

## Research paper

## Transforming norm-based to graph-based spatial representation for spatio-temporal epidemiological models

Teddy Lazebnik \*

Department of Information Systems, University of Haifa, Haifa, Israel  
 Department of Computing, Jonkoping University, Jonkoping, Sweden

## ARTICLE INFO

## Keywords:

Transformative algorithms  
 Computational epidemiology  
 Agent-based simulation  
 Heuristic optimization algorithms  
 Machine learning

## ABSTRACT

Pandemics, with their profound societal and economic impacts, pose significant threats to global health, mortality rates, economic stability, and political landscapes. In response to these challenges, numerous studies have employed spatio-temporal models to enhance our understanding and management of these complex phenomena. These spatio-temporal models can be roughly divided into two main spatial categories: norm-based and graph-based. Norm-based models are usually more accurate and easier to model, but are more computationally intensive and require more data to fit. On the other hand, graph-based models are less accurate and harder to model, but are less computationally intensive and require fewer data to fit. As such, ideally, one would like to use a graph-based model while preserving the representation accuracy obtained by the norm-based model. In this study, we explore the ability to transform from norm-based to graph-based spatial representation for these models. We first show that no analytical mapping between the two exists, requiring one to use numerical approximation methods instead. We introduce a novel framework for this task, together with twelve possible implementations using a wide range of heuristic optimization approaches. Our findings show that by leveraging agent-based simulations and heuristic algorithms for the graph node's location and population's spatial walk dynamics approximation, one can use graph-based spatial representation without losing much of the model's accuracy and expressiveness. We investigate our framework for three real-world cases, achieving 93% accuracy preservation, on average, while obtaining 86% relative computational time reduction. Moreover, an analysis of synthetic cases shows the proposed framework is relatively robust for changes in both spatial and temporal properties.

## 1. Introduction

Pandemics throughout history caused intense impact on societies across centuries, leaving an indelible mark on mortality rates, economic stability, and political landscapes (Andrea Alberto Conti, 2020). In the last decades, the persistent threat of emerging and reemerging pandemics and epidemics looms large over humanity (Brodeur et al., 2020). Indeed, recent outbreaks such as the severe acute respiratory syndrome (SARS) in 2003, the H1N1 influenza pandemic in 2009, the Middle East respiratory syndrome coronavirus (MERS-CoV) in 2012, the Ebola virus in 2014, and the ongoing global severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) pandemic, underscore the urgency of comprehending, preparing, and managing pandemics (Euros-surveillance Editorial Team, 2020; Mackey and Liang, 2012; Shami and Lazebnik, 2023; De Groot et al., 2013; Iuliano et al., 2018).

To this end, multiple mathematical and computational models have been proposed to ameliorate our understanding of the pandemic spread

and advance our ability to control it (Jones et al., 2021; Acemoglu et al., 2021; Wiratsudakul et al., 2018; Adiga et al., 2020). These models can be roughly divided into two primary groups: statistical models and mechanistic models. The former group employs a data-driven approach without making explicit assumptions about the underlying dynamics, while the latter relies on theoretical principles to elucidate the mechanisms governing pandemic dynamics. Statistical models, on the other hand, commonly leverage statistical and machine learning (ML) methods to project a diverse array of outcomes. Within this category, a multitude of models has been developed utilizing various techniques, including auto-regressive time series methods, Bayesian optimization techniques, and, more recently, deep learning models (Desai et al., 2019; Ivorra et al., 2020; Long and Ehrenfeld, 2020; Salgotra et al., 2020; Agarwal and Jhajharia, 2021). Recently, a growing body of work uses neural dynamic models, in which deep neural networks, in particular Graph Neural Networks (GNNs) and Transformer-based

\* Correspondence to: Department of Computing, Jonkoping University, Jonkoping, Sweden.  
 E-mail address: [lazebnik.teddy@gmail.com](mailto:lazebnik.teddy@gmail.com).

architectures, are trained end-to-end to approximate complex spatio-temporal dynamics directly from data (Liu et al., 2024; Tomy et al., 2022). In these models, interactions between agents or locations are typically encoded through message-passing mechanisms on graphs, and the learned latent representations implicitly capture mobility patterns, contact structures, and non-linear temporal dependencies (Jin et al., 2023; Yu et al., 2023b; Wang et al., 2022). These models have been shown to provide promising results on the pandemic they were trained on, but they usually do not generalize well to other cases, require large amounts of data to obtain reasonable performance, and are very sensitive to noise. The latter group, exemplified by the SIR model, operates based on a predefined set of rules or processes governing interactions among individuals within a population (Tuite et al., 2020; Miller, 2017a; Berge et al., 2017; Miller, 2017b). At the forefront of this approach stands the Susceptible–Infected–Recovered (SIR) model (Al-Raei, 2020; Fernández-Villaverde and Jones, 2020; Ellison, 2020), originally proposed by Kermack and McKendrick (1927). Models belonging to this group require more effort from the modeler, as one needs to manually detect, define, and formalize the main interactions and processes occurring in a pandemic for the model to capture the dynamics correctly. That said, once done efficiently, models from this group are more generalizable, adaptable, explainable, and require less data.

Focusing on the mechanistic models, a growing number of works adopted a spatiotemporal model formalization as the spatial nature of pandemics is repeatedly shown to take a significant part in the dynamics (Byrd et al., 1995; Friedman et al., 2021; Lazebnik et al., 2021b). Intuitively, the first SIR model (Kermack and McKendrick, 1927) assumes a well-mixed property of the population, which means that for each point in time  $t$ , the probability two random individuals in the population would interact is uniformly distributed. This assumption is problematic as it is known not to be accurate for large-scale scenarios such as cities or countries (Baber, 2020; Andraud et al., 2012; Conti, 2020). Commonly, modelers mistakenly assume one can use the well-mixture assumption for short periods of time or small spatial locations, such as rooms or buildings, but recent studies show that even for these cases, the approximation is sub-optimal if one desires reasonably accurate predictions (Peng et al., 2020; Lazebnik and Alexi, 2022; Fukuoka and Ito, 2010; Chen et al., 2017). Therefore, spatial dynamics are important in order to capture pandemic spread. One can categorize the spatial component into two main groups: Graph-based and Norm-based. The graph-based approach assumes a graph,  $G = (V, E)$  such that  $E \subseteq V \times V$ , where  $V$  are the nodes and  $E$  are the edges between them. Interestingly, the nodes ( $V$ ) can either represent locations where individuals are located at in a given time (Masuda and Holme, 2017; Holme, 2021; Kiss et al., 2017) or the individuals themselves. For the first case, the edges are usually the possible connections between the physical location and the individuals performing some kind of walk between the nodes (Ellison, 2020; Holme, 2021). This approach abstracts the physical domain of the pandemic in favor of a more abstract and computationally attractive representation allowing the modeler to focus on “conceptual” locations, often ignoring the transformation between them (Alexi et al., 2023b). On the other hand, for the second case, the edges indicate the interaction between the individuals, often associated with infectious routes (Viguerie et al., 2021; Bognanni et al., 2020). In a complementary manner, the norm-based approach assumes a continuous space, rather than a discrete one enforced in the graph-based approach, where individuals are moving and interacting over time. This setting is usually more computationally expensive while providing a more physically accurate presentation of reality (Connell et al., 2009).

One can easily notice that the norm-based representation is more expressive than the graph-based one. Nonetheless, this expressiveness can be computationally overwhelming while relatively easy to model and comprehend compared to the more abstract graph-based approach.

Therefore, it is interesting to pose the question of whether we can transform models from norm-based to graph-based representations, in a way that preserves the model’s accuracy in representing the epidemiological dynamics. Hence, in this study, we proposed a novel framework that utilizes an agent-based simulation (ABS) (Chumachenko et al., 2018; Priest et al., 2021) representation of a norm-based spatio-temporal model to obtain a numerically similar graph-based spatio-temporal model. We evaluate the proposed model on three different temporal epidemiological models as well as three different pandemic settings to explore the algorithm’s performance.

The rest of the paper is organized as follows. Section 2 outlines the model definition and ABS representation. Section 3 provides a formal description of the norm-based to graph-based representation conversion task and possible heuristic numerical-based solutions to this task. Section 4.1 describes the experimental setup used to evaluate the proposed algorithms and the obtained experimental results. Finally, Section 5 discusses the results, the proposed algorithm’s limitations, and provides possible future work.

## 2. Generic spatio-temporal model formalization

In this section, we outline the formalization of the temporal component of the epidemiological model based on the SIR model’s structure, the norm-based spatial component, the graph-based spatial component formalization, and the ABS representation of the model.

### 2.1. Temporal component formalization

For the temporal component, let us assume a population of timed finite state machines (Alagar and Periyasamy, 2011) agents,  $P$ , such that each agent in the population,  $p \in P$ , is defined by a tuple  $p := (\xi, \tau)$  where  $\xi \in \Xi$  is the current epidemiological state of the agent out of the set of all possible epidemiological states ( $\Xi$ ) and  $\tau$  is a set of variables used to determinate the additional properties of the agent.

In addition, the temporal component contains a function  $I_s : P \rightarrow P$  which gets the population of agents and returns the same population of agents after altering the state of each agent in the population based on their current state and change over time. For example, let us consider the ordinary differential equation (ODE) representation of the SIR model (Kermack and McKendrick, 1927):

$$\frac{dS(t)}{dt} = -\beta S(t)I(t); \frac{dI(t)}{dt} = \beta S(t)I(t) - \gamma I(t); \frac{dR(t)}{dt} = \gamma I(t), \quad (1)$$

where  $S(t)$ ,  $I(t)$ , and  $R(t)$  are the number of susceptible, infected, and recovered agents in the population,  $\beta$  is the average infection rate, and  $\gamma$  is the average recovery rate. In this example, the second term of the  $\frac{dI(t)}{dt}$  and the third equation is part of the  $I_s$  function while the term  $\beta S(t)I(t)$  is not as the change in the agents’ state is not exclusively depended on time.

Importantly, we allow  $I_s$  to both add and remove agents to the population. This allows the temporal model to include the birth and death of agents into and from the dynamic, respectively.

### 2.2. Norm-based spatial component formalization

For the norm-based spatial component, one should consider three main elements: the environment ( $E$ ), agents’ walk dynamics, and infection dynamics. In this formalization, we assume that the information related to the agent’s location, movement, and infection dynamics is described as part of the  $\tau$  parameter.

Intuitively, the environment element,  $E$ , outlines the physical location where the pandemic takes place. For example, for small-scale, the environment can be a specific building, and for larger-scale, the environment can be a city or even a country. On top of that, the resolution of the environment and its accuracy are also important parts of it. Following the same example, a one-meter resolution of a building is feasible, while the same resolution for a city can be computationally

infeasible for the average modern computation system. Similarly, in a building or even room setting, one may take into consideration the agents' breathing patterns (Lazebnik and Alexi, 2023) as the population size is small, while on the city level, the time of year may be more relevant (and computationally feasible) property to consider. Formally, The environment is presented by continuous space,  $L \in \mathbb{R}^k$  ( $k \in \{1, 2, 3\}$ ) is the spatial dimension of the environment, such that point,  $l \in L$ , is represented by some set of parameters denoted by  $\{\alpha_1, \dots, \alpha_j\}$  where  $j \in \mathbb{N}$  is the number of parameters. Theoretically, an infinite number of locations are presented in the environment.

The agent's walk dynamics is a function  $I_w : (p, P, E) \rightarrow (p, E)$  that gets a specific individual, the entire population, and the environment and returns the same individual with the relevant alter to the  $\tau$  parameter related to its location and other variables which related to the fact it moved, as well as updating the relevant points in the environment.

Finally, the agents' interaction dynamics is a function  $I_i : (P, E) \rightarrow (P, E)$  that gets the population and the environment and returns the population with the relevant alter to the  $\xi$  epidemiological parameter as well as to  $\tau$  while also updating the relevant points in the environment. This function is responsible for capturing the infection performed between agents in the population, as well as between an agent and the environment itself.

### 2.3. Graph-based spatial component formalization

The graph-based spatial component is similar to the norm-based spatial component, where the differentiation is present only in the formalization of the environment itself. Namely, the environment,  $E$ , is represented by a graph  $G := (V, E)$  where  $V$  are a set of locations and  $E \subset V \times V$  is a set of edges between these locations. Importantly, the size of  $|V|$  is finite. For the agent's walk dynamics ( $I_w$ ) we allow individuals to move from a current location ( $v_c \in V$ ) to a neighbor location ( $v_n \in V$ ) in a single step in time if and only if  $e = (v_c, v_n) \in E$ . Finally, for the interaction dynamics,  $I_i$ , we assume that at each location ( $v \in V$ ) the individuals are well-mixed.

### 2.4. Agent-based simulation representation

Following the structure proposed by Lazebnik (2023a), an ABS representation of a spatio-temporal model is presented in Algo 1. Initially, a population of agents ( $P$ ) and the environment ( $E$ ) are generated. Then, in an iterative manner, until a pre-defined stop condition ( $C$ ) is met, three operators take place — the spontaneous operator ( $I_s$ ), the walk operator ( $I_w$ ), and the interaction operator ( $I_i$ ). These operators alter the agents' state over time. Commonly, the population's epidemiological state (denoted by  $P_\xi$ ), defined to be the vector of agents' states, over time is returned for further analysis.

**Algorithm 1** Generic agent-based simulation representation of spatio-temporal epidemiological model.

---

```

1: Input: population ( $P$ ), environment ( $E$ )
2: Output: Population's state over time ( $A$ )
3:  $t \leftarrow 1$ 
4: while  $C(P, t, E)$  do
5:    $P \leftarrow I_s(P)$ 
6:   for  $p \in P$  do
7:      $p \leftarrow I_w(p, P, E)$ 
8:   end for
9:    $(P, E) \leftarrow I_i(P, E)$ 
10:   $A[t] \leftarrow P_\xi$ 
11:   $t \leftarrow t + 1$ 
12: end while
13: return  $A$ 

```

---

## 3. Norm-based to graph-based spatial component translation

Following the formalization in Section 2, one can notice that the translation of the norm-based to graph-based spatial component is trivial without constraints as one can pick a discretization to the continuous environment which is at least as small as the smallest interaction-related distance and generate a mesh graph. However, this solution ignores the motivation of reducing computational time when transforming from norm-based to graph-based spatial component representation while also not necessarily resulting in analytically isomorphic results. To prove this point, let us again consider the SIR model in Eq. (1) but this time we define the infection rate at time  $t$  between two agents in the population ( $i, j$ ) to be  $\beta_{i,j} = \frac{1}{|L_i - L_j|}$  where  $L_x \in \mathbb{R}^k$  is the locations of the  $x$ 's agents at time  $t$  (Lazebnik, 2023b). Moreover, let us assume the agents in the population apply the following walking dynamics:  $\forall j : L(t+1) = \frac{1}{|P|-1} \sum_{i=1, i \neq j}^{|P|} L_i(t)$ . In this case, there is no existing fixed discretization size that is at least as small as the smallest interaction-related distance as the latter converges to zero.

At a high level, our goal is to construct a coarse-grained spatial representation that preserves the epidemiologically relevant dynamics of the original norm-based model. In the norm-based setting, infection events are driven by spatio-temporal co-location patterns and local densities of agents in continuous space. The graph-based representation can be viewed as a partition of this continuous space into a finite set of "regions" (graph nodes) within which individuals are approximately well-mixed at the time scale of infection, together with a transition kernel describing how agents move between these regions. If (i) the partition groups together locations that are dynamically similar with respect to contact patterns, and (ii) the induced transition kernel reproduces the empirical flows of agents between regions, then the resulting graph-based model will, by construction, approximate the norm-based epidemic dynamics as measured by the population state trajectories  $A_n$  and  $A_g$ .

### 3.1. Problem definition

Following this example, the desired graph-based representation should be both computationally appealing and preserve as much as possible the norm-based dynamics. We treat the norm-based to graph-based translation as a constrained optimization problem:

$$\min_{|V|} \min |A_n - A_g| \quad (2)$$

where  $A_n$  and  $A_g$  are the population's state over time for the norm-based and graph-based spatial components, respectively. From the computational point of view, one should define the graph,  $G = (V, E)$ , and the walk dynamics on the graph.

Below, the algorithms used for both sub-tasks are briefly described with extended technical description is provided in Appendix.

### 3.2. Stage 1: Heuristic numerical-based solution

Since the problem is challenging and solving it analytically can be computationally expensive (or even infeasible) even for relatively small cases, we propose a heuristic approach to solve the problem. The methods presented below are not independent heuristics but different numerical strategies to approximate this optimization problem by (i) searching over spatial partitions (graph search) and (ii) learning the induced transition kernel (walk dynamics approximation - WDA). Namely, let us consider a given case with a norm-based spatial component with some walk dynamics. In order to convert it into a graph-based representation, one would need to find  $G$  as well as transform the walk dynamics to the constraints enforced by  $G$ . Below, we describe five methods for the graph search and four for the WDA, resulting in 20 possible models.

### 3.2.1. Graph search

For the graph search component, we assume each algorithm gets the locations of the agent population at each step in time and returns the spatial graph ( $G$ ).

**Uniform square grid (Grid).** As a simple, fixed spatial discretization baseline, we consider a uniform square grid over the environment. Let the walkable area be contained in an axis-aligned bounding box  $[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$ . We discretize this domain into  $N_x \times N_y$  rectangular cells of equal size, where  $N_x$  and  $N_y$  are chosen using a grid search (Lerman, 1980) optimizing Eq. (2). Nodes are connected by an undirected edge if their corresponding cells share a common side (4-neighborhood connectivity). In environments with obstacles or non-rectangular floorplans, we remove cells that fall entirely outside the walkable region; this yields an irregular subset of the original grid but preserves the local regularity of node spacing where agents can actually move.

**Quadtree.** A Quadtree is a hierarchical data structure used for efficient spatial partitioning of an environment. It recursively subdivides a region of dimension  $k$  into  $2^k$  sub-sections, each representing a smaller portion of the environment (Chien and Aggarwal, 1986). This tree-like structure is commonly employed in computer graphics, image processing, and geographical information systems for efficient spatial indexing and searching (Samet, 1984). In our case, we first build a Quadtree at each simulation time step,  $t$ , given the agents' locations. Formally, let the root node be the entire environment. Recursively subdivide a node into  $2^k$  children (for a  $k$ -dimensional space) until one of the following conditions holds for that node (Krekov et al., 2015): (i) it contains no agents; (ii) it contains a single agent that cannot interact with any other agent within the node; or (iii) all agents inside the node can interact with each other (i.e., their pairwise distances are below the interaction radius). The resulting structure at time  $t$  is a Quadtree  $Q_t$ . To obtain a single, time-aggregated spatial partition from the sequence  $\{Q_t\}_{t=1}^T$ , we construct what we refer to as the “average Quadtree” as follows. First, we align all  $Q_t$  to a common maximum depth, interpreting missing children in shallower trees as empty cells at that level. Then, for each cell  $c$  at each level, we compute an occupancy score

$$\omega(c) = \frac{1}{T} \sum_{t=1}^T \mathbf{1}\{c \text{ exists in } Q_t \text{ and satisfies the interaction condition (ii) or (iii)}\},$$

which is the fraction of time steps in which  $c$  is a leaf in  $Q_t$  and is “interaction-coherent”. We then prune the tree by selecting as leaves those cells whose occupancy score exceeds a threshold  $\omega(c) \geq \theta$  and whose children (if any) have strictly lower scores; all such leaves constitute the nodes of the final, time-aggregated Quadtree. Finally, each selected leaf cell becomes a vertex  $v \in V$  of the graph  $G = (V, E)$ , and an undirected edge  $e = (v_i, v_j)$  is added between two vertices if their corresponding cells share a common face or parent in the Quadtree. This procedure yields a single graph-based spatial discretization that summarizes how the continuous environment is populated and used over the entire simulation horizon, rather than at a single snapshot in time.

**Genetic algorithm (GA).** A GA is an optimization technique inspired by the process of natural selection, where populations of potential solutions evolve over generations (Bo and Rein, 2005; Ghaheri et al., 2005). It involves the use of genetic operators such as selection, crossover, and mutation to iteratively improve candidate solutions towards achieving optimal or near-optimal outcomes for a given problem (Davis, 1985a; Bo et al., 2006a; Hassanat and Alkafaween, 2017). In order to use the GA approach, one is required to answer seven modeling decisions: (1) How to define a member (also known as “gene”) in the population; (2) how to generate the initial population; (3) what is the stop condition for the evolution process; (4) how to define the mutation operator; (5) how to define the cross-over operator; (6) how to define the selection

operator; and (7) how to define the loss function of each member (also known as the “fitness function”) (Nix and Vose, 1992; Alexi et al., 2023a). To this end, we represent each member in the GA's population as a vector of location such that each location is the node's location in the environment. An agent is associated with the node that is the closest to it at any given point in time. Next, the GA's population initialization involves generating an initial population with a random number of nodes as well as their locations, denoted by  $\delta_i$  for the  $i_{th}$  agent. Hence, the loss function (i.e., the *fitness* function) is defined to be:

$$L(m) := \frac{1}{T} \sum_{t=0}^T \left( \frac{1}{N} \sum_{i=1}^n \delta_i \right). \quad (3)$$

In order to obtain a more computationally appealing initial condition, for an initial number of nodes  $x \in \mathbb{N}$ , we compute the set cover using a greedy algorithm for the initial state of the agents' population (i.e.,  $t = 0$ ) (Grossman and Wool, 1997). The stop condition for the algorithm is a pre-defined number of iterations (also known as “generation”) (Holland, 1992). For each iteration, three operators are performed. First, the mutation operator picks a node's index randomly, with a uniform distribution, and changes its location by adding to its current location a random vector sampled from a pre-defined  $n$ -dimensional normal distribution. Second, the cross-over operator accepts two node location vectors of identical size ( $V_1, V_2$ ), picks an index,  $j \in [1, \dots, |V_1|]$  in a uniform distribution and returns  $V_1^{new} = V_1[1, \dots, j] \cup V_2[j+1, \dots, |V_2|]$  and  $V_2^{new} = V_2[1, \dots, j] \cup V_1[j+1, \dots, |V_2|]$  (Davis, 1985b). Finally, for the selection operator, we adopted the *royalty tournament* operator (Bo et al., 2006b), which selects the top  $\alpha \in [0, 1]$  graphs from the population according to their loss function. Afterward, the population is sampled (with repetitions) according to their loss function values, i.e., with probability  $p_{select}(m) = \frac{L(m)}{\sum_{m' \in P_i} L(m')}$  where  $P_i$  is the population in the  $i_{th}$  iteration.

**Time series X-means (TSxM).** Time series K-means (TSKM) is a clustering algorithm tailored for time-dependent data, aiming to identify homogeneous groups or clusters based on temporal patterns (Cai et al., 2021). It employs the traditional K-means approach but incorporates dynamic time warping or other distance metrics suitable for capturing similarities in time series data. On top of that, it automatically finds the number of clusters,  $k$ , using a hyperparameter optimization process. To use the TSxM approach, the location of each agent in the population is treated as a time-series vector, with each value representing the location of the agent in the environment in a single step in time. The locations are normalized to the environment's size to ensure comparability. The number of clusters (i.g., the nodes of the graph  $G$ ,  $x$ , is set to 1 and increases up to a pre-defined number  $\epsilon \in \mathbb{N}$ . The final value of  $x$  is obtained using the elbow point method (Syakur et al., 2018). For each value of  $x$ , a Dynamic Time Warping K-means algorithm (Soheily-Khah et al., 2016) is used to cluster the time-series data. As the Dynamic Time Warping K-means algorithm is sensitive to its random initial condition, we run it for each simulation for  $a \in \mathbb{N}$  times, taking the configuration with the smallest  $L_1$  metric value between the agent's true location and the cluster's center of mass.

**Recurrent Neural Network (RNN).** A Recurrent Neural Network (RNN) is a type of neural network architecture designed to process sequential data by maintaining a hidden state that captures information from previous time steps and enables the modeling of temporal dependencies in sequences, making it suitable for tasks such as time-series prediction and classification (Zhang and Man, 1998). In particular, for our case where the number of agents in the population may change over time, we cannot assume a fixed input space's size. As such, we use two layers of Gated Recurrent Unit (GRUs) such that one obtains the agent population as a vector with arbitrary size, followed by two fully connected layers to obtain a fixed-sized and with a more computationally appealing feature space, while the latter accepts sequences of the processed agent's population locations and also followed by two layers



of fully connected layers. As such, the input of the RNN model is the agent population's locations over time, and the output is the graph's nodes' locations. For the loss function, we adopted Eq. (3) and for the optimization algorithm, we adopted ADAM (Kingma and Ba, 2017).

Notably, in the GA- and RNN- based variants, we do not optimize Eq. (2) directly inside the search loop. Instead, we use the following spatial surrogate: we assign each agent to its closest node and minimize the average distance between agents and their assigned node over all agents and time steps (Eq. (3)). Intuitively, this encourages each node to represent a compact, approximately well-mixed region of space, and pushes the graph towards a low-distortion partition of the continuous environment, which in turn is expected to preserve the pairwise proximity patterns that drive infection events.

### 3.2.2. Stage 2: Walk dynamics approximation

Given a fixed graph-based environment  $G = (V, E)$ , we refer to “WDA” as the task of learning a graph-based walk operator  $I_w^g$  that mimics the movement behavior of the original norm-based walk operator  $I_w$ . Formally,  $I_w^g$  defines, at each time step, a transition policy that maps from an agent's current node  $v \in V$ , its internal and epidemiological state, and possibly aggregate properties of the population and environment, to a probability distribution over feasible next nodes  $v' \in V$  such that  $(v, v') \in E$ . This transition policy induces a Markov kernel on  $V$  and is chosen so that the resulting trajectories on the graph reproduce, as closely as possible, the trajectories generated by  $I_w$  in the continuous space.

**Markov chain (MC).** A Markov chain is a stochastic process where the probability of transitioning to a particular state depends only on the current state, disregarding the sequence of events leading to that state. Intuitively, it is characterized by a set of states and transition probabilities between those states (Privault, 2018; Sharma, 2017). In our case, for each node in the graph ( $v \in V$ ) and agent ( $p \in P$ ) that is currently located in the node, let us assume the MC's state is represented by the agent's state, the environment's state, and the epidemiological's state distribution of the agents in each node  $v_n \in V$  such that  $e = (v, v_n) \in E$ . Notably, this approach ignores any additional properties the agents might have in favor of the common (epidemiological) ground.

**Multi-agent classification with AutoEncoder (MAC-AE).** The multi-agent classification approach stands for the idea that a classification task can be solved by multiple models, each one governing a sub-set of the task, and they are autonomous from each other. In our case, each node of the graph ( $v \in V$ ) is associated with an ML classification model. Each such ML model obtains the state of an agent currently located in the node, the environment, and the entire population. However, as the size of the entire population can alter over time, a fully connected attention AutoEncoder model (Bank et al., 2023; Lazebnik and Simon-Keren, 2023) is used to obtain a fixed-length representation of the population data. As such, the model is trained on tuples of the agent's state, environment, and encoded population representation, while the target column is the agent's location, as represented by the graph node's index, in the next step in time. In order to find the best ML model for each node, we used an automatic machine learning (AutoML) approach. Namely, we adopted the Tree-Based Pipeline Optimization Tool (TPOT), a genetic algorithm-based automatic ML library (Olson and Moore, 2016). TPOT produces a full ML pipeline, including feature selection engineering, model selection, model ensemble, and hyperparameter tuning, which is close to optimal for relatively long enough computational time (Lazebnik et al., 2023).

**Deep reinforcement learning (DRL).** Deep Reinforcement Learning (DRL) is a data-driven algorithm that combines reinforcement learning with deep neural networks to enable agents to learn complex behaviors and decision-making processes (Lazebnik, 2023c; Witty et al., 2021; Zhang et al., 2020; Veturi et al., 2023). It involves training artificial agents to

interact with an environment, receive feedback in the form of rewards or penalties, and leverage deep neural networks to approximate optimal strategies for maximizing cumulative rewards over time (Arulkumaran et al., 2017). For our case, the walk dynamics are represented in the same manner as the MAC-AE method. However, rather than using an AutoML model with AE for each node of the graph ( $v \in V$ ), we use a single DRL model for the entire graph ( $G$ ). To be exact, we adopt the Proximal Policy Optimization (PPO) (Schulman et al., 2017), which considers one of the state-of-the-art DRL algorithms for a wide range of tasks. Specifically, we use a fully connected neural network at the base of the PPO algorithm.

**Spatio-temporal graph neural network (STGNN)** Spatio-temporal graph neural networks extend message-passing GNNs with temporal sequence models (e.g., recurrent or attention-based modules) to capture how node states evolve over time on a fixed graph, and have been successfully applied to pandemic forecasting from regional case data (Hy et al., 2022; Nguyen et al., 2023). We adapt this idea to WDA by treating the graph  $G = (V, E)$  as given and constructing, at each time step, node features describing the current occupancy and epidemiological composition. A stack of spatial GNN layers first propagates information along edges to encode the local neighborhood of each node; these node embeddings are then processed by a temporal encoder that summarizes recent history and outputs, for each node  $v \in V$ , a probability distribution over its neighbors representing the likelihood of agents moving to each adjacent node at the next step. Agents sample their next node from this distribution during simulation, and the model parameters are learned by training on the norm-based trajectories so that the induced flows between nodes match the observed movements as closely as possible.

### 3.3. Evaluation metric

We quantify the approximation quality of the graph-based spatial representation by comparing the time series of epidemiological compartment counts produced by the norm-based and graph-based models, denoted by  $A_n(t)$  and  $A_g(t)$ , respectively. Let  $A_\bullet(t) \in \mathbb{R}^d$  be the vector of compartment counts (e.g.,  $S, I, R, \dots$ ) at time  $t$ , and let  $T$  be the total number of time steps. We define the normalized agreement (i.e., performance) between the two models as

$$1 - \frac{1}{T} \sum_{t=1}^T \frac{\|A_n(t) - A_g(t)\|}{\|A_n(t)\|}, \quad (4)$$

so that the result is between 0 and 1, with higher values indicating better preservation of the epidemiological dynamics of the norm-based model by the graph-based model.

## 4. Results

Based on the algorithms presented in Section 3, an experimental setup is defined for both real-world and synthetic cases such that the real-world cases are used to evaluate the algorithms' performances in realistic scenarios while the synthetic cases are used for further sensitivity analysis of different spatial conditions.

### 4.1. Experimental setup

In order to evaluate the proposed algorithms, we adopted three temporal component dynamics as well as three norm-based spatial components with increasing levels of complexity for both. For the temporal component, we used SIR (Kermack and McKendrick, 1927), SIRD with two age groups (Lazebnik et al., 2021a), and two-strain SIR models (Lazebnik and Bunimovich-Mendrazitsky, 2022). Simply put, SIR is considered the most simplistic model with only three states — susceptible (S), infected (I), and recovered (R) individuals, where

**Table 1**

A summary of the epidemiological model's parameters' values for the synthetic and three real-world cases.

Symbol	Parameter	Synthetic	Airport	Restaurant	Bus
$\beta$	Infection rate in a hour [ $t$ ]	0.037–0.1	0.05–0.1	0.062–0.079	0.048–0.069
$\gamma$	Recovery duration in a hour [ $t$ ]	120–240	144–170	144–170	144–170
$\rho$	Recovery rate (Andrea Alberto Conti, 2020)	0.9–0.99	0.98–1	0.98–1	0.98–1
$\psi$	Symptomatic rate (Andrea Alberto Conti, 2020)	0.05–0.20	0.1–0.15	0.1–0.15	0.1–0.15

susceptible individuals become infected from interaction with infected individuals while infected slowly recover over time until they fully recover and transform to the recovered epidemiological state where they are immune to re-infection. The SIRD with two age groups introduces a dead state (denoted by  $D$ ) where individuals who are infected have some probability of dying rather than recovering. Moreover, it extends the SIR model by dividing the population into two age groups (for example, children and adults) with asymmetric infection rates and unique recovery durations. Finally, the two-strain SIR assumes two pathogens are causing a pandemic in parallel and individuals can be infected and/or recovered from either or both of them at any given moment. A formal ODE and ABS descriptions of each temporal model are provided in [Appendix](#). In all experiments, the norm-based spatial component is simulated via an ABS, which yields complete trajectories for all agents over time. These fully observed trajectories serve as “privileged information” for the translation framework: they are used by the graph search stage to infer a spatial partition and by the WDA stage to estimate transition behavior between regions.

Since each epidemiological scenario is known to have unique parameter values, we set a biologically relevant range for the synthetic cases while adopting the values presented in the original works of the real-world cases, following the values for COVID-19 ([Adiga et al., 2020](#)). [Table 1](#) summarizes the parameters and their values for all four cases.

Moreover, the proposed algorithms require that some hyperparameters would be configured before being used. These hyperparameters are closely related to the algorithm's performance and commonly different for different tasks to obtain the algorithm's optimal performance ([Yang and Shami, 2020](#)). As such, we used the default suggested hyperparameters' values from the code libraries we adopted in Python. [Fig. 1](#) presents a schematic view of the experimental process.

All experiments were conducted on the same workstation to ensure comparable runtime measurements. The machine is equipped with an Intel Xeon Gold 6230R, 2.1 GHz, 40 cores with 128 GB RAM, running *Ubuntu 18.04* as the operating system. All algorithms were implemented in Python 3.9 with PyTorch 2.1 and executed in single-precision floating point, where applicable. For each epidemiological scenario, we report wall-clock times averaged over three runs with different random seeds. The reported translation times should therefore be interpreted as indicative of relative, rather than absolute, computational cost under this hardware setup.

#### 4.1.1. Real-world cases

For the real-world cases, we adopted three locations and walks: airport ([Harweg et al., 2023](#)), restaurant ([Kwon et al., 2020](#)), and bus ([Shen et al., 2020](#)). [Fig. 2](#) shows schematic views of the three locations as presented by [Harweg et al. \(2023\)](#), [Shen et al. \(2020\)](#), [Kwon et al. \(2020\)](#). In order to make the representation of all three locations identical, we used pixel-level measurements of the provided images alongside the scale declared by the authors, generating a grid of (a) 2.5 m, (b) 0.5 m, and (c) 0.5 m. Despite the fact that height plays a role in the infection dynamics ([Lazebnik and Alexi, 2023](#)), it is shown only when taking airflow dynamics into account, which we do not do in this experiment. Hence, we ignore any height-related information that may be in these settings.

In addition, we adopted the walk dynamics presented in the original papers. Simply put, each agent has a unique set of locations in the space, which is influenced by the simulation's step in time as well as the presence of other agents. In the airport, agents randomly obtain a path from the entrance and a gate and move at a constant speed until they arrive at the gate. For the restaurant, some agents were allocated to sit, at random, according to the time of the day in the simulation, while small, pre-defined agents performed random walks. In a similar manner, agents in the bus case enter from the top door, picking a free sit at random, walking to the seat, waiting for a random number of simulation steps, moving to the exit door, and exiting.

Importantly, as not the entire population is present in the dynamics at any point in time, as required by the SIR-based temporal component, we assume individuals getting and leaving the scene are not interacting with each other and only can recover (or die) if they are infected. We pick the individuals to enter the scene at random, in a uniformly distributed manner.

These three scenarios can be seen as “semi-real” or “proxy” validation settings. The floorplans and walk dynamics are directly adapted from published investigations of real environments and outbreaks, and thus reflect empirically grounded geometries and movement patterns. However, we emphasize that the infection and recovery processes in our experiments are still simulated. We do not have access to jointly observed high-resolution mobility and infection time series for these environments.

#### 4.1.2. Synthetic cases

In order to explore the proposed algorithms' sensitivity to multiple spatial components of the space, we decided to keep the spatial configuration simple and highly adaptable. To this end, we allocate an on a plane, a set of non-pairwise distinct circles. Each circle is defined by its radius and center location. As such, the spatial configuration of each synthetic case is defined by a set of circles and their properties. This allows the generation of a wide space of configurations while keeping the seed relatively simple. Moreover, it allows for control of the average density of the population, enforces walking patterns, etc. An example of the simplest configuration, a configuration with different sub-locations with different densities, and spatially-enforced walk patterns are shown in panels a, b, and c, respectively, in [Fig. 3](#). For simplicity, in these examples, we assume individuals perform random walks and can occupy the same location. To capture complex enough settings, we set the number of circles to be between 1 and 40 and the radius to be between 2 and 25 m.

We assume individuals follow random walks, but have a “pulling” power towards their original spawn location which is reversibly proportional to their distance up to some distance  $\delta$ .

#### 4.2. Performance analysis

We start by exploring the performance of all twelve possible combinations of the graph search and the WDA for the synthetic cases. As such, we generate  $n = 10000$  random synthetic cases where the population size is set to 1000 individuals. [Table 2](#) summarizes the results, where the performance represents the average agreement between the norm-based and graph-based population states over time, and the relative time reduction quantifies the speed-up of the translated model.

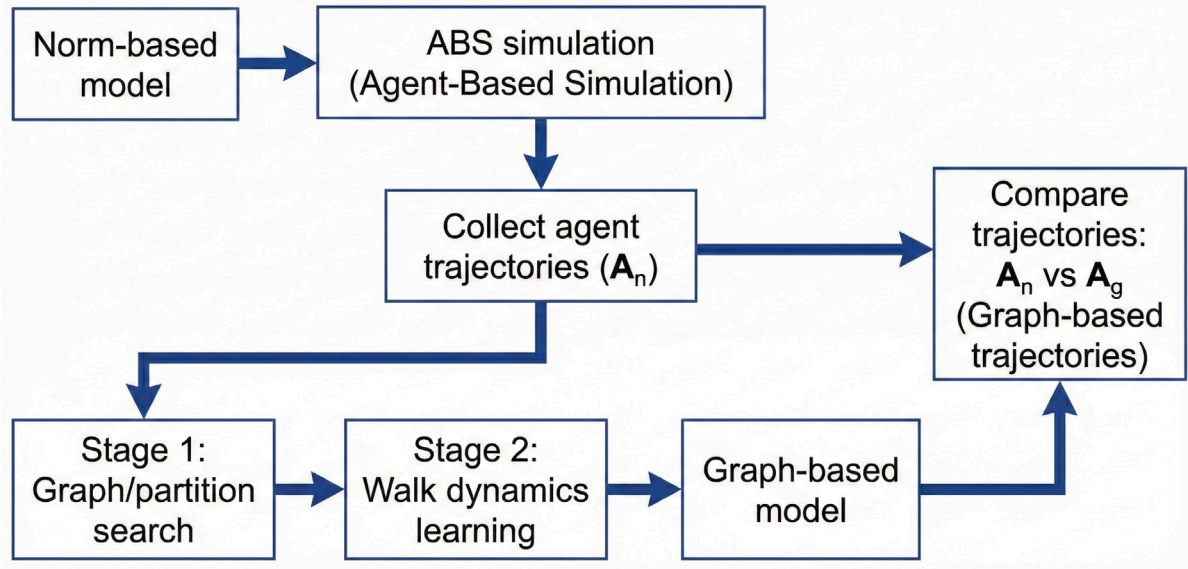


Fig. 1. A schematic view of the experimental process.

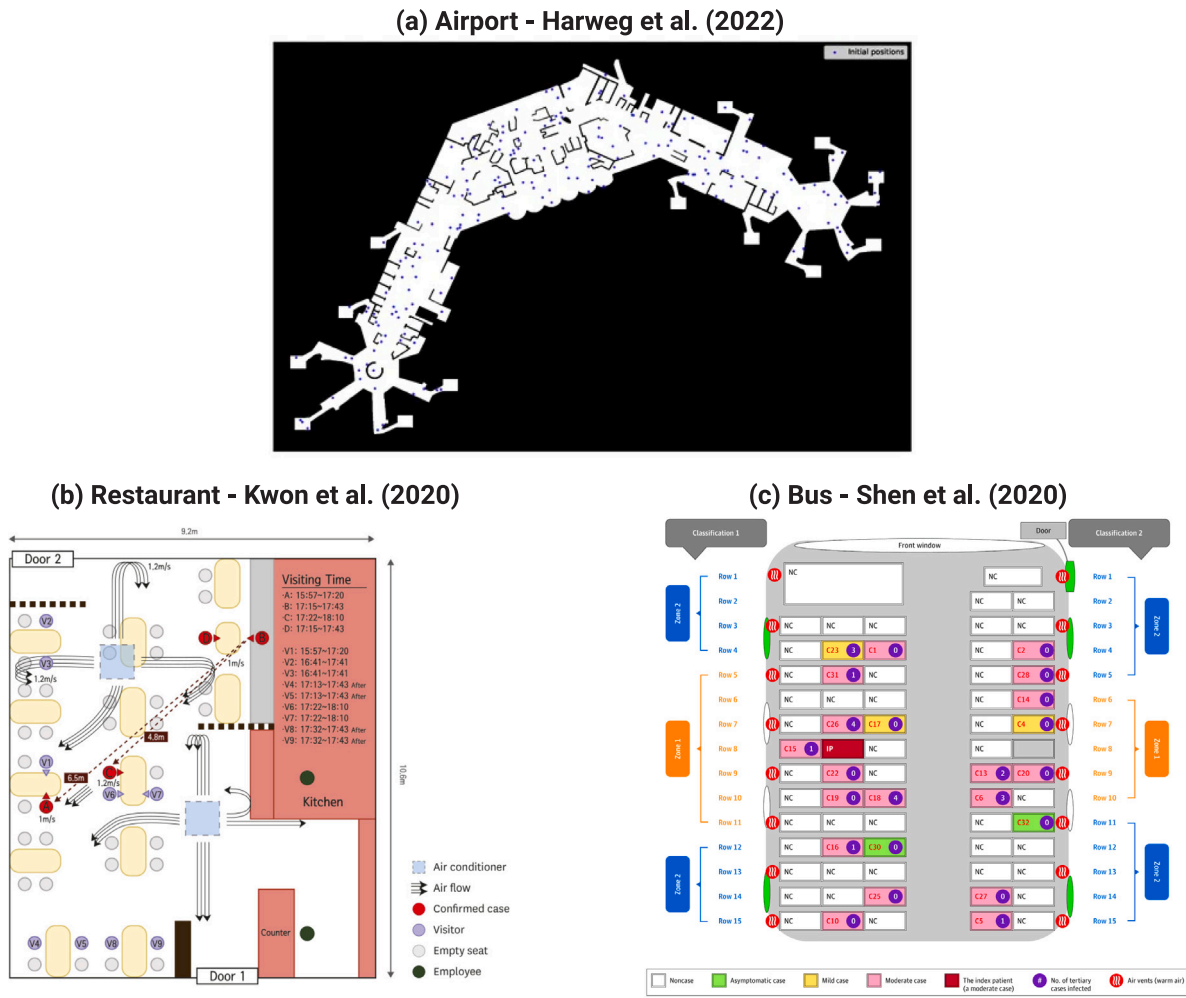
Table 2

Performance analysis for the synthetic cases divided into the graph search and WDA options. All models were trained for  $n_{train} = 900$  random cases and evaluated on  $n_{test} = 100$  random cases; the performance,  $|V|$ , and  $|E|$  are shown as the mean  $\pm$  standard deviation over the test set.

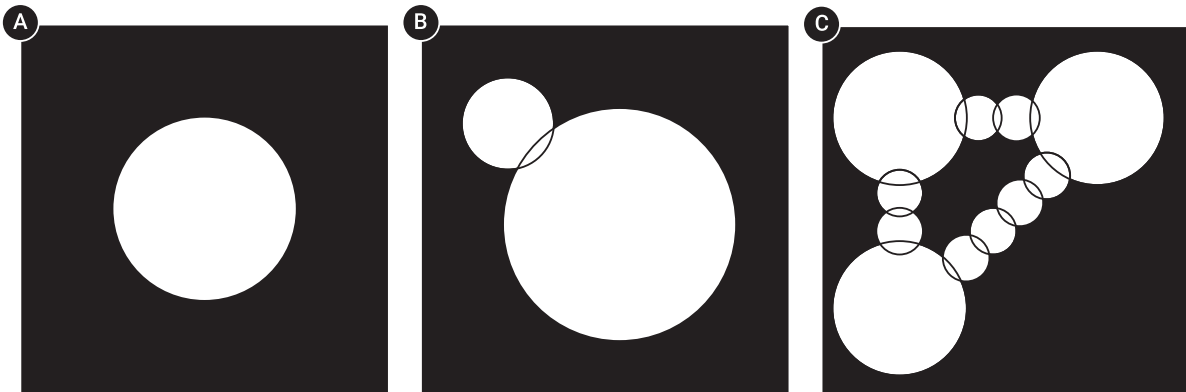
Graph search	WDA	Performance	Rel. time reduction [%]	$ V $	$ E $
Grid	MC	$0.70 \pm 0.07$	$41.43 \pm 7.31$	$145.32 \pm 70.15$	$3.72 \pm 0.05$
	MAC-AE	$0.69 \pm 0.08$	$52.37 \pm 10.46$	$138.91 \pm 65.44$	$3.80 \pm 0.05$
	DRL	$0.63 \pm 0.08$	$49.82 \pm 9.73$	$142.08 \pm 67.29$	$3.75 \pm 0.06$
	STGNN	$0.71 \pm 0.09$	$55.68 \pm 10.25$	$140.55 \pm 66.82$	$3.78 \pm 0.05$
Quadtree	MC	$0.73 \pm 0.06$	$68.29 \pm 13.47$	$72.14 \pm 39.14$	$5.14 \pm 3.88$
	MAC-AE	$0.76 \pm 0.05$	$70.84 \pm 12.15$	$65.38 \pm 35.15$	$5.48 \pm 3.26$
	DRL	$0.68 \pm 0.10$	$65.39 \pm 14.92$	$46.03 \pm 51.09$	$5.95 \pm 4.78$
	STGNN	$0.78 \pm 0.07$	$71.56 \pm 13.08$	$69.02 \pm 37.98$	$5.32 \pm 3.51$
GA	MC	$0.84 \pm 0.07$	$74.12 \pm 12.37$	$75.19 \pm 37.13$	$6.02 \pm 3.70$
	MAC-AE	$0.88 \pm 0.08$	$78.95 \pm 11.42$	$62.29 \pm 42.08$	$4.89 \pm 2.90$
	DRL	$0.82 \pm 0.07$	$73.58 \pm 12.16$	$67.58 \pm 40.37$	$5.12 \pm 3.23$
	STGNN	$0.87 \pm 0.08$	$79.63 \pm 11.07$	$64.15 \pm 39.62$	$5.05 \pm 3.10$
TSxM	MC	$0.84 \pm 0.07$	$82.13 \pm 11.28$	$68.14 \pm 40.05$	$4.19 \pm 3.05$
	MAC-AE	$0.90 \pm 0.09$	$87.52 \pm 10.63$	$77.08 \pm 34.40$	$4.65 \pm 3.15$
	DRL	$0.89 \pm 0.10$	$85.27 \pm 10.94$	$75.19 \pm 37.11$	$4.95 \pm 3.80$
	STGNN	$0.89 \pm 0.09$	$86.04 \pm 10.37$	$76.02 \pm 35.72$	$4.58 \pm 3.21$
RNN	MC	$0.83 \pm 0.05$	$81.16 \pm 10.95$	$81.06 \pm 43.72$	$4.22 \pm 2.96$
	MAC-AE	$0.85 \pm 0.06$	$84.31 \pm 10.72$	$83.71 \pm 41.18$	$4.08 \pm 3.53$
	DRL	$0.86 \pm 0.05$	$83.09 \pm 10.88$	$68.26 \pm 40.01$	$4.21 \pm 3.04$
	STGNN	$0.87 \pm 0.06$	$85.38 \pm 10.41$	$80.12 \pm 42.05$	$4.15 \pm 3.18$

The uniform Grid baseline, despite using the largest number of nodes, attains the lowest performance overall and relatively modest time savings. The Quadtree also performs worse than the learned graph-search methods, although it already yields substantial runtime reductions. In contrast, GA, TSxM, and RNN achieve clearly higher agreement with the norm-based dynamics (typically 0.84–0.90), with TSxM and RNN slightly outperforming GA on average. For the WDA, the simple MC baseline consistently underperforms the learned methods, while MAC-AE, DRL, and STGNN achieve comparable performance, with MAC-AE and STGNN often giving the best trade-off between accuracy and cost. Across methods, higher performance generally correlates with a moderate increase in  $|V|$  and  $|E|$ , but even the best-performing combinations (e.g., TSxM or RNN with MAC-AE or STGNN) still provide large relative time reductions, often above 80%.

Table 3 presents the performance of the best-performing combination of graph search and WDA for each real-world case. As in the synthetic setting, there is no single graph search or WDA algorithm that dominates across all three environments: the airport case is best captured by the combination of RNN with TSGNN, whereas both the restaurant and bus cases favor TSxM, paired with MAC-AE and DRL, respectively. Despite this heterogeneity, all three best combinations achieve high agreement with the norm-based dynamics, with average performance above 0.92 in every case. Importantly, these accuracies are obtained together with substantial computational gains, with relative time reductions between approximately 83% and 89%. The largest and most complex geometrical configuration, the airport, requires an order of magnitude more nodes ( $|V| = 182.28$ ) than the smaller restaurant and bus layouts ( $|V| = 18.13$  and  $|V| = 26.41$ , respectively). The



**Fig. 2.** A schematic view of the locations used for the experiments.  
Source: Images taken from Harweg et al. (2023), Shen et al. (2020), Kwon et al. (2020).



**Fig. 3.** Examples for (a) the simplest configuration; (b) configuration with different sub-locations with different densities; and (c) spatially-enforced walk patterns using the synthetic spatial component formalization.

average node degree is also slightly higher in the airport case ( $|E| \approx 4.4$ ) than in the restaurant ( $|E| = 2.89$ ) and bus ( $|E| = 3.77$ ), reflecting the greater diversity of movement paths and contact opportunities in that environment.

#### 4.3. Sensitivity analysis

While the results above do not reveal a single winner across all graph search and WDA combinations, TSxM with MAC-AE consistently



**Table 3**

Performance analysis for real-world cases. All 20 model combinations are trained on  $n_{train} = 900$  random initial conditions of each real-world case and evaluated on  $n_{test} = 100$  other random initial conditions such that the performance,  $|V|$ ,  $|E|$ , and the relative time reduction are shown as the mean  $\pm$  standard deviation of the test set for the best combination of graph search and WDA.

Case	Airport	Restaurant	Bus
Best graph search	RNN	TSxM	TSxM
Best WDA	TSGNN	MAC-AE	DRL
Performance	$0.94 \pm 0.04$	$0.95 \pm 0.04$	$0.92 \pm 0.02$
Rel. time reduction [%]	$86.47 \pm 8.12$	$83.35 \pm 8.74$	$89.02 \pm 9.95$
$ V $	$182.28 \pm 13.08$	$18.13 \pm 2.04$	$26.41 \pm 3.16$
$ E $	$4.43 \pm 1.35$	$2.89 \pm 0.66$	$3.77 \pm 0.34$

exhibits strong performance and favorable computational cost, on average. We therefore examine its sensitivity to several key temporal and spatial properties. First, we consider a circular spatial configuration with radius 50 m and vary the density by changing the population size from  $3.14 \cdot 10^2$  to  $3.14 \cdot 10^4$ . Second, we fix the radius of all circles at 2 m and vary the number of circles defining the geometrical configuration between 10 and 100. Third, we study the effect of different infection levels by varying the infection rate  $\beta$  from 0.037 to 0.37. Fourth, we vary the recovery duration  $\gamma$  between 60 and 380 h. Table 4 summarizes the results for  $n = 1000$  random cases, sampled uniformly over the specified ranges. The spatially related perturbations (first two rows) lead to slightly lower performance (0.94–0.95 with standard deviations 0.02–0.03) and more variable time reductions ( $\sim 66\%$ – $70\%$  with standard deviations around 13%–15%). By contrast, the temporally related perturbations (last two rows) maintain very high performance ( $\approx 0.97 \pm 0.01$ ) together with consistently large time savings (around 90% with small standard deviations  $\approx 2\%$ – $3\%$ ). Taken together, these results indicate that the proposed method is relatively robust to changes in temporal parameters, while being somewhat more sensitive to variations in the spatial configuration and density.

Fig. 4 visualizing the trade-off between the performance and reduction in computational time. Panel A shows, for the synthetic configurations, how the average relative time reduction increases as the performance of TSxM with MAC-AE decreases, together with a shaded band indicating one standard deviation across random cases while Panel B shows the same relationship for the three real-world environments, where each dashed line summarizes the empirical performance–cost frontier obtained by varying the graph-search and WDA hyperparameters. Overall, both panels highlight that substantial reductions in runtime (often above 80%) can be achieved while preserving a high level of agreement with the norm-based dynamics.

## 5. Discussion

In this study, we proposed a novel approach for translating norm-based spatial components to graph-based representations in the context of epidemiological modeling. Our approach balances computational efficiency with the preservation of norm-based dynamics, thereby facilitating more efficient simulations while maintaining analytical fidelity.

Our experimental results demonstrated the effectiveness of various algorithms for translating norm-based spatial components to graph-based representations. Through extensive experimentation with synthetic and real-world scenarios, we evaluated twelve combinations of graph search and WDA algorithms. The analysis revealed promising performance across different scenarios. Notably, the Time Series X-means (TSxM) algorithm combined with Multi-agent Classification with AutoEncoder (MAC-AE) for WDA consistently demonstrated robust performance across both synthetic and real-world cases.

Practically, the proposed approach has significant implications for public health policy and epidemic management. By enabling more

efficient simulations of epidemic spread in real-world environments such as airports, restaurants, and buses, our methodology can inform decision-making processes aimed at mitigating disease transmission (Shami and Lazebnik, 2022; Silva et al., 2023). Moreover, the ability to accurately model spatial dynamics in diverse settings enhances our understanding of epidemic dynamics and facilitates the development of targeted intervention strategies (Yu et al., 2023a; Orozco-Acosta et al., 2023).

The study's proposed methodology introduces promising advancements but comes with several limitations. Mainly, the proposed method is not validated against real-world data, leaving uncertainties about the accuracy and reliability of the proposed approach in predicting actual pandemic dynamics. However, this limitation is in the field of epidemiological modeling as the required data is usually scarce or even unavailable. Second, the proposed method is not interpretable nor explainable as it solves a multi-dimensional and complex task using a heuristic or ML approach, making the proposed approach a *black-box*. This fact may limit the ability of policymakers to put their faith in this solution by itself or as part of a larger epidemiological model (Yin et al., 2023; McDowell, 2023). Moreover, the study predominantly assumes a static environment throughout the spatio-temporal simulations, disregarding potential changes in the spatial landscape that could influence epidemic dynamics. Third, we assume fully observed, noise-free agent trajectories. Both the graph search and WDA stages are trained and evaluated on complete simulated paths, whereas in many practical deployments, only partially observed or noisy mobility traces are available (e.g., temporally sparse location pings, aggregated origin–destination matrices, or trajectories with missing segments). Importantly, the proposed framework does not fundamentally rely on exact continuous paths, but rather on the empirical co-location and flow patterns they induce. This makes it, in principle, compatible with partially observed data after appropriate pre-processing steps such as temporal subsampling, interpolation or smoothing of trajectories, or direct estimation of transition frequencies between candidate regions. A systematic robustness analysis across different observation regimes is a prerequisite for large-scale deployment in real-world surveillance settings. As such, future research should aim to refine the proposed solutions and address these limitations for a more comprehensive understanding of their practical implications. One particular promising future venue is to investigate multimodal data and its influence on the proposed model, as it has been shown to significantly influence pandemic spread dynamics (Tran et al., 2024; Lazebnik et al., 2022).

## CRedit authorship contribution statement

**Teddy Lazebnik:** Writing – review & editing, Writing – original draft, Visualization, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization.

## Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

## 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.

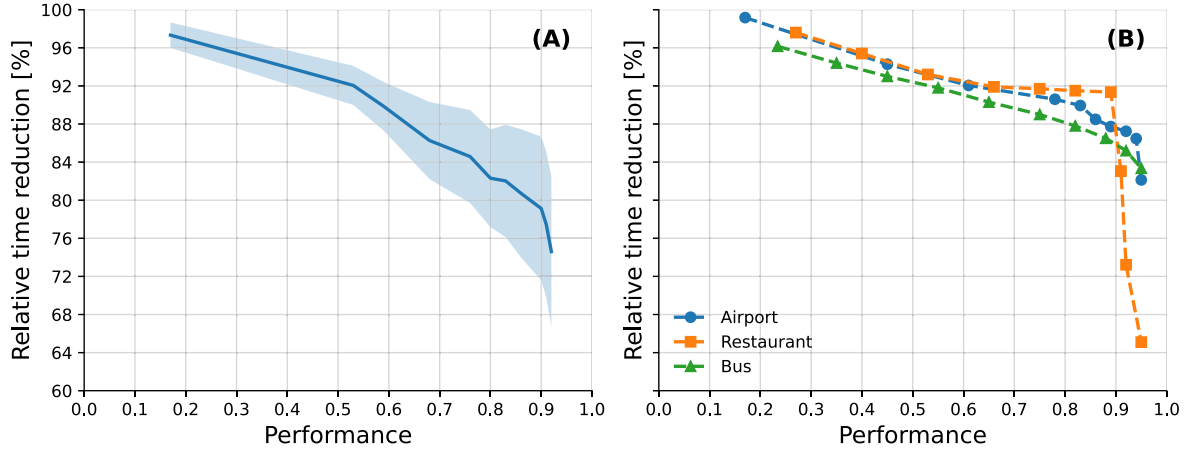
## Acknowledgment

The author wishes to thank Sveta Hardak-Nissan for teaching me the professional maturity required for fundamental research.

**Table 4**

Sensitivity analysis of the TSxM with MAC-AE to spatial and temporal changes. The results are shown as the mean  $\pm$  standard deviation of  $n = 1000$  random cases.

Parameter	Range	Performance	Rel. time reduction [%]
Population size ( $N$ )	314–31400	$0.95 \pm 0.02$	$69.73 \pm 13.58$
Number of circles defining the geometrical configuration	10–100	$0.94 \pm 0.03$	$66.15 \pm 14.97$
Infection rate ( $\beta$ )	0.037–0.37	$0.97 \pm 0.01$	$91.42 \pm 2.31$
Recovery duration in hours ( $\gamma$ )	60–380	$0.97 \pm 0.01$	$89.96 \pm 2.74$



**Fig. 4.** Trade-off between norm-based to graph-based performance and relative time reduction. Panel A shows the average relative time reduction as a function of performance for the synthetic configurations when using TSxM with MAC-AE; the shaded region denotes one standard deviation. Panel B presents the corresponding performance-cost profiles for the three real-world cases (airport, restaurant, and bus), where each dashed line is obtained by varying the underlying graph-search and WDA hyperparameters.

## Appendix

### Temporal component for epidemiological models

In this sub-section, the formal ordinary differential equation (ODE), as well as the agent-based simulation (ABS) representation, are provided for the three models used in the experiments.

#### SIR

For the SIR model, the ODE representation takes the form:

$$\begin{aligned} \frac{dS(t)}{dt} &= -\beta S(t)I(t), \\ \frac{dI(t)}{dt} &= \beta S(t)I(t) - \gamma I(t), \\ \frac{dR(t)}{dt} &= \gamma I(t), \end{aligned} \quad (5)$$

where  $S(t)$ ,  $I(t)$ , and  $R(t)$  are the number of susceptible, infected, and recovered agents in the population,  $\beta$  is the average infection rate, and  $\gamma$  is the average recovery rate.

For the ABS representation, let us assume each agent has an inner clock,  $\theta \in \mathbb{N}$ , that indicates the number of steps in time passed since the agent's epidemiological state changed last time. Moreover, the epidemiological state ( $\xi$ ) belongs to  $\xi \in \{S, I, R\}$ .

$$\begin{cases} \xi = S \times \xi = I \rightarrow \xi = I \times \xi = I, & \beta \\ \xi = I, \theta = \gamma \rightarrow \xi = R \wedge \theta = 0, & 1. \end{cases} \quad (6)$$

Any other case that is not specifically mentioned in Eq. (6), is the identity function with probability 1. Moreover, for all transactions,  $\theta = 0$ .

### SEIRD with two age groups

For the SEIRD with two age groups model, the ODE representation takes the form:

$$\begin{aligned} \frac{dS_c(t)}{dt} &= -(\beta_{cc}^s I_c^s(t) + \beta_{cc}^a I_c^a(t) + \beta_{ca}^s I_a^s(t) + \beta_{ca}^a I_a^a(t)) S_c(t), \\ \frac{dS_a(t)}{dt} &= -(\beta_{ac}^s I_c^s(t) + \beta_{ac}^a I_c^a(t) + \beta_{aa}^s I_a^s(t) + \beta_{aa}^a I_a^a(t)) S_a(t), \\ \frac{dI_c^s(t)}{dt} &= (1 - \psi_c)(\beta_{cc}^s I_c^s(t) + \beta_{cc}^a I_c^a(t) + \beta_{ca}^s I_a^s(t) + \beta_{ca}^a I_a^a(t)) S_c(t) - \gamma_c I_c^s(t), \\ \frac{dI_c^a(t)}{dt} &= \psi_c(\beta_{cc}^s I_c^s(t) + \beta_{cc}^a I_c^a(t) + \beta_{ca}^s I_a^s(t) + \beta_{ca}^a I_a^a(t)) S_c(t) - \gamma_c I_c^a(t), \\ \frac{dI_a^s(t)}{dt} &= \psi_a(\beta_{ac}^s I_c^s(t) + \beta_{ac}^a I_c^a(t) + \beta_{aa}^s I_a^s(t) + \beta_{aa}^a I_a^a(t)) S_a(t) - \gamma_a I_a^s(t), \\ \frac{dI_a^a(t)}{dt} &= (1 - \psi_a)(\beta_{ac}^s I_c^s(t) + \beta_{ac}^a I_c^a(t) + \beta_{aa}^s I_a^s(t) + \beta_{aa}^a I_a^a(t)) S_a(t) - \gamma_a I_a^a(t), \\ \frac{dR_c(t)}{dt} &= \gamma_c \rho_c (I_c^s(t) + I_c^a(t)), \\ \frac{dR_a(t)}{dt} &= \gamma_a \rho_a (I_a^s(t) + I_a^a(t)), \\ \frac{dD_c(t)}{dt} &= \gamma_c (1 - \rho_c) (I_c^s(t) + I_c^a(t)), \\ \frac{dD_a(t)}{dt} &= \gamma_a (1 - \rho_a) (I_a^s(t) + I_a^a(t)), \end{aligned} \quad (7)$$

where  $S(t)$ ,  $I^a(t)$ ,  $I^s(t)$ ,  $R(t)$ , and  $D(t)$  are the number of susceptible, asymptomatic infected, symptomatic infected, recovered, and dead agents in the population,  $\beta$  is the average infection rate,  $\gamma$  is the average recovery rate,  $\rho$  is the recovery probability.

For the ABS representation, let us assume each agent has an inner clock,  $\theta \in \mathbb{N}$ , that indicates the number of steps in time passed since the agent's epidemiological state changed last time and an age group,  $\zeta \in \{c, a\}$ . Moreover, the epidemiological state ( $\xi$ ) belongs to  $\xi \in$

$\{S, I^a, I^s, R, D\}$ .

$$\begin{aligned}
 \xi = S, \zeta = c \times \xi = I^a, \zeta = c \rightarrow \xi = I^a, \zeta = c \times \xi = I^a, \zeta = c, & \quad \psi_c \beta_{cc}^a \\
 \xi = S, \zeta = c \times \xi = I^a, \zeta = c \rightarrow \xi = I^s, \zeta = c \times \xi = I^a, \zeta = c, & \quad (1 - \psi_c) \beta_{cc}^a \\
 \xi = S, \zeta = c \times \xi = I^s, \zeta = c \rightarrow \xi = I^a, \zeta = c \times \xi = I^s, \zeta = c, & \quad \psi_c \beta_{cc}^s \\
 \xi = S, \zeta = c \times \xi = I^s, \zeta = c \rightarrow \xi = I^a, \zeta = c \times \xi = I^s, \zeta = c, & \quad (1 - \psi_c) \beta_{cc}^s \\
 \xi = S, \zeta = c \times \xi = I^a, \zeta = a \rightarrow \xi = I^a, \zeta = c \times \xi = I^a, \zeta = a, & \quad \psi_c \beta_{ca}^a \\
 \xi = S, \zeta = c \times \xi = I^a, \zeta = a \rightarrow \xi = I^s, \zeta = c \times \xi = I^a, \zeta = a, & \quad (1 - \psi_c) \beta_{ca}^a \\
 \xi = S, \zeta = c \times \xi = I^s, \zeta = a \rightarrow \xi = I^a, \zeta = c \times \xi = I^s, \zeta = a, & \quad \psi_c \beta_{ca}^s \\
 \xi = S, \zeta = c \times \xi = I^s, \zeta = a \rightarrow \xi = I^a, \zeta = c \times \xi = I^s, \zeta = a, & \quad (1 - \psi_c) \beta_{ca}^s \\
 \xi = S, \zeta = a \times \xi = I^a, \zeta = c \rightarrow \xi = I^a, \zeta = a \times \xi = I^a, \zeta = c, & \quad \psi_a \beta_{ac}^a \\
 \xi = S, \zeta = a \times \xi = I^a, \zeta = c \rightarrow \xi = I^s, \zeta = a \times \xi = I^a, \zeta = c, & \quad (1 - \psi_a) \beta_{ac}^a \\
 \xi = S, \zeta = a \times \xi = I^s, \zeta = c \rightarrow \xi = I^a, \zeta = a \times \xi = I^s, \zeta = c, & \quad \psi_a \beta_{ac}^s \\
 \xi = S, \zeta = a \times \xi = I^s, \zeta = c \rightarrow \xi = I^a, \zeta = a \times \xi = I^s, \zeta = c, & \quad (1 - \psi_a) \beta_{ac}^s \\
 \xi = S, \zeta = a \times \xi = I^a, \zeta = a \rightarrow \xi = I^a, \zeta = a \times \xi = I^a, \zeta = a, & \quad \psi_a \beta_{aa}^a \\
 \xi = S, \zeta = a \times \xi = I^a, \zeta = a \rightarrow \xi = I^s, \zeta = a \times \xi = I^a, \zeta = a, & \quad (1 - \psi_a) \beta_{aa}^a \\
 \xi = S, \zeta = a \times \xi = I^s, \zeta = a \rightarrow \xi = I^a, \zeta = a \times \xi = I^s, \zeta = a, & \quad \psi_a \beta_{aa}^s \\
 \xi = S, \zeta = a \times \xi = I^s, \zeta = a \rightarrow \xi = I^a, \zeta = a \times \xi = I^s, \zeta = a, & \quad (1 - \psi_a) \beta_{aa}^s \\
 \xi = I^s, \zeta = c, \theta = \gamma_c \rightarrow \xi = R & \quad \rho_c \\
 \xi = I^s, \zeta = a, \theta = \gamma_a \rightarrow \xi = R & \quad \rho_a \\
 \xi = I^s, \zeta = c, \theta = \gamma_c \rightarrow \xi = D & \quad 1 - \rho_c \\
 \xi = I^s, \zeta = a, \theta = \gamma_a \rightarrow \xi = D & \quad 1 - \rho_a \\
 \xi = I^a, \zeta = c, \theta = \gamma_c \rightarrow \xi = R & \quad 1 \\
 \xi = I^a, \zeta = a, \theta = \gamma_a \rightarrow \xi = R & \quad 1
 \end{aligned} \tag{8}$$

Any other case that is not specifically mentioned in Eq. (10), is the identical function with chance 1. Moreover, for all transactions,  $\theta = 0$ .

### Two-strain SIR

For the two-strain SIR model, the ODE representation takes the form:

$$\begin{aligned}
 \frac{dR_\emptyset I_1(t)}{dt} &= \beta_{\emptyset,1}(R_\emptyset I_1(t) + R_{\{2\}} I_1(t))R_\emptyset(t) - \gamma_{\emptyset,1} R_\emptyset I_1(t), \\
 \frac{dR_{\{2\}} I_1(t)}{dt} &= \beta_{\{2\},1}(R_{\{2\}} I_1(t) + R_\emptyset I_1(t))R_{\{2\}}(t) - \gamma_{\{2\},1} R_{\{2\}} I_1(t), \\
 \frac{dR_\emptyset I_2(t)}{dt} &= \beta_{\emptyset,2}(R_\emptyset I_2(t) + R_{\{1\}} I_2(t))R_\emptyset(t) - \gamma_{\emptyset,2} R_\emptyset I_2(t), \\
 \frac{dR_{\{1\}} I_2(t)}{dt} &= \beta_{\{1\},2}(R_{\{1\}} I_2(t) + R_\emptyset I_2(t))R_{\{1\}}(t) - \gamma_{\{1\},2} R_{\{1\}} I_2(t), \\
 \frac{dR_\emptyset(t)}{dt} &= -R_\emptyset(t)(\beta_{\emptyset,1}(R_\emptyset I_1(t) + R_{\{2\}} I_1(t)) + \beta_{\emptyset,2}(R_\emptyset I_2(t) + R_{\{1\}} I_2(t))), \\
 \frac{dR_{\{1\}}(t)}{dt} &= \gamma_{\emptyset,1} \rho_{\emptyset,1} R_\emptyset I_1(t) - \beta_{\{1\},2}(R_{\{1\}} I_2(t) + R_\emptyset I_2(t))R_{\{1\}}(t), \\
 \frac{dR_{\{2\}}(t)}{dt} &= \gamma_{\emptyset,2} \rho_{\emptyset,2} R_\emptyset I_2(t) - \beta_{\{2\},1}(R_{\{2\}} I_1(t) + R_\emptyset I_1(t))R_{\{2\}}(t), \\
 \frac{dR_{\{1,2\}}(t)}{dt} &= \gamma_{\{2\},1} \rho_{\{2\},1} R_{\{2\}} I_1(t) + \gamma_{\{1\},2} \rho_{\{1\},2} R_{\{1\}} I_2(t), \\
 \frac{dD(t)}{dt} &= \gamma_{\emptyset,1}(1 - \rho_{\emptyset,1})R_\emptyset I_1(t) + \gamma_{\{2\},1}(1 - \rho_{\{2\},1})R_{\{2\}} I_1(t) \\
 &\quad + \gamma_{\emptyset,2}(1 - \rho_{\emptyset,2})R_\emptyset I_2(t) + \gamma_{\{1\},2}(1 - \rho_{\{1\},2})R_{\{1\}} I_2(t).
 \end{aligned} \tag{9}$$

where  $R_\emptyset(t), R_{\{1\}}, R_{\{2\}}, R_{\{1,2\}}, R_\emptyset I_1(t), R_\emptyset I_2(t), R_{\{1\}} I_2, R_{\{2\}} I_1$ , and  $D(t)$  are the number of recovered from strains  $\emptyset, \{1\}, \{2\}, \{1, 2\}$ , infected with the first and second strains, and dead agents in the population,  $\beta$  is the average infection rate,  $\gamma$  is the average recovery rate,  $\psi$  is the recovery probability.

For the ABS representation, let us assume each agent has an inner clock,  $\theta \in \mathbb{N}$ , that indicates the number of steps in time passed since the agent's epidemiological state changed last time and an age group. Moreover, the epidemiological state ( $\xi$ ) belongs to  $\xi \in \{R_\emptyset(t), R_{\{1\}}, R_{\{2\}},$

$R_{\{1,2\}}, R_\emptyset I_1(t), R_\emptyset I_2(t), R_{\{1\}} I_2, R_{\{2\}} I_1, D\}$ .

$$\begin{aligned}
 \xi = \{\emptyset, 1\} \times \theta = \gamma_{\emptyset,1} \rightarrow \xi = D, & \quad 1 - \rho_{\emptyset,1} \\
 \xi = \{\emptyset, 2\} \times \theta = \gamma_{\emptyset,2} \rightarrow \xi = D, & \quad 1 - \rho_{\emptyset,2} \\
 \xi = \{\{1\}, 2\} \times \theta = \gamma_{\{1\},2} \rightarrow \xi = D, & \quad 1 - \rho_{\{1\},2} \\
 \xi = \{\{2\}, 1\} \times \theta = \gamma_{\{2\},1} \rightarrow \xi = D, & \quad 1 - \rho_{\{2\},1} \\
 \xi = \{\emptyset, 1\} \times \theta = \gamma_{\emptyset,1} \rightarrow \xi = \{\{1\}, \emptyset\}, & \quad \rho_{\emptyset,1} \\
 \xi = \{\emptyset, 2\} \times \theta = \gamma_{\emptyset,2} \rightarrow \xi = \{\{2\}, \emptyset\}, & \quad \rho_{\emptyset,2} \\
 \xi = \{\{1\}, 2\} \times \theta = \gamma_{\{1\},2} \rightarrow \xi = \{\{1, 2\}, \emptyset\}, & \quad \rho_{\{1\},2} \\
 \xi = \{\{2\}, 1\} \times \theta = \gamma_{\{2\},1} \rightarrow \xi = \{\{1, 2\}, \emptyset\}, & \quad \rho_{\{2\},1} \\
 \xi = \{\emptyset, \emptyset\} \times \xi = \{\emptyset, 1\} \rightarrow \xi = \{\emptyset, 1\} \times \xi = \{\emptyset, 1\}, & \quad \beta_{\emptyset,1} \\
 \xi = \{\emptyset, \emptyset\} \times \xi = \{\emptyset, 2\} \rightarrow \xi = \{\emptyset, 2\} \times \xi = \{\emptyset, 2\}, & \quad \beta_{\emptyset,2} \\
 \xi = \{\{1\}, \emptyset\} \times \xi = \{\emptyset, 2\} \rightarrow \xi = \{\{1\}, 2\} \times \xi = \{\emptyset, 2\}, & \quad \beta_{\{1\},2} \\
 \xi = \{\{2\}, \emptyset\} \times \xi = \{\emptyset, 1\} \rightarrow \xi = \{\{2\}, 1\} \times \xi = \{\emptyset, 1\}, & \quad \beta_{\{2\},1}
 \end{aligned} \tag{10}$$

Any other case that is not specifically mentioned in Eq. (10), is the identical function with chance 1. Moreover, for all transactions,  $\theta = 0$ .

### Description of the algorithms' implementation

In this section, we provide implementation details for all algorithms used in the norm-to-graph transformation framework. Recall that the norm-based agent-based simulator defines a walk operator  $I_w^{\text{norm}} : (p, P, E) \rightarrow (p, E)$  acting on an individual  $p$ , the full population  $P$ , and the spatial environment  $E$  in continuous space  $L \subset \mathbb{R}^k$  (Section 2.2). In the graph-based simulator, the environment is a graph  $G = (V, E)$  and the walk operator  $I_w^{\text{graph}}$  moves agents between graph nodes according to a learned transition policy (Section 2.3). All algorithms below are trained on trajectories generated by  $I_w^{\text{norm}}$  and are used either to construct the graph  $G$  (graph search algorithms) or to approximate  $I_w^{\text{norm}}$  by a graph-based walk operator  $I_w^{\text{graph}}$  (walk dynamics approximation algorithms). Throughout, the norm-based simulator produces agent locations  $\ell_p(t) \in L$  and epidemiological states  $\xi_p(t)$  for  $p \in P$  and discrete time indices  $t = 0, \dots, T$ . Graph search algorithms consume only the locations  $\{\ell_p(t)\}$ , whereas walk dynamics approximation algorithms use both the graph  $G$  and the discrete node assignments  $v_p(t) \in V$  induced by the chosen graph search method.

### Integration into the framework

Given a fixed norm-based model and initial condition, we first run the norm-based ABS and record the continuous trajectories  $\{\ell_p(t)\}_{p,t}$  for all agents. A graph search algorithm then constructs a finite graph  $G = (V, E)$  together with a mapping  $c : L \rightarrow V$  that associates every continuous location  $\ell$  with a node  $v = c(\ell)$ . For each recorded trajectory, we obtain a discrete node sequence  $v_p(t) = c(\ell_p(t))$ . These discrete trajectories, together with the corresponding epidemiological states  $\xi_p(t)$  and aggregate population statistics (e.g., counts of susceptible, infected, recovered agents per node), form the training data for the walk dynamics approximation algorithms. A walk dynamics model learns, from these data, a probabilistic transition rule of the form

$$\pi_\theta(v' | s_t(p)) \approx \mathbb{P}(v_p(t+1) = v' | s_t(p)),$$

where  $s_t(p)$  is a state vector summarizing the information available to the agent at time  $t$ . The learned transition probabilities define the graph-based walk operator  $I_w^{\text{graph}}$  used in Algorithm 1: at each time step and for each agent  $p$  located at node  $v_p(t)$ , we sample a new node  $v_p(t+1)$  from  $\pi_\theta(\cdot | s_t(p))$  and update the agent's location accordingly. The infection and temporal operators are kept identical to the original norm-based model; only the spatial part of  $I_w$  is replaced.

### Graph search algorithms

**Uniform square grid.** The uniform square grid provides the simplest graph discretization of the continuous environment. Let  $[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$  be the axis-aligned bounding box of all observed locations

$\{\ell_p(t)\}$  in two dimensions. We overlay this domain with an axis-aligned square grid of cell side length  $h > 0$ . Each grid cell that is visited by at least one agent at any time step becomes a node  $v \in V$  whose spatial coordinate is taken as the cell's center of mass. Two nodes are connected by an edge  $(v, u) \in E$  if the corresponding cells share a side (4-neighborhood). The mapping  $c : L \rightarrow V$  assigns each continuous location  $\ell$  to the unique cell that contains it, and therefore to the corresponding grid node. This construction yields a mesh-like discretization where the number of nodes  $|V|$  is controlled primarily by the grid spacing  $h$ , and the average node degree  $|E|/|V|$  is close to four. The resulting graph  $G$  and mapping  $c$  are subsequently used to generate training data for the walk dynamics approximation algorithms and to define  $I_w^{\text{graph}}$ .

**Quadtree.** The Quadtree graph search algorithm constructs an adaptive spatial discretization that refines the grid where the agent density is high and keeps it coarse elsewhere. We start from the same bounding box as for the grid and recursively subdivide each square cell into four equal subcells until either the maximum tree depth  $d_{\max}$  is reached or the number of agents contained in the cell at a given time step falls below a threshold  $n_{\text{leaf}}$ . For each time step  $t$ , this procedure yields a Quadtree  $Q_t$  whose leaves partition the domain into square cells of varying size. To obtain a single graph  $G$  over the entire simulation horizon, we construct an “average Quadtree” by aligning all Quadtrees to the same maximal depth and counting how often each leaf cell at that depth is occupied. Formally, for each potential leaf cell  $c$  at depth  $d_{\max}$ , we define an occupancy frequency

$$\phi(c) = \frac{1}{T+1} \sum_{t=0}^T \mathbf{1}\{\exists p \in P : \ell_p(t) \in c\},$$

and keep only those cells whose occupancy  $\phi(c)$  exceeds a small threshold  $n_{\min}/(T+1)$ . Each retained cell becomes a node  $v \in V$  whose coordinate is the geometric center of the cell. Two nodes are connected by an edge if their corresponding cells share a side. The mapping  $c(\ell)$  assigns each location to the unique retained cell that contains it (or to the nearest retained cell if  $\ell$  falls into a pruned region). This adaptive discretization tends to produce more nodes in crowded regions and fewer nodes elsewhere, with relatively heterogeneous degree distribution, and is again used only to define  $G$  and the mapping  $c$ ;  $I_w^{\text{graph}}$  is entirely determined by the WDA algorithms.

**Genetic algorithm (GA).** The GA-based graph search treats the node locations as a continuous optimization variable and searches over candidate graphs by minimizing the proxy loss in Eq. (3). A candidate solution (an individual in the GA population) is a vector of  $K$  node coordinates  $(x_1, \dots, x_K) \in L^K$ , where  $K$  is implicitly controlled by the encoding and mutation operators. Given a candidate graph  $G_m$  defined by these nodes, we connect nodes using a simple proximity rule: for each node  $v$ , we connect it to its  $k_e$  nearest neighbors by undirected edges, which results in a sparse, locally connected graph. The fitness of  $G_m$  is measured by the loss  $L(m)$  in Eq. (3), which averages, over all agents and time steps, the Euclidean distance between each continuous location  $\ell_p(t)$  and its closest node in the candidate graph. A standard generational GA is used to minimize  $L(m)$ : at each generation, a population of  $N_{\text{pop}}$  graphs is updated by tournament selection (parameterized by a fraction  $\alpha$  of the best individuals), one-point or uniform crossover with probability  $p_{\text{cross}}$ , and Gaussian mutation of node coordinates with standard deviation  $\sigma_{\text{mut}}$  and probability  $p_{\text{mut}}$ . After a fixed number of generations  $G_{\text{gen}}$ , the best-performing candidate graph is selected, and its nodes and edges define  $G$ , while the mapping  $c(\ell)$  is taken as nearest-neighbor assignment to the learned node coordinates. The GA itself does not define  $I_w^{\text{graph}}$ ; it only produces a graph  $G$  that is later used by the WDA algorithms.

**Time-series X-means (TSxM).** The TSxM graph search algorithm constructs  $G$  by clustering entire agent trajectories rather than individual locations. Each agent  $p$  is represented by its discrete-time trajectory

$\{\ell_p(t)\}_{t=0}^T$ , treated as a multivariate time series. We use dynamic time warping (DTW) as the distance between trajectories and perform DTW  $k$ -means clustering for a range of cluster counts  $k = 1, \dots, \epsilon$ . For each  $k$ , we run the DTW  $k$ -means algorithm  $a$  times with different random initializations and keep the solution that minimizes an  $L_1$ -type trajectory reconstruction error. For each resulting clustering, we compute the proxy loss in Eq. (3) by assigning each trajectory point to the centroid of its cluster and measuring its distance to that centroid's spatial center of mass. The number of clusters  $k^*$  is then selected by a simple elbow-like rule that balances this loss against model complexity or, in the implementation used here, by directly choosing the  $k$  that yields the smallest value of  $L(m)$  among  $k = 1, \dots, \epsilon$ . Each cluster  $c$  defines a node  $v_c \in V$  whose spatial coordinate is the average of all  $\ell_p(t)$  that belong to that cluster. Edges between nodes are induced by observed temporal transitions: we add an edge  $(v_c, v_{c'})$  if there exists an agent  $p$  and time  $t$  such that  $\ell_p(t)$  is assigned to cluster  $c$  and  $\ell_p(t+1)$  to cluster  $c'$ . The mapping  $c(\ell)$  assigns each location to the cluster centroid that is closest under DTW distance. In this way, TSxM constructs  $G$  and  $c$  by capturing typical movement patterns;  $I_w^{\text{graph}}$  is subsequently learned from the induced discrete trajectories.

**Recurrent neural network (RNN) graph search.** The RNN-based graph search algorithm learns the node locations directly from the trajectories via a sequence model trained with the proxy loss in Eq. (3). At each time step  $t$ , the continuous agent locations  $\{\ell_p(t)\}_{p \in P}$  are embedded into a fixed-length vector representation by an order-invariant encoder (for example, an average or attention-based pooling layer) that produces a population-level vector  $z_t \in \mathbb{R}^{d_h}$ . The sequence  $\{z_t\}_{t=0}^T$  is then fed into a stacked GRU-based recurrent network with  $L_{\text{RNN}}$  layers and hidden dimension  $d_h$ . The final hidden state is passed through a small fully connected network to output  $K$  node coordinates  $\hat{x}_1, \dots, \hat{x}_K$ , where  $K$  is fixed a priori based on the desired graph resolution. During training, we construct the candidate graph  $G_m$  using these coordinates and a proximity-based edge rule (as in the GA case) and minimize Eq. (3) with respect to the RNN parameters using the Adam optimizer. After convergence, the learned node coordinates define  $V$ , edges are added by  $k$ -nearest neighbors, and the mapping  $c(\ell)$  again assigns each location to its nearest node. This method leverages temporal correlations in the population-level dynamics to produce a compact and data-adaptive graph structure that is subsequently used by the WDA algorithms to define  $I_w^{\text{graph}}$ .

#### Walk dynamics approximation algorithms

**Markov chain (MC).** The MC-based walk dynamics approximation models  $I_w^{\text{graph}}$  as a first-order Markov chain on the graph whose state summarizes the epidemiological and local spatial context. For each agent  $p$  and time  $t$ , we define a discrete state

$$s_t(p) = (v_p(t), \xi_p(t), C_{v_p(t)}(t)),$$

where  $v_p(t) \in V$  is the node occupied by  $p$ ,  $\xi_p(t)$  is its epidemiological state, and  $C_v(t)$  is a vector of counts of agents in each epidemiological state across  $v$  and its neighbors  $\{u : (v, u) \in E\}$ . We then estimate, from the discrete trajectories, the empirical transition probabilities

$$\hat{P}(v' | s) = \frac{N(s \rightarrow v') + \alpha_{\text{MC}}}{\sum_{u \in V} N(s \rightarrow u) + \alpha_{\text{MC}}|V|},$$

where  $N(s \rightarrow v')$  is the number of observed transitions from state  $s$  to node  $v'$  and  $\alpha_{\text{MC}} > 0$  is a Laplace smoothing parameter. The walk operator  $I_w^{\text{graph}}$  is then defined by this Markov chain: at simulation time, for an agent with current state  $s_t(p)$ , the next node is sampled from the categorical distribution  $\hat{P}(\cdot | s_t(p))$ , and the agent's location component in  $\tau$  is updated accordingly. This approach is purely tabular and does not involve any trainable neural network parameters; its hyperparameters are the definition of the context vector  $C_v(t)$  and the smoothing constant  $\alpha_{\text{MC}}$ .



**Multi-agent classification with AutoEncoder (MAC-AE).** The MAC-AE algorithm learns a set of node-specific classifiers that map the current local and global state to a distribution over neighboring nodes. For each time step  $t$ , we construct a population-level representation by feeding the features of all agents (node index, epidemiological state, and possibly additional attributes) into a fully connected attention-based autoencoder. The encoder part of this autoencoder outputs a fixed-length embedding  $z_t \in \mathbb{R}^{d_z}$  that summarizes the current configuration of the entire population. For an agent  $p$  at node  $v_p(t) = v$ , we then build a feature vector  $x_t(p)$  by concatenating its local features (e.g., its epidemiological state  $\xi_p(t)$  and node index  $v$ ), relevant aggregate counts in its neighborhood, and the global embedding  $z_t$ . For each node  $v \in V$ , we train a separate supervised classifier  $f_v$  that receives  $x_t(p)$  and predicts the node index  $v_p(t+1)$  at the next time step. To avoid hand-tuning the classifier architecture and hyperparameters, we use the TPOT AutoML library to search over a space of pipelines comprising feature preprocessing, model families, and their hyperparameters, with population size  $N_{\text{TPOT}}$  and number of generations  $G_{\text{TPOT}}$ . The training objective is a cross-entropy loss between the predicted distribution over nodes and the true next node  $v_p(t+1)$ . At simulation time, the graph-based walk operator  $I_w^{\text{graph}}$  is defined by these classifiers: given the current state  $s_t(p)$ , we compute  $x_t(p)$ , query the corresponding classifier  $f_{v_p(t)}$ , and either sample the next node from the predicted categorical distribution or take its argmax. The hyperparameters for MAC-AE comprise the latent dimension  $d_z$  of the autoencoder, its architecture and training epochs, and the TPOT search parameters.

**Deep reinforcement learning (DRL).** The DRL-based approach treats the walk-dynamics learning problem as an imitation-style reinforcement learning task on the graph. We define, for each fixed graph  $G = (V, E)$ , an episodic Markov decision process in which all agents share a single stochastic policy  $\pi_\theta$ , parameterized by a fully connected neural network. The state space  $S$  is continuous and consists of feature vectors  $s_t(p)$  constructed similarly to the MAC-AE method: for each agent  $p$  at time  $t$  we encode its current node as a one-hot vector  $e_{v_p(t)} \in \mathbb{R}^{|V|}$ , its epidemiological state  $\xi_p(t)$  as a one-hot vector over compartments, its local neighborhood configuration as a vector of counts  $C_{v_p(t)}(t)$  of agents in each compartment in  $v_p(t)$  and its neighbors  $\mathcal{N}(v_p(t))$ , and a global embedding  $z_t \in \mathbb{R}^{d_z}$  of the entire population obtained from the autoencoder described in MAC-AE. These components are concatenated, normalized (zero mean and unit variance per feature), and form the state vector  $s_t(p) = [e_{v_p(t)}, \xi_p(t), C_{v_p(t)}(t), z_t]$  fed into the policy.

The action space  $\mathcal{A}(v)$  for an agent located at node  $v$  is the finite set of indices corresponding to the neighbors of  $v$  plus a “stay” action, i.e.,  $\mathcal{A}(v) = \mathcal{N}(v) \cup \{v\}$ , where  $\mathcal{N}(v) = \{u : (v, u) \in E\}$ . In practice, we implement this by building, for each node  $v$ , a fixed ordering of its neighbors and padding with a dummy “stay” index when necessary; the policy network outputs a categorical distribution over this local action set via a softmax layer. Given  $s_t(p)$ , the policy computes logits  $g_\theta(s_t(p)) \in \mathbb{R}^{|\mathcal{A}(v_p(t))|}$  and the action  $a_t(p)$  is sampled from  $\pi_\theta(\cdot | s_t(p)) = \text{softmax}(g_\theta(s_t(p)))$  during training; at evaluation time, we use the greedy action  $\arg \max_a \pi_\theta(a | s_t(p))$ .

To define the reward function, we leverage the norm-based trajectories as expert demonstrations. For each agent  $p$  and time step  $t$  in the recorded trajectories, we know the “expert” next node  $v_p(t+1)$  induced by applying the mapping  $c$  to the continuous norm-based location. We then define a per-step imitation reward

$$r_t(p) = \mathbf{1}\{a_t(p) = v_p(t+1)\} - \lambda \mathbf{1}\{a_t(p) \neq v_p(t+1)\},$$

where  $a_t(p)$  is the node selected by the policy at time  $t$  and  $\lambda \geq 0$  is a penalty parameter (controlling how strongly deviations from the expert are discouraged; in our experiments we set  $\lambda = 0.1$ ). Intuitively, this reward encourages the policy to choose actions that match the expert next-node assignments while mildly penalizing exploratory deviations.

Episodes are finite-horizon sequences obtained from the norm-based simulation. For each training episode, we sample an agent  $p$  and a starting time index  $\tau$  from the dataset of recorded trajectories and roll out the policy for a fixed horizon  $H_{\text{ep}}$  steps (or until the end of the trajectory), using the expert transitions to update the state from  $s_t(p)$  to  $s_{t+1}(p)$  while giving the policy credit or penalty via  $r_t(p)$ . In the experiments we use a discount factor of  $\gamma_{\text{RL}} = 0.99$  and an episode length of  $H_{\text{ep}} = 32$  steps. Rollouts are batched across multiple agents and starting times: each PPO update uses a batch of  $B = 8192$  transitions, obtained by concatenating several episodes until the batch size is reached.

The policy  $\pi_\theta$  and the value function  $V_\phi$  are parameterized by separate multilayer perceptrons (MLPs) with the same architecture: two hidden layers of width  $d_p = 128$  with ReLU activations, followed by either a softmax output over actions (policy) or a linear scalar output (value). We train both networks using the Proximal Policy Optimization (PPO) algorithm, which maximizes the clipped surrogate objective

$$L^{\text{PPO}}(\theta) = \mathbb{E} \left[ \min(\rho_t(\theta) \hat{A}_t, \text{clip}(\rho_t(\theta), 1 - \epsilon_{\text{PPO}}, 1 + \epsilon_{\text{PPO}}) \hat{A}_t) \right],$$

where  $\rho_t(\theta) = \pi_\theta(a_t | s_t) / \pi_{\theta_{\text{old}}}(a_t | s_t)$  is the importance sampling ratio,  $\hat{A}_t$  is a generalized advantage estimate computed from the value network with  $\lambda_{\text{GAE}} = 0.95$ , and  $\epsilon_{\text{PPO}} = 0.2$  is the clipping parameter. Each PPO iteration runs  $N_{\text{epochs}} = 4$  epochs of minibatch stochastic gradient ascent with the Adam optimizer (learning rate  $\eta = 10^{-3}$ , minibatch size  $M = 512$ ) on the collected batch of  $B$  transitions.

Training proceeds for a maximum of  $N_{\text{iter}}$  PPO iterations (we use  $N_{\text{iter}} = 200$ ) or until an early stopping criterion is met. For early stopping, we hold out a validation set of trajectories from the norm-based simulation and periodically measure the imitation accuracy, defined as the fraction of time steps for which the greedy policy's action  $\arg \max_a \pi_\theta(a | s_t(p))$  matches the expert next node  $v_p(t+1)$ . If this validation accuracy does not improve for  $N_{\text{patience}} = 10$  consecutive evaluations, we stop training and keep the policy parameters from the best validation epoch. After training, the learned policy  $\pi_\theta(a | s)$  defines the graph-based walk operator  $I_w^{\text{graph}}$ : in the translated simulation, for each agent and time step we build the state  $s_t(p)$  from the current graph-based population, sample or select the next node from  $\pi_\theta(\cdot | s_t(p))$ , and update the agent's location in  $G$ , while the infection and temporal operators remain identical to those of the original norm-based model.

**Spatio-temporal graph neural network (STGNN).** The STGNN-based walk dynamics approximation adapts recent spatio-temporal graph neural network architectures for epidemic prediction to our local transition-learning setting. Inspired by the Temporal Multiresolution Graph Neural Network (TMGNN) and transfer GNN models for pandemic forecasting, the STGNN operates on sequences of graph snapshots and learns to predict node-level flows between neighbors over time. At each time step  $t$ , we construct node features  $\phi_v(t)$  consisting of the counts of agents at node  $v$  in each epidemiological state, possibly concatenated with static node attributes such as spatial coordinates. We then build a temporal window of length  $H$  for each node,  $(\phi_v(t-H+1), \dots, \phi_v(t))$ , and feed the sequence of graph snapshots  $\{(\phi_v(\tau))_{v \in V}\}_{\tau=t-H+1}^t$  into a multiresolution GNN encoder. Following Hy et al. and Panagopoulos et al. the encoder stacks several message-passing GNN layers at multiple spatial resolutions and couples them with temporal recurrence or attention to capture both local and long-range dynamics. Its output is a collection of node-level embeddings  $h_v(t)$ , which are then passed through a decoder that outputs, for each node  $v$ , a vector of non-negative flows  $f_{v \rightarrow u}(t)$  for all  $u \in \mathcal{N}(v) \cup \{v\}$ .

To train the model, we derive, from the discrete trajectories, the empirical normalized flows

$$\hat{q}_{v \rightarrow u}(t) = \begin{cases} \frac{\#\{p: v_p(t)=v, v_p(t+1)=u\}}{\#\{p: v_p(t)=v\}}, & \text{if } \#\{p: v_p(t)=v\} > 0, \\ 0, & \text{otherwise,} \end{cases}$$

**Table 5**

Summary of the main hyperparameters for all graph search and walk dynamics approximation algorithms.

Hyperparameter	Short description	Default value
Quadtree: $d_{\max}$	Maximum depth of the Quadtree	$d_{\max} = 16$
Quadtree: $n_{\text{leaf}}$	Max. agents per cell before stopping subdivision	$n_{\text{leaf}} = 5$
GA: $N_{\text{pop}}$	Population size (number of candidate graphs)	$N_{\text{pop}} = 100$
GA: $G_{\text{gen}}$	Number of GA generations	$G_{\text{gen}} = 50$
GA: $p_{\text{mut}}$	Mutation probability for each individual	$p_{\text{mut}} = 0.05$
GA: $p_{\text{cross}}$	Crossover probability for selected parents	$p_{\text{cross}} = 0.1$
TSxM: $\epsilon$	Maximum number of clusters considered	$\epsilon = 200$
TSxM: $a$	Number of DTW $k$ -means restarts per $k$	$a = 10$
RNN: $d_h$	GRU hidden dimension for population encoder	$d_h = 128$
RNN: $L_{\text{RNN}}$	Number of stacked GRU layers	$L_{\text{RNN}} = 3$
MC: $\alpha_{\text{MC}}$	Laplace smoothing added to transition counts	$\alpha_{\text{MC}} = 10^{-3}$
MAC-AE: $d_z$	Latent dimension of population autoencoder	$d_z = 128$
MAC-AE: $G_{\text{TPOT}}$	TPOT generations for node-wise classifier search	$G_{\text{TPOT}} = 20$
MAC-AE: $N_{\text{TPOT}}$	TPOT population size per node	$N_{\text{TPOT}} = 50$
DRL (PPO): $\gamma_{\text{RL}}$	Discount factor for cumulative reward	$\gamma_{\text{RL}} = 0.98$
DRL (PPO): $\epsilon_{\text{PPO}}$	Clipping parameter in PPO objective	$\epsilon_{\text{PPO}} = 0.20$
DRL (PPO): $d_p$	Hidden dimension of policy/value networks	$d_p = 128$
STGNN: $d_{\text{ST}}$	Node embedding dimension in STGNN	$d_{\text{ST}} = 128$
STGNN: $H$	Temporal window length (number of past steps)	$H = 8$
STGNN: $L_{\text{ST}}$	Number of spatial resolutions (including finest)	$L_{\text{ST}} = 4$
STGNN: $E_{\text{ST}}$	Number of STGNN training epochs	$E_{\text{ST}} = 100$
Neural models: $\eta$	Adam learning rate (RNN, AE, DRL, STGNN)	$\eta = 10^{-3}$

which represent the fraction of agents at node  $v$  that move to neighbor  $u$  at the next time step. The STGNN is trained to minimize the discrepancy between the predicted probabilities  $q_{v \rightarrow u}(t) = \text{softmax}_u(f_{v \rightarrow u}(t))$  and the empirical  $\hat{q}_{v \rightarrow u}(t)$ , for example using a cross-entropy or Kullback–Leibler loss aggregated over nodes and time. After training, the walk operator  $I_{\text{walk}}^{\text{graph}}$  uses these predicted flow distributions: when simulating the graph-based model, for each node  $v$  and time  $t$  we compute  $q_{v \rightarrow u}(t)$ , and for every agent currently at  $v$  we sample its next node  $v_p(t+1)$  from the categorical distribution over  $\mathcal{N}(v) \cup \{v\}$  given by  $q_{v \rightarrow \cdot}(t)$ . Core hyperparameters include the node embedding dimension  $d_{\text{ST}}$ , the number of resolutions  $L_{\text{ST}}$ , the number of GNN layers per resolution, the temporal window length  $H$ , the learning rate, and the number of training epochs.

Table 5 summarizes the main hyperparameters used across all algorithms. When possible, we adopt the default values recommended by the corresponding Python libraries; for custom components (e.g., grid spacing, Quadtree thresholds) we select values that provide a reasonable trade-off between accuracy and computational cost.

## Data availability

No data was used for the research described in the article.

## References

- Acemoglu, D., Chernozhukov, V., Werning, I., Whinston, M.D., 2021. Optimal targeted lockdowns in a multigroup sir model. *Am. Econ. Rev.: Insights* 3 (4), 487–502.
- Adiga, A., Dubhashi, D., Lewis, B., Marathe, M., Venkatramanan, S., Vollikanti, A., 2020. Mathematical models for covid-19 pandemic: A comparative analysis. *J. Indian Inst. Sci.* 100 (4), 793–807.
- Agarwal, P., Jhajharia, K., 2021. Data analysis and modeling of covid-19. *J. Stat. Manag. Syst.* 24 (1), 1.
- Al-Raei, M., 2020. The forecasting of covid-19 with mortality using SIRD epidemic model for the united states, russia, china, and the syrian arab republic. *AUO Adv.* 10 (6).
- Alagar, V.S., Periyasamy, K., 2011. *Extended Finite State Machine*. Springer London, pp. 105–128.
- Alexi, A., Lazebnik, T., Shami, L., 2023a. Microfounded tax revenue forecast model with heterogeneous population and genetic algorithm approach. *Comput. Econ.*
- Alexi, A., Rosenfeld, A., Lazebnik, T., 2023b. A security games inspired approach for distributed control of pandemic spread. *Adv. Theory Simulations* 6 (2), 2200631.
- Andraud, M., Hens, N., Marais, C., Beutels, P., 2012. Dynamic epidemiological models for dengue transmission: A systematic review of structural approaches. *Plos One* 7 (11), 1–14, 11.
- Andrea Alberto Conti, 2020. Historical and methodological highlights of quarantine measures: from ancient plague epidemics to current coronavirus disease (COVID-19) pandemic. *Acta Bio-Medica : Atenei Parm.* 91 (2), 226–229.
- Arulkumaran, K., Deisenroth, M.P., Brundage, M., Bharath, A.A., 2017. Deep reinforcement learning: A brief survey. *IEEE Signal Process. Mag.* 34 (6), 26–38.
- Baber, R., 2020. Pandemics: learning from the past. *Climacteric* 23 (3), 211–212.
- Bank, D., Koenigstein, N., Giryas, R., 2023. *Autoencoders*. Springer International Publishing, Cham, pp. 353–374.
- Berge, T., Lubuma, J.M.-S., Moremedi, G.M., Morris, N., Kondera-Shava, R., 2017. A simple mathematical model for ebola in africa. *J. Biol. Dyn.* 11 (1), 42–74.
- Bo, Z.W., Hua, L.Z., Yu, Z.G., 2006a. Optimization of process route by genetic algorithms. *Robot. Comput.-Integr. Manuf.* 22, 180–188.
- Bo, Z.W., Hua, L.Z., Yu, Z.G., 2006b. Optimization of process route by genetic algorithms. *Robot. Comput.-Integr. Manuf.* 22, 180–188.
- Bo, L., Rein, L., 2005. Comparison of the luus-jaakola optimization procedure and the genetic algorithm. *Eng. Optim.* 37 (4), 381–396.
- Bognanni, M., Doug, H., Kolliner, D., Mitman, K., 2020. Economics and epidemics: Evidence from an estimated spatial econ-sir model. In: *Finance and Economics Discussion Series 2020-091*. Board of Governors of the Federal Reserve System, Washington.
- Brodeur, A., Gray, D., Islam, A., Bhuiyan, S., 2020. A literature review of the economics of covid-19. IZA Discussion Paper No. 13411, Available at SSRN: <https://ssrn.com/abstract=3636640>.
- Byrd, R.H., Lu, P., Nocedal, J., Zhu, C., 1995. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.* 16 (5), 1190–1208.
- Cai, B., Huang, G., Samadiani, N., Li, G., Chi, C.-H., 2021. Efficient time series clustering by minimizing dynamic time warping utilization. *IEEE Access* 9, 46589–46599.
- Chen, Y., Liang, X., Hong, T., Luo, X., 2017. Simulation and visualization of energy-related occupant behavior in office buildings. *Build. Simul.* 10, 785–798.
- Chien, C.H., Aggarwal, J.K., 1986. Identification of 3d objects from multiple silhouettes using quadrees/octrees. *Comput. Vis. Graph. Image Process.* 36 (2), 256–273.
- Chumachenko, D., Dobriak, V., Mazorchuk, M., Meniaillov, I., Bazilevych, K., 2018. On agent-based approach to influenza and acute respiratory virus infection simulation. In: *2018 14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering. (TCSET)*, pp. 192–195.
- Connell, R., Dawson, P., Alex, S., 2009. Comparison of an agent-based model of disease propagation with the generalised sir epidemic model. *ADA510899*.
- Conti, A.A., 2020. Historical and methodological highlights of quarantine measures: from ancient plague epidemics to current coronavirus disease (COVID-19) pandemic. *Acta Bio-Medica : Atenei Parm.* 91 (2), 226–229.
- Davis, L., 1985a. Applying adaptive algorithms to epistatic domains. In: *Proceedings of the International Joint Conference on Artificial Intelligence*. pp. 162–164.
- Davis, L., 1985b. Applying adaptive algorithms to epistatic domains. In: *Proceedings of the International Joint Conference on Artificial Intelligence*. pp. 162–164.
- De Groot, R.J., Baker, S.C., Baric, R.S., Brown, C.S., Drosten, C., Enjuanes, L., Fouchier, R.A.M., Galiano, M., Gorbalenya, A.E., Memish, Z.A., Perlman, S., Poon, L.L.M., Snijder, E.J., Stephens, G.M., Woo, P.C.Y., Zaki, A.M., Zambon, M., Ziebuhr, J., 2013. Middle east respiratory syndrome coronavirus (mers-cov): announcement of the coronavirus study group. *J. Virol.* 87, 7790–7792.
- Desai, A., Kraemer, M., Bhatia, S., Cori, A., Nouvellet, P., Herringer, M., Cohn, E., Carrion, M., Brownstein, J., Madoff, L., Lassmann, B., 2019. Real-time epidemic forecasting: challenges and opportunities. *Health Secur.* 17 (8), 268–275.

- Ellison, G., 2020. Implications of Heterogeneous SIR Models for Analyses of COVID-19. National Bureau of Economic Research, page Working paper 27373.
- Eurosurveillance Editorial Team, 2020. Note from the editors: world health organization declares novel coronavirus (2019-ncov) sixth public health emergency of international concern. *Euro Surveill* 25, 200131e.
- Fernández-Villaverde, J., Jones, C.I., 2020. Estimating and Simulating a SIRD Model of COVID-19 for Many Countries, States, and Cities. National Bureau of Economic Research, Working Paper 27128.
- Friedman, J., Liu, P., Troeger, C.E., Carter, A., Reiner, R.C., Barber, R.M., Collins, J., Lim, S.S., Pigott, D.M., Vos, T., Hay, S.I., Murray, C.J.L., Gakidou, E., 2021. Predictive performance of international covid-19 mortality forecasting models. *Nat. Commun.* 12, 2609.
- Fukuoka, T., Ito, K., 2010. Exposure Risk Assessment By Coupled Analysis of Cfd and Sir Model in Enclosed Space. *AIVC*.
- Ghaeri, A., Shoar, S., Naderan, M., Hoseini, S.S., 2005. The applications of genetic algorithms in medicine. *Oman Med J* 30 (6), 406–416.
- Grossman, T., Wool, A., 1997. Computational experience with approximation algorithms for the set covering problem. *European J. Oper. Res.* 101 (1), 81–92.
- Harweg, T., Wagner, M., Weichert, F., 2023. Agent-based simulation for infectious disease modelling over a period of multiple days, with application to an airport scenario. *Int. J. Environ. Res. Public Health* 20 (1).
- Hassanat, A.B.A., Alkafaween, E., 2017. On enhancing genetic algorithms using new crossovers. *Int. J. Comput. Appl. Technol.* 55 (3).
- Holland, J.H., 1992. Genetic algorithms. *Sci. Am.* 267 (1), 66–73.
- Holme, P., 2021. Fast and principled simulations of the sir model on temporal networks. *Plos One* 16 (2), e0246961.
- Hy, Truong Son, Nguyen, Viet Bach, Tran-Thanh, Long, Kondor, Risi, 2022. Temporal multiresolution graph neural networks for epidemic prediction. In: *Workshop on Healthcare AI and COVID-19. PMLR*, pp. 21–32.
- Iuliano, A.D., et al., 2018. Estimates of global seasonal influenza-associated respiratory mortality: a modelling study. *Lancet* 391 (10127), 1285–1300.
- Ivorra, B., Ferrandez, M.R., Vela-Perez, M., Ramos, A.M., 2020. Mathematical modeling of the spread of the coronavirus disease 2019 (covid-19) taking into account the undetected infections. the case of china. *Commun Nonlinear Sci Numer. Simulat.*
- Jin, Guangyin, Liang, Yuxuan, Fang, Yuchen, Shao, Zezhi, Huang, Jincai, Zhang, Junbo, Zheng, Yu, 2023. Spatio-temporal graph neural networks for predictive learning in urban computing: A survey. *IEEE Trans. Knowl. Data Eng.* 36 (10), 5388–5408.
- Jones, C., Philippon, T., Venkateswaran, V., 2021. Optimal mitigation policies in a pandemic: Social distancing and working from home. *Rev. Financ. Stud.* 34 (11), 5188–5223.
- Kermack, W.O., McKendrick, A.G., 1927. A contribution to the mathematical theory of epidemics. *Proc. R. Soc.* 115, 700–721.
- Kingma, D.P., Ba, J., 2017. Adam: A method for stochastic optimization. *arXiv*.
- Kiss, I.Z., Miller, J.C., Simon, P.L., 2017. *Mathematics of Epidemics on Networks*. Springer, Cham.
- Krekov, A., Gruninger, J., Schlönvoigt, R., Kruger, J., 2015. Towards in situ visualization of extreme-scale, agent-based, worldwide disease-spreading simulations. *SIGGRAPH Asia 2015 Vis. High Perform. Comput.* 7, 1–4.
- Kwon, K.S., Park, J.I., Park, Y.J., Jung, D.M., Ryu, K.W., Lee, J.H., 2020. Evidence of long-distance droplet transmission of sars-cov-2 by direct air flow in a restaurant in Korea. *J. Korean Med. Sci.* 35 (46), e415.
- Lazebnik, T., 2023a. Computational applications of extended sir models: A review focused on airborne pandemics. *Ecol. Model.* 483, 110422.
- Lazebnik, T., 2023b. Cost-optimal seeding strategy during a botanical pandemic in domesticated fields. *arXiv*.
- Lazebnik, T., 2023c. Data-driven hospitals staff and resources allocation using agent-based simulation and deep reinforcement learning. *Eng. Appl. Artif. Intell.* 126, 106783.
- Lazebnik, T., Alexi, A., 2022. Comparison of pandemic intervention policies in several building types using heterogeneous population model. *Commun. Nonlinear Sci. Numer. Simul.* 107 (4), 106176.
- Lazebnik, T., Alexi, A., 2023. High resolution spatio-temporal model for room-level airborne pandemic spread. *Mathematics* 11 (2), 426.
- Lazebnik, T., Bunimovich-Mendrazitsky, S., 2022. Generic approach for mathematical model of multi-strain pandemics. *Plos One* 17 (4), e0260683.
- Lazebnik, Teddy, Bunimovich-Mendrazitsky, Svetlana, Ashkenazi, Shai, Levner, Eugene, Benis, Arriel, 2022. Early detection and control of the next epidemic wave using health communications: Development of an artificial intelligence-based tool and its validation on COVID-19 data from the US. *Int. J. Environ. Res. Public Health* 19 (23), 16023.
- Lazebnik, T., Bunimovich-Mendrazitsky, S., Shami, L., 2021a. Pandemic management by a spatio-temporal mathematical model. *Int. J. Nonlinear Sci. Numer. Simul.* 107 (4), 106176.
- Lazebnik, T., Fleischer, T., Yaniv-Rosenfeld, A., 2023. Benchmarking biologically-inspired automatic machine learning for economic tasks. *Sustainability* 15 (14).
- Lazebnik, T., Shami, L., Bunimovich-Mendrazitsky, S., 2021b. Spatio-temporal influence of non-pharmaceutical interventions policies on pandemic dynamics and the economy: The case of covid-19. *Res. Econ.*
- Lazebnik, T., Simon-Keren, L., 2023. Cancer-inspired genomics mapper model for the generation of synthetic dna sequences with desired genomics signatures. *Comput. Biol. Med.* 164, 107221.
- Lerman, P.M., 1980. Fitting segmented regression models by grid search. *J. R. Stat. Soc. Ser. C. Appl. Stat.* 29 (1), 77–84.
- Liu, Zewen, Wan, Guancheng, Aditya Prakash, B., Lau, Max S.Y., Jin, Wei, 2024. A review of graph neural networks in epidemic modeling. In: *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. pp. 6577–6587.
- Long, J.B., Ehrenfeld, J.M., 2020. The role of augmented intelligence (ai) in detecting and preventing the spread of novel coronavirus. *J. Med. Syst.* 44.
- Mackey, T.K., Liang, B.A., 2012. Lessons from sars and h1n1/a: employing a who-wto forum to promote optimal economic-public health pandemic response. *J. Public Health Policy* 33, 119–139.
- Masuda, N., Holme, P., 2017. *Temporal Network Epidemiology*. Springer, Singapore.
- McDowell, I., 2023. *Explanation and Causal Models for Social Epidemiology*. Springer International Publishing, pp. 37–88.
- Miller, J.C., 2017a. Mathematical models of sir disease spread with combined non-sexual and sexual transmission routes. *Infect. Dis. Model.* 2, 35–55.
- Miller, Joel C., 2017b. Mathematical models of SIR disease spread with combined non-sexual and sexual transmission routes. *Infect. Dis. Model.* 2 (1), 35–55.
- Nguyen, Viet Bach, Hy, Truong Son, Tran-Thanh, Long, Nghiem, Nhung, 2023. Predicting covid-19 pandemic by spatio-temporal graph neural networks: A new zealand's study. *arXiv preprint arXiv:2305.07731*.
- Nix, A.E., Vose, M.D., 1992. Modeling genetic algorithms with Markov chains. *Ann. Math. Artif. Intell.* 5 (1).
- Olson, R.S., Moore, J.H., 2016. Tpot: A tree-based pipeline optimization tool for automating machine learning. In: *Workshop on Automatic Machine Learning. PMLR*, pp. 66–74.
- Orozco-Acosta, E., Adin, A., Ugarte, M.D., 2023. Big problems in spatio-temporal disease mapping: Methods and software. *Cpmputer Methods PPrograms Biomed.* 231 (107403).
- Peng, Z., Rogas, P.A.L., Kropff, E., Bahnfleth, W., Buonanno, G., Dancer, S.J., Kur-nitski, J., Li, Y., Loomans, M.G.L.C., Marr, L.C., Morawska, L., Nazaroff, W., Noakes, C., Querol, X., Sekhar, C., Tellier, R., Greenhalgh, T., Bourouiba, L., Boerstra, A., Tang, J.W., Miller, S.L., Jimenez, J.L., 2020. Practical indicators for risk of airborne transmission in shared indoor environments and their application to covid-19 outbreaks. *Environ. Sci. Technol.* 56, 1125–1137.
- Priest, J.D., Kishore, A., Machi, L., Kuhlman, C.J., Machi, D., Ravi, S.S., 2021. Csonnet: An agent-based modeling software system for discrete time simulation. In: *2021 Winter Simulation Conference. (WSC)*, pp. 1–12.
- Privault, N., 2018. *Understanding Markov Chains*. Springer Singapore.
- Salgotra, R., Gandomi, M., Gandomi, A.H., 2020. Time series analysis and forecast of the covid-19 pandemic in india using genetic programming. *Chaos Solitons Fractals* 138, 109945.
- Samet, H., 1984. The quadtree and related hierarchical data structures. *ACM Comput. Surv.* 16 (2), 187–260.
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O., 2017. Proximal policy optimization algorithm. *arXiv*.
- Shami, L., Lazebnik, T., 2022. Economic aspects of the detection of new strains in a multi-strain epidemiological-mathematical model. *Chaos Solitons Fractals* 165, 112823.
- Shami, L., Lazebnik, T., 2023. Financing and managing epidemiological-economic crises: Are we ready for another outbreak?. *J. Policy Model.* 45 (1), 74–89.
- Sharma, S., 2017. Markov chain monte carlo methods for bayesian data analysis in astronomy. *Annu. Rev. Astron. Astrophys.* 55 (1), 213–259.
- Shen, Y., Li, C., Dong, H., Wang, Z., Martinez, L., Sun, Z., Handel, A., Chen, Z., Chen, E., Ebell, M.H., Wang, F., Yi, B., Wang, H., Wang, X., Wang, A., Chen, B., Qi, Y., Liang, L., Li, Y., Ling, F., Chen, J., Xu, G., 2020. Community outbreak investigation of sars-cov-2 transmission among bus riders in eastern china. *JAMA Intern. Med.* 180 (12), 1665–1671.
- Silva, V.L.S., Heaney, C.E., Li, Y., Pain, C.C., 2023. Data assimilation predictive gan (da-predgan) applied to a spatio-temporal compartmental model in epidemiology. *J. Sci. Comput.* 94 (25).
- Soheili-Khah, S., Douzal-Chouakria, A., Gaussier, E., 2016. Generalized k-means-based clustering for temporal data under weighted and kernel time warp. *Pattern Recognit. Lett.* 75, 63–69.
- Syakur, M.A., Khotimah, B.K., Rochman, E.M.S., Satoto, B.D., 2018. Integration k-means clustering method and elbow method for identification of the best customer profile cluster. *IOP Conf. Ser.: Mater. Sci. Eng.* 336 (1), 012017.
- Tomy, Abhishek, Razzanelli, Matteo, Lauro, Francesco Di, Rus, Daniela, Santina, Cosimo Della, 2022. Estimating the state of epidemics spreading with graph neural networks. *Nonlinear Dynam.* 109 (1), 249–263.
- Tran, Khanh-Tung, Hy, Truong Son, Jiang, Lili, Vu, Xuan-Son, 2024. Mglep: Multimodal graph learning for modeling emerging pandemics with big data. *Sci. Rep.* 14 (1), 1–16.
- Tuite, A.R., Fisman, D.N., Greer, A.L., 2020. Mathematical modelling of covid-19 transmission and mitigation strategies in the population of ontario, canada. *CMAJ* 192, E497–E505.

- Veturi, Y.A., Woof, W., Lazebnik, T., Moghul, I., Woodward-Court, P., Wagner, S.K., Cabral de Guimarães, T. a., Varela, D., Liefers, M.B., Patel, P.J., Beck, S., Webster, A.R., Mahroo, O., Keane, P.A., Michaelides, M., Balaskas, K., Ponikos, N., 2023. Syntheye: Investigating the impact of synthetic data on artificial intelligence-assisted gene diagnosis of inherited retinal disease. *Ophthalmol. Sci.* 3, 100258.
- Viguerie, A., Lorenzo, G., Auricchio, F., Baroli, D., Hughes, T.J.R., Patton, A., Reali, A., Yankeelov, T.E., Veneziani, A., 2021. Simulating the spread of covid-19 via a spatially-resolved susceptible–exposed–infected–recovered–deceased (seird) model with heterogeneous diffusion. *Appl. Math. Lett.* 111, 106617.
- Wang, Lijing, Adiga, Aniruddha, Chen, Jiangzhuo, Sadilek, Adam, Venkatramanan, Srinivasan, Marathe, Madhav, 2022. Causalgnn: Causal-based graph neural networks for spatio-temporal epidemic forecasting. In: *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, pp. 12191–12199.
- Wiratsudakul, A., Suparit, P., Modchang, C., 2018. Dynamics of zika virus outbreaks: an overview of mathematical modeling approaches. *PeerJ*.
- Witty, S., Lee, J.K., Tosch, E., Atrey, A., Clary, K., Littman, M.L., Jensen, D., 2021. Measuring and characterizing generalization in deep reinforcement learning. *Appl. AI Lett.* 2 (4), e45.
- Yang, L., Shami, A., 2020. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing* 415, 295–316.
- Yin, S., Wu, J., Song, P., 2023. Optimal control by deep learning techniques and its applications on epidemic models. *J. Math. Biol.* 86 (36).
- Yu, X., Fang, M., Yu, J., Cheng, L., Ding, S., Kou, Z., 2023a. Epidemiological characteristics and spatio-temporal analysis of brucellosis in shandong province, 2015–2021. *BMC Infect. Dis.* 23 (669).
- Yu, Shuo, Xia, Feng, Li, Shihao, Hou, Mingliang, Sheng, Quan Z., 2023b. Spatio-temporal graph learning for epidemic prediction. *ACM Trans. Intell. Syst. Technol.* 14 (2), 1–25.
- Zhang, J., Man, K.F., 1998. Time series prediction using rnn in multi-dimension embedding phase space. In: *SMC'98 Conference Proceedings 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*, vol. 2, pp. 1868–1873.
- Zhang, C., Song, W., Cao, Z., Zhang, J., Tan, P.S., Xu, C., 2020. Learning to dispatch for job shop scheduling via deep reinforcement learning. In: *34th Conference on Neural Information Processing Systems*.