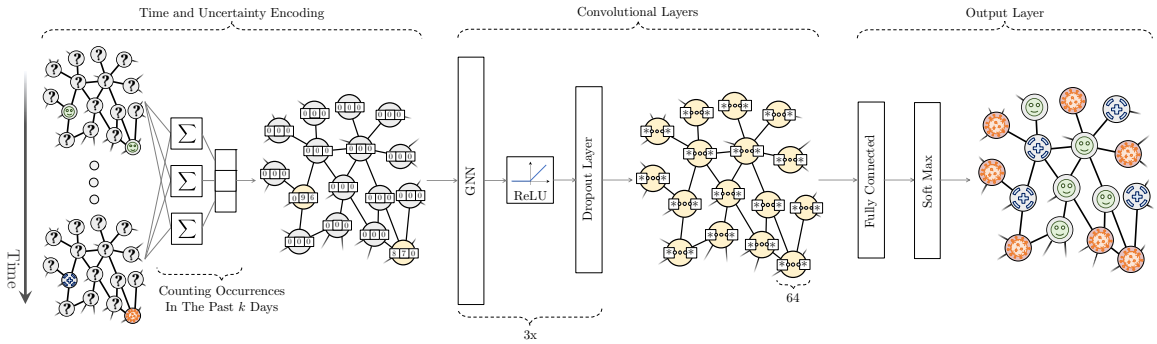


Fig. 3 The proposed architecture is made up of three stages. The first one samples data from the evolution of the known nodes y in the past k days, and counts the occurrences of the three classes. In this way it creates labels l encoding the temporal information. The second stage performs most of the computation, and it is made of three graph convolutional layers. Finally, the high dimensional internal information is compressed again by the output layer, i.e. a fully connected network and a soft max. The output is an estimation of the current full state of the epidemics spreading

the unlabeled nodes given the labeled ones. As mentioned before we want to predict the full state of the pandemics spreading x, see Fig. 1.

The central part of Fig. 3 shows the core GNN layers in our architecture. The target of our GNN is to learn the state embedding $l_i \in \mathbb{N}^3$ for $i = 1, 2, 3$, which contains the information of neighborhood for each node. The initial node feature corresponds to the node state itself, encoded in binary vector $\in \mathbb{N}^3$ that contains only one element equal to 1. We preprocess this information by integrating l_i along the time horizon of k. We cannot use k too big to avoid that the neural network leverages on this pattern to recognize that the state coincide with the node label. Due to the fact that it is a classification problem setup, we then mask a certain percentage of node $(95 - 90 - 80\%)$ depending on the scenario we are considering. We finalize the preprocess by loading data by batch by using the Dataloader class defined inside the Pytorch library [37]. Thanks to a specific variable, named 'batch', the data loader can associates node and edges to a specific graph. Since a DataLoader aggregates nodes, edges and the features from different graphs into batches during the message passing layers, the GNN model needs this information to know which nodes belong to the same graph. For what concern message passing layer, it describes how l_i is passed through the layers of the network to create the node embedding. As we know, the message passing layer is the result of the generalization of the convolution operator by extending the concept of the neighbourhood from pixels to nodes [24]. Given the state of the node i at the layer h, l_i^h we find the l_i^{h+1} by applying the activation function of the message passing layer to l_i^h and the aggregation of l_j^h where $j \in \mathcal{N}_i$ is a neighbour of node i (and $\mathcal{N}_i$ is the neighbourhood of node i). As the node embedding evolve through the message passing layers, as the knowledge of the neighborhood of each single node increases. Thus, the message passing layers enlarge, in general, the size of the node feature. The number of the message passing layers could be considered again as a hyperparameter. Without loss of generality, in our case, three message passing layers with a rectifier as the activation function (ReLu) are considered. The first message passing layer has an input size of 3 (i.e. the number of features), and output size of 64. The second and the third message passing layers have an input and output sizes of dimension 64. Between layers there the dropout regularization method is used during training to avoid over-fitting.

As a result of this processing, each node is equipped with a rich description of its possible state as inferenced by its neighbours own representations. This state need to be converted into one of the three states $\{S, E + I, R + D\}$. This is done through the Output Layer (right part of Fig. 3). First we have a fully connected layer. Its input size is 64 and output size of 3. It is defined with a linear activation function. Then, a $softmax$ function in introduced as defined in the Pytorch library. It is applied to the observation l_i so to retrieve the highest probability that the node will be labeled with a certain class.

For what concerns the training, an Adam optimizer with a fixed learning rate is defined and we select loss $L_1(\cdot)$ as the cross entropy. Given the unbalanced classes $c \in \{S, E+I, R+D\}$ we compute weight w_c to normalize observation l_i. The weight of each class is determined by the $\frac{N_{\max}}{N_c}$ where $N_{\max}$ is the number of observations in the class with maximum occurrence and N_c is the number of observations or nodes belonging to class c in the training set. We use the loss function for measuring the performance of the algorithm as described by the Pytorch library. The losses are then averaged across observations:

$$L_1(\cdot) = \frac{\sum_{c=1}^{3} \text{loss}(x_c)}{\sum_{c=1}^{3} w_c}. \tag{5}$$

Given a fixed number of epochs (250 in our case) we train our network and we measure the loss function as previously defined to measure the loss. TWe represent the model's predictions in a confusion matrix under four categories. We label true positives into the considered class as TP. False positives (FP) refer to nodes from other classes incorrectly labeled as belonging to a particular class. True negatives (TN) refer to nodes correctly classified as belonging to other classes. False negatives (FN) refer to nodes incorrectly labeled as negative or belonging to another class. We evaluate the goodness of our GNN through precision and accuracy as Precision $=$ TP/(TP + FP), Accuracy $=$ TP/(TP + FN). Hyperparameter optimization is done using balanced accuracy. Balanced accuracy is calculated as the average of the proportion corrects of each class individually. Balanced accuracy is suitable for datasets with class imbalance unlike other metrics which may favour results from the majority class.

4 Simulations

We test the proposed architecture in two scenarios with different topological characteristics. The first one is a homogeneous network, in which any node has the same probability of being connected with all the others. We use this scenario to test the effectiveness and scalability of the method extensively. The second scenario is instrumental in testing the neural architecture in a more challenging setting, closer to a real-world scenario.

4.1 Scenario 1: random network

We consider Erdős-Rényi networks, which are a class of well-known network models. Such random networks are relatively simple to describe, and at the same time offer some heterogeneity in terms of the degree distribution. The generative algorithm can be described as follows: we start with N isolated nodes, then we place a link between any two nodes with probability $0 < p < 1$. The degree distribution of the network is therefore binomial $\mathbb{B}(N, p)$. We showcase results for networks with average degree $\langle k \rangle = 30$. This value is comparable with the number of daily contacts at risk as measured in a recent survey [33].

We generate training set from 80 realizations, each one happening on a different and randomly generate social network with a population of 500 nodes. The epidemic spreads between 0 and 120 days. Yet in the initial month, the behavior is quite stationary due to the well-know slow increase of the total number of infected subjects. Therefore, a few samples from the initial days is enough to learn the pattern during that period. Only 3 random days are selected from the first month of each realization. All the remaining days from 30 to 120 are used for training. The hyperparameters are 0.3 for dropout, 64 hidden units, 3 layers. We use a learning rate of 0.0002, we train for 250 epochs, with a batch size of 256.

At first, we test the trained architecture on a set of 40 realizations, representing evolutions on randomly generated social networks with 500 nodes (same size of the training set). We repeat the analysis for the cases in which the size of $\mathcal{M}$ (i.e. monitored subjects) is 5%, 10%, 20% of the size of V (i.e. the total amount of subjects in the considered population). It is worth to notice that this is a very sparse amount of information. Indeed, 10% of tests with an average connectivity of 10 means that any node has on average just a single neighbor whose the state is known (see Fig. 1 to get a visual sense of this ratio). Accuracy and precision of the predictions are provide in Table 1. Note that these values are evaluated only on the nodes which are not part of $\mathcal{M}$ since they are always perfectly known. Thus, we prefer to leave them out to not artificially increase the performance of the neural network. Interestingly, the quality of the predictions do not change significantly with the size of $\mathcal{M}$. In general classes with larger amount of subjects have better performance. This can be due to the higher amount of examples which are available from

Table 1 Accuracy (Ac.) and precision (Pr.) of the classification for Scenario 1, evaluated only on the nodes which are not in $\mathcal{M}$. The testing set is generated with networks of 500 nodes. Three levels of supervision are considered - i.e., number of nodes of V which are in $\mathcal{M}$ as well

	Ac., 5%	Pr., 5%	Ac., 10%	Pr., 10%	Ac., 20%	Pr., 20%
S	0.93	0.84	0.93	0.84	0.93	0.85
$E+I$	0.52	0.56	0.51	0.57	0.51	0.57
$R+D$	0.74	0.78	0.75	0.77	0.76	0.77
All	0.75	–	0.75	–	0.76	–

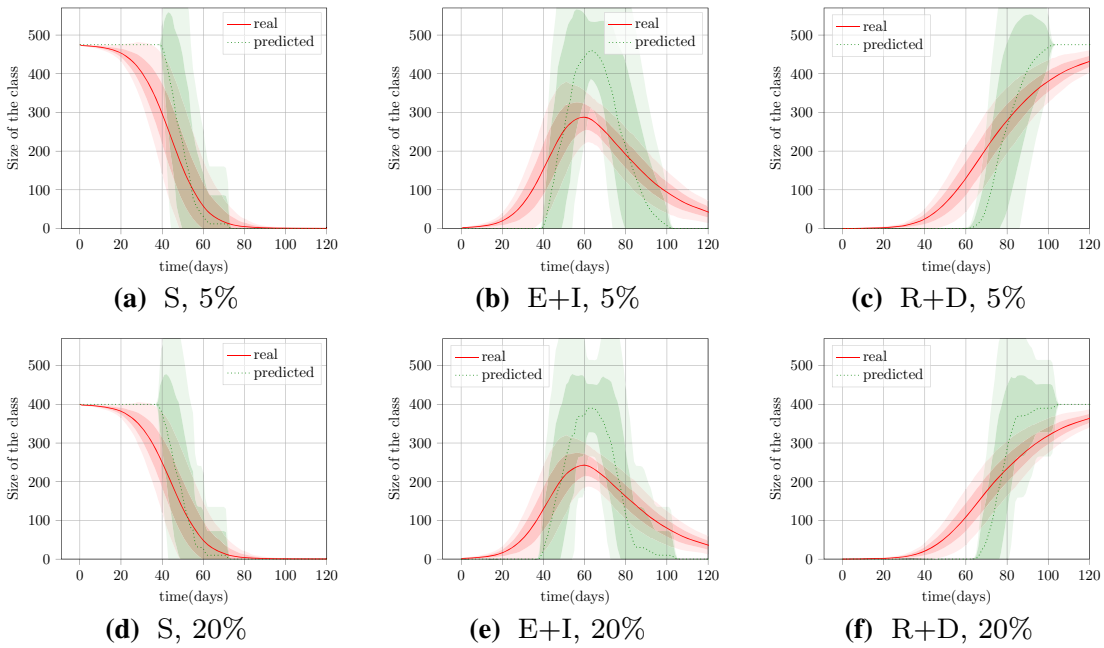


Fig. 4 Evolution of overall statistics associated to the epidemics, when evolving on a medium-small homogeneous network (scenario 1). More specifically, we show the amount of nodes which are susceptible - Panels **a**,**d** - which got infected by the pathogen - Panels **b**,**e** - and which either recovered or died - Panels **c**,**f**. Actual evolutions are in red, while estimations are in green. The solid lines represent the mean, while the translucent areas the variance. Training and testing sets are made of realizations produced by simulating the epidemic spreading on random social networks of 500 subjects. In Panels **a**–**c** only 5% of subjects is tested at any time, while in Panels **d**–**f** this number reaches 20%

the training set. Overall the performance is satisfactory, with a general accuracy always higher than 0.75. To get a sense of how these results reflect in the estimation of cumulative statistics of the pandemic evolution, in Fig. 4 we plot the total size of each class against the amount of nodes which are classified to be part of that class. The match is good. The network is not sensitive to small deviations of S and $R + D$ from the maximum and the minimum value. This may be due to the fact that so small variations may not be captured by changes in $\mathcal{M}$. Also, the neural network tends to overestimate the presence of subjects which got infected at the peak. A possible interpretation could be derived from the fact that the neural network learns that after a certain amount of days, without any kind of restriction, the neighborhood of an infected node is more likely to be all infected. It is very important to stress here that these overall statistics serve here only to get a sense of the overall quality of the network predictions. The goal of the neural architecture is indeed not to estimate these

values directly, but the exact way in which each class is spread over the social network. This is an important distinction because the direct estimations of the size of the three classes is a relatively simple task, as discussed in the Introduction.

A nice property that our architecture inherits from Graph Convolutional Neural network is that once trained it can be applied to graphs of any size. This is because we directly learn the weights of the convolution operator, which can then be applied to networks of all sizes. There is however no guarantee that the classification will keep being effective. Indeed, the way in which the pandemic evolves is clearly affected by the size of the social network despite the local rules remaining the same. We therefore tested the ability of the architecture to generalize to larger populations by building an additional testing set of 10 realizations with a total number of nodes which is several orders of magnitude larger than before: 10^5 subjects. It is very important to stress that no re-training is performed. Therefore, we are training the neural architecture with a small-village community, and testing it with a medium size city. Table 2 and Fig. 5 show the result of this analysis. No essential differences can be observed. Overall the performance is still satisfactory, with a general accuracy which is even higher than the previous test set and always equal to 0.83. This may be due to the fact that larger social networks generate more homogeneous distributions of the illness since the border-effects are less dominant.

4.2 Scenario 2: Boston

The second scenario we consider aims at modeling a more realistic social structure, such as the one of a relatively big city. We need to consider both a model that takes into account the existence of different neighborhoods and the age distribution of people living in that area. We take as a reference the City of Boston and Cambridge, Massachusetts, USA . (Fig. 6)

The generative model, which takes inspiration from the work in [34], is divided into three steps, as in fig. 6. Initially, we outline a map of the neighborhoods of the urban area we focus on. At this stage, each neighborhood is a network on its own. The size of each neighborhood is taken from the official website of the city of Boston[1] and Cambridge.[2] Within each neighborhood, the topology reflects the contact patterns between different age-classes, as described in the Supplementary material of [34]. To do so, we cohort the population into age groups of size 5 years, and we model the contact patterns among groups based on their age with a stochastic block model [19]. Stochastic block models are generative models for random graphs that are used to generate topologies that have a community-like structure. Each node is given a unique label (the age cohort). Then, we define a symmetric matrix (known as Affinity matrix) whose elements are $\mathcal{A}_{ij} = p_{ij}$, where p_{ij} is the probability that a node whose label is i is in contact with a node whose label is j. The Affinity matrix we use is the Massachusetts age-contact matrix, as described in the supplementary material of [34]. The age-contact matrices, available online[3] are built with a data-driven approach, that relies on detailed census and survey data from publicly available sources, and incorporates key features like the socio-economic status and the household composition. Such matrices represent the inferred per-capita probabilities of contact between individuals divided by age in a particular area or country. The last step is to connect different neighborhoods by allowing nodes in each neighborhood to have links with nodes from other neighborhoods. To do so, we consider a diffusion-like procedure: for each couple of neighborhoods we place a random number of links between randomly selected nodes from both communities, depending on the length of the shortest path connecting the two on the geographical level: neighborhoods at distance d from each other will share, on average, $1/d$ links with respect to neighborhoods at distance 1. The number of links shared between any two communities is drawn from a Binomial with probability $p = \frac{1}{50}\frac{1}{d}$.

We generated a training composed of 20 realizations, each one happening on a different and randomly generated social network with a population of 10^4 nodes. This is one order of magnitude less than the actual population of that area. This choice has been imposed by limits on the hardware resources available. In this scenario, the epidemic spreads over a relatively long period

[1] https://www.bostonplans.org/getattachment/7987d9b4-193b-4749-8594-e41f1ae27719.

[2] https://www.cambridgema.gov/-/media/Files/CDD/FactsandMaps/profiles/demo_profile_neighborhood_2019.pdf.

[3] https://github.com/mobs-lab/mixing-patterns.

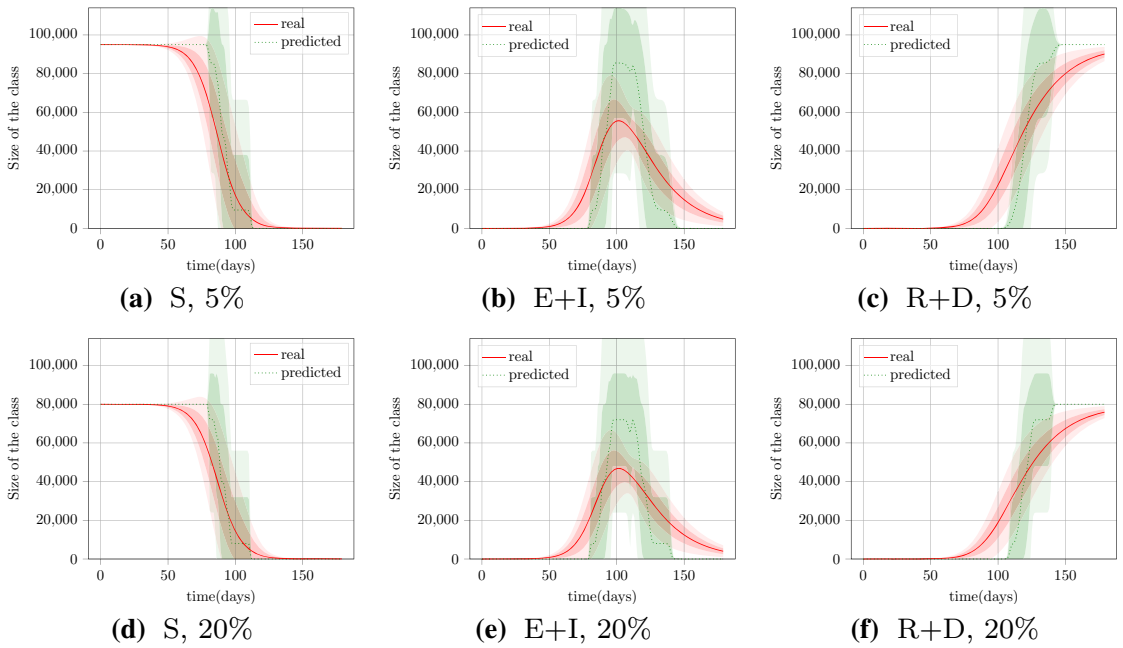


Fig. 5 Evolution of overall statistics associated to the epidemics, when evolving on an homogeneous network (scenario 1). Actual evolutions are in red, and estimations in green. The solid lines represent the mean, while the translucent areas the variance. The testing set is made of realizations produced by random social networks of 10^5 subjects. Instead the training set contains only networks which are 500 nodes big. In Panels **a–c** only 5% of subjects is tested at any time, while in Panels **d–f** this number reaches 20%

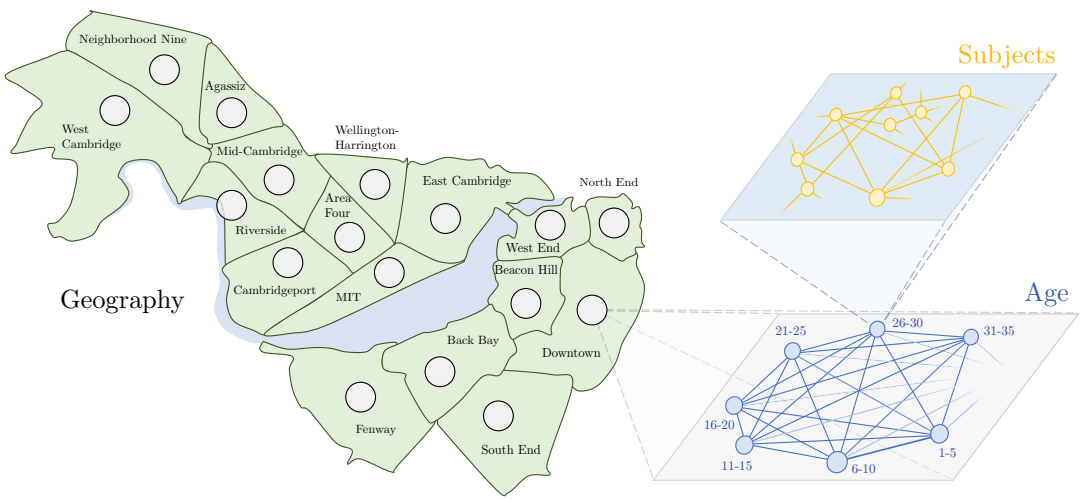


Fig. 6 The second scenario is a toy model inspired by the spreading of CoVid in the Boston and Cambridge, Massachusetts. The topology of the graph is built on three layers, which integrate the geographical distribution of the population, the demographics, and the subject level variability

Table 2 Accuracy (Ac.) and precision (Pr.) of the classification for Scenario 1, evaluated only on the nodes which are not in $\mathcal{M}$. The testing set is generated with networks of 10^5 nodes. Three levels of supervision are considered - i.e., number of nodes of V which are in $\mathcal{M}$ as well

	Ac., 5%	Pr., 5%	Ac., 10%	Pr., 10%	Ac., 20%	Pr., 20%
S	0.97	0.92	0.96	0.92	0.93	0.85
$E+I$	0.50	0.58	0.50	0.57	0.51	0.57
$R+D$	0.79	0.81	0.80	0.81	0.76	0.77
All	0.83	–	0.83	–	0.83	–

of time, with each day being of importance and different. Hence we select a total of 201 of days from each realizations, starting 100 days before the peak of the infection, and ending 100 days after. No sample is removed. The hyperparameters are 0.4 for dropout, 64 hidden units, 3 layers. We use a learning rate of 0.0002, we train for 250 epochs, with a batch size of 64.

We test the effectiveness of the proposed approach by collecting a testing set made of 10 realizations. Each realization is an evolution of the epidemic on a different and randomly generated social network (following the same statistical characteristics of the testing set). As for scenario 1, also here we test the case of size of $\mathcal{M}$ (i.e. tested subjects) being 5%, 10%, or 20% of the total population. Results are shown in Table 3 and Fig. 7. Although lower in the easier scenario 1, the accuracy is consistently good across classes and conditions. Yet, the accuracy of S is a bit lower than before, and the precision of $E + I$ is very low. This is because the neural architecture tends to wrongly label a number of nodes which are susceptible as infected. Yet, it is important to underline here that the neural network is working with a quite small amount of information on the spread of infected subjects. Indeed, at its peak $E+I$ is less than the 10% of the population, which with 10% of measures means that the algorithm can rely on the knowledge of 10^2 infected nodes. This behavior is also evident in Fig. 7, where the total number of susceptible subjects is higher than estimated, and vice versa the infected subjects are lower than the neural network thinks. By considering Table 3, we can notice that precision and accuracy slightly increase as $\mathcal{M}$ increase. This is a different trend than the one observed in Table 1, and it may be due to our algorithm overestimating the number of nodes that will be infected in a homogeneous network. In a more complex network topology, GNN fits well with this kind of hierarchical model due to its intrinsic ability to retrieve features of a higher level given from the convolutional operator. It is again important to stress that the proposed algorithm is optimized to estimate the distribution of subjects rather than the total size of each class, which should therefore be regarded as a secondary index. It is also interesting that the algorithm rarely does the opposite error, i.e. classifying S as $E + I$. The precision of S is indeed above 97%. Although not explicitly forced in training phase, this behavior makes very much sense in the practice since it is better to isolate healthy subjects than not to act on infected ones.

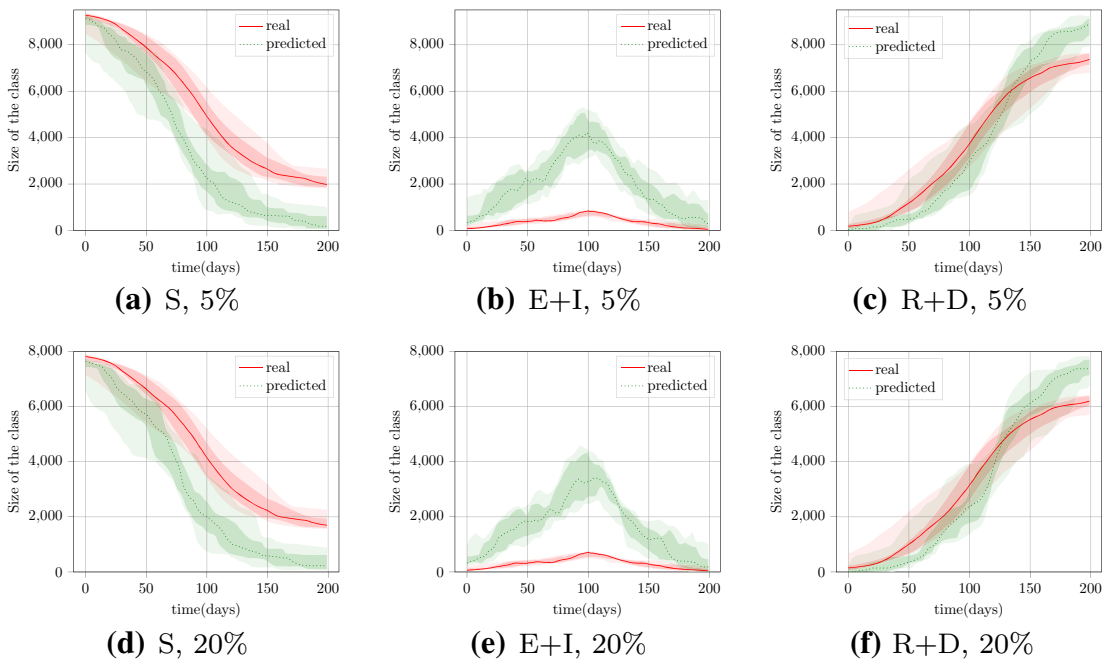
From an alternative standpoint, Fig. 8 shows a comparison of the Cumulative Distribution Function (CDF) across all the scenarios. For each level of accuracy ($0 \leq x \leq 1.0$), the CDF represents the fraction of algorithms execution that results in an accuracy of x or less. This way, we can more clearly observe the entire distribution beyond only mean and variance. We also directly compare how the distribution function behaves in the different scenarios. From this perspective, we can notice that our algorithm seems to perform better in the Boston scenario than in the similarly large-scale random network. The network solves 80% of the executions with an accuracy of about 80%. This may be because the deep convolutional nature of our algorithm performs better when a the social network follows a non-trivial topology. In this case, there are more information to exploit.

5 Discussion and conclusions

We investigated the use of Graph Neural Networks to develop state observes for epidemics evolving on social networks with this work. The results are promising. The proposed architecture can approximate the overall state with an accuracy above 70% on the nodes that are not directly monitored. We consistently observe these

Table 3 Accuracy (Ac.) and precision (Pr.) of the classification for Scenario 2 (Boston), evaluated only on the nodes which are not in $\mathcal{M}$. Three levels of supervision are considered—i.e., number of nodes of V which are in $\mathcal{M}$ as well

	Ac., 5%	Pr., 5%	Ac., 10%	Pr., 10%	Ac., 20%	Pr., 20%
S	0.67	0.97	0.67	0.98	0.67	0.98
$E+I$	0.75	0.14	0.78	0.14	0.80	0.15
$R+D$	0.78	0.79	0.79	0.79	0.80	0.80
All	0.72	–	0.72	–	0.73	–



(a) S, 5% **(b)** E+I, 5% **(c)** R+D, 5%

(d) S, 20% **(e)** E+I, 20% **(f)** R+D, 20%

Fig. 7 Evolution of overall statistics associated to the epidemics, when evolving on a toy model inspired by the Boston and Cambridge (MA,USA) areas. We show the amount of number of susceptibles in Panels **a**,**d**, the infected and exposed in Panels **b**,**e**, the recovered or dead in Panels **c**,**f**. Actual evolutions are in red, while estimations are in green. The solid lines represent the mean, while the translucent areas the variance. Training and testing sets are made of realizations produced by simulating the epidemic spreading on random social networks of 500 subjects. In Panels **a**–**c** only 5% of subjects is tested at any time, while in Panels **d**–**f** this number reaches 20%

performance even when the sample space is as small as 5% of the total population. In these conditions, the network must perform inference on subjects that divided by several degrees of separation from the nodes that we are measuring. Also, it should be considered that no retraining of the network is performed when changing the network.

Nonetheless, there are several directions towards which we may further improve our results. First, future work will be devoted to adding explicit dynamic reasoning within the neural network, for example, introducing recurrent layers [28]. This change should help boost the capability of the neural network of discerning between exposed and infected and between infected and recovered (or dead). Indeed, these transitions are essentially time-dependent and can be extracted from associating internal dynamics to the initial recognition that a node entered in the exposed state. Nevertheless, it is worth mentioning that stacking LSTMs layers between the GNNs did not produce a statistically relevant increase in the network performance and, as such, has not been included in the present work. Similarly, attention mechanisms [47] have been tested but

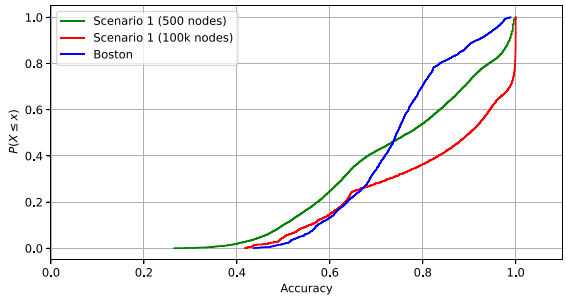


Fig. 8 Cumulative Distribution Function of Scenario 1 (500 and 100.000 nodes) and of the Boston scenario. We show only the more challenging case, in which 5% of the population is monitored. The other two are qualitatively similar. Interestingly, the neural network appears to exploit the richer structure of the more realistic social network topology in the Boston case

not included due to the negligible increment of performance that they resulted in.

Finally, we believe that a very important assumption to be relaxed is the full knowledge of the social network (see Sect. 3.1). Indeed, we have assumed a complete knowledge of the social structure $(V, \mathcal{E})$ as input for the proposed algorithm. This assumption can be relaxed and introduced in the model as a time-variant probability for the edges. Several algorithms are being proposed that can extract the social structure from GPS localization and other mobility information provided by contact tracing apps [9,14]. Data driven methods can then possibly be used to infer the graph topology itself [16,41].

Funding Information This work is supported by the TU Delft CoVid-19 response fund, and by the Leverhulme Trust through the Research Project (Grant number RPG2017-370).

Data availability The dataset used to train and test the network are large and not easy to share. Being the datasets synthetically generated, we have decided to share the code instead (https://github.com/DiLauroF/gnninferenceepid). Running them should be sufficient to replicate the study since we found no statistically significant difference between different datasets generated. We will share the datasets upon reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Bacciu, D., Errica, F., Micheli, A., Podda, M.: A gentle introduction to deep learning for graphs. Neural Netw. **5**, 87 (2020)
2. Bahr, D.B., Browning, R.C., Wyatt, H.R., Hill, J.O.: Exploiting social networks to mitigate the obesity epidemic. Obesity **17**(4), 723–728 (2009)
3. Battistelli, G., Chisci, L.: Stability of consensus extended kalman filter for distributed state estimation. Automatica **68**, 169–178 (2016)
4. Battistelli, G., Benavoli, A., Chisci, L.: Data-driven communication for state estimation with sensor networks. Automatica **48**(5), 926–935 (2012)
5. Block, P., Hoffman, M., Raabe, I.J., Dowd, J.B., Rahal, C., Kashyap, R., Mills, M.C.: Social network-based distancing strategies to flatten the covid-19 curve in a post-lockdown world. Nature Human Behav. **4**(6), 588–596 (2020)
6. Britton, T., Pardoux, E., Ball, F., Laredo, C., Sirl, D., Tran, V.C.: Stochastic epidemic models with inference. Springer, New York (2019)
7. Bronstein, M.M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P.: Geometric deep learning: going beyond euclidean data. IEEE Signal Process. Magaz. **34**(4), 18–42 (2017)
8. Brunton, S.L., Kutz, J.N.: Data-driven science and engineering: machine learning, dynamical systems, and control. Cambridge University Press, USA (2019)
9. Cheng, H.Y., Jian, S.W., Liu, D.P., Ng, T.C., Huang, W.T., Lin, H.H., et al.: Contact tracing assessment of Covid-19 transmission dynamics in Taiwan and risk at different exposure periods before and after symptom onset. JAMA Internal Med. **180**(9), 1156–1163 (2020)
10. Cutura, G., Li, B., Swami, A., Segarra, S.: Deep Demixing: Reconstructing the Evolution of Epidemics Using Graph Neural Networks. http://arxiv.org/abs/201109583 (2020)
11. Di Lauro, F., Kiss, I.Z., Rus, D., Della Santina, C.: Covid-19 and flattening the curve: a feedback control perspective. IEEE Control Syst. Lett. **5**(4), 1435–1440 (2020)
12. Ding, D., Wang, Z., Ho, D.W., Wei, G.: Distributed recursive filtering for stochastic systems under uniform quantizations and deception attacks through sensor networks. Automatica **78**, 231–240 (2017)
13. Ding, D., Han, Q.L., Wang, Z., Ge, X.: A survey on model-based distributed control and filtering for industrial cyber-physical systems. IEEE Trans. Indus. Inf. **15**(5), 2483–2499 (2019)
14. Ferretti, L., Wymant, C., Kendall, M., Zhao, L., Nurtay, A., Abeler-Dörner, L., Parker, M., Bonsall, D., Fraser, C.: Quantifying sars-cov-2 transmission suggests epidemic control with digital contact tracing. Science **368**, 6491 (2020)
15. Gao, J., Sharma, R., Qian, C., Glass, L.M., Spaeder, J., Romberg, J., Sun, J., Xiao, C.: STAN: spatio-temporal attention network for pandemic prediction using real-world evidence. J. Am. Med. Inf. Assoc. **28**(4), 733–743 (2021)
16. Giannakis, G.B., Shen, Y., Karanikolas, G.V.: Topology identification and learning over graphs: accounting for nonlinearities and dynamics. Proc. IEEE **106**(5), 787–807 (2018)

17. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. J. Phys. Chem. **81**(25), 2340–2361 (1977)
18. He, S., Peng, Y., Sun, K.: Seir modeling of the Covid-19 and its dynamics. Nonlinear Dyn. **101**(3), 1667–1680 (2020)
19. Holland, P.W., Laskey, K.B., Leinhardt, S.: Stochastic blockmodels: first steps. Social Netw. **5**(2), 109–137 (1983)
20. Iacopini, I., Petri, G., Barrat, A., Latora, V.: Simplicial models of social contagion. Nature Commun. **10**(1), 1–9 (2019)
21. Jain, A., Liu, I., Sarda, A., Molino, P.: Food Discovery with Uber Eats: Using Graph Learning to Power Recommendations (2019)
22. Kapoor, A., Ben, X., Liu, L., Perozzi, B., Barnes, M., Blais, M., O'Banion, S.: Examining covid-19 forecasting using spatio-temporal graph neural networks. http://arxiv.org/abs/200703113 (2020)
23. Keshtegar, B., Bagheri, M., Fei, C.W., Lu, C., Taylan, O., Thai, D.K.: Multi-extremum-modified response basis model for nonlinear response prediction of dynamic turbine blisk. Eng. Computers **5**, 1–12 (2021)
24. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. http://arxiv.org/abs/160902907 (2016)
25. Kiss, I.Z., Miller, J.C., Simon, P.L., et al.: Mathematics of epidemics on networks. Springer, Cham (2017)
26. Kompella, V., Capobianco, R., Jong, S., Browne, J., Fox, S., Meyers, L., Wurman, P., Stone, P.: Reinforcement learning for optimization of covid-19 mitigation policies. http://arxiv.org/abs/201010560 (2020)
27. Kutz, J.N.: Deep learning in fluid dynamics. J. Fluid Mech. **814**, 1–4 (2017)
28. Liang, X., Shen, X., Feng, J., Lin, L., Yan, S.: Semantic object parsing with graph lstm. In: European Conference on Computer Vision, Springer, pp 125–143 (2016)
29. Linka, K., Peirlinck, M., Sahli Costabal, F., Kuhl, E.: Outbreak dynamics of Covid-19 in Europe and the effect of travel restrictions. Computer Methods Biomech. Biomed. Eng. **23**(11), 710–717 (2020)
30. Liu, Q., Wang, Z., He, X., Zhou, D.: On kalman-consensus filtering with random link failures over sensor networks. IEEE Trans. Autom. Control **63**(8), 2701–2708 (2017)
31. Liu, Y., Wang, Z., Liang, J., Liu, X.: Synchronization and state estimation for discrete-time complex networks with distributed delays. IEEE Trans. Syst., Man,Cybern., Part B (Cybern.) **38**(5), 1314–1325 (2008)
32. Lusch, B., Kutz, J.N., Brunton, S.L.: Deep learning for universal linear embeddings of nonlinear dynamics. Nature Commun. **9**(1), 1–10 (2018)
33. Melegaro, A., Jit, M., Gay, N., Zagheni, E., Edmunds, W.J.: What types of contacts are important for the spread of infections? Using contact survey data to explore European mixing patterns. Epidemics **3**(3–4), 143–151 (2011)
34. Mistry, D., Litvinova, M., Piontti, A.P., et al.: Inferring high resolution human mixing patterns for disease modeling. Nature Commun. **12**(1), 1–12 (2021)
35. Nande, A., Adlam, B., Sheen, J., Levy, M.Z., Hill, A.L.: Dynamics of Covid-19 under social distancing measures are driven by transmission network structure. PLOS Comput. Biol. **17**(2), 1008 (2021)
36. Pastor-Satorras, R., Castellano, C., Van Mieghem, P., Vespignani, A.: Epidemic processes in complex networks. Rev. Modern Phys. **87**(3), 925 (2015)
37. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. http://arxiv.org/abs/191201703 (2019)
38. Péni, T., Csutak, B., Szederkényi, G., Röst, G.: Nonlinear model predictive control with logic constraints for Covid-19 management. Nonlinear Dyn. **102**(4), 1965–1986 (2020)
39. Saha, P., Mukherjee, D., Singh, P.K., Ahmadian, A., Ferrara, M., Sarkar, R.: Graphcovidnet: a graph neural network based model for detecting Covid-19 from ct scans and x-rays of chest. Scientif. Rep. **11**(1), 1–16 (2021)
40. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Trans. Neural Netw. **20**(1), 61–80 (2008)
41. Segarra, S., Marques, A.G., Mateos, G., Ribeiro, A.: Network topology inference from spectral templates. IEEE Trans. Signal Inf. Process. Netw. **3**(3), 467–483 (2017)
42. Shah, C., Dehmamy, N., Perra, N., Chinazzi, M., Barabási, A.L., Vespignani, A., Yu, R.: Finding patient zero: Learning contagion source with graph neural networks. http://arxiv.org/abs/200611913 (2020)
43. Shim, E., Tariq, A., Choi, W., Lee, Y., Chowell, G.: Transmission potential and severity of Covid-19 in South Korea. Int. J. Infect. Dis. **93**, 339–344 (2020)
44. Soatti, G., Nicoli, M., Savazzi, S., Spagnolini, U.: Consensus-based algorithms for distributed network-state estimation and localization. IEEE Trans. Signal Inf. Process. Netw. **3**(2), 430–444 (2016)
45. Tizzoni, M., Bajardi, P., Poletto, C., Ramasco, J.J., Balcan, D., Gonçalves, B., Perra, N., Colizza, V., Vespignani, A.: Real-time numerical forecast of global epidemic spreading: case study of 2009 a/h1n1pdm. BMC Med. **10**(1), 1–31 (2012)
46. Valle, J.A.M.: Predicting the number of total Covid-19 cases and deaths in brazil by the gompertz model. Nonlinear Dyn. **102**(4), 2951–2957 (2020)
47. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. http://arxiv.org/abs/171010903 (2017)
48. Wang, W., Liu, Q.H., Liang, J., Hu, Y., Zhou, T.: Coevolution spreading in complex networks. Phys. Rep. **820**, 1–51 (2019)
49. Wang, Z., Ho, D.W., Liu, X.: State estimation for delayed neural networks. IEEE Trans. Neural Netw. **16**(1), 279–284 (2005)
50. Weinberger, K., Dasgupta, A., Langford, J., Smola, A., Attenberg, J.: Feature hashing for large scale multitask learning. In: Proceedings of the 26th annual international conference on machine learning, pp 1113–1120 (2009)
51. Xu, Y., Lu, R., Shi, P., Tao, J., Xie, S.: Robust estimation for neural networks with randomly occurring distributed delays and markovian jump coupling. IEEE Trans. Neural Netw. Learn. Syst. **29**(4), 845–855 (2017)
52. Yang, Z., Zeng, Z., Wang, K., Wong, S.S., Liang, W., Zanin, M., Liu, P., Cao, X., Gao, Z., Mai, Z., et al.: Modified seir and ai prediction of the epidemics trend of Covid-19 in china under public health interventions. J. Thoracic Dis. **12**(3), 165 (2020)

53. Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W.L., Leskovec, J.: Graph convolutional neural networks for web-scale recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp 974–983 (2018)
54. Zhang, D., Wang, Q.G., Srinivasan, D., Li, H., Yu, L.: Asynchronous state estimation for discrete-time switched complex networks with communication constraints. IEEE Trans. Neural Netw. Learn. Syst. **29**(5), 1732–1746 (2017)
55. Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M.: Graph neural networks: A review of methods and applications. http://arxiv.org/abs/181208434 (2018)