Generating Synthetic Mobility Networks with Generative Adversarial Networks

Giovanni Mauro¹²³, Massimiliano Luca⁴⁶, Antonio Longa⁵⁶, Bruno Lepri⁶, and Luca Pappalardo¹

¹Institute of Information Science and Technologies, National Research Council (ISTI- CNR), Italy
²University of Pisa
³IMT School for Advanced Studies, Lucca, Italy
⁴Free University of Bolzano, Italy
⁵University of Trento, Italy
⁶Fondazione Bruno Kessler (FBK), Italy

Abstract

The increasingly crucial role of human displacements in complex societal phenomena, such as traffic congestion, segregation, and the diffusion of epidemics, is attracting the interest of scientists from several disciplines. In this article, we address mobility network generation, i.e., generating a city’s entire mobility network, a weighted directed graph in which nodes are geographic locations and weighted edges represent people’s movements between those locations, thus describing the entire mobility set flows within a city. Our solution is MoGAN, a model based on Generative Adversarial Networks (GANs) to generate realistic mobility networks. We conduct extensive experiments on public datasets of bike and taxi rides to show that MoGAN outperforms the classical Gravity and Radiation models regarding the realism of the generated networks. Our model can be used for data augmentation and performing simulations and what-if analysis.
1 Introduction

The increasing complexity of urban environments \[1, 2\] and the crucial role played by human displacements in the diffusion of epidemics, not least the COVID-19 pandemic \[3, 4, 5, 6, 7, 8\], have created a great deal of interest around the study of individual and collective human mobility \[9, 10, 11\]. The prevention of detrimental collective phenomena such as traffic congestion, air pollution, segregation, and epidemics spread, which is crucial to make our cities inclusive, safe, resilient, and sustainable \[12, 13, 14\], depends on how accurately we can predict and simulate people’s movements within an urban environment.

In this regard, a particularly challenging task is generating realistic mobility flows, i.e., flows of people among a set of geographic locations given their demographic and geographic characteristics (e.g., population and distance) \[9, 15, 11, 10, 16\]. Traditionally, flow generation is addressed through the Gravity model \[17, 18, 19, 11, 20\], the Radiation model \[21, 11, 10\], and their variants \[15, 11, 22, 23\]. The Gravity model assumes that the number of travelers between two locations (flow) increases with the locations’ populations while decreasing with the distance between them. The Radiation model is a parameter-free model that only requires information about geographic locations (e.g., population) and their intervening opportunities. The Gravity and the Radiation models are designed to generate single flows between pairs of locations and are typically used to complete a network in which some mobility flows are missing.

In this paper, we address mobility network generation, a variation of flow generation that consists in generating a city’s entire mobility network. A mobility network is a weighted directed graph in which nodes are geographic locations and weighted edges represent people’s movements between those locations, thus describing the entire set of mobility flows within a city.

Our solution to mobility network generation – MoGAN (Mobility Generative Adversarial Network) – is based on Generative Adversarial Networks (GANs) \[24\], deep learning architectures composed of a discriminator, which maximizes the probability to classify real and artificial mobility networks correctly, and a generator, which maximizes the probability to fool the discriminator producing artificial mobility networks classified by the discriminator as real. The choice of GANs is motivated by the fact that mobility networks can be represented as weighted adjacency matrices, similarly to how images are typically represented, and considering that GANs are tremendously effective in generating realistic images \[24, 25, 26, 27\]. While several papers show that GANs can generate individual mobility trajectories \[9, 28, 29, 30, 31, 32, 33, 34\] with a realism comparable to or better than mechanistic mobility models \[35, 36, 37, 11\], to what extent GANs can generate realistic mobility flows has never been explored in the literature.

We train MoGAN on a set of real mobility networks and develop a tailored evaluation methodology to test the model’s effectiveness in generating realistic mobility...
networks. We conduct extensive experiments on four public mobility datasets, describing flows of bikes and taxis in New York City and Chicago, US, to demonstrate that MoGAN generates synthetic mobility networks that are way more realistic than those generated by the Gravity and the Radiation model. Our results prove that our model can synthesize aggregated movements within a city into a realistic generator, which can be used for data augmentation and performing simulations and what-if analysis.

2 Mobility network generation

Mobility network generation consists of generating a realistic mobility network, i.e., a weighted directed graph in which nodes are locations and edges represent flows between those locations. The locations are defined by a discretization of the geographic space defined by a spatial tessellation, i.e., a covering of the bi-dimensional space using a countable number of geometric shapes called tiles, with no overlaps and no gaps. In mobility networks, nodes are tiles of the spatial tessellation and edges flows of people among these tiles.

Formally, we define a mobility network as a weighted directed graph $G = (V, E, w)$, where:

- $V$ is the set of nodes, i.e., tiles of the spatial tessellation;
- $w : V \times V \mapsto \mathbb{N}$ is a function that assigns to each pair of nodes the number of people moving between the two nodes (mobility flow);
- $E = \{(x, y) | (x, y) \in V \times V \land w(x, y) \neq 0\}$ is the set of the weighted directed edges in the network.

A mobility network may contain self-loops (edges in which the origin and destination coincide), which describe movements of people within the same tile. Here, we represent a mobility network as a weighted adjacency matrix $A_{n \times n}$ with $n = |V|$. Thus, an element $a_{i,j} \in A$ represents the number of people moving from node $i$ to node $j$, with $i, j \in V$.

A generative model of mobility networks $M$ is any algorithm able to generate a set of $n$ synthetic mobility networks $\mathcal{X}_M = \{\hat{G}_1, \ldots, \hat{G}_n\}$, which describe the set of mobility flows on a given spatial tessellation. The realism of $M$ is evaluated with respect to:

1. A set of network patterns $\mathcal{K} = \{s_1, \ldots, s_m\}$ that describe some statistical properties of mobility networks. A realistic set $\mathcal{T}_M$ of synthetic mobility networks is expected to reproduce as many of these mobility patterns as possible.
2. A set $\mathcal{X} = \{G_1, \ldots, G_n\}$ of real mobility networks that describe real flows on the same spatial tessellation. Typically, a portion $\mathcal{X}_{\text{train}} \subset \mathcal{X}$ is used to train $M$ or to fit its parameters. The remaining part $\mathcal{X}_{\text{test}} \subset \mathcal{X}$ is used to compute the set $\mathcal{K}$ of patterns, which are compared with the patterns computed on $\mathcal{X}_M$.

3. A function $D$ that computes the dissimilarity between two distributions. Specifically, for each measure in $f \in \mathcal{K}$, $D(P_{(f,X_M)}||P_{(f,X_{\text{test}})})$ indicates the dissimilarity between $P_{(f,X_M)}$, the distribution of the measures computed on the synthetic mobility networks in $\mathcal{X}_M$, and $P_{(f,X_{\text{test}})}$, the distribution of the measures computed on the mobility networks in $\mathcal{X}_{\text{test}}$. The lower $D(P_{(f,X_M)}||P_{(f,X_{\text{test}})})$, the more realistic model $M$ is with respect to $f$ and $\mathcal{X}_{\text{test}}$.

3 MoGAN: A Mobility Generative Adversarial Network

To solve the problem of mobility network generation, we design MoGAN (Mobility Generative Adversarial Network), a deep learning architecture based on Deep Convolutional Generative Adversarial Networks (DCGANs) [27]. MoGAN consists of a generator $G$, which learns how to produce new synthetic mobility networks, and a discriminator $D$, which has the task of distinguishing between real and fake (artificial) mobility networks. $G$ and $D$ are trained in an adversarial manner: $D$ maximizes the probability to correctly classify real and fake mobility networks; $G$ maximizes the probability to fool $D$, i.e., to produce fake mobility networks classified by $D$ as real. Both $D$ and $G$ are Convolutional Neural Networks (CNNs), which are proven to be effective in capturing spatial patterns in the data [9].

During the training phase, $G$ repeatedly takes a $1 \times 100$ noise vector as input and operates a series of transposed convolutions, which perform upsampling of the input vector to generate a $64 \times 64$ adjacency matrix representing a mobility network. Then, $D$ takes a set of real and generated $64 \times 64$ matrices as input and performs a binary classification task to classify these matrices as real or fake. The above process is repeated for a certain number of epochs and stopped when some criteria are met (see Supplementary Note 1). Once MoGAN is trained, $G$ can be used to generate as many mobility networks as desired. A visual representation of the networks generated during the training phase can be found in Supplementary Note 2. Figure 1 schematizes and describes MoGAN’s architecture. Further details on MoGAN’s architecture and training can be found in Supplementary Note 1.
Figure 1: **Architecture of MoGAN**: The generator (a Convolutional Neural Network or CNN) performs transposed convolution operations that upsample the input random noise vector, transforming it into a $64 \times 64$ adjacency matrix representing a mobility network. The discriminator (a CNN) takes as input both the generated mobility networks and the real ones from the training set and performs a series of convolutional operations that end up with a probability, for each sample, to be fake or real. Both the discriminator and generator weights are then backpropagated.

### 4 Baseline models

We compare MoGAN with the Gravity and the Radiation models, two classical approaches for mobility flows’ generation [11, 9, 15, 21].

The singly-constrained Gravity model [17, 18, 19, 11] prescribes that the expected flow, $\bar{y}$, between an origin location $l_i$ and a destination location $l_j$ is generated according to the following equation:

$$
\bar{y}(l_i, l_j) = O_i p_{ij} = O_i \frac{m_j^{\beta_1} f(r_{ij})}{\sum_k m_k^{\beta_1} f(r_{ik})}
$$

(1)

where $m_j$ is the population of location $l_j$, $p_{ij}$ is the probability to observe a trip (unit flow) from location $l_i$ to location $l_j$, $\beta_1$ is a parameter and $f(r_{ij})$ is the deterrence function, which is a function of the distance $r_{ij}$ between two locations. Typically, the deterrence function $f(r_{ij})$ can be either an exponential, $f(r) = e^{\beta_2 r}$, or a power-law function, $f(r) = r^{\beta_2}$, where $\beta_2$ is another parameter. These parameters can be fitted from a subset of available flows.

The Radiation model [21, 11] is a parameter-free model that aims to generate flows between locations given their characteristics (e.g., population) and the intervening opportunities among them. The choice of the destination consists of two
steps: (i) we assign a fitness $z$ to each location opportunity sampled from a distribution $p(z)$ that represents the quality of the opportunity for each travel; (ii) the traveler ranks the opportunities according to their distance from the origin location and chooses the nearest location with a fitness higher than a certain threshold. As a result, the mean flow between two locations $l_i$ and $l_j$ is calculated as:

$$\bar{y}(l_i, l_j) = O_i \frac{1}{1 - \frac{m_i}{M}} \frac{m_i m_j}{(m_i + m_j + s_{ij})(m_i + m_j + s_{ij})} \quad (2)$$

where $O_i$ is the number of people leaving location $l_i$, $m_i$ and $m_j$ are the opportunities in $l_i$ and $l_j$, $M$ is the sum of all the opportunities, and $s_{ij}$ is the number of opportunities in a circle of radius $r_{i,j}$.

Note that the Gravity and the Radiation models do not solve mobility network generation directly. While MoGAN, once trained, can generate an entire mobility network, the Gravity and the Radiation models are designed to generate single flows between pairs of locations. To generate a mobility network using the Gravity and the Radiation models, we proceed as follows: (i) we take a real mobility network; (ii) for each node, we compute its relevance $m_i$ and total outflow $O_i$; and (iii) we use $m_i$ and $O_i$ in Equations 1 and 2. For the Gravity model, we also fit parameters $\beta_1$ and $\beta_2$ from the real mobility network assuming a power-law deterrence function. For both the Gravity and Radiation models, we use the implementations available in the library scikit-mobility [38], which provides methods to fit parameters and generate flows from locations’ relevance and outflow.

5 Experimental setup

5.1 Datasets

To train/test MoGAN and the baselines we use four real-world public datasets, which describe trips with taxis and bikes in New York City and Chicago during 2018 and 2019 (730 days). Two datasets contain daily information regarding the use of bike-sharing services: the City Bike Dataset for New York City [39] and the Divvy Bike Dataset for Chicago [40]. Each record describes the coordinates of each ride’s starting and ending station, and the starting and ending times. We remove trips with a duration lower than 60 seconds because they could be false starts or users trying to re-dock a bike to ensure it is secure [39, 40]. We also use two datasets containing daily information about the movements of taxis: the New York City taxi dataset [41] and the Chicago taxi dataset [42]. A record describes each ride’s starting and ending location and the starting and ending times. Both datasets are already preprocessed to remove dummy and noisy rides. In the Chicago taxi dataset, we know the GPS points corresponding to the starting and ending points of each taxi trajectory. In the New York City taxi dataset, we only know the trajectories’
starting and ending zones, i.e., administrative areas in New York City. We use an administrative area’s centroid as a taxi ride’s reference starting or ending point. We select the island of Manhattan for New York City and the central districts for Chicago (see Supplementary Figure S3) and split the selected zones into 64 equally-sized squared tiles (1840 meters per side for New York City, 1405 meters per side for Chicago). For each dataset, we count the daily number of taxis or bikes moving between each pair of tiles to obtain an origin-destination matrix representing the daily mobility network. We compute the relevance of each location (tile), which is needed for generating flows in the Gravity and the Radiation models, as the total number of daily drop-offs in that location. Table 1 shows some statistics about the dataset used in our study. As an example, Figure 2 visualizes where bike stations concentrate and a mobility network representing daily flows in Manhattan, New York City.

| dataset    | rides       | locations | #bikes/taxis |
|------------|-------------|-----------|--------------|
| CHI bikes  | 3,505,03    | 198       | 6293         |
| NYC bikes  | 29,294,326  | 509       | 19,514       |
| CHI taxis  | 11,050,936  | 96        | 5,668        |
| NYC taxis  | 157,485,483 | 68        | N.D.         |

Table 1: Statistics of the four datasets used in our study. For each dataset, we provide the link to download it, the number of rides, the different locations, and the number of bikes/taxis. CHI = Chicago, NYC = New York City. For the NYC taxi dataset, taxi identifiers are not available and we do not know the total number of taxis. All datasets refer to trips in 2018 and 2019.

5.2 Validation

We develop a tailored approach to evaluate the realism of the mobility networks generated by MoGAN. For each dataset, we construct a mobility network for each day obtaining 730 real mobility networks in total. We split the 730 networks into a training set (584 networks) and a test set (146 networks). We train MoGAN on the training set and generate 146 synthetic mobility networks (synthetic set). We evaluate the model’s realism computing the difference between each network in the synthetic set and each network in the test set, so obtaining $146 \times 146 = 21,316$ values. If the generated mobility networks are realistic, they should differ from the real networks to the same extent real networks differ between themselves. To stress this aspect, we create a set of 146 mobility networks (mixed set), in which half of them are chosen uniformly at random from the test set, and the other half is
chosen uniformly at random from the synthetic set. We then compute the pairwise difference between any possible pair of mobility networks in the mixed set.

A crucial aspect is how to compute the difference between two mobility networks, considering that directed weighted networks are hard to compare, even in the case of known-node correspondence (i.e., networks with the same nodes but different edges) [43]. We compute this difference in two ways.

The first one consists of computing an error metric between two networks’ adjacency matrices. In our experiments, we try three error metrics: RMSE, CPC, and CD. The Root Mean Square Error (RMSE) [9, 15] is defined as:

$$RMSE(A, B) = \sqrt{\frac{1}{n} \sum_{i,j=1}^{n} (a_{ij} - b_{ij})^2}$$
where \( a_{ij} \) and \( b_{ij} \) are the elements (flows) in position \((i, j)\) in the two networks’ adjacency matrices of \( A \) and \( B \) and \( n \) is the number of elements of the matrices \((64 \times 64)\). Note that RMSE is substantially equivalent to the Frobenious norm (see Supplementary Note 4).

The Common Part of Commuters (CPC), also known as Sørensen-Dice index \([11, 15, 44, 15]\), a well-established measure to compute the similarity between real and generated, is defined as:

\[
CPC(A, B) = \frac{2 \left( \sum_{i,j=1}^{n} \min(a_{ij}, b_{ij}) \right)}{\sum_{i,j=1}^{n} a_{ij} + \sum_{i,j=1}^{n} b_{ij}}
\]

The Cut Distance (CD) \([45]\) is based on the notion of cut weight, widely used in network theory \([43]\), and measures how much a network is bipartite. The cut norm \( \| A \|_C \) of a real matrix \( A = (a_{ij}), i \in R, j \in S \) with a set of rows indexed by \( R \) and a set of columns indexed by \( S \), is the maximum over all \( I \subset R, J \subset S \) of the quantity \( |\sum_{i \in I, j \in J} a_{ij}| \). The Cut Distance (CD) between two adjacency matrices \( A \) and \( B \) is the cut norm of their difference:

\[
CD(A, B) = \max_{S \in V} \frac{1}{|V|} |e_A(S, S^C) - e_B(S, S^C)|
\]

with \( V \) being the number of nodes (64, in our case), \( e_G(S, T) = \sum_{i \in S, j \in T} w_{ij} \) is the cut weight of adjacency matrix \( G \) with weights \( w_{ij} \), i.e., the sum of the weights of the edges that starts in \( S \) and ends in \( T \) and \( S^C = V \setminus S \). \([46]\). Maximizing this quantity is a computationally heavy problem, so we use the Semidefinite Program (SDP) approximation proposed by Chan and Sun \([47]\). For calculating CD, we use the python implementation available in the library cutnorm \([48]\).

The second approach to computing the difference between two mobility networks consists of comparing their distributions of edge weights and weight-distances. Edge weights indicate the values (flows) of the adjacency matrices describing the two mobility networks. Weight-distances indicate the combination of an edge’s weight (flow) and the distance between the two nodes composing the edge. We compute the weighted-distance adjacency matrix of a mobility network as \( \hat{A} = A/(d+\epsilon) \), where \( A \) is the network’s weighted adjacency matrix, \( d \) is the distance matrix having the same dimension and node ordering of \( A \) and representing the geographic distances between all pair of nodes\(^1\). We add the residual term \( \epsilon = 0.8 \) to the denominator to avoid dividing by zero only for elements on the diagonal of the adjacency matrices. Given two mobility networks, the more similar their distribution of edge weights or weight-distances are, the more similar the two mobility networks are. We measure the similarity between two distributions using the Jensen-Shannon divergence \([49, 36]\):

\[
JS(P||Q) = \frac{1}{2} KL(P||M) + \frac{1}{2} KL(Q||M)
\]

\(^1\)The geographic distance between two nodes is calculated as the distance between the centroids of the tile that represents that node.
where \( P \) and \( Q \) are two density distributions, \( M = \frac{1}{2}(P + Q) \), and \( KL \) is the Kullback–Leibler divergence (KL) \([50, 51]\), defined as:

\[
KL(P||Q) = \sum_{x \in X} P(x) \log \left( \frac{P(x)}{Q(x)} \right)
\]

6 Results

Figure 3 shows the distribution of the Cut Distance (CD) in the four datasets’ test (red), synthetic (blue), and mixed sets (green) for MoGAN (left), the Gravity model (center), and the Radiation model (right). MoGAN’s CD distributions overlap almost entirely in all four datasets, meaning that MoGAN generates mobility networks that are indistinguishable from real ones and way more realistic than those generated by the baselines (except in two cases, see Supplementary Note 5). Similar results hold for the other metrics: MoGAN typically outperforms the baselines regarding CPC (Figure 4) and RMSE (Supplementary Note 6). Table 2 shows, for each model, the JS-divergence between the CPC distribution of the mixed and test sets and the JS-divergence between the CPC distribution of the synthetic and test sets.

To compute the improvement in performance of MoGAN with respect to the Gravity and the Radiation models, for each metric, each set and each baseline, we define the quantity:

\[
\Delta = -\left( \frac{JS(\text{MoGAN}) - JS(\text{baseline})}{JS(\text{baseline})} \right) \times 100
\]

where \( JS(\text{MoGAN}) \) is the JS divergence between the set (synthetic or mixed) of networks generated by MoGAN and the test set, while \( JS(\text{baseline}) \) is the JS divergence between the set (synthetic or mixed) of networks generated by the baselines (Gravity or Radiation) and the test set.

Table 2 shows that, according to the CPC, MoGAN outperforms the other models on all datasets, with a relative improvement of up to 86% on the Gravity model and 91% on the Radiation model over the mixed set, and a relative improvement of up to 49% on the Gravity model and 37% on the Radiation model over the synthetic set. We report the results for RMSE, CD, weights distribution and weight-distances distribution in Supplementary Notes 6, 7 and 8.

MoGAN’s JS-divergences between the mixed and test sets and between the synthetic and test sets are the lowest for each dataset, meaning that our model produces the most overlapping distributions (see Table 2). Our results also show that the difference (either in terms of CD, CPC, or RMSE) between a real network and a synthetic one is similar to the difference between two real networks or two synthetic...
networks. This means that MoGAN generates realistic mobility networks that are, to a certain extent, indistinguishable from real ones.

Figure 5 shows the distributions of the pairwise similarities among the edge weights for the synthetic, mixed, and test sets built over the four datasets. For each dataset, we report the performances of MoGAN, the Gravity model, and the Radiation model. Again, MoGAN significantly outperforms the baselines, except for two cases (mixed set of NYC and CHI taxi) in which the Gravity model and MoGAN achieve similar performance. We find a similar result for the weight-distances (see Supplementary Note 7).

7 Conclusion

In this work, we presented MoGAN, a deep-learning-based model for generating realistic urban mobility networks. Our results, conducted on four public datasets representing flows of bikes and taxis in New York City and Chicago, show that the realism of the networks generated by MoGAN outperforms those generated by the Gravity and the Radiation model.

Although MoGAN’s performance is encouraging, it has some limitations. Being based on Deep Convolutional Generative Adversarial Networks (DCGAN) [27], MoGAN can generate 64 × 64 adjacency matrices, that is, mobility networks with 4096 locations. We plan to extend MoGAN’s architecture to generate mobility networks with an arbitrary number of nodes as future improvements. Other technical improvements may be the use of Graph Neural Networks (GNNs) [52] for the generator and the discriminator, which would better capture the network dependencies and include other information related to the locations (e.g., population or relevance), and the use of the Wasserstein loss [53], which has been proven to improve GANs in several

| data     | MoGAN | Gravity | Radiation | Rel. Improvement |
|----------|-------|---------|-----------|------------------|
|          | JS_m | JS_s   | JS_m | JS_s | Δ_m,G | Δ_s,G | Δ_m,R | Δ_s,R |
| NYC_bike | 0.06  | 0.08   | 0.46 | 0.15 | 0.72  | 0.12  | 86%   | 49%   | 91%   | 37%   |
| NYC_taxi | 0.09  | 0.11   | 0.53 | 0.14 | 0.83  | 0.15  | 83%   | 22%   | 89%   | 29%   |
| CHI_bike | 0.14  | 0.16   | 0.29 | 0.25 | 0.56  | 0.26  | 51%   | 35%   | 75%   | 38%   |
| CHI_taxi | 0.08  | 0.09   | 0.39 | 0.11 | 0.79  | 0.13  | 80%   | 21%   | 90%   | 30%   |

Table 2: JS divergences of the distributions of the CPC scores. For each model, we report the JS divergence between mixed set and test set (column JS_m) and the JS divergence between synthetic set and the test set (column JS_s). The last four Δ_x,Z-like columns represent the improvement of MoGAN compared to the Gravity model on the mixed and the synthetic sets (columns Δ_m,G an Δ_s,G) and the improvement of MoGAN compared to the Radiation model on the mixed and synthetic sets (columns Δ_m,R and Δ_s,R).
contextual flows [54, 55]. Finally, it would also be interesting to test MoGAN’s effectiveness on cities of different sizes and shapes as well as on the generation of individual mobility trajectories or [56, 57, 58], which represent the aggregated movements of single individuals among a city’s locations.

In the meantime, our study demonstrates the great potential of artificial intelligence to improve solutions to crucial problems in human mobility, such as the generation of realistic mobility networks. Our model can synthesize aggregated movements within a city into a realistic generator, which can be used for data augmentation and performing simulations and what-if analysis. Given the flexibility of the training phase, our model can be easily extended to synthesize specific types of mobility, such as aggregated movements during workdays, weekends, specific periods of the year, or in presence of pandemic-driven mobility restrictions, events, and natural disasters.
Figure 4: **Results for the CPC.** Distributions of the pairwise CPC distances between mobility networks in the test set (red), synthetic set (blue), and mixed set (green), for the four datasets. For each dataset, we compare the overlap of the distributions of MoGAN and the two baselines (Gravity and Radiation). For both the Gravity model and the Radiation model, the three distribution are significantly different, especially for the latter.

### Availability of data and material

The code to train/test MoGAN and reproduce our analyses, and the links to the datasets used in our experiments, can be found at [https://github.com/jonpappal/crd/GAN-flow](https://github.com/jonpappal/crd/GAN-flow).

### Funding

Luca Pappalardo and Giovanni Mauro have been supported by EU project SoBigData++ grant agreement #871042.

### Acknowledgments

Luca Pappalardo and Giovanni Mauro thank EU project SoBigData++ grant agreement #871042 for supporting this research. We thank Ramon Ferrer-i-Cancho, Matteo Böhm, Giuliano Cornacchia, and Vasiliki Voukelatou for the useful suggestions.
Figure 5: **Results for weights distribution.** Distributions of the pairwise JS distance between the distribution of weights of the mobility networks in the test set (red), synthetic set (blue), and mixed set (green), for the four different datasets. For each dataset, we compare the overlap of the distributions of MoGAN and the two baselines (Gravity and Radiation). The Radiation model’s mixed and synthetic sets distributions significantly differs from the test set for all datasets. The situation is similar for the Gravity model performances. MoGAN distributions are almost overlapping for all four datasets.

We thank Daniele Fadda and Eleonora Cappuccio for the visualization suggestions. We also thank René Ferretti and Dante Milonga for the inspiration.

**References**

[1] Batty, M.: The New Science of Cities. MIT press, ??? (2013)

[2] Andrienko, G., Andrienko, N., Boldrini, C., Caldarelli, G., Cintia, P., Cresci, S., Facchini, A., Giannotti, F., Gionis, A., Guidotti, R., Mathioudakis, M., Muntean, C.I., Pappalardo, L., Pedreschi, D., Pournaras, E., Pratesi, F., Tesconi, M., Trasarti, R.: (so) big data and the transformation of the city. International Journal of Data Science and Analytics (2020)

[3] Lucchini, L., Centellegher, S., Pappalardo, L., Gallotti, R., Privitera, F., Lepri, B., De Nadai, M.: Living in a pandemic: changes in mobility routines, social
activity and adherence to COVID-19 protective measures. Scientific Reports 11(1), 24452 (2021). doi:10.1038/s41598-021-04139-1

[4] Pepe, E., Bajardi, P., Gauvin, L., Privitera, F., Lake, B., Cattuto, C., Tizzoni, M.: Covid-19 outbreak response, a dataset to assess mobility changes in italy following national lockdown. Scientific data 7(1), 1–7 (2020)

[5] Lai, S., Farnham, A., Ruktanonchai, N.W., Tatem, A.J.: Measuring mobility, disease connectivity and individual risk: a review of using mobile phone data and health for travel medicine. Journal of travel medicine 26(3) (2019)

[6] Ruktanonchai, N.W., Floyd, J.R., Lai, S., Ruktanonchai, C.W., Sadilek, A., Rente-Lourenco, P., Ben, X., Carioli, A., Gwinn, J., Steele, J.E., Prosper, O., Schneider, A., Oplinger, A., Eastham, P., Tatem, A.J.: Assessing the impact of coordinated covid-19 exit strategies across europe. Science 369(6510), 1465–1470 (2020)

[7] Kraemer, M.U., Yang, C.-H., Gutierrez, B., Wu, C.-H., Klein, B., Pigott, D.M., Du Plessis, L., Faria, N.R., Li, R., Hanage, W.P., et al.: The effect of human mobility and control measures on the covid-19 epidemic in china. Science 368(6490), 493–497 (2020)

[8] Oliver, N., Lepri, B., Sterly, H., Lambiotte, R., Deletaille, S., De Nadai, M., Letouzé, E., Salah, A.A., Benjamins, R., Cattuto, C., et al.: Mobile phone data for informing public health actions across the COVID-19 pandemic life cycle (2020)

[9] Luca, M., Barlacchi, G., Lepri, B., Pappalardo, L.: A survey on deep learning for human mobility. ACM Computing Surveys (CSUR) 55(1), 1–44 (2021)

[10] Wang, J., Kong, X., Xia, F., Sun, L.: Urban human mobility: Data-driven modeling and prediction. ACM SIGKDD Explorations Newsletter, 1–19 (2019)

[11] Barbosa, H., Barthelemy, M., Ghoshal, G., James, C.R., Lenormand, M., Louail, T., Menezes, R., Ramasco, J.J., Simini, F., Tomasini, M.: Human mobility: Models and applications. Physics Reports 734, 1–74 (2018). doi:10.1016/j.physrep.2018.01.001

[12] Le Blanc, D.: Towards integration at last? the sustainable development goals as a network of targets. Sustainable Development 23(3), 176–187 (2015)

[13] Kroll, C., Warchold, A., Pradhan, P.: Sustainable development goals (sdgs): Are we successful in turning trade-offs into synergies? Palgrave Communications 5(1), 1–11 (2019)
[14] United Nations General Assembly: Transforming our world: the 2030 agenda for sustainable development. Technical report (2015). Accessed: 2021-02-23

[15] Simini, F., Barlacchi, G., Luca, M., Pappalardo, L.: A deep gravity model for mobility flows generation. Nature Communications 12(1), 1–13 (2021)

[16] Masucci, A.P., Serras, J., Johansson, A., Batty, M.: Gravity versus radiation models: On the importance of scale and heterogeneity in commuting flows. Physical Review E 88(2), 022812 (2013)

[17] Carey, H.C.: Principles of Social Science vol. 3. JB Lippincott & Company, ??? (1867)

[18] Zipf, G.K.: The p 1 p 2/d hypothesis: on the intercity movement of persons. American sociological review 11(6), 677–686 (1946)

[19] Lenormand, M., Bassolas, A., Ramasco, J.J.: Systematic comparison of trip distribution laws and models. Journal of Transport Geography 51, 158–169 (2016)

[20] Erlander, S., Stewart, N.F.: The Gravity Model in Transportation Analysis: Theory and Extensions vol. 3. Vsp, ??? (1990)

[21] Simini, F., González, M.C., Maritan, A., Barabási, A.-L.: A universal model for mobility and migration patterns. Nature 484(7392), 96–100 (2012)

[22] Prieto Curiel, R., Pappalardo, L., Gabrielli, L., Bishop, S.R.: Gravity and scaling laws of city to city migration. PLOS ONE 13(7), 1–19 (2018). doi:10.1371/journal.pone.0199892

[23] Yan, X.-Y., Wang, W.-X., Gao, Z.-Y., Lai, Y.-C.: Universal model of individual and population mobility on diverse spatial scales. Nature Communications 8(1), 1639 (2017). doi:10.1038/s41467-017-01892-8

[24] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Proceedings of the 27th International Conference on Neural Information Processing Systems, pp. 2672–2680 (2014)

[25] Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., Bharath, A.A.: Generative adversarial networks: An overview. IEEE Signal Processing Magazine 35(1), 53–65 (2018)

[26] Goodfellow, I.: Nips 2016 tutorial: Generative adversarial networks. arXiv preprint arXiv:1701.00160 (2016)
[27] Radford, A., Metz, L., Chintala, S.: Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks (2016). 1511.06434

[28] Liu, X., Chen, H., Andris, C.: trajgans: Using generative adversarial networks for geo-privacy protection of trajectory data (vision paper). In: Location Privacy and Security Workshop, pp. 1–7 (2018)

[29] Yin, D., Yang, Q.: Gans based density distribution privacy-preservation on mobility data. Security and Communication Networks 2018 (2018)

[30] Kulkarni, V., Tagasovska, N., Vatter, T., Garbinato, B.: Generative models for simulating mobility trajectories. arXiv preprint arXiv:1811.12801 (2018)

[31] Ouyang, K., Shokri, R., Rosenblum, D.S., Yang, W.: A non-parametric generative model for human trajectories. In: IJCAI, pp. 3812–3817 (2018)

[32] Huang, D., Song, X., Fan, Z., Jiang, R., Shibasaki, R., Zhang, Y., Wang, H., Kato, Y.: A variational autoencoder based generative model of urban human mobility. In: 2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), pp. 425–430 (2019)

[33] Feng, J., Yang, Z., Xu, F., Yu, H., Wang, M., Li, Y.: Learning to simulate human mobility. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. KDD ’20, pp. 3426–3433. Association for Computing Machinery, New York, NY, USA (2020). doi:10.1145/3394486.3412862. https://doi.org/10.1145/3394486.3412862

[34] Moreira-Matias, L., Gama, J., Ferreira, M., Mendes-Moreira, J., Damas, L.: Predicting taxi–passenger demand using streaming data. IEEE Transactions on Intelligent Transportation Systems 14(3), 1393–1402 (2013)

[35] Cornacchia, G., Pappalardo, L.: A mechanistic data-driven approach to synthesize human mobility considering the spatial, temporal, and social dimensions together. ISPRS International Journal of Geo-Information 10(9) (2021). doi:10.3390/ijgi10090599

[36] Pappalardo, L., Simini, F.: Data-driven generation of spatio-temporal routines in human mobility. Data Mining and Knowledge Discovery 32(3), 787–829 (2018)

[37] Jiang, S., Yang, Y., Gupta, S., Veneziano, D., Athavale, S., Gonzalez, M.C.: The timegeo modeling framework for urban mobility without travel surveys. Proceedings of the National Academy of Sciences 113, 201524261 (2016)
[38] Pappalardo, L., Simini, F., Barlacchi, G., Pellungrini, R.: scikit-mobility: a Python library for the analysis, generation and risk assessment of mobility data (2019). 1907.07062

[39] CitiBike System Data (2013–). https://www.citibikenyc.com/system-data

[40] Divvy System Data (2016–). https://www.divvybikes.com/system-data

[41] TLC Trip Record Data (2009–). https://www1.nyc.gov/site/tlc/about/tlc-trip-records/data

[42] TLC Trip Record Data (2013–). https://data.cityofchicago.org/Transportation/Taxi-Trips/wrvz-psew/data

[43] Tantardini, M., Ieva, F., Tajoli, L., Piccardi, C.: Comparing methods for comparing networks. Scientific reports 9(1), 1–19 (2019)

[44] Lenormand, M., Bassolas, A., Ramasco, J.J.: Systematic comparison of trip distribution laws and models. Journal of Transport Geography 51, 158–169 (2016). doi:10.1016/j.jtrangeo.2015.12.008

[45] Liu, Q., Dong, Z., Wang, E.: Cut based method for comparing complex networks. Scientific reports 8(1), 1–11 (2018)

[46] Alon, N., Naor, A.: Approximating the cut-norm via grothendieck’s inequality. In: Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing, pp. 72–80 (2004)

[47] O’Donnell, R., Wu, Y.: An optimal sdp algorithm for max-cut, and equally optimal long code tests. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, pp. 335–344 (2008)

[48] Chiu, P.-K.: cutnorm package (2018–). https://pypi.org/project/cutnorm/

[49] Fuglede, B., Topsoe, F.: Jensen-shannon divergence and hilbert space embedding. In: International Symposium on Information Theory, 2004. ISIT 2004. Proceedings., p. 31 (2004). IEEE

[50] Kullback, S.: Information Theory and Statistics. Courier Corporation, ??? (1997)

[51] Van Erven, T., Harremos, P.: Rényi divergence and kullback-leibler divergence. IEEE Transactions on Information Theory 60(7), 3797–3820 (2014)
[52] Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE transactions on neural networks 20(1), 61–80 (2008)

[53] Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: International Conference on Machine Learning, pp. 214–223 (2017). PMLR

[54] Weng, L.: From gan to wgan. arXiv preprint arXiv:1904.08994 (2019)

[55] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.: Improved training of wasserstein gans. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 5769–5779 (2017)

[56] Berke, A., Doorley, R., Larson, K., Moro, E.: Generating synthetic mobility data for a realistic population with rnns to improve utility and privacy. CoRR abs/2201.01139 (2022).

[57] Rinzivillo, S., Gabrielli, L., Nanni, M., Pappalardo, L., Pedreschi, D., Giannotti, F.: The purpose of motion: Learning activities from individual mobility networks. In: 2014 International Conference on Data Science and Advanced Analytics (DSAA), pp. 312–318 (2014)

[58] Schneider, C.M., Belik, V., Couronné, T., Smoreda, Z., González, M.C.: Unravelling daily human mobility motifs. Journal of The Royal Society Interface 10(84), 20130246 (2013)

[59] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc., (2019). http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[60] Hagberg, A., Swart, P., S Chult, D.: Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States) (2008)
Supplementary Note 1: Architecture and training of MoGAN

We modify the Deep Convolutional Generative Adversarial Network (DCGAN) model [27] to work over bi-dimensional matrices of dimension 64 × 64 as if they were gray-scale mono-channel images, but without constraining the values to be in the range [0, 255]. We use the Rectified Linear Unit (ReLU) as activation function.

Consistently with the proposed architecture, a Generator $G$ takes a noisy vector $z$ as input and operates a series of “Transposed Convolutions” with fixed values for the parameters, in order to perform the upsampling operations and generate a 64 × 64 matrix. As stated in the original DCGAN architecture, each block of Upsample-Convolution-LeakyReLU of the model is followed by a batch normalization.

A discriminator $D$ performs a binary classification task using a sigmoid activation function as last layer. The convolutional process gradually reduces the size of the matrix, learning different peculiarities at each epoch. We train MoGAN using a batch size of 146 to create $\frac{584}{146} = 4$ equally spaced batches and avoid an uncontrollable variance in both $G$ and $D$. We fix the two parameters controlling the Adam optimizer, responsible for the backward optimization, to $b1 = 0.5$ and $b2 = 0.999$. We fix the latent dimension of the noise vector $z$ to 100. We find no significant improvement when changing the values of the above parameters. The minimum number of epochs required for our model to perform at its best over the four datasets is 6000. We use PyTorch [59] for implementing MoGAN. An explanation of the training loop can be found in Supplementary Note V.

In detail, iterating through the epochs and, for each epoch, iterating through the minibatch we:

- Define a vector of valid labels (ones) and a vector of synthetic labels (zeros) having the same dimension of the current minibatch.
- Set the $G$’s weights’ gradients to zero. We do this before starting the backpropagation, since PyTorch accumulates the gradients on subsequent backward passes.
- Generate a minibatch of synthetic images (output of the Generator).
- Calculate $G$’s loss as the loss between the output of $D$ over the generated images and the valid labels vector.
- Backpropagate the error calculated with this latter loss and update $G$’s weights.
- Set $D$ gradients to zero.
- Calculate $D$’s loss as an average loss between two losses. The first one is the loss between the output of $D$ over the real images of the minibatch and the
Supplementary Figure S1: Left: Score of the Discriminator over real (blue) and synthetic (red) samples during the training. Right: Loss of the Generator (red) and Discriminator (blue) during the training. Iterations indicate the quantity $n_{\text{epochs}} \times n_{\text{batches}}$, where $n_{\text{epochs}}$ is the number of epochs and $n_{\text{batches}}$ is the number of batches. For sake of simplicity, we only report the losses and the scores of the first 2000 epochs of our model applied to the City Bike dataset of New York City.

valid labels. The second one is the loss between the output of $D$ over generated images and the synthetic labels.

- Backpropagate the error calculated with this latter loss and update D’s weights.

Note that we use only one loss, both for G and D, as in [24]. Nevertheless, changing the values over which this loss is calculated (generated/real images, true/synthetic labels) originates different and feasible values, given the nature of the Mini-Max game solution of the GAN’s loss.

Supplementary Figure S1 (left) shows the evolution of the scores with the number of epochs. For each epoch, we define as score the average label assigned by $D$ to the mobility networks. $D$ assigns label 1 to the real mobility networks and 0 to the generated (fake) ones. Therefore, the score is a measure of accuracy. In particular, the real score is the average label assigned by $D$ to real mobility networks; the synthetic score is the average label assigned to generated mobility networks. Given an epoch, the score is 0 (1) if $D$ labels all the mobility networks as real (fake). Ideally, at the end of the training process, both the real score and the synthetic score should be around 0.5, meaning that $D$ can no longer determine whether a sample comes from the real distribution or the synthetic one. In the initial phase, $D$ cannot discriminate between real and fake mobility networks, assigning to both of them an average score of 0.5 (Supplementary Figure S1). After about 100 epochs, $D$ learns to classify the generated mobility networks. Then, there is a progressive convergence to 0.5, meaning that $D$ can no longer distinguish real mobility networks from synthetic ones.
Supplementary Figure S1 (right) shows how the D’s and G’s losses change with the number of iterations. Initially, G’s loss is high and unstable, meaning that it cannot fool D. In contrast, D’s loss is close to 0, meaning that it well discriminates between a real and a fake mobility network. After almost 6000 iterations (1500 epochs), G’s loss decreases, and the two losses stabilize, converging to a similar value. This result seems to suggest that G has becomes capable of fooling D, i.e., to generate synthetic mobility networks indistinguishable from real ones.

Supplementary Note 2: Networks generated during training phase

Supplementary Figure S2 shows four snapshots of the mobility networks generated during the training: the network at the end of the first epoch, and those after 30%, 60%, and 90% of the epochs. After the first training epoch, mobility flows are light or non-existing (Supplementary Figure S2a). As the epochs go by, MoGAN starts identifying the most connected nodes, generating heavier flows between them (Supplementary Figure S2 b-d).

Supplementary Note 3: Tessellations of New York City and Chicago

In Supplementary Figure S3, we provide a visualization of the tessellations we use for our analysis, for both of the analyzed cities.

Supplementary Note 4: Equivalence between RMSE and Frobenius Norm

We show that calculating the RMSE between two adjacency matrices is equivalent to calculate the Frobenius norm of the difference matrix. As suggested by [43], a norm of the difference matrix is a metric for evaluate how much two weighted networks with the same node-correspondence are similar. The Frobenius norm of a matrix A is defined as:

$$
\|A\|_F = \sqrt{\sum_{i,j=1}^{n} |a_{ij}|^2}
$$

Therefore, the Frobenius norm of the difference matrix A − B can be calculated.
Supplementary Figure S2: Mobility networks generated during the training process, after the first epoch (a), after 30% of the epochs (b), after 60% of the epochs (c), and after 90% of the epochs (d). The nodes represent the centroids of the 64 tiles that compose the tessellation of Manhattan. The thickness of the edges is proportional to the magnitude of the corresponding flows. Visualization made using library networkx [60].

as:

$$\|A - B\|_F = \sqrt{\sum_{i,j=1}^{n}|(a_{ij} - b_{ij})|^2} \quad (3)$$

Root Mean Square Error (RMSE) is defined as:

$$RMSE(A, B) = \sqrt{\frac{1}{n}\sum_{i,j=1}^{n}(a_{ij} - b_{ij})^2} \quad (4)$$

The right hand sides of Equations (1) and (2) only differ for the 1/n term. Nevertheless, given that in our case $n$ is fixed (and equal to the number of elements
Supplementary Figure S3: (Left) Tessellation of New York City. The city is divided in 64 equally-spaced squared tiles. Each side of each tile is 1840 meters long. (Right) Tessellation of Chicago. The city is divided in 64 equally-spaced squared tiles. Each side of each tile is 1405 meters long.

of the same matrices), studying the distribution of the RMSE or the distribution of the Frobenius norm is equivalent.

**Supplementary Note 5: Cut Distance tabular comparison**

In Table 3 we show the numerical improvements in terms of JS divergence among distributions per each dataset and model. As it can be inferred from the latest four columns, MoGAN achieves the best performance in almost all the cases, with an improvement on the baselines ranging from 12% to 87%. Only in two cases, the baselines produce a synthetic and a mixed set that are more similar to the actual test set.
Table 3: JS divergences of the distributions of the CD scores. For each model, we report the JS divergence between mixed and test set (column $JS_m$) and the JS divergence between synthetic and test set (column $JS_s$). The last four $\Delta_{x/y,z}$-like columns represent, respectively the improvement of our MoGAN model with respect to the Gravity model in terms of mixed and synthetic set (columns $\Delta_{m,G}$ and $\Delta_{s,G}$) and the improvement of our models with respect to the Radiation model in terms of mixed and synthetic set (columns $\Delta_{m,R}$ and $\Delta_{s,R}$).

Supplementary Note 6: RMSE evaluation

Supplementary Figure S4 shows the RMSE scores over the four datasets. For each dataset, we show the distributions of the RMSE among mobility networks in the test set (red), the synthetic set (blue), and the mixed set (green), for the Radiation model (left), Gravity model (center), and MoGAN (right). MoGAN’s RMSE distributions over the three sets overlaps almost completely in all the four scenarios. Both Gravity and Radiation distributions are not overlapping, especially for the latter. We also report the numerical improvements in terms of JS divergence among distributions per each dataset and model in Table 4.

Table 4: JS divergences of the distributions of the RMSE scores. For each model, we report the JS divergence between mixed and test set (column $JS_m$) and the JS divergence between synthetic and test set (column $JS_s$). The last four $\Delta_{x/y,z}$-like columns represent, respectively the improvement of our MoGAN model with respect to the Gravity model in terms of mixed and synthetic set (columns $\Delta_{m,G}$ and $\Delta_{s,G}$) and the improvement of our models with respect to the Radiation model in terms of mixed and synthetic set (columns $\Delta_{m,R}$ and $\Delta_{s,R}$).
Supplementary Figure S4: Distributions of the pairwise RMSE distances between mobility networks in the test set (red), synthetic set (blue), and mixed set (green), for the four different datasets. For each dataset, we compare the overlapping level of the distributions of our two baseline models (Radiation and Gravity) and of our MoGAN model. Radiation’s mixed set and synthetic set distributions significantly differ from the test set one in any of the four scenarios. The situation is similar for the Gravity model’s performances. MoGAN distributions are, instead, almost overlapping.

**Supplementary Note 7: Weight distributions’ tabular comparison**

In Table 5 we show the numerical improvements in terms of JS divergence among distributions per each dataset and model. As it can be inferred from the latest four columns, MoGAN’s performance is the best in almost all the cases, with an improvement on the baselines ranging from 22% to 90%. Only in two cases the baselines produce a mixed set that is more similar to the actual test set. The same results hold for weight-distances distribution.
MoGAN
Gravity Radiation Rel. Improvement

|        | J_SM | JS_m | J_SM | JS_s | J_SM | JS_s | ∆_m, G | ∆_s, G | ∆_m, R | ∆_s, R |
|--------|------|------|------|------|------|------|--------|--------|--------|--------|
| NYCbike | 0.18 | 0.44 | 0.25 | 0.36 | 0.52 |      | 60%    | 56%    | 50%    | 78%    |
| NYCtaxi | 0.06 | 0.05 | 0.10 | 0.13 | 0.82 |      | -10%   | 22%    | 57%    | 90%    |
| CHIbike | 0.06 | 0.35 | 0.24 | 0.28 | 0.21 |      | 82%    | 51%    | 78%    | 45%    |
| CHITaxi | 0.12 | 0.11 | 0.13 | 0.23 | 0.23 |      | -7%    | 62%    | 49%    | 78%    |

Table 5: JS divergences of the distributions of the weights. For each model, we report the JS divergence between mixed and test set (column JS_m) and the JS divergence between synthetic and test set (column JS_s). The last four Δ_{x/y,Z}-like columns represent, respectively, the improvement of our MoGAN model with respect to the Gravity model in terms of mixed and synthetic set (columns ∆_m,G and ∆_s,G) and the improvement of our models with respect to the Radiation model in terms of mixed and synthetic set (columns ∆_m,R and ∆_s,R).

Supplementary Note 8: Weight-Distances’ Evaluation

In Supplementary Figure S5, we repeat the analysis for the weight-distances. MoGAN’s good performance suggests that it can reproduce a notable law of mobility networks: the closer two nodes are, the higher the weight (flow) between them tends to be.
Supplementary Figure S5: Distributions of the pairwise JS divergences between the weight-distances of the mobility networks in the test set (red), synthetic set (blue), and mixed set (green), for the four datasets. For each dataset, we compare the overlapping level of the distributions of our two baseline models (Radiation and Gravity) and our MoGAN model. Radiation’s mixed set and synthetic set distributions significantly differ from the test set one in all the four investigated scenarios. The situation is similar for the Gravity model performances. MoGAN distributions are instead almost overlapping.