



STORM-GAN+: spatio-temporal meta-GAN for cross-city estimation of heterogeneous human mobility responses to COVID-19

Han Bao¹ · Xun Zhou¹ · Yiqun Xie² · Yanhua Li³ · Xiaowei Jia⁴

Received: 6 February 2023 / Revised: 14 June 2023 / Accepted: 21 June 2023 /

Published online: 17 July 2023

© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2023

Abstract

Estimating human mobility is essential during the COVID-19 pandemic because it provides policymakers with important information for non-pharmaceutical actions. Deep learning methods perform better on tasks with enough training data than traditional estimating techniques. However, estimating human mobility during the rapidly developing pandemic is challenging because of data non-stationarity, a lack of observations, and complicated social situations. Prior studies on estimating mobility either concentrate on a single city or cannot represent the spatio-temporal relationships across cities and time periods. To address these issues, we solve the cross-city human mobility estimation problem using a deep meta-generative framework. Recently, we proposed the Spatio-Temporal Meta-Generative Adversarial Network (STORM-GAN) model, which estimates dynamic human mobility responses under social and policy conditions relevant to COVID-19 and is facilitated by a novel spatio-temporal task-based graph (STTG) embedding. Although STORM-GAN achieves a good average estimation accuracy, it creates higher errors and exhibits over-fitting in particular cities due to spatial heterogeneity. To address these issues, in this paper, we extend our prior work by introducing an improved spatio-temporal deep generative model,

✉ Xun Zhou
xun-zhou@uiowa.edu

Han Bao
han-bao@uiowa.edu

Yiqun Xie
xie@umd.edu

Yanhua Li
yli15@wpi.edu

Xiaowei Jia
xiaowei@pitt.edu

¹ University of Iowa, Iowa City, IA, USA

² University of Maryland, College Park, MD, USA

³ Worcester Polytechnic Institute, Worcester, MA, USA

⁴ University of Pittsburgh, Pittsburgh, PA, USA

namely STORM-GAN+. STORM-GAN+ deals with the difficulties by including a distance-based weighted training technique into the STTG embedding component to better represent the variety of knowledge transfer across cities. Furthermore, to mitigate the issue of overfitting, we modify the meta-learning training objective to teach estimated mobility. Finally, we propose a conditional meta-learning algorithm that explicitly tailors transferable knowledge to various task clusters. We perform comprehensive evaluations, and STORM-GAN+ approximates real-world human mobility responses more accurately than previous methods, including STORM-GAN.

Keywords Meta-learning · Generative adversarial networks · Spatio-temporal · Graph embedding · COVID-19

1 Introduction

The COVID-19 pandemic's evolving developments (e.g., spread, mutation, and vaccination) raised the pressure on policymakers to develop flexible and adaptable regulations that can safeguard public health while preventing economic collapses and supporting basic needs in daily life. The staged reopening has been established as a solution to this problem to prevent illnesses brought on by loosened social distance policies. Estimating how dynamic human mobility will respond to pandemic conditions and policies is therefore still important in policymaking.

It is particularly challenging for cities in the early stages of an outbreak or wave (for example, the Omicron variant) to forecast future human mobility responses under unprecedented severity levels or unforeseen policies due to the asynchronous spread of the disease and the absence of past data. In order to resolve this issue, it is essential for these cities to be able to utilize the past experiences and knowledge of other cities for their own evaluation. In this regard, there is an imperative need for estimation methods of mobility response that can leverage cross-city knowledge to produce promising results.

We attempt to solve the *cross-city heterogeneous* human mobility responses estimation problem in this paper. We intend to develop a model that can rapidly adapt to previously unobserved cities and time periods and estimate the dynamics of human mobility response under any projected conditions. Our inputs consist of contextual (e.g., population, POI) counts, epidemic (e.g., COVID-19 cases), policy (e.g., stay-at-home orders), and corresponding human mobility responses (e.g., POI visit counts, home dwell time) measures from multiple locations.

Challenges There are four main challenges with the cross-city human mobility response estimation problem. *First*, human mobility responses are influenced by various intricate social–physical factors that are either unknown or uncertain. Responses may be impacted, for instance, by people's willingness to comply with rules, decisions made by service providers (such as whether a restaurant will open or offer dine-in options), modifications to public transportation, supply, and a variety of other factors [1]. *Second*, spatial–temporal non-stationarity is frequently observed in human mobility responses. For instance, the impact of various factors on mobility can quickly change over time and tends to vary from region to region due to cultural and economic differences. The availability of training data for each estimation task is severely constrained by such spatial and temporal non-stationarity, making it challenging to take advantage of the approximation power of data-driven approaches. *Third*, additionally, there are complex spatial and temporal dependencies among varying estimation

tasks, i.e., cities and times, which must be explicitly taken into account for reliable parameter sharing. Cities may, for instance, have similar mobility dynamic patterns based on their geographic proximity (e.g., distances, travel connections like airlines) and their pandemic stage. *Fourth*, task heterogeneity and uncertainty are two major obstacles to meta-learning that cannot be overcome by globally sharing information among tasks, because the majority of the meta-learning algorithms in use nowadays presume that all tasks globally share the same transferable knowledge. As a result, they struggle to manage a series of tasks that come from multiple distributors.

Related work Numerous recent attempts have used machine learning techniques to perform spatio-temporal estimate tasks. In a recent paper [2], a COVID-GAN has been proposed for estimating human mobility, where policies (e.g., school closure) are employed as restrictions to enhance estimation outcomes. This study does not model the spatial and temporal relationships between cities or phases; instead, it solely considers estimation in a single city. As a result, they are unable to adjust to unknown cities quickly. COVID-GAN is a spatial model that does not explicitly characterize the temporal dynamics of human mobility responses.

The few-shot learning problem, which arises when there are few training samples for new tasks, has been addressed in recent years by a number of meta-learning approaches [3, 4]. However, few of them are designed for spatio-temporal tasks. Zhang et al. [5] is one of the exceptions; it generates traffic volume using a model-based meta-learning technique and a variational autoencoder structure. Another study on traffic prediction incorporates CNNs and the attention mechanism [4]. In addition, functionality zones are used to divide cities into tasks before the model-agnostic meta-learning (MAML) framework is implemented, as described in [6].

We recently proposed a Spatio-TempOral (conditional) Meta-Generative Adversarial Network (STORM-GAN [7]), a deep spatial-temporal meta-generative model [8, 9], to estimate cross-cities mobility under various real-world conditions such as COVID-19 severity and local policy interventions. STORM-GAN effectively captures mobility pattern similarities among cities by modeling the spatio-temporal dependencies and correlations across different tasks with a meta-learning paradigm and spatio-temporal task-based graph (STTG) embedding learning. However, it does not consider the effects of spatial heterogeneity over different urban contexts. For example, more activities are concentrated in urban areas (e.g., Boston, NYC) than rural areas, which can be attributed to different population densities in different regions, producing higher errors in certain regions. Also, STORM-GAN suffers from overfitting when testing in rural areas. However, these regions indeed have a high significance in understanding the mobility patterns due to data scarcity issues.

Proposed work This paper is a significant extension of our recent work [7]. In this paper, we address the above limitations by proposing an improved conditional meta-learning-based spatio-temporal generative adversarial network model to take into consideration of spatial heterogeneity across tasks when estimating human mobility response patterns to COVID-19. We call the new meta-learning-based conditional generative adversarial network model STORM-GAN+. The use of a conditional GAN [8] allows consideration of unknown and uncertain factors (i.e., modeled as latent factors). Building on top of STORM-GAN, the STORM-GAN+ model learns to generate spatio-temporal mobility dynamics in different cities under a set of geographic, epidemic, social, and other factors. It utilizes an improved STTG embedding learning strategy to capture region heterogeneity from a distribution of tasks (i.e., mobility estimation for each city over a time period) for fast adaption to new spatio-temporal tasks (e.g., new cities, future projection). Moreover, we adopt a new conditional meta-learning strategy to explicitly solve the task heterogeneity issue, and modify the

training objective to learn the estimated mobility faster and more accurately. We proposed the STORM-GAN+ with the new graph embedding method and meta-learning structure for better model generalization and adaptation, further improving STORM-GAN+ performance.

The new contributions of this paper are as follows:

- We quantify the spatial heterogeneity of human mobility tasks using an enhanced spatio-temporal task-based graph (STTG) to improve the model's capacity for capturing task diversity. Through a distance-based weighted learning technique, this augmentation can assist in learning the spatial heterogeneity of the task, hence increasing the shared knowledge learning across tasks.
- We quantify the spatial heterogeneity of human mobility tasks using an enhanced spatio-temporal task-based graph (STTG) to improve the model's capacity for capturing task diversity. Through a distance-based weighted learning technique, this augmentation can assist in learning the spatial heterogeneity of the task, hence increasing the shared knowledge learning across tasks.
- We modify the objective function of meta-learning by substituting the fixed meta-learning rates with a dynamic meta-hyperparameter adaptation scheme. This dynamic learning mechanism can eliminate the need to tune meta-learning rates and improve solution quality by addressing the issue of overfitting when training on sparse regions.
- We conduct intensive experiments on real-world data to validate the solution quality improvements achieved by the proposed approach under various scenarios. Extensive experiments on real-world datasets demonstrate that STORM-GAN+ is capable of generating human mobility patterns that closely approximate the ground truth and significantly outperform other baseline methods.

The rest of the paper is organized as follows: Section 2 defines the problem. Sections 3 and 4 detail the methodology and model structure of STORM-GAN and STORM-GAN+, respectively. We present evaluation results in Sect. 5, and related works are summarized in Sect. 6, and the paper is concluded in Sect. 7.

2 Problem statement

This section introduces a set of basic concepts about our data modeling and then provides a formal problem statement.

2.1 Basic concepts

Definition 1 *Spatial grid* S is a grid-discretization of a spatial field (e.g., a city), where each grid cell s_i represents an equally sized squared area. Given S , the location of any POI can be mapped into a grid cell. For simplicity, in this work we choose the grid cells to be $1\text{km} \times 1\text{km}$.

Definition 2 *Temporal period* T is a temporal period (e.g., a 7-day window) containing equal-length slots (e.g., a day), denoted as $T = \{t_1, t_2, \dots, t_n\}$, where each slot t represents the finest temporal resolution of the data.

Definition 3 *Mobility-related conditions* All conditions that will influence human mobility responses are mobility-related conditions including contextual conditions (e.g., population, household income), epidemic conditions (e.g., COVID-19 confirmed cases and deaths), and policy conditions (e.g., strict stay-at-home or shelter-in-place orders). We denote a list of $\mathbf{k}$

conditions as $\mathbf{F} = \{f_1, f_2, \dots, f_k\}$. For a grid cell s , we denote $f^{s,t}$ as all the conditions of s in time slot t .

Definition 4 *Human mobility responses* The human mobility responses $\mathbf{M}$ are a two-dimensional tensor, representing the total number of visits to POIs (e.g., grocery stores, hardware stores, restaurants, gas stations) in each grid cell s for time slot t .

In the scope of the present study, we use point-of-interest (POI) visit counts simply as an example to demonstrate the solution framework. Our model can also use other mobility measures (e.g., median home dwell time). The choice of the measure is not the focus of this paper.

Definition 5 *Spatio-temporal task-based graph (STTG)* A STTG is defined as a directed weighted graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents spatial locations of tasks (e.g., cities), and $\mathcal{E}$ represents the relevance among spatial locations. The graph is attributed, meaning that the nodes are associated with attributes $f(v_i)$ to describe the characteristics of each spatial location in the task space. The attributes of STTG are not fixed and can be specified by users in various scenarios. We illustrate two STTG scenarios in Sect. 5.4.

Definition 6 *Generator G* A deep neural network model is used to generate a series of human mobility response maps $\mathbf{M}'_G$ given a set of conditions.

Definition 7 *Discriminator D* A deep neural network model that outputs a probability p_{real} at which a map of human mobility responses is classified as from real-world rather than from a generator G .

Definition 8 *Spatio-temporal mobility estimation tasks* A task $\mathcal{T}_i$ consists of a series of pairs $(\mathbf{M}^t, \mathbf{F}^t)$ for a few consecutive time periods T (e.g., 5 weeks) in a partitioned area $\mathbf{S}$ (e.g., 10×10 grids of a city). Each sample is a 4D tensor with size $l \times l \times k \times T$, where $l \times l$ is the size of the spatial window, and k is the number of conditions. Each $\mathcal{T}_i$ is divided into a training set $\mathcal{D}_i^{train}$ and a testing set $\mathcal{D}_i^{test}$.

2.2 Problem definition

We construct the tasks (Definition 8) by a spatio-temporal partition of all conditions $\mathbf{F}$ and mobility responses $\mathbf{M}$. Each spatio-temporal task $\mathcal{T}_i$ contains data from the grid S of a single city for w consecutive time periods $\{T_1, \dots, T_w\}$, and tasks are mutually exclusively (i.e., no overlap along the temporal dimension). Each data sample in a task contains a time-series of length $|T|$ with any start time (but the time span of a sample must be completely within the span of a task).

Inputs:

- A time-series of conditions $\{\mathbf{F}^{t-|T|+1}, \dots, \mathbf{F}^t\}$ for each data sample in training tasks;
- Mobility response $\mathbf{M}^t$ for each sample in training tasks;
- Spatio-temporal task-based graph **STTG** for tasks.

Outputs:

- A generator **G** to generate/estimate mobility responses;
- A meta-initialization θ for **G** for fast adaptation to training and new testing tasks.

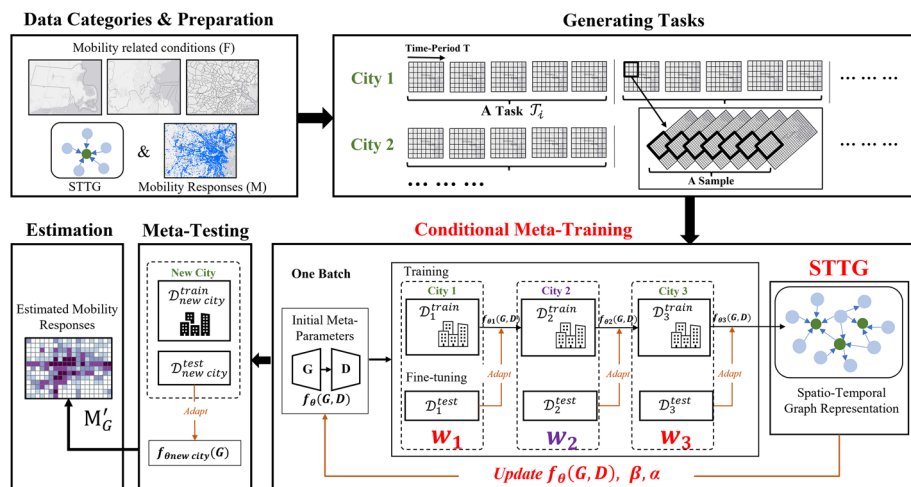


Fig. 1 STORM-GAN and STORM-GAN+

Objective:

- Minimize average generation error on new testing tasks.

In this work, given a series of tasks sampled from multiple cities (e.g., Boston and NYC), we train a meta-generative model. Then, when a new city (e.g., Houston) comes in with a small training set, we quickly fine-tune the meta-model parameters to obtain a tailored model for the new city to generate its mobility responses.

3 Baseline: STORM-GAN

In this section, we present STORM-GAN details from our recent work [7] and use STORM-GAN as the building block for the new method proposed in this paper. Our previous work makes the first attempt to formulate the cross-city COVID-19 human mobility estimation problem through a meta-learning-based deep data generation problem.

3.1 STORM-GAN architecture

The following section presents our model structure to address the cross-city human mobility response estimation problem. Figure 1 shows the overall workflow of the proposed STORM-GAN starting from the data preprocessing step to the human mobility estimation step. To help capture the task similarity, we propose a spatio-temporal task-based graph (STTG) in the generator to improve the estimated values using domain knowledge. Next, we will discuss the three key components in our STORM-GAN, i.e., the spatio-temporal generator, discriminator, and STTG. Note, Fig. 1 illustrates the structures of STORM-GAN+, and differences made by STORM-GAN+ based on STORM-GAN are highlighted in red.

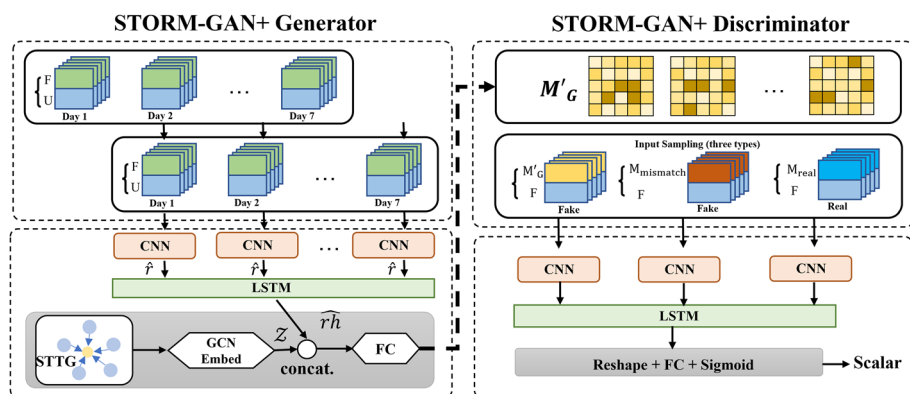


Fig. 2 STORM-GAN and STORM-GAN+ Network Architecture

3.1.1 Spatio-temporal generator

The spatio-temporal generator aims to generate human mobility responses while capturing spatial patterns and temporal dependencies. The utilization of the GAN structure allows known factors to be learned as conditions and unknown factors to be represented by latent noise, which helps the model to express these uncertainties (i.e., mobility response estimations may have some degree of variations). As shown in Fig. 2, the generator uses a stack of CNN and LSTM elements where CNN captures local spatial patterns and maintains the spatial representation (e.g., neighbor relationships). LSTM is able to capture temporal trends in a given sequence. The generator takes a condition tensor $\mathbf{F} \in \mathbb{R}^{l \times l \times k \times T}$ (we skip the batch dimension here for simplicity) and a latent code tensor $\mathbf{U} \in \mathbb{R}^{l \times l \times u \times T}$, where k is the number of conditions (e.g., policy, COVID statistics, and contextual conditions), u is the dimension of the noise vector for modeling the uncertainties, and T is the length of a time period.

In the generator, denote the CNN output as $\hat{r} \in \mathbb{R}^{d \times T}$, where d is the number of output features. Next, to capture temporal patterns and trends, $\hat{r}$ is fed into an LSTM layer, where the memory vector is concatenated to $\hat{r}$. Then, the output from the last timestamp of the LSTM layer $r\hat{h}$ will be concatenated with the graph embedding and further passes through a fully connected layer to generate the final output. This $r\hat{h}$ is not yet the estimated mobility response $\mathbf{M}'_G$.

For more robust estimation, the spatio-temporal generator additionally uses a proposed spatio-temporal task-based graph embedding to characterize task-level spatio-temporal features and potential dependency across multiple cities and their mobility patterns, as discussed in the next section.

3.1.2 Spatio-temporal task-based graph (STTG) embedding

In real-world scenarios, spatial meta-learning tasks may have a very diverse distribution. For example, in our problem, tasks sampled from multiple cities can have significantly different human mobility patterns due to different urban contexts. Meanwhile, there may also exist underlying dependencies among cities due to traffic connections, geo-socio similarities, etc. Such spatial distribution of tasks, if properly utilized, would significantly enhance the performance of the learned meta-learning model.

To better model heterogeneity and dependency across spatio-temporal tasks, we propose a novel spatio-temporal task-based graph (STTG) to incorporate such information and facilitate the learning of transferable knowledge from related tasks. In the following part, we will first introduce STTG's construction rules and then discuss STTG-based embedding learning.

The STTG in our proposed STORM-GAN framework is a directed weighted graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where nodes represent the spatial locations of tasks (e.g., cities) and edges (and weights) represent the relevance among spatial locations. Furthermore, the graph is attributed, meaning that the nodes are associated with attributes $f(v_i)$ to describe the characteristics of each spatial location in the task space.

STTG can be defined in various ways depending on the underlying analysis goal and the network data used. In our particular application, we define each node v_i as a major metropolitan area in the USA, which contains features $f(v_i)$ of the city such as the current stage of the pandemic. Each edge e_{ij} connecting cities v_i and v_j indicates that there is geo-socio similarity between v_i and v_j in the pandemic, where the edge weight represents the strengths of such similarity. Depending on how "similarity" is measured, we can define the edge and weights differently. Examples of such measures may include the infection spreading between cities [10], the correlation between cities' mobility patterns, etc.

In this thesis, we present two examples of STTG construction cases, although other definitions can also be used with our method. In the first case, we define the edges and their weights based on physical reachability, i.e., the number of direct flights and driving distance between cities, with the assumptions that the COVID spreading is tightly related to traveling and that cities with stronger transportation connections tend to have more relevance in COVID situation. In the second case, we define the edges based on the similarity of historical mobility pattern distribution measured by the Kullback–Leibler (KL) divergence [11] between cities. We provide details on the STTG construction in Sect. 5.4 and show effectiveness in Sect. 5.5.

Next, we use the built STTG in the meta-training phase to help learn more helpful knowledge across tasks. As Fig. 2 shows, during the training on the generator, we first sample a task-specific 1-hop subgraph $\mathbf{H}$ for the corresponding node (a city) on the STTG. Then, we obtain a sub-graph embedding using variational graph autoencoder (VGAE), which consists of graph convolution neural networks (GCNs) [12] by solving:

$$f(\mathbf{X}^L, A) = \alpha \left(\hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} \mathbf{X}^{L-1} W^{L-1} \right), \quad (1)$$

where A is the adjacency matrix, $\hat{A} = A + I$, I is the identity matrix, $\hat{D}$ is the diagonal node degree matrix of $\hat{A}$, $\alpha(\cdot)$ is an activation function (e.g., ReLU), X is the feature matrix of each node from the graph, and W^{L-1} is a weight matrix for the $L - 1$ th layer. The encoder takes A and X as inputs and generates the latent variable $\mathcal{Z}$ as output. The decoder reconstructs an adjacency matrix defined by the inner product between latent variable $\mathcal{Z}$.

The graph feature representation $\mathcal{Z}$ is concatenated with the output $\hat{r}h$ (Fig. 2) and flows through a final fully connected layer in the spatio-temporal generator to achieve $\mathbf{M}'_G$. The new STTG- and GCN-based embedding, being part of the generation process, will also help the meta-learner to incorporate the similarity and dependency among tasks.

3.1.3 Spatio-temporal discriminator

Figure 2 shows the structure of the discriminator, which takes a tensor of size $\mathbb{R}^{I \times I \times (k+1) \times T}$, where k is the number of conditions (same as that for generator) and the added one dimension is for the mobility response layer.

To create training data for “fake” or “real” labels, the input tensors are created in three ways: (1) generated mobility $\mathbf{M}'_G$ concatenated with conditions; (2) real mobility $\mathbf{M}_{real}$ concatenated with corresponding conditions; (3) conditions concatenated with mismatched real mobility $\mathbf{M}_{mismatch}$. Finally, only samples from the second combination are labeled “real”. Using these inputs, the discriminator learns to determine whether an input is “real” or “fake”.

STORM-GAN training on a single city is performed through adversarial configuration between the generator and discriminator. A min-max objective function is used to train G and D jointly by solving:

$$\begin{aligned} \mathcal{L}_{G,D} = & \mathbb{E}_{\mathbf{M} \sim P_{data}} [\log D(\mathbf{M}, \mathbf{F})] \\ & + \mathbb{E}_{\mathbf{U} \sim P_U} [\log(1 - D(G(\mathbf{F}, \mathbf{U}, STTG), \mathbf{M}))] \end{aligned} \quad (2)$$

where $\mathcal{L}_{G,D}$ is the binary cross-entropy loss.

3.2 STORM-GAN training and testing

3.2.1 MAML-based outer loop updates

As defined in Sect. 1, our goal is to learn the shared knowledge or initialization across tasks drawn from multiple cities. To transfer the structural knowledge from the graph and spatio-temporal knowledge from mobility data in multiple cities, we adopt the model-agnostic meta-learning (MAML) framework to learn the meta parameter θ_D and θ_G , specifically in our case for all spatio-temporal tasks. The learned initialization is expected to contain common knowledge that can be fast-adapted to new tasks.

With MAML, we sample a batch of tasks in each step, where each task $\mathcal{T}_i$ consists of $(\mathbf{F}, \mathbf{M})$ and their corresponding one-hop subgraph in STTG. The general optimization formulation is as follows. Given a set of tasks $\{\mathcal{T}_1, \mathcal{T}_2, \dots\}$ drawn from a task distribution $p(\mathcal{T})$, where each task $\mathcal{T}_i \sim p(\mathcal{T})$ consists of a training and a test set $\{\mathcal{D}_i^{train}, \mathcal{D}_i^{test}\}$, we optimize the G and D with parameters θ_G and θ_D to minimize the expected empirical loss across all tasks during meta-training. The meta-update rules are given by:

$$\theta_D = \theta_D - \beta \nabla_{\theta_D} \mathcal{L}_{G,D}(f_{\theta'_D}) \quad (3)$$

$$\theta_G = \theta_G - \beta \nabla_{\theta_G} \mathcal{L}_{G,D}(f_{\theta'_G}) \quad (4)$$

where β is the learning rate for meta-update, and θ'_G and θ'_D represent temporary task-specific parameters. Following the recommendation in [6], we use the first-order MAML for the meta-weight update.

3.2.2 STORM-GAN inner loop updates

Algorithm 1 shows the detailed meta-training procedure. The training of discriminator uses the three types of combinations: $(\mathbf{M}'_G, \mathbf{F})$, $(\mathbf{M}_{real}, \mathbf{F})$ and $(\mathbf{M}_{mismatch}, \mathbf{F})$. Denote α as the learning rate of the discriminator, θ'_D as the parameters of the discriminator, the loss function and the update rule of D are shown in Eqs. (5) and (6), respectively.

$$f_D = -\frac{1}{m} \sum_{i=1}^m \left(\log(1 - D((\mathbf{M}'_G)^i, \mathbf{F}^i)) + \log(D(\mathbf{M}_{real}^i, \mathbf{F}^i)) \right)$$

$$+ \log(1 - D(\mathbf{M}_{mismatch}^i, \mathbf{F}^i)) \quad (5)$$

$$\theta'_D = \theta'_D - \alpha \nabla f_D(\theta'_D) \quad (6)$$

where m is the total number of samples in a batch, and index i refers to the i th sample. Denote θ'_G as the parameters in G , we have the loss function and update rule of G as:

$$\begin{aligned} f_G &= \frac{1}{m} \sum_{i=1}^m \left(\log(1 - D((\mathbf{M}'_G)^i, \mathbf{F}^i)) \right) \\ &= \frac{1}{m} \sum_{i=1}^m \left(\log(1 - D(G(\mathbf{F}_i, \mathbf{U}_i, STTG), \mathbf{F}^i)) \right) \end{aligned} \quad (7)$$

$$\theta'_G = \theta'_G + \alpha \nabla f_G(\theta'_G) \quad (8)$$

3.2.3 STORM-GAN adaptation on new tasks

During the model adaptation phase (e.g., updating the optimal initialization for a new task from a new city), we first copy θ_G and θ_D from the meta-training phase as the initialization for fast-adaptation and then use training samples from the new task to perform STORM-GAN for updating the meta-parameter θ_D and θ_G . Finally, the updated model outputs the estimated mobility using testing samples. The tasks used for meta-testing adaptation are held out from meta-training.

4 Mobility estimation through STORM-GAN+

Although STORM-GAN achieves a good average estimation accuracy, it creates higher errors and exhibits over-fitting in particular cities due to the presence of spatial heterogeneity (e.g., urban vs. suburban). To address these issues, we propose a new model, namely **STORM-GAN+**, to improve the estimation accuracy.

4.1 STORM-GAN+ architecture

As discussed above, STORM-GAN does not handle spatial heterogeneity in different urban contexts. To help improve the model capacity in dealing with this issue, we reform the objective of the STTG by introducing a distance-based weighted training technique into the STTG embedding component in order better to represent the variety of knowledge transfer across cities. Furthermore, to mitigate the issue of overfitting, we modify the meta-learning training objective to teach estimated mobility. Note, Fig. 1 illustrates the overall structure of STORM-GAN+.

4.1.1 STORM-GAN+ generator

The spatio-temporal generator aims to generate human mobility responses while capturing spatial patterns and temporal dependencies. As shown in Fig. 2, the generator uses a stack of CNN and LSTM elements where CNN captures local spatial patterns and maintains the spatial representation (e.g., neighbor relationships). LSTM is able to capture temporal trends in a given sequence. The generator G takes a condition tensor $\mathbf{F} \in \mathbb{R}^{l \times l \times k \times T}$ (we skip the

Algorithm 1 STORM-GAN Training and Testing**Require:**

- 1: • Set of training cities $\mathbf{T}_{\text{train}}$; set of testing cities $\mathbf{T}_{\text{test}}$
- 2: • Conditions $\mathbf{F}$, mobility $\mathbf{M}_{\text{real}}$, a STTG $\mathcal{G}$
- 3: • Inner learning rate α ; outer learning rate β ; number of epochs *epoch*

Ensure: θ_G, θ_D , estimated mobility $\mathbf{M}'_G$ for $\mathbf{T}_{\text{test}}$

```

4: G = initG(); D = initD()
5: Randomly initialize meta  $\theta_G, \theta_D$ 
6: for  $e = 1$  to epoch do
7:   Sample a batch of  $\mathcal{T}$  from  $\mathbf{T}_{\text{train}}$ 
8:   Sample the subgraph  $\mathbf{H}$  of  $\mathcal{T}$  from  $\mathcal{G}$ 
9:   for  $\mathcal{T}_i$  in  $\{\mathbf{F}, \mathbf{M}_{\text{real}}, \mathbf{H}\}$  do
10:    Sample a set of disjoint  $\mathcal{D}_i^{\text{train}}, \mathcal{D}_i^{\text{test}}$ 
11:    Generate graph embedding  $\mathbf{E}$  of  $\mathbf{H}$ 
12:     $\mathbf{M}'_G = G(\mathbf{F}, \mathbf{E}, \text{rand}(P_U))$ 
13:    Update D using  $\mathcal{D}_i^{\text{train}}$  by Eqs. (5) and (6)
14:    Update G using  $\mathcal{D}_i^{\text{train}}$  by Eqs. (7) and (8)
15:    Evaluate estimation loss using  $\mathcal{D}_i^{\text{test}}$  by Eq. (2)
16:   end for
17:   Update  $\theta_D$  and  $\theta_G$  by Eqs. (3) and (4)
18: end for
19: Return  $\theta_G, \theta_D$ 
20: Sample batch of testing tasks  $\mathcal{T}$  from  $\mathbf{T}_{\text{test}}$ 
21: for  $\mathcal{T}_i$  in  $\{\mathbf{F}, \mathbf{M}_{\text{real}}, \mathbf{H}\}$  do
22:   Sample a disjoint  $\mathcal{D}_i^{\text{train}}, \mathcal{D}_i^{\text{test}}$  from  $\mathbf{T}_{\text{test}}$ 
23:   Generate graph embedding  $\mathbf{E}$  of  $\mathbf{H}$ 
24:   Copy  $\theta_G, \theta_D$ 
25:   Evaluate performance by Eq. (2) using  $\mathcal{D}_i^{\text{train}}$ 
26:   Update G through Eqs. (7) and (8)
27:   Estimate  $\mathbf{M}'_G$  using updated G and  $\mathcal{D}_i^{\text{test}}$ 
28: end for

```

batch dimension here for simplicity) and a latent code tensor $\mathbf{U} \in \mathbb{R}^{l \times l \times u \times \mathbb{T}}$, where k is the number of conditions (e.g., policy, COVID statistics, and contextual conditions), u is the dimension of the noise vector for modeling the uncertainties, and $\mathbb{T}$ is the length of a time period.

In G , denote the CNN output as $\hat{r} \in \mathbb{R}^{d \times \mathbb{T}}$, where d is the number of output features. Next, to capture temporal patterns and trends, $\hat{r}$ is fed into a LSTM layer, where the memory vector is concatenated to $\hat{r}$. Then, the output from the last timestamp of the LSTM layer $\hat{r}h$ will be concatenated with the graph embedding and further passes through a fully connected layer to generate the final output. This $\hat{r}h$ is not yet the estimated mobility response $\mathbf{M}'_G$.

4.1.2 Enhanced spatio-temporal task-based graph (STTG) embedding

In real-world situations, spatial meta-learning tasks may have a highly variable distribution. For instance, due to the heterogeneous metropolitan environments in our problem, tasks sampled from diverse locations may exhibit dramatically varied human mobility patterns. Furthermore, due to traffic linkages, geo-socio similarities, etc., there could also be underlying relationships among cities. If utilized appropriately, such a geographical distribution of activities would significantly improve how well the acquired meta-learning model performed.

Phase-I: Define STTG To better model heterogeneity and dependency across spatio-temporal tasks, we propose a novel spatio-temporal task-based graph (STTG) to incorporate such information and facilitate the learning of transferable knowledge from related tasks.

In the following part, we will first introduce STTG's construction rules and then discuss STTG-based embedding learning. Finally, we will introduce the distance-based weighted learning strategy.

The STTG in our proposed STORM-GAN+ framework is a directed weighted graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where nodes represent the spatial locations of tasks (e.g., cities) and edges (and weights) represent the relevance among spatial locations. Furthermore, the graph is attributed, meaning that the nodes are associated with attributes $f(v_i)$ to describe the characteristics of each spatial location in the task space.

STTG can be defined in various ways depending on the underlying analysis goal and the network data used. In our particular application, we define each node v_i as a major metropolitan area in the USA, which contains features $f(v_i)$ of the city such as the current stage of the pandemic. Each edge e_{ij} connecting cities v_i and v_j indicates that there is geo-socio similarity between v_i and v_j in the pandemic, where the edge weight represents the strengths of such similarity. Depending on how "similarity" is measured, we can define the edge and weights differently. Examples of such measures may include the infection spreading between cities [10], the correlation between cities' mobility patterns, etc.

In this paper, we present two examples of STTG construction cases, although other definitions can also be used with our method. In the first case, we define the edges and their weights based on physical reachability, i.e., the number of direct flights and driving distance between cities, with the assumptions that the COVID spreading is tightly related to traveling and that cities with stronger transportation connections tend to have more relevance in COVID situation. In the second case, we define the edges based on the similarity of historical mobility pattern distribution measured by the Kullback–Leibler (KL) divergence [11] between cities. We provide details on the STTG construction in Sect. 5.4 and show effectiveness in Sect. 5.5.

Phase-2: Engage STTG with generator Next, we use the built STTG in the meta-training phase to help learn more useful knowledge across tasks. As Fig. 2 shows, during the training on generator, we first sample a task-specific 1-hop subgraph $\mathbf{H}$ for the corresponding node (a city) on the STTG. Then, we obtain a sub-graph embedding using variational graph autoencoder (VGAE) which consists of graph convolution neural network (GCNs) [12] by solving:

$$f(\mathbf{X}^L, A) = \alpha \left(\hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} \mathbf{X}^{L-1} W^{L-1} \right), \quad (9)$$

where A is the adjacency matrix, $\hat{A} = A + I$, I is the identity matrix, $\hat{D}$ is the diagonal node degree matrix of $\hat{A}$, $\alpha(\cdot)$ is an activation function (e.g., ReLU), X is the feature matrix of each node from the graph, and W^{L-1} is a weight matrix for the $L - 1$ th layer. The encoder takes A and X as inputs and generates the latent variable $\mathcal{Z}$ as output. The decoder reconstructs an adjacency matrix defined by the inner product between latent variable $\mathcal{Z}$.

The graph feature representation $\mathcal{Z}$ is concatenated with the output $\hat{r}h$ (Fig. 2) and flows through a final fully connected layer in the spatio-temporal generator to achieve $\mathbf{M}'_G$. The STTG- and GCN-based embedding, being part of the generation process, will also help the meta-learner to incorporate the similarity and dependency across tasks.

Phase-3: Measure STTG heterogeneity across tasks As shown in Fig. 1, tasks in each mini-batch are randomly selected from a source city during the meta-training phase. According to the generation rule of the spatio-temporal mobility estimation tasks, there are no temporal overlaps across tasks from the same source city; however, there are spatial overlaps. Due to this, it is likely that one mini-batch can contain two or more tasks from identical cities, and each mini-batch task is treated equally in the graph learning phase. Consequently, from a spatial perspective, the meta-training on such mini-batches cannot well capture the spatial heterogeneity across tasks. Our initial STTG does not handle these mini-batch sampling

drawbacks explicitly; as a result, model performance is greatly affected by spatially duplicated tasks. Therefore, estimating mobility in sparse regions (e.g., sub-urban, rural areas) presents one difficulty. Due to data scarcity issues, these regions have a high level of importance for comprehending mobility patterns, but they are difficult to estimate due to their small sample sizes.

To overcome this problem, we present an enhanced STTG by integrating a weighted mini-batch learning technique to increase estimate accuracy. The primary motivation for weighting tasks is to provide lower weights to tasks that are geographically near to the first sampled task in each mini-batch and allocate greater weights to tasks that are spatially far from the first sampled task.

We introduce a distance-based weighted mini-batch tasks learning strategy as a constraint to preserve the spatial heterogeneity across tasks in each mini-batch. As a natural generalization, given a set of tasks $t = \{1, 2, \dots, i\}$ in a mini-batch, we can assign a weight w_i to each task by solving:

$$\mathbf{d}_{1i} = \frac{d_o - \min(d_o)}{\max(d_o) - \min(d_o)}, (\mathbf{d}_{1i} \in [0, 1]) \quad (10)$$

$$\mathbf{w}_i = 1 - e^{-\mathbf{d}_{1i}}, (\mathbf{w}_i \in [0, 1]) \quad (11)$$

where d_{1i} represents the spatial distance between the 1st sampled task and i th sampled task, d_o represents the original spatial distance calculated through the Haversine formula using the coordinates of city centroid coordinates. The d_{1i} has the range $[0, 1]$, with a unity value implying the maximum normalization. When tasks are sampled spatially identical, $d_{1i} = 0$ and $w_i = 0$. In this situation, duplicated tasks only contribute to spatial heterogeneity learning once. Contrariwise, greater distance will lead to greater w_i which facilitates the acquisition of spatial heterogeneity across tasks.

4.1.3 STORM-GAN+ discriminator

Figure 2 shows the structure of the discriminator, which takes a tensor of size $\mathbb{R}^{l \times l \times (k+1) \times |T|}$, where k is the number of conditions (same as that for generator) and the added one dimension is for the mobility response layer.

To create training data for “fake” or “real” labels, the input tensors are created in three ways: (1) generated mobility $\mathbf{M}'_G$ concatenated with conditions; (2) real mobility $\mathbf{M}_{real}$ concatenated with corresponding conditions; (3) conditions concatenated with mismatched real mobility $\mathbf{M}_{mismatch}$. Only samples from the second combination are labeled “real”. Using these inputs, the discriminator learns to determine whether an input is “real” or “fake”.

STORM-GAN+ employed the adversarial setup between the generator and discriminator of GAN model training framework. Each task t_i receives a min-max objective function, which is combined with the **new** distance-based weights $\mathbf{w}_i$ for tasks in each mini-batch. To train G and D together, a new goal function is provided:

$$\begin{aligned} \mathcal{L}_{t_i(G,D)} = \min_G \max_D V(G, D) = & \mathbb{E}_{\mathbf{M} \sim P_{data}} [\log D(\mathbf{M}, \mathbf{F})] \\ & + \mathbb{E}_{\mathbf{U} \sim P_U} [\log(1 - D(G(\mathbf{F}, \mathbf{U}), STTG), \mathbf{M}))] \end{aligned} \quad (12)$$

where $\mathcal{L}_{t_i(G,D)}$ is the binary cross-entropy loss of each tasks. Thus, the final objective function can be expressed as the sum loss over tasks sampled in each mini-batch:

$$\mathcal{L}_{G,D} = \sum_{i=1}^j \mathbf{w}_i \mathcal{L}_{t_i(G,D)} \quad (13)$$

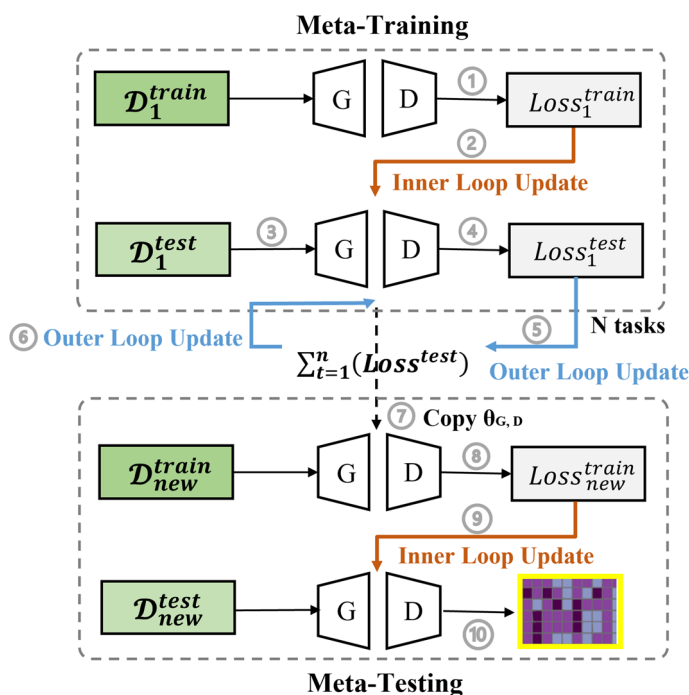


Fig. 3 Meta-learning training process

where $\mathcal{L}_{G,D}$ is the sum of loss over tasks in each mini-batch.

4.2 STORM-GAN+ training and testing

We deployed the model-agnostic meta-learning (MAML [13]) framework for learning the STORM-GAN+ meta parameters θ_D and θ_G to transfer the structural information from graphs and the spatio-temporal knowledge from mobility data in multiple cities. Two loops—the inner loop and the outer training loop—are used by MAML to carry out the training procedure, as shown in Fig. 3. In our particular case, MAML attempts to direct the generative model for the inner loop so that the training loss for a specific spatio-temporal task is minimized. In the outer loop, the objective function is used to determine the optimal parameters that can be generalized to a new task in an unknown city.

There are two fundamental drawbacks of MAML, first, when the number of parameters is significantly expanded from a base network, as indicated by the inner loop learning rate α and the outer loop meta-learning rate β , the performance and stability of this technique vary drastically. Compared to the [14] approach, choosing learning rates can be a difficult operation that takes up much GPU time. We discovered that this restriction causes a problem with overfitting in the STORM-GAN estimate results in sparse regions. We decide to learn learning rates as part of the optimization process because only a few samples are available for meta-testing and fine-tuning. Thus, the computational power required to locate fairly good learning rates decreases while also assuring that learning rates function very well. Second, in standard meta-learning, most gradient-based meta-learning algorithms assume an initialization that is globally shared across all tasks. These existing methods may fail to adapt

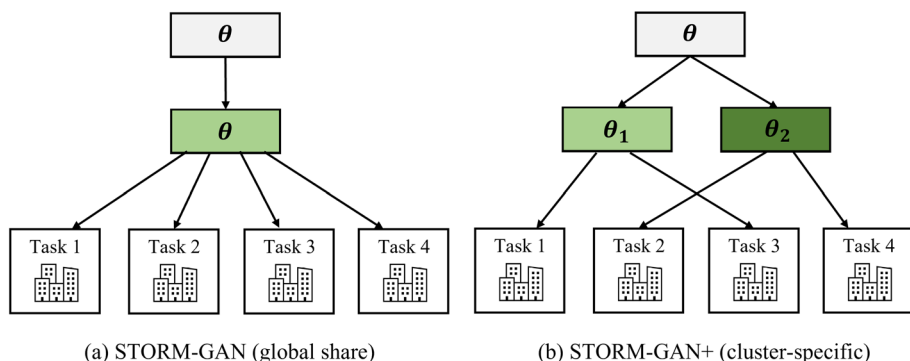


Fig. 4 Illustration of meta-parameter sharing of STORM-GAN and STORM-GAN+

to heterogeneous environments of tasks, in which the complexity of the tasks' distribution cannot be captured by a single meta-parameter vector. In STORM-GAN, tasks are trained with a single meta-parameter that disregards task heterogeneity. Since spatio-temporal tasks are diverse, explicit modeling of task diversity in parameter sharing is essential.

To mitigate the problem of overfitting and globally shared meta-parameter, particularly in sparse regions, we first propose a conditional meta-learning strategy to customize a shared initialization to each cluster using a task clustering framework. Second, we propose an adaptive meta-learning scheme with dynamic learning rates as opposed to fixed learning rates. The solution we propose is founded on the premise that the optimal learning rate does not vary significantly between iterations [15], so we can optimize the learning rates from the previous optimization step.

Section 4.2.1 describes the new conditional meta-learning-based STORM-GAN+ inner loop generation strategy and learning rate α update rules and fine-tuning processes, followed by a more in-depth discussion (Sect. 4.2.2) of the outer loop learning rate β updates. Section 4.2.3 demonstrates the new task adaptation and detailed Algorithm 2.

4.2.1 STORM-GAN+ inner loop updates

The optimization-based meta-learning for the model's inner loop contains the update for Discriminator D and the Generator G , so the inner loop learning rate is applied to both components. As Fig. 3 shows, we first construct a batch of meta-training tasks and divide each task into a training set $\mathcal{D}^{train}$ and testing set $\mathcal{D}^{test}$. For each task, the training set is fed into a GAN parameterized by θ_D and θ_G ; the training set loss $Loss^{train}$ is computed and propagated to update the GAN parameters. The testing set then feeds into the updated GAN to generate mobility. Using the generated mobility to compute the testing set loss $Loss^{test}$ for the given task. Steps (1–4) are repeated, and the same process repeats for the other n sampled tasks, starting from the same GAN model.

Conditional meta-learning: cluster-based parameter sharing STORM-GAN adopts a global parameter as shown in Fig. 4. To accommodate heterogeneous tasks, as illustrated in Fig. 4, STORM-GAN+ tailors the global shared initialization to each task by taking advantage of task-specific information. First, based on the diversity of spatial-temporal tasks, we cluster all training cities' tasks into C categories. Category c of assignments represents various urban contexts. If task $\mathcal{T}_i$ belongs to cluster c , the clustering result $rc_{t_i} = 1$, otherwise $rc_{t_i} = 0$. In the meta-training phase, tasks in the same category are trained together, and each

cluster learns a set of unique parameters for the generator and discriminator, respectively, $\theta_{G_{rc}}$ and $\theta_{D_{rc}}$.

The training loss for each task is subjected to the standard stochastic gradient descent during the meta-training inner loop. Following that, the testing set is used to evaluate the updated parameters. Denote α_D as the inner loop learning rate and θ'_D as the parameter for the discriminator of a particular task, the loss function of D , update rule of α_D , and the update rule of θ'_D are shown in Eqs. (14), (15), and (16), respectively.

$$f_D = -\frac{1}{m} \sum_{i=1}^m \left(\log(1 - D((\mathbf{M}'_G)^i, \mathbf{F}^i)) + \log(D(\mathbf{M}'_{real}, \mathbf{F}^i)) \right. \\ \left. + \log(1 - D(\mathbf{M}'_{mismatch}, \mathbf{F}^i)) \right) \quad (14)$$

$$\alpha_D = \alpha_{D_{n-1}} + \eta_D \nabla f_{D_n}(\theta'_{D_n}) \quad (15)$$

$$\theta'_D = \theta'_D - \alpha_D \nabla f_D(\theta'_D) \quad (16)$$

where m is the total number of samples in a batch, n is the iteration number, η_D is the hyper learning rate for α_D and index i refers to the i th sample.

Similarly, denoting α_G as the inner loop learning rate, η_G as the hyper learning rate, and θ'_G as the parameters in generator G , we have the loss function of G , update rule of α_G and update rule of θ'_G as shown in Eqs. (17), (18), and (19), respectively.

$$f_G = \frac{1}{m} \sum_{i=1}^m \left(\log(1 - D((\mathbf{M}'_G)^i, \mathbf{F}^i)) \right) \\ = \frac{1}{m} \sum_{i=1}^m \left(\log(1 - D(G(\mathbf{F}_i, \mathbf{U}_i, STTG), \mathbf{F}^i)) \right) \quad (17)$$

$$\alpha_G = \alpha_{G_{n-1}} + \eta_G \nabla f_{G_n}(\theta'_{G_n}) \quad (18)$$

$$\theta'_G = \theta'_G - \alpha_G \nabla f_G(\theta'_G) \quad (19)$$

Equations (14) to (19) show the GAN model learning process of each task. As the inner meta-training is trained on batch level, thus, we revise the loss function in Eq. (13) by summing the loss of each task cluster in one batch. Then, the final objective function is:

$$\mathcal{L}_{T_i \sim p(\mathcal{T})_{G,D}} = \sum_{rc}^C \sum_{i=1}^j \mathbf{w}_i r c_i \mathcal{L}_{t_i}(f_{G_{rc}, D_{rc}}) \quad (20)$$

Here we note that $\mathcal{L}_{T_i \sim p(\mathcal{T})_{G,D}}$ denotes the training loss over all tasks from different clusters. Each cluster has different $\theta_{G_{rc}}$ and $\theta_{D_{rc}}$. The structure of training STORM-GAN+ at the inner loop is conditioned by task-related information used for clustering.

4.2.2 STORM-GAN+ outer loop updates

The general optimization procedure is given a set of tasks $\{\mathcal{T}_1, \mathcal{T}_2, \dots\}$ drawn from a task distribution $p(\mathcal{T})$, where each task $T_i \sim p(\mathcal{T})$ consists of a training and a test set $\{\mathcal{D}_i^{train}, \mathcal{D}_i^{test}\}$, we optimize the G and D with parameters θ'_G and θ'_D to minimize the expected empirical loss across all tasks during meta-training for each task cluster. The purpose of the outer loop is to sum up the last update step's testing loss from all the tasks and use the summation to update the $\theta_{G_{rc}}$ and $\theta_{D_{rc}}$. Then, another batch of tasks is sampled, and steps (1–6) in Fig. 3

are repeated. Then, for meta-testing tasks, steps (8–9) are applied to the GAN using the meta-learned parameter $\theta_{G_{rc}}$ and $\theta_{D_{rc}}$ copied from the meta-training phase, which enables generalization over unseen tasks (step 10). During meta-testing, an unseen task is categorized into one cluster, and meta-updated parameters are learned starting from initialization parameters from the corresponding cluster across meta-training tasks, and the new optimal parameters are used to adapt to unseen tasks quickly.

To improve the estimation in unseen cities, we transfer initialization $\theta_{G_{rc}}$ and $\theta_{D_{rc}}$ of STORM-GAN+. Utilize the same assumption that the optimal value of learning rate β does not vary significantly between iterations, we compute the outer loop learning rate β_D , β_G , and meta-update rules θ_D , θ_G through one gradient descent step over each cluster c_i , the previous value of learning rate given by:

$$\beta_D = \beta_{D_{n-1}} + \eta_D \Sigma_{\mathcal{T}_{irc} \sim p(\mathcal{T})} \nabla f_{D_n}(\theta'_{D_n}) \quad (21)$$

$$\theta_{D_{rc}} = \theta_{D_{rc}} - \beta_D \Sigma_{\mathcal{T}_{irc} \sim p(\mathcal{T})} \nabla_{\theta_D} \mathcal{L}_{G,D}(f_{\theta'_D}) \quad (22)$$

$$\beta_G = \beta_{G_{n-1}} + \eta_G \Sigma_{\mathcal{T}_{irc} \sim p(\mathcal{T})} \nabla f_{G_n}(\theta'_G) \quad (23)$$

$$\theta_{G_{rc}} = \theta_{G_{rc}} - \beta_G \Sigma_{\mathcal{T}_{irc} \sim p(\mathcal{T})} \nabla_{\theta_G} \mathcal{L}_{G,D}(f_{\theta'_G}) \quad (24)$$

where θ'_G and θ'_D represent temporary task-specific parameters which acquired from Eqs. (16) and (19). And, $\mathcal{L}_{G,D}$ is acquired from Eq. (20) which represents the summation loss over tasks sampled in each cluster. Note, each cluster has unique $\theta_{G_{rc}}$ and $\theta_{D_{rc}}$.

4.2.3 STORM-GAN+ adaptation on new tasks

During the model adaptation phase (e.g., updating the optimal initialization for a new task from a new city), we first cluster the unseen task, then output (Fig. 3—step 7) $\theta_{G_{rc}}$ and $\theta_{D_{rc}}$ from the meta-training phase as the initialization for fast-adaptation from corresponding clusters. We then use training samples from the new city to perform STORM-GAN+ for updating the meta-parameters θ_D and θ_G in the meta-testing phase. Lastly, the updated model estimates mobility based on testing samples. The tasks used for meta-testing adaptation are held out from meta-training.

The goal of this adaptation step is to generate the mobility map of the entire study area (e.g., a city or county) from the estimations of $s \times s$ spatial unit windows defined in Definition 8. Since STORM-GAN+ would generate the mobility maps of spatial unit windows in different time slots and areas, we present *multiple-draw based sliding window* scheme to generate the final map of mobility estimation. This scheme takes w draws of the same window from the generator G instead of a single draw. The results of the multiple draws are averaged before being integrated into the estimation of the original study area. This scheme can reduce random effects in comparison. The overall learning process is demonstrated in Algorithm 2;

5 Evaluation

Through the experiments, we aim to answer the following questions:

- Whether STORM-GAN+ can outperform baseline methods in terms of solution quality, including STORM-GAN?
- How does the proposed spatio-temporal network impact the solution quality compared to non-spatial-temporal models?
- What is the effect of the improved meta-learning framework on model performance?

Algorithm 2 STORM-GAN+ Training and Testing

Require:

```

1: • Set of training cities  $\mathbf{T}_{\text{train}}$ ; set of testing cities  $\mathbf{T}_{\text{test}}$ 
2: • Conditions  $\mathbf{F}$ , mobility  $\mathbf{M}_{\text{real}}$ , a STTG  $\mathcal{G}$ 
3: • Inner learning rate  $\alpha$ ; outer learning rate  $\beta$ ; number of epochs epoch
Ensure:  $\theta_G, \theta_D$ , estimated mobility  $\mathbf{M}'_G$  for  $\mathbf{T}_{\text{test}}$ 
4:  $\mathbf{G} = \text{initG}()$ 
5:  $\mathbf{D} = \text{initD}()$ 
6: Randomly initialize meta  $\theta_G, \theta_D$ 
7: Cluster all tasks and into  $\mathbf{C}$  clusters, and get  $rc$  for each task;
8: for  $e = 1$  to epoch do
9:   Sample a batch of  $\mathcal{T}$  from  $\mathbf{T}_{\text{train}}$ 
10:  Sample the subgraph  $\mathbf{H}$  of  $\mathcal{T}$  from  $\mathcal{G}$ 
11:  for  $\mathcal{T}_i$  in  $c_i$  do
12:    for  $\mathcal{T}_{c_i}$  in  $\{\mathbf{F}, \mathbf{M}_{\text{real}}, \mathbf{H}\}$  do
13:      Sample a set of disjoint  $\mathcal{D}_i^{\text{train}}, \mathcal{D}_i^{\text{test}}$ 
14:      Generate graph embedding  $\mathbf{E}$  of  $\mathbf{H}$ 
15:       $\mathbf{M}'_G = \mathbf{G}(\mathbf{F}, \mathbf{E}, \text{rand}(\mathbf{P}_{\mathbf{U}}))$ 
16:      Update  $\alpha_D$  and  $\mathbf{D}$  using  $\mathcal{D}_i^{\text{train}}$  by Eqs. (14), (15), and (16)
17:      Update  $\alpha_G$  and  $\mathbf{G}$  using  $\mathcal{D}_i^{\text{train}}$  by Eqs. (17), (18), and (19)
18:      Evaluate estimation loss using  $\mathcal{D}_i^{\text{test}}$  by Eq. (20)
19:    end for
20:  Update  $\beta_D, \beta_G$  and  $\theta_{D_{rc}}$  and  $\theta_{G_{rc}}$  by Eqs. (21), (23), and (22), (24)
21:  Copy  $\theta_{D_{rc}}$  and  $\theta_{G_{rc}}$ 
22: end for
23: end for
24: Return  $\theta_G, \theta_D$ 
25: Sample batch of testing tasks  $\mathcal{T}_{\text{new}}$  from  $\mathbf{T}_{\text{test}}$ 
26: Cluster  $\mathcal{T}_{\text{new}}$  into  $c_i$ 
27: for  $\mathcal{T}_{c_i}$  in  $\{\mathbf{F}, \mathbf{M}_{\text{real}}, \mathbf{H}\}$  do
28:  Sample a disjoint  $\mathcal{D}_i^{\text{train}}, \mathcal{D}_i^{\text{test}}$  from  $\mathbf{T}_{\text{test}}$ 
29:  Generate graph embedding  $\mathbf{E}$  of  $\mathbf{H}$ 
30:  Copy  $\theta_{D_{rc}}$  and  $\theta_{G_{rc}}$ 
31:  Evaluate performance by Eq. (20) using  $\mathcal{D}_i^{\text{train}}$ 
32:  Update  $\mathbf{G}$  through Eqs. (18), (17) and (19)
33:  Estimate  $\mathbf{M}'_G$  using updated  $\mathbf{G}$  and  $\mathcal{D}_i^{\text{test}}$ 
34: end for

```

- Can the new proposed STTG embedding contribute to the model performance?
- What are the effects of training on temporally seen vs. temporally unseen tasks and spatially seen vs. spatially unseen tasks?
- How does conditional meta-learning impact the solution quality compared to standard-meta learning?

5.1 Dataset description

5.1.1 Data sources

We elaborate on four types of data as described in Definitions 3 and 4 (pandemic, contextual, policy, and mobility). The pandemic-related conditions are collected from Centers for Disease Control and Prevention [16], demographic and socio-economic conditions are collected from Census Bureau [17], the date of disease prevention policies (e.g., stay-at-home, mask policies) are collected from the corresponding city government websites. Finally, the human mobility

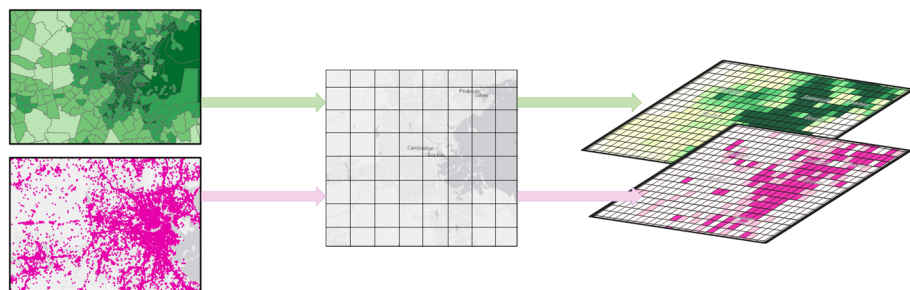


Fig. 5 Data preprocessing for STORM-GAN+

responses, which are represented by the total POIs visit count data in this paper, are collected from SafeGraph [18]. SafeGraph provides free access to data for academic purposes upon request, and all the other data are publicly available.

5.1.2 Data granularity

The original POI dataset from SafeGraph is obtained by collecting the location from cell phone records with latitude and longitude information. Then, the location information is used to determine the visits to POIs [18]. The POI visit counts data is in point data format. Figure 5 pink color data illustrates the discretization of one city, where we sum the total POI visit counts that fall into each grid cell, and use this aggregated visit counts value to represent the human mobility responses of each grid cell. Moreover, each condition data are associated with different geographic units (e.g., census block groups, counties) due to different data sources or privacy protection concerns. Thus, further spatio-temporal data processing is needed before training and estimation.

5.1.3 Data preprocessing

Figure 5 shows a summary of two types of data we gathered from multiple sources based on definitions in Sect. 2.1 (i.e., contextual, epidemic, and policy conditions; human mobility responses). As we can see, most of the data are associated with different geographic units (e.g., census block groups, counties) due to different data sources or privacy protection concerns. Thus, further spatio-temporal data processing is needed before training and estimation.

First, we need to integrate all the data of various types and geographic units into the same format, and feed them into the STORM-GAN+. To construct the list of conditions for our input, for each grid cell, we preprocess data collected from different sources with different geographic units. As Fig. 5 shows, we first adopt a commonly used space-partitioning method to segment each spatial domain into grid cells of size of $1 \text{ km} \times 1 \text{ km}$, and segment all mobility-related conditions using the same grid cells. Then, each spatial region (or unit spatial window) we used to create a data sample is a 10×10 spatial window on the grid. For each grid cell, the value of human mobility response is the total number of POI visit counts in a day. Note that some conditions are re-scaled during this process. For example, population and median household income data are collected at the census tract level, and we linearly re-scaled the data using the corresponding area ratios between the area of the original census tract polygon and the proposed 10×10 grid cells. Similarly, COVID-19 statistics and policy

Table 1 Detailed data statistics

City	Average number of POIs per grid cell	POIs	Size
Boston	28	26,054	37×48
NYC	64	133,520	58×72
LA	52	86,721	52×64
Chicago	30	47,356	50×40
Houston	24	37,315	50×60
Iowa City	6	1401	20×32

data are collected at the county level. We assign each grid cell with the corresponding data on which county it belongs.

5.1.4 Training data description

We collect mobility-related datasets from six cities. The dataset spans over six cities in different states located from the west coast, midwest to the east coast (i.e., Boston, Chicago, Houston, Iowa City, Los Angeles, and NYC). The list of cities also covers regions from large metropolitan areas to less populous places. Detailed statistics of these datasets for each city (e.g., number of POIs, number of cells covered for each city) are listed in Table 1. The duration of data is from 02/24/2020 to 10/25/2020 for all cities, covering 35 weeks in total. As discussed in Sect. 2.1, the data are segmented into a spatio-temporal distribution of tasks, where each task contains one single city for five consecutive weeks (no mutual overlaps among tasks). The candidate methods are trained on five cities (meta-training) with one left out as the new city for meta-testing. Specifically, we selected Houston (large metropolitan area) and Iowa City (small urban area) as the two test cities in two separate experiments. Adaptation on test cities is performed with data samples from the most recent two weeks (out of 35 weeks in total). As the duration of data is from 02/24/2020 to 10/25/2020, we divide each city into 7 tasks along temporal dimensions, and each task maintains the full spatial domain of the corresponding city. Overall, we have **35** spatio-temporal estimation tasks in total. For methods with meta-learning, 80% of data in each task is used for meta-training, and the rest for testing (Definition 8).

5.2 Evaluation metrics

We evaluate the performance of STORM-GAN+ by using the following measures: mean absolute error (MAE) and rooted mean square error (RMSE).

$$MAE = \frac{1}{n} \sum_{i=0}^n |M_G - \hat{M}_G| \quad (25)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (M_G - \hat{M}_G)^2} \quad (26)$$

where M_G is the real mobility response and $\hat{M}_G$ is the generated mobility response values by candidate approach. Since the model generates the spatial unit windows multiple times for

each grid cell during estimation, the outputs of the generator are averaged before comparing with the ground truth.

To evaluate the model performance of learning the data distribution, we calculate the KL divergence to indicate the similarity between the learned human mobility responses distribution $\hat{\mathbf{P}}$ and real human mobility responses distribution $\mathbf{P}$ on different bin sizes. The KL divergence is defined as follows:

$$\mathbf{D}_{\text{KL}}(\mathbf{P} \hat{\mathbf{P}}) = \sum_{i=1}^N P(M'_G) \log \left(\frac{P(M'_G)}{\hat{P}(M_{\text{real}})} \right) \quad (27)$$

5.3 Baseline methods

We compare our proposed method with the following baseline methods, and fine-tune each method using Houston and Iowa City as testing cities, respectively.

- *HA: Historical Average* The average of human mobility responses was calculated using observed values from the same location in the past two weeks (same weekday).
- *Spatial smoothing with neighborhood regions* [19]. This method uses the mobility response values in a local 3×3 window to compute a mean as the estimated value. The values for smoothing are from the same weekday in the most recent week.
- *Ridge* [20] We use ridge regression with the same input features and mobility responses.
- *cGAN* [21] A conditional GAN where the generator and discriminator use three fully connected layers (no layer structure to learn spatial or temporal patterns).
- *COVID-GAN* [2] COVID-GAN has the same structure as the above cGAN, and it adds a correction layer, which is used to add constraints based on policy to refine the results.
- *MAML-DAWSON* [3] An optimization-based meta-learning approach using MAML. As DAWSON originally works on music generation tasks, we modify its inner structure with a regression-focused conditional GAN.
- *MetaST* [4] MetaST fuses CNN, LSTM, and attention mechanism to predict urban traffic volume through the MAML framework.
- *STORM-GAN* [7] STORM-GAN is a spatio-temporal meta-generative model that creates human mobility responses in various cities based on a variety of geographic, epidemiological, social, and other aspects.

5.4 STTG construction examples

In this section, we provide two different STTG construction scenarios to evaluate the effectiveness of graph embedding in human mobility estimation.

Scenario 1 (S1) We assume that cities of similar sizes, socio-economic environments, and land-use design may share similar human mobility patterns that could help the estimation of new cities. To build the graph $\mathcal{G}_{s1}(\mathcal{V}, \mathcal{E})$, we enumerate major metropolitan cities from every region in the U.S. and define each city as a node v_i . Next, we extract human mobility maps for all the cities from the same date, and calculate the pairwise distribution similarity score between cities using KL-divergence. The KL divergence indicates the strength of human mobility correlation. Each edge is added if the correlation is ≤ 0.5 and is weighted by the correlation. $\mathcal{G}_{s1}$ contains 55 nodes and 682 edges. Node attributes store the outbreak stage of COVID-19. Each stage value is in $\{1, 2, 3\}$, where a smaller value means earlier in the

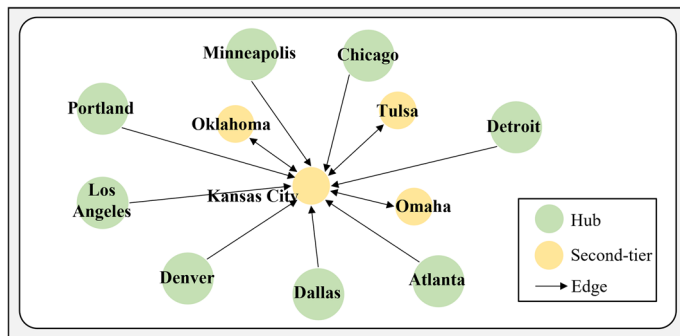


Fig. 6 Subgraph of a node in STTG S_2

COVID-19 outbreak. The stage value is assigned based on the month when exponential growth first appeared.

Scenario 2 (S_2) Intuitively, urban environment increases the chance of infection as people move around and interact with others and the environment. As a hub for migration and travel, urban areas may quickly spread infections to nearby places through short-distance travel, and to major cities through connection flights.

To construct S_2 , similar to S_1 , we enumerate major metropolitan areas in USA and define a graph $\mathcal{G}_{S_2}(\mathcal{V}, \mathcal{E})$ to represent the relationships of these cities.

We divide the nodes $\mathcal{V}$ into two categories: the hub nodes $\mathcal{V}^h$ are major cities with more than 100 airlines; the second-tier nodes $\mathcal{V}^s$ are cities with more than 35 but less than 100 airlines. Moreover, each directed edge $v_i \rightarrow v_j \in \mathcal{E}$ is added if its two nodes are: (1) both major cities that have direct flights or (2) within a spatial proximity threshold (500 km in this paper). Our graph contains 69 nodes and 776 edges.

The graph is then weighted by spatio-temporal attributes associated with nodes and edges. Edge attributes contain the number of directed flights between the cities and their geographic distance. Node attributes store the sum of flights from connected edges as well as the outbreak stage of COVID-19 which is the same as scenario 1. Figure 6 shows an example of our S_2 , which is a 1-hop subgraph for Kansas City, a second-tier city by the above-mentioned classification. Major cities that have direct flights to Kansas City (e.g., Denver, Atlanta, Minneapolis) and second-tier cities (e.g., Oklahoma, Omaha) within the spatial proximity threshold are shown on the subgraph.

We use both of the two STTG construction scenarios with our STORM-GAN+ (namely, STORM-GAN+(S_1) and STORM-GAN+(S_2)) as well as the original STORM-GAN (S_1 and S_2) to compare their performances in the next subsection.

5.5 Estimation quality evaluation

During the adaptation phase, we evaluate the performance of the candidate methods on the two test cities (i.e., Houston and Iowa City) using their last two weeks (Monday to Sunday) of data, respectively. The length of a time period we use is 7 days since human mobility pattern is influenced by strong weekly periodicity. The benefit of meta-learning is that the model can quickly update the model parameters and generate good results on a new task by seeing a small fraction of new data. So, for each testing city and week, we use 2 consecutive

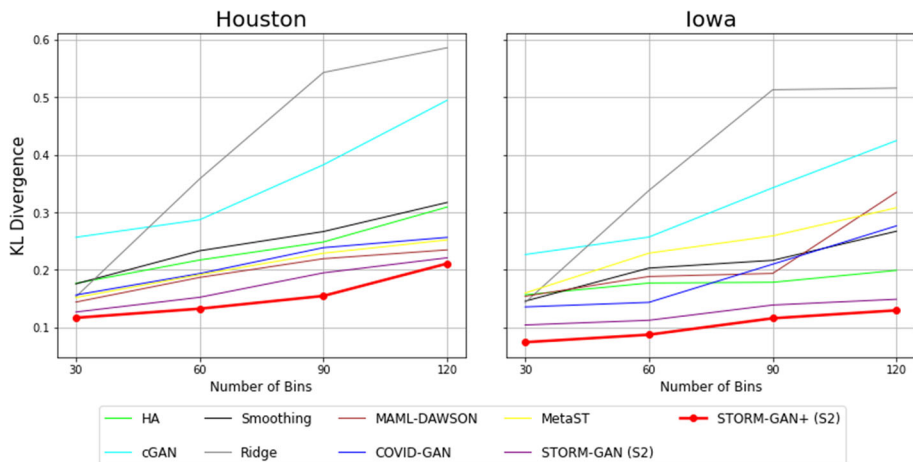


Fig. 7 Kullback–Leibler divergence

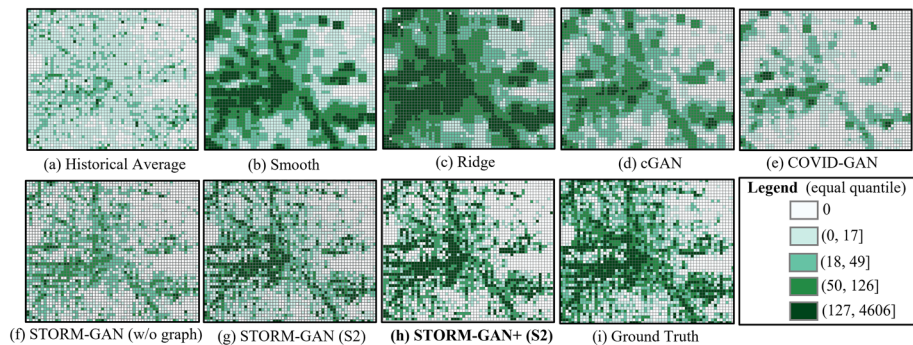


Fig. 8 Mobility estimation results of the Houston study area

weeks of data ahead of the week for adaptation and then use the parameters to generate the next 7-day (one week) human mobility responses.

Furthermore, for all generated mobility maps, we apply the correction filtering proposed in [2], which helps remove all mobility responses at cells with no valid POIs (i.e., no POI or no POIs that are open according to the policy feature) after generating the estimated mobility maps to mitigate spurious results during training.

In the following subsections, we aim to answer the first three questions summarized at the beginning of Sect. 5.5.1. Figures 8 and 9 show a comparison of results from different approaches in different scenarios for Houston and Iowa City, respectively. Furthermore, we answer the fourth question at Sect. 5.5.3. Table 4 shows the ablation study of STORM-GAN+. In Tables 2 and 3, we demonstrate the statistical results of the candidate methods for Houston and Iowa City, respectively. The results for the KL-Divergences are shown in Fig. 7.

The colors used in map symbologies for Figs. 8 and 9 are classified using quantiles extracted from the ground truth (i.e., 0th, 25th, 50th, 75th and 100th), a typical approach for enhanced map visualization. To reduce random effects in the comparison, all the results are based on multiple-draw based approach (10 repetitive runs) described in 4.2.3.

Table 2 Human mobility responses estimation by candidate methods for Houston

Model	RMSE							MAE						
	Mon	Tues	Wed	Thu	Fri	Sat	Sun	Mon	Tues	Wed	Thu	Fri	Sat	Sun
HA	194.8	193.2	193.5	196.3	193.6	194.5	195.1	81.2	80.9	80.4	80.8	79.6	81.3	81.2
Smoothing	150.9	168.1	169.2	177.1	187.3	202.4	162.1	82.3	90.2	90.4	94.6	100.1	99.8	104.2
cGAN	278.2	283.4	284.6	279.5	286.3	286.1	286.2	118.7	122.4	125.3	120.6	128.3	130.6	129.3
Ridge	189.4	192.1	182.3	181.5	187.3	188.6	195.7	95.8	99.5	101.9	95.3	99.4	98.1	95.2
COVID-GAN	171.5	175.6	174.3	172.2	176.4	171.8	170.6	75.1	80.1	73.2	77.1	80.5	81.7	82.8
MAML-DAWSON	169.5	169.7	166.1	168.9	164.6	166.7	168.1	68.3	67.4	70.3	69.4	68.2	75.5	78.2
MetaST	170.2	171.4	173.8	170.6	169.5	170.3	169.4	72.8	76.1	71.5	74.2	72.9	80.2	81.3
STORM-GAN (S1)	151.2	150.5	149.3	162.9	163.9	164.2	167.2	66.7	66.4	64.7	63.1	70.4	71.8	72.2
STORM-GAN (S2)	145.1	142.6	141.9	141.6	152.5	156.7	160.2	61.7	60.4	59.3	53.8	58.4	64.2	67.2
STORM-GAN+ (S1)	141.3	140.1	140.9	140.4	148.8	154.6	157.6	57.9	59.2	50.9	55.5	54.1	60.2	62.1
STORM-GAN+ (S2)	140.1	138.9	138.6	140.4	147.1	153.8	157.2	57.3	58.1	49.6	54.4	53.7	59.0	61.3

New methods and best results are highlighted in bold

Table 3 Human mobility responses estimation by candidate methods for Iowa City

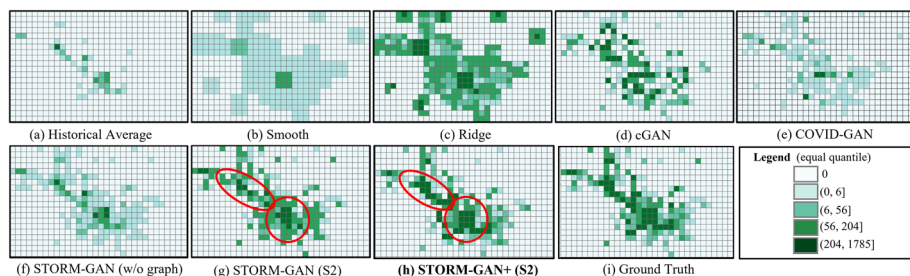
Model	RMSE							MAE						
	Mon	Tues	Wed	Thu	Fri	Sat	Sun	Mon	Tues	Wed	Thu	Fri	Sat	Sun
HA	22.2	23.1	21.3	24.2	22.3	25.2	26.5	13.4	13.1	15.2	12.6	14.2	16.1	13.2
Smoothing	18.4	16.3	18.6	21.7	18.3	19.2	19.5	11.2	10.3	10.6	9.1	11.6	9.2	11.6
cGAN	34.3	33.7	34.8	36.5	32.8	31.2	34.4	21.1	19.5	21.3	18.7	17.8	19.9	20.3
Ridge	20.6	22.3	21.1	20.5	19.8	19.2	20.1	12.4	11.6	13.3	13.6	12.2	13.5	12.2
COVID-GAN	17.1	17.3	16.6	16.3	15.2	14.3	15.6	11.3	10.7	13.1	12.3	14.7	13.8	13.3
MAML-DAWSON	16.5	17.6	17.7	15.2	15.3	13.6	13.4	10.3	11.6	10.1	9.5	8.1	10.9	9.2
MetaST	17.1	17.4	17.8	17.6	16.5	16.3	16.1	10.8	10.6	10.4	10.5	9.9	10.1	9.7
STORM-GAN (S1)	15.8	16.2	16.1	15.8	15.9	15.4	14.3	8.9	9.6	9.9	10.2	9.1	9.3	9.3
STORM-GAN (S2)	14.1	15.6	14.9	14.4	14.2	13.6	13.3	8.2	7.4	9.1	8.3	7.8	9.1	8.5
STORM-GAN+ (S1)	12.1	12.8	12.3	11.7	11.1	10.8	10.9	7.3	6.0	6.9	6.2	7.3	7.4	6.5
STORM-GAN+ (S2)	11.5	11.8	12.0	11.3	10.7	10.1	9.9	7.0	5.5	6.2	6.0	6.7	6.9	6.1

New methods and best results are highlighted in bold

Table 4 Comparison among STORM-GAN+ variations

Method	RMSE	MAE
Base	202.2	80.6
Base + S	171.5	75.1
Base + ST + Meta	149.8	67.6
Base + ST + Meta + Graph (S2)	145.1	61.7
Base + ST + new Meta + new Graph (S2)	140.5	57.8
Base + ST + conditional Meta + new Graph (S2)	140.1	57.3

New methods are highlighted in bold

**Fig. 9** Mobility estimation results of the Iowa City study area

5.5.1 Performance comparison of proposed STORM-GAN+ and other candidate methods

Tables 2 and 3 show the results of the candidate methods obtained using Houston and Iowa City as the testing city, respectively. We apply S1 and S2 graph construction scenarios on STORM-GAN+, as well as STORM-GAN. The overall evaluation results from Houston show that STORM-GAN+ achieves the lowest RMSE and MAE for each day in the week, with major improvements from 7.5% to 100% with both STTG scenarios.

It is interesting to observe that historical average and spatial smoothing methods perform better than the basic cGAN, which to some degree shows the spatio-temporal auto-correlation effects. However, these methods can mainly estimate a rough base but are limited in capturing complex spatio-temporal relationships between features and mobility responses.

Compared to COVID-GAN and MAML-DAWSON, our model outperforms COVID-GAN by 21.8% (RMSE) and 26.1% (MAE) on average, and MAML-DAWSON by 18.8% (RMSE) and 25.3% (MSE) in Houston. The results show that the design of spatio-temporal architecture (i.e., CNN and LSTM substructures and the STTG) and meta-learning adaptation can significantly improve the solution quality. Furthermore, our model achieves 18.1% (RMSE) and 22.2% (MSE) better than MetaST, demonstrating that task-based graph embedding can contribute to model performance by learning the inter-task similarities.

We also evaluate the model performance on less populous areas using Iowa City as a testing city, the POI numbers and city size of Iowa City are significantly smaller than large metropolitan areas according to Table 1. In Table 3, the improvements are relatively smaller due to the smaller number of POI visit counts in a less populated city. Moreover, the study area for sparse regions is small, which lead to data scarcity issue. However, both STORM-GAN+ scenarios still achieve the lowest errors in all of the testing days.

We calculate the KL-divergence using results from Houston and Iowa City (Fig. 7). The X-axis represents the number of equal-size bins used to discretize the value needed for the computation, and the Y-axis shows the KL divergence values. A lower KL divergence value means the result better matches the real distribution. As shown in Fig. 7, STORM-GAN+ achieves the lowest KL-divergence compared to the baseline methods consistently for all numbers of bins.

5.5.2 Impact of STTG choice

Our results show that both of the two STTG constructed can significantly improve the performance of STORM-GAN+ in Houston and Iowa City. This proves that the spatio-temporal task-graph embedding design is effective and robust, rather than tailored for a specific STTG definition. Between the two choices, S2 slightly achieves better performance as it uses more information that are directly related to the spreading of COVID-19.

5.5.3 Ablation study

We study the effect of different components proposed in our method using Houston as the testing city on one day (Monday).

- **Base:** Baseline conditional GAN.
- **Base + Spatial (S):** Equivalent to COVID-GAN, which has a correction layer to add policy constraints, but purely a spatial model.
- **Base + Spatio-Temporal (ST) + Meta:** STORM-GAN with spatio-temporal meta-learning, but without the STTG graph.
- **Base + ST + Meta + Graph(S2):** Complete STORM-GAN.
- **Base + ST + new Meta + new Graph(S2):** STORM-GAN+.
- **Base + ST + conditional Meta + new Graph(S2):** Complete STORM-GAN+ with conditional meta-learning.

Table 4 shows the estimation performance of STORM-GAN+ and its variants. First, the base + spatial (S) achieves a lower RMSE and MAE (a reduction of 15.3% and 15.2%, respectively) compared to cGAN, showing the effectiveness of the correction layer from COVID-GAN. Next, we can see that the addition of spatio-temporal meta-learning further reduces RMSE and MAE by 12.7% and 10%, respectively. This result demonstrates that CNN, LSTM and meta-learning can better capture the complex spatio-temporal relationships across multiple cities. Furthermore, the complete STORM-GAN improves the RMSE and MAE by 3.1% and 8.6%, respectively. Finally, the complete STORM-GAN+ achieves the lowest RMSE and MAE with the spatio-temporal task-based graph with an improvement of 3.2% and 6.3%, and the conditional meta-learning strategy achieves better performance than the standard meta-learning training paradigm.

5.5.4 Visualization

We compare the solution quality of eight candidate approaches through map visualization. Figures 8 and 9a–g show the results of baseline methods, and (h), (i) display the STORM-GAN+ (S2) and ground truth. The results show the full Houston and Iowa City study areas for a day in the data. Here STORM-GAN+ generates fine-scale mobility values that are closer to the ground truth. As we can see, the mobility pattern generated by the STORM-GAN+ can

capture the spatial pattern of human mobility responses better than other baselines. The reason may be that similar functionality zones in different cities may have similar mobility patterns. The meta-learning framework successfully learns this shared knowledge from training tasks. Moreover, the utilization of CNN and LSTM helps capture the spatio-temporal correlation from region to region.

5.5.5 STORM-GAN+ vs. STORM-GAN

When comparing the results between STORM-GAN and STORM-GAN+, as we can see in Figs. 8g, h and 9g, h, the generated human mobility patterns of STORM-GAN+ can better approximate the mobility distributions in the ground truth than the results of the STORM-GAN method in (g), where a pattern with a large number of cells around the downtown dense region is significantly improved and more accurately depicted in both cities. Furthermore, Fig. 9 shows the Iowa City results of STORM-GAN+ in (h), STORM-GAN (g), and ground truth (i), respectively. The red circles highlight two examples of the differences between STORM-GAN and STORM-GAN+. It is evident that downtown dense zones of STORM-GAN+ are much closer to the ground truth compared to STORM-GAN result.

According to the statistical results in Tables 2 and 3, the RMSE of STORM-GAN+ improved by 7% in Houston and improved by 17% in Iowa City. As previously mentioned, the limitations of the original STORM-GAN are more apparent in sparse regions. The results demonstrate that STORM-GAN underestimates human mobility responses in sparse regions due to the smaller dataset. The results prove that adopting the new meta-learning objective functions and weighted STTG learning strategy can significantly improve the model's capacity for capturing spatial heterogeneity; at the same time, these improvements can reduce the difficulty of learning in sparse regions and enhance the estimation quality overall.

5.5.6 Temporally seen vs. temporally unseen

In this comparison, temporally seen refers to the time periods of task (i.e., days, weeks) in the training data and have been seen by STORM-GAN+ during the meta-training phase. In contrast, temporally unseen refers to data of days or weeks that are outside the training data in the meta-training phase.

Given different condition combinations, we expect our model to generalize reliable cross-city human mobility. The purpose of temporally unseen vs. temporally seen is to examine the model estimation accuracy given new temporally unseen conditions. To make the test more realistic, the timestamp of unseen data must also be strictly after all timestamps in the training data so that the model does not try to estimate the past based on the future. Therefore, we always feed prior conditions to estimate the following day's mobility.

According to Sect. 5.1.4, there are seven temporally consecutive tasks for each training city and 35 tasks in total for meta-training. In the temporally unseen vs. temporally seen experiment, we use the first six tasks of each training city and 30 tasks in total for meta-training to train the candidate models and leave the 7th tasks out for each city as "temporally unseen" data. We use 7th tasks from the testing city to evaluate the model performance with unseen conditions temporally.

The results are shown in Figs. 10 and 11; we evaluate the results for Houston and Iowa City, respectively. The first row is the temporally unseen results for (a) ground truth, (b) STORM-GAN+, and (c) STORM-GAN, and the second row is the temporally seen results. We can see that STORM-GAN+ can maintain a good description of the overall pattern given

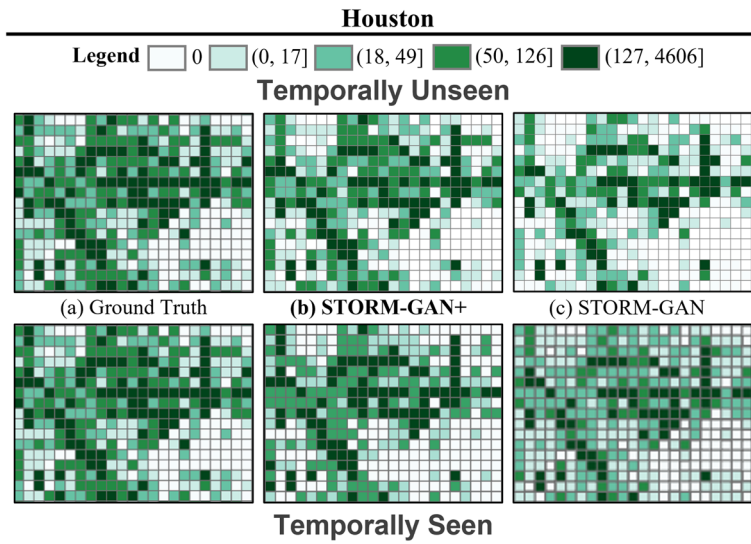


Fig. 10 Mobility estimation results in a sub-region of Houston for temporally unseen (first row) and seen regions (second row) data

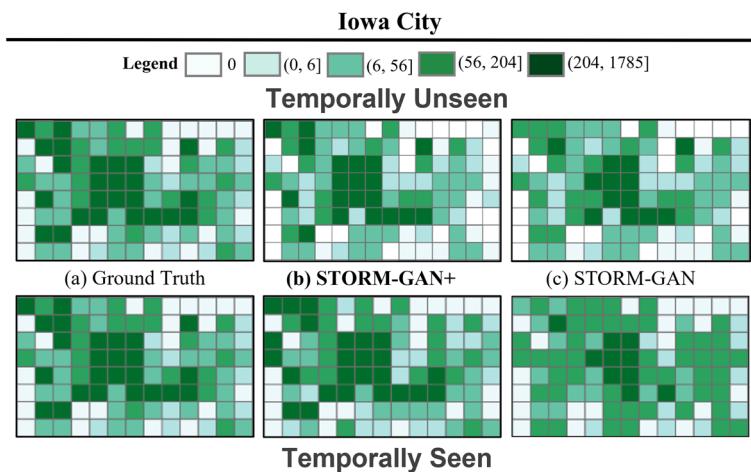


Fig. 11 Mobility estimation results in a sub-region of Iowa City for temporally unseen (first row) and seen regions (second row) data

temporally unseen conditions. The grid cells on the center of the sub-region in Figs. 10 and 11 from the STORM-GAN+ model have a higher solution quality, while STORM-GAN gives a blur description and underestimation of this sub-region. This is especially important in assisting policy-making when a model is used to estimate human mobility responses given unseen conditions from unseen cities.

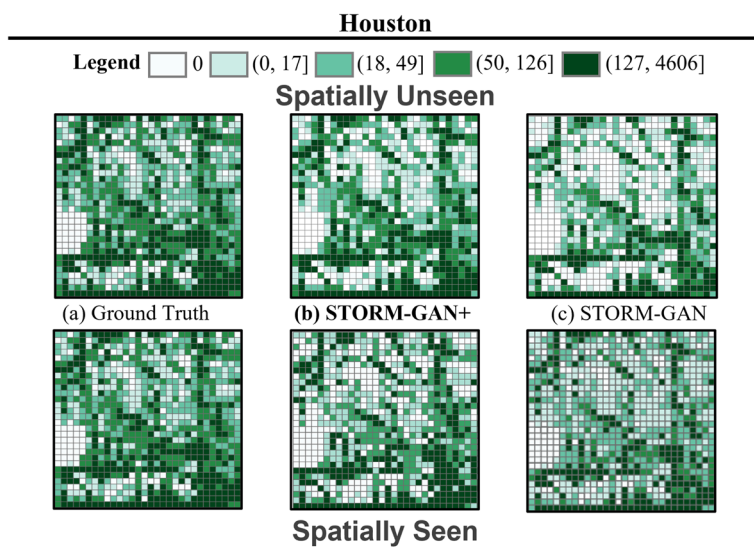


Fig. 12 Mobility estimation results in spatially unseen (size of $30\text{ km} \times 30\text{ km}$) and seen regions of Houston

5.5.7 Spatially seen vs. spatially unseen

In this experiment, spatially seen refers to the study areas of the tasks from testing cities and has been seen by STORM-GAN+ during the meta-testing fine-tuning phase. In contrast, spatially unseen refers to study areas that are outside the training data in the meta-testing fine-tuning phase. This experiment aims to explore the STORM-GAN+ generalization capacity in a zero-shot learning scenario.

To generate spatially unseen regions, we crop a $30\text{ km} \times 30\text{ km}$ sub-region (i.e., a 30×30 sub-grid) off the total geographic space in Houston and crop a $15\text{ km} \times 15\text{ km}$ sub-region from Iowa City, and condition combinations in these sub-regions are not seen by STORM-GAN+ during the fine-tuning in the meta-testing phase. More specifically, the cropped study areas are taken out from $\mathcal{D}_{new}^{train}$, and the study area can be seen in $\mathcal{D}_{new}^{test}$ referring to Fig. 3. This eliminates about one-third of the total amount of training samples (overlaps with the sub-grid are not allowed) for both cities. Figures 12 and 13 show the comparison of results by STORM-GAN+ and STORM-GAN in Houston and Iowa City, respectively. The first row displays the spatially unseen results for (a) ground truth, (b) STORM-GAN+, and (c) STORM-GAN, and the second row shows the spatially seen results.

When comparing the two rows of results in Houston, STORM-GAN+ can still produce a good estimate for the first row, and STORM-GAN's results are unable to accurately approximate the mobility distribution's finer details when the data are spatially hidden. It is evident that STORM-GAN+ can capture human mobility patterns and clearly generate high-mobility grid cells, and STORM-GAN underestimates the mobility of the overall sub-region. If we omit a portion of the city's region, the STORM-GAN model would be unable to function because of the lack of data. Thus, STORM-GAN should be utilized when prior data are provided for fine-tuning the entire city. However, STORM-GAN+ can estimate the human mobility pattern for the unseen city when no prior data are available because the cropped areas are hidden during the meta-training and fine-tuning process.

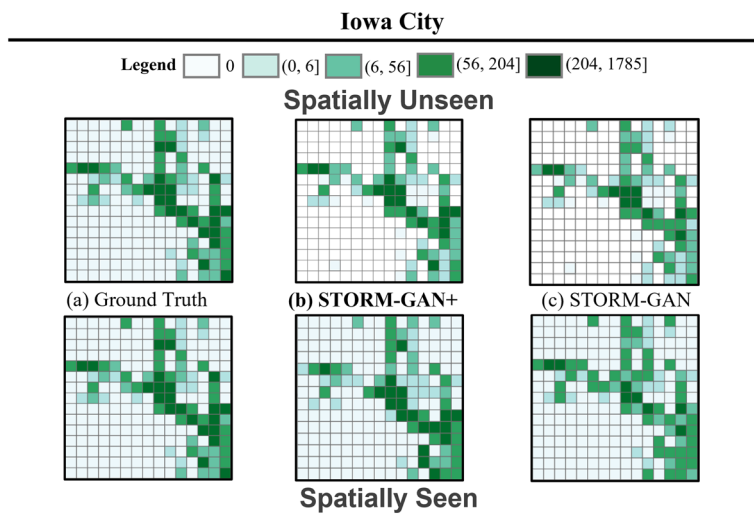


Fig. 13 Mobility estimation results in spatially unseen (size of 15 km × 15 km) and seen regions of Iowa City

5.6 Conditional meta-learning vs. unconditional meta-learning

To mitigate the impact of task heterogeneity, STORM-GAN+'s conditional meta-learning strategy assigns a cluster to every task and trains a conditional meta-parameter based on the cluster assignment to improve estimation performance. Since the granularity of human mobility data is daily, we depict the spatial-temporal pattern of each task using the average daily human mobility pattern over a week. We cluster these patterns into four categories using K-means. In this experiment, we examine the effectiveness of conditional meta-learning and test under different scenarios.

Overall, the performance of STORM-GAN+ improves the estimation performance by 3.4% for RMSE and 7.2% for MAE compared to STORM-GAN. In addition, conditional cluster-based meta-learning facilitates the learning of a further improved and specialized initialization for the unobserved task. The ideal degree of knowledge transfer stability is achieved. Since the data were collected throughout the same time period, cluster similarity regularization provides a flexible and effective method of knowledge transfer.

6 Related works

6.1 Understand the mobility pattern during COVID-19

There have been many studies [1, 22–25] exploring the interplay between human mobility responses, social distancing policies, and transmission dynamics in response to the COVID-19 pandemic. For example, it was shown by [1] that strict implementation of social distancing policies can reduce mobility and substantially mitigate the spread of COVID-19. A US mobility change map was created in [24] to increase public risk awareness and visualize dynamic changes in mobility as the COVID-19 situation and policy evolve. Study [25] measures the effectiveness of non-pharmaceutical interventions (NPIs) introduced by governments across Europe using the changes in mobility. Studies [26, 27] have also explored the feasibility of

utilizing contact tracing to control the spread of the disease through simulated synthetic data and real-world smartphone trajectories.

These studies are timely in showing the critical role played by mobility in the spread of COVID-19, but they do not address the challenges in real-world mobility estimation/simulation (e.g., effects of unknown, uncertain, and random factors), and they analyze the mobility changes in city or country scale. A study [28] simulated human mobility, allowing policymakers to inspect mobility changes under different policies. But this approach utilizes a traditional epidemiological model and does not transfer the simulation from city to city by shared knowledge. Another study [29] proposed a deep neural network model to capture spatio-temporal information from human mobility data through a straight forward parameter-sharing method and transfer from one city to another. However, these studies have yet to explore the potential use of deep learning based generative models and meta-learning to assist the estimation.

6.2 Deep learning for spatio-temporal prediction

Many deep learning-based techniques have been developed for spatio-temporal data. For example, LSTMs were widely used in traffic accident prediction [30] and flow prediction [31], due to their capability to capture spatio-temporal correlation and thus provide good prediction results. Geospatial object mapping [32–34], taxi driver behavior imitation [35], taxi demand [36], travel time estimation [37], etc, they all combine the deep learning model with spatio-temporal perspective in their model design and obtain good performance. Most of these techniques typically are stationary predictors (i.e., the same result from two runs on the same data) rather than generative models. They do not consider unknown factors in prediction; their performance relies on large data sets. Besides, they do not leverage domain knowledge-based constraints to assist learning (e.g., cGAN [2, 21]).

6.3 Generative adversarial networks (GANs)

GANs were proposed by [9] and have achieved great performance in the image generation domain, including image-to-image translation [38], image super-resolution [39], and text-to-image synthesis [40]. Despite the success, a critical issue for GANs is known to be the unstable and sensitivity to the choices of hyperparameters in the learning process. Several works have attempted to address the GANs training problem and to improve the stability by designing new network architectures [41], modifying the learning objectives and dynamics [42], adding regularization methods to obtain stable gradients [43]. Besides image generation, recently, deep graph generative adversarial structure has been developed based on the concept of unsupervised learning. Existing architectures build upon generative models including GraphVAE [44] and GraphGAN [45], and they achieved good performance. However, the generative model has not been applied to estimating human mobility problems as well as other human-related movement research. For example, [21] uses a conditional generative adversarial network (cGAN) to estimate traffic volume. A POI embedding transfer learning approach is proposed [29] to predict urban traffic from one city to another city. This approach adapts model parameters without using the meta-learning method.

6.4 Meta-learning in spatio-temporal data mining

Meta-learning learns new tasks quickly and effectively with a few examples. Existing optimization-based meta-learning algorithms such as MAML [6] and Reptile [46] rely on optimization through gradient descent, and both are compatible with any model. MAML produces a good initialization toward a new task with a few steps of gradient updates, and Reptile is iteratively trained on a sampled task by multiple gradient steps. Recently, the idea of optimization-based meta-learning has been applied to many domains including classification and reinforcement learning. However, there only a few works address the spatial and temporal problems simultaneously. In traffic prediction, a recent work [5] focuses on knowledge transfer in a single city, which only deals with temporal tasks with no spatial-based tasks. [47] proposes a transfer learning framework for traffic prediction through a learning region matching function. Another work [4], which is based on multiple cities, does not consider temporal patterns and dynamic scenarios. This model is designed with no time-based tasks, which is insufficient to model the continued and dynamic changes.

7 Conclusions

We tackled the **heterogeneous** human mobility estimation problem through a spatio-temporal meta-generative framework. Specifically, we proposed a STORM-GAN+ model to capture complex spatio-temporal patterns as an extension of STORM-GAN. Building upon STORM-GAN, we introduced an enhanced spatio-temporal task-based graph (STTG) to represent the spatio-temporal relationships across tasks, with a graph convolution network to learn embedding of its subgraph for cross-task learning enhancements. Moreover, STORM-GAN+ utilized a redesigned conditional meta-learning structure and improved the objective to learn shared knowledge from a spatio-temporal distribution of estimation tasks and can quickly adapt to new tasks (e.g., new cities). The experiment results showed that our proposed approach could significantly improve the estimation performance compared to baselines, including STORM-GAN. The model can assist policymakers in better understand the dynamic mobility pattern changes under different social and policy conditions and can potentially be leveraged to inform decisions in resource allocation and provisioning, event planning, response management, etc.

Acknowledgements This paper is funded in part by Safety Research using Simulation University Transportation Center (SAFER-SIM). SAFER-SIM is funded by a grant from the U.S. Department of Transportation's University Transportation Centers Program (69A3551747131). However, the U.S. Government assumes no liability for the contents or use thereof. Yiqun Xie is supported in part by NSF Grants 2105133, 2126474, 2147195, Google's AI for Social Good Impact Scholars program, and the DRI award at the University of Maryland; and Xiaowei Jia is supported in part by NSF award 2147195, USGS award G21AC10207, Pitt Momentum Funds award, and CRC at the University of Pittsburgh. Yanhua Li was supported in part by NSF Grants IIS-1942680 (CAREER), CNS-1952085, CMMI-1831140, and DGE-2021871.

Author Contributions Han Bao wrote the main manuscript, implemented the algorithm, and conducted experiments. Han Bao, Xun Zhou, Yiqun Xie, Yanhua Li, and Xiaowei Jia contributed to the development of the proposed method. Xun Zhou, Yiqun Xie, Yanhua Li, and Xiaowei Jia reviewed and improved the article.

Declarations

Competing interests The authors declare no competing interests.

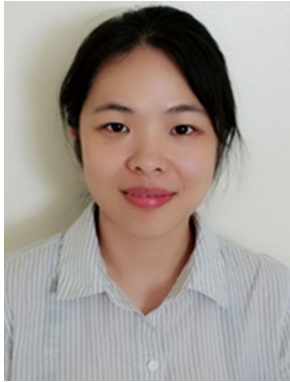
References

1. Kraemer MU, Yang C-H, Gutierrez B, Wu C-H, Klein B, Pigott DM, Du Plessis L, Faria NR, Li R, Hanage WP et al (2020) The effect of human mobility and control measures on the covid-19 epidemic in china. *Science* 368(6490):493–497
2. Bao H, Zhou X, Zhang Y, Li Y, Xie Y (2020) Covid-gan: estimating human mobility responses to covid-19 pandemic through spatio-temporal conditional generative adversarial networks. In: *Proceedings of the 28th international conference on advances in geographic information systems. SIGSPATIAL '20*, pp 273–282. Association for Computing Machinery, New York, NY, USA. <https://doi.org/10.1145/3397536.3422261>
3. Liang W, Liu Z, Liu C (2020) Dawson: A domain adaptive few shot generation framework. *arXiv preprint arXiv:2001.00576*
4. Yao H, Liu Y, Wei Y, Tang X, Li Z (2019) Learning from multiple cities: a meta-learning approach for spatial-temporal prediction. In: *The World Wide Web conference*, pp 2181–2191
5. Zhang Y, Li Y, Zhou X, Luo J (2020) cst-ml: Continuous spatial-temporal meta-learning for traffic dynamics prediction. In: *2020 IEEE international conference on data mining (ICDM)*, pp 1418–1423. IEEE
6. Finn C, Xu K, Levine S (2018) Probabilistic model-agnostic meta-learning. In: *Advances in neural information processing systems*, 31
7. Bao H, Zhou X, Xie Y, Li Y, Jia X (2022) Storm-gan: spatio-temporal meta-gan for cross-city estimation of human mobility responses to covid-19. In: *2022 IEEE international conference on data mining (ICDM)*, pp 1–10. IEEE
8. Gauthier J (2014) Conditional generative adversarial nets for convolutional face generation. *Class Project for Stanford CS231N: convolutional neural networks for visual recognition. Winter semester 2014(5):2*
9. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: *Advances in neural information processing systems*, pp 2672–2680
10. Kang Y, Gao S, Liang Y, Li M, Rao J, Kruse J (2020) Multiscale dynamic human mobility flow dataset in the us during the covid-19 epidemic. *Sci data* 7(1):1–13
11. Kullback S (1997) *Information theory and statistics*. Courier Corporation
12. Chen M, Wei Z, Huang Z, Ding B, Li Y (2020) Simple and deep graph convolutional networks. In: *International conference on machine learning*, pp 1725–1735
13. Finn C, Abbeel P, Levine S (2017) Model-agnostic meta-learning for fast adaptation of deep networks. In: *International conference on machine learning*, pp 1126–1135
14. Antoniou A, Edwards H, Storkey A (2018) How to train your maml. *arXiv preprint arXiv:1810.09502*
15. Baydin AG, Cornish R, Rubio DM, Schmidt M, Wood F (2017) Online learning rate adaptation with hypergradient descent. *arXiv preprint arXiv:1703.04782*
16. Centers for Disease Control and Prevention. <https://www.cdc.gov/> (2020)
17. American Community Survey (ACS). <https://www.census.gov/programs-surveys/acs> (2020)
18. SafeGraph. <https://www.safegraph.com/> (2020)
19. Getis A (2008) A history of the concept of spatial autocorrelation: a geographer's perspective. *Geogr Anal* 40(3):297–309
20. Hoerl RW (2020) Ridge regression: a historical context. *Technometrics* 62(4):420–425
21. Zhang Y, Li Y, Zhou X, Kong X, Luo J (2019) Trafficgan: off-deployment traffic estimation with traffic generative adversarial networks. In: *2019 IEEE international conference on data mining (ICDM)*, pp 1474–1479. IEEE
22. Soucy J-PR, Sturrock SL, Berry I, Daneman N, MacFadden DR, Brown KA (2020) Estimating the effect of physical distancing on the covid-19 pandemic using an urban mobility index. *medRxiv*
23. Bryant P, Elofsson A (2020) Estimating the impact of mobility patterns on covid-19 infection rates in 11 European countries. *medRxiv*
24. Gao S, Rao J, Kang Y, Liang Y, Kruse J (2020) Mapping county-level mobility pattern changes in the united states in response to covid-19. *SIGSPATIAL Special* 12(1):16–26
25. Chinazzi M, Davis JT, Ajelli M, Gioannini C, Litvinova M, Merler S, AP y Piontti, Mu K, Rossi L, Sun K (2020) The effect of travel restrictions on the spread of the 2019 novel coronavirus (covid-19) outbreak. *Science* 368(6489):395–400
26. Hellewell J, Abbott S, Gimma A, Bosse NI, Jarvis CI, Russell TW, Munday JD, Kucharski AJ, Edmunds WJ, Sun F et al (2020) Feasibility of controlling covid-19 outbreaks by isolation of cases and contacts. *The Lancet Global Health*
27. Cho H, Ippolito D, Yu YW (2020) Contact tracing mobile apps for covid-19: privacy considerations and related trade-offs. *arXiv preprint arXiv:2003.11511*

28. Chang S, Wilson ML, Lewis B, Mehrab Z, Dudakiya KK, Pierson E, Koh PW, Gerardin J, Redbird B, Grusky D, et al (2021) Supporting covid-19 policy response with large-scale mobility-based modeling. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery and data mining, pp 2632–2642
29. Jiang R, Song X, Fan Z, Xia T, Wang Z, Chen Q, Cai Z, Shibasaki R (2021) Transfer urban human mobility via poi embedding over multiple cities. *ACM Trans Data Sci* 2(1):1–26
30. Yuan Z, Zhou X, Yang T (2018) Hetero-convlstm: a deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining, pp 984–992
31. Pan Z, Wang Z, Wang W, Yu Y, Zhang J, Zheng Y (2019) Matrix factorization for spatio-temporal neural networks with applications to urban flow prediction. In: Proceedings of the 28th ACM international conference on information and knowledge management, pp 2683–2691
32. Xie Y, Bao H, Shekhar S, Knight J (2018) A timber framework for mining urban tree inventories using remote sensing datasets. In: 2018 IEEE international conference on data mining (ICDM), pp 1344–1349. IEEE
33. Li Z, Xie Y, Jia X, Stuart K, Delaire C, Skakun S (2023) Point-to-Region Colearning for Poverty Mapping at High Resolution Using Satellite Imagery. *Proceedings of the AAAI Conference on Artificial Intelligence*. 37:14321–14328
34. Xie Y, Cai J, Bhojwani R, Shekhar S, Knight J (2020) A locally-constrained yolo framework for detecting small and densely-distributed building footprints. *Int J Geogr Inf Sci* 34(4):777–801
35. Zhang X, Li Y, Zhou X, Luo J (2019) Unveiling taxi drivers' strategies via cgail: Conditional generative adversarial imitation learning. In: 2019 IEEE international conference on data mining (ICDM), pp 1480–1485. IEEE
36. Zhang C, Zhu F, Lv Y, Ye P, Wang F-Y (2021) Mlrnn: taxi demand prediction based on multi-level deep learning and regional heterogeneity analysis. *IEEE Trans Intell Transport Syst*
37. Xu S, Zhang R, Cheng W, Xu J (2020) Mtlm: a multi-task learning model for travel time estimation. *GeoInformatica*, 1–17
38. Isola P, Zhu J-Y, Zhou T, Efros AA (2017) Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1125–1134
39. Ledig C, Theis L, Huszár F, Caballero J, Cunningham A, Acosta A, Aitken A, Tejani A, Totz J, Wang Z et al (2017) Photo-realistic single image super-resolution using a generative adversarial network. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4681–4690
40. Reed S, Akata Z, Yan X, Logeswaran L, Schiele B, Lee H (2016) Generative adversarial text to image synthesis. *arXiv preprint [arXiv:1605.05396](https://arxiv.org/abs/1605.05396)*
41. Karras T, Laine S, Aila T (2019) A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4401–4410
42. Mao X, Li Q, Xie H, Lau RY, Wang Z, Paul Smolley S (2017) Least squares generative adversarial networks. In: Proceedings of the IEEE international conference on computer vision, pp 2794–2802
43. Che T, Li Y, Jacob AP, Bengio Y, Li W (2016) Mode regularized generative adversarial networks. *arXiv preprint [arXiv:1612.02136](https://arxiv.org/abs/1612.02136)*
44. Simonovsky M, Komodakis N (2018) Graphvae: towards generation of small graphs using variational autoencoders. In: International conference on artificial neural networks, pp 412–422. Springer
45. Wang H, Wang J, Wang J, Zhao M, Zhang W, Zhang F, Xie X, Guo M (2017) Graphgan: graph representation learning with generative adversarial nets. *arXiv preprint [arXiv:1711.08267](https://arxiv.org/abs/1711.08267)*
46. Nichol A, Schulman J (2018) Reptile: a scalable metalearning algorithm. *arXiv preprint [arXiv:1803.02999](https://arxiv.org/abs/1803.02999)*
47. Wang L, Geng X, Ma X, Liu F, Yang Q (2018) Cross-city transfer learning for deep spatio-temporal prediction. *arXiv preprint [arXiv:1802.00386](https://arxiv.org/abs/1802.00386)*

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

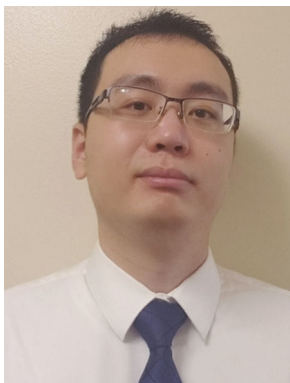
Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.



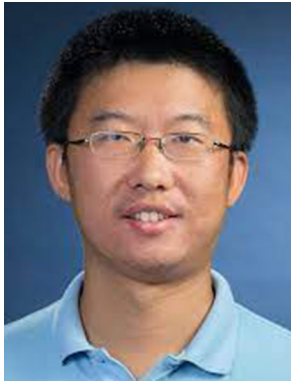
Han Bao is working towards a PhD degree in Informatics at the University of Iowa. Her research focuses on developing novel data mining and AI techniques for spatio-temporal big data, with applications in smart cities, transportation, public health, etc. She received the Best Paper Award from IEEE ICDM 2021.



Xun Zhou is an Associate Professor and Henry B. Tippie Research Fellow in the Department of Business Analytics at the University of Iowa. He received a Ph.D. degree in Computer Science from the University of Minnesota in 2014. His research interests include big data analytics and management, spatial and spatio-temporal data mining, machine learning, and business analytics. He has received best paper awards from several conferences including ICDM'21, SDM'19, and SSTD'11. He also co-edited the Encyclopedia of GIS, 2nd Edition.



Yiqun Xie received his Ph.D. degree in Computer Science from the University of Minnesota. He is currently an Assistant Professor in Geospatial Information Science at the University of Maryland, College Park. His research focuses on data mining and artificial intelligence methods for spatial data, such as satellite remote sensing data, UAV imagery, trajectories, etc. His recent research results have received the Best Application Paper Award from SDM 2023, the Best Paper Award from IEEE ICDM 2021, the Best Vision Paper Award from SIGSPATIAL 2019, and the Best Paper Award from SSTD 2019. His work was also highlighted by the Great Innovative Ideas program at the Computing Community Consortium at CRA.



Yanhua Li received two Ph.D. degrees in computer science from University of Minnesota at Twin Cities in 2013, and in electrical engineering from Beijing University of Posts and Telecommunications in China in 2009, respectively. He is currently an Associate Professor in the Department of Computer Science at Worcester Polytechnic Institute (WPI) in Worcester, MA. Prior to this, he worked in Bell Labs in New Jersey, and Microsoft Research Asia in Beijing. His research interests are artificial intelligence and spatial temporal data mining in many contexts, including urban intelligence, smart cities, and urban planning and optimization. He is a recipient of NSF CAREER award and NSF CRII award.



Xiaowei Jia is an Assistant Professor in the Department of Computer Science at the University of Pittsburgh. His research interests include knowledge-guided data science and spatio-temporal data mining for societally important applications. A major highlight of his research is a general paradigm Physics-Guided Machine Learning for combining machine learning and physics-based modeling approaches. He is the recipient of the University of Minnesota Best Dissertation Award and Best Paper Awards from SDM 22, ICDM 21, SDM 21, ASONAM 16, and BIBE 14.