Mauro et al. *EPJ Data Science* (2022) 11:58
https://doi.org/10.1140/epjds/s13688-022-00372-4

EPJ Data Science
a SpringerOpen Journal

EPJ.org

REGULAR ARTICLE Open Access

Generating mobility networks with generative adversarial networks

Giovanni Mauro[1,2,3], Massimiliano Luca[4,6], Antonio Longa[5,6], Bruno Lepri[6] and Luca Pappalardo[1*]

*Correspondence:
luca.pappalardo@isti.cnr.it
[1]Institute of Information Science and Technologies, National Research Council (ISTI-CNR), Pisa, Italy
Full list of author information is available at the end of the article

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.

Keywords: Human mobility; Artificial intelligence; Flow generation; GANs

1 Introduction

The increasing complexity of urban environments [2, 5] and the crucial role played by human displacements in the diffusion of epidemics, not least the COVID-19 pandemic [24, 28, 36, 43, 47, 51], have created a great deal of interest around the study of individual and collective human mobility [4, 34, 60]. 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 [7, 25, 29, 57], 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) [4, 34, 37, 54, 60]. Traditionally, flow generation is addressed through the Gravity model [4, 8, 15, 30, 35, 65], the Radiation model [4, 55, 60], and their variants [4, 48, 54, 63]. 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)

Springer

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) [20], 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 [13, 18, 20, 49]. While several papers show that GANs can generate individual mobility trajectories [16, 22, 26, 33, 34, 39, 44, 64] with a realism comparable to or better than mechanistic mobility models [4, 12, 23, 45], 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 several baseline models, i.e., the Gravity, the Radiation, and the Random Weighted models. Our results prove that our solution 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 [34]. 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 $\mathcal{G} = (V, E, w)$, 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);
- $E = \{(x, y) | (x, y) \in V \times V \wedge w(x, y) \neq 0\}$ 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 $\mathcal{A}_{n\times n}$ with $n = |V|$. Thus, an element $a_{i,j} \in \mathcal{A}$ 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 $\mathcal{X}_M = \{\hat{\mathcal{G}}_1, \ldots, \hat{\mathcal{G}}_n\}$, 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 $\mathcal{X} = \{\mathcal{G}_1, \ldots, \mathcal{G}_n\}$ 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 $f \in \mathcal{K}$, $D(P_{(f,\mathcal{X}_M)}||P_{(f,\mathcal{X}_{\text{test}})})$ indicates the dissimilarity between $P_{(f,\mathcal{X}_M)}$, the distribution of the measures computed on the synthetic mobility networks in $\mathcal{X}_M$, and $P_{(f,\mathcal{X}_{\text{test}})}$, the distribution of the measures computed on the mobility networks in $\mathcal{X}_{\text{test}}$. The lower $D(P_{(f,\mathcal{X}_M)}||P_{(f,\mathcal{X}_{\text{test}})})$, 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) [49]. 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 [34].

During the training phase, G repeatedly takes a 1×100 noise vector as input and operates a series of transposed convolutions, which perform upsampling of the input vector to generate a 64×64 adjacency matrix representing a mobility network. Then, D takes a set of real and generated 64×64 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).

MoGAN leverages the architecture of DCGAN [49] and, as highlighted above, this implies that the shape of adjacency matrices must be 64×64. MoGAN could easily be extended to geographic areas with less than 64 zones, for example testing MoGAN ability of working with 0-padded mobility networks. On the other hand, working with more than 64 zones would necessarily require some form of aggregation of the zones, or a totally different GAN structure.

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.

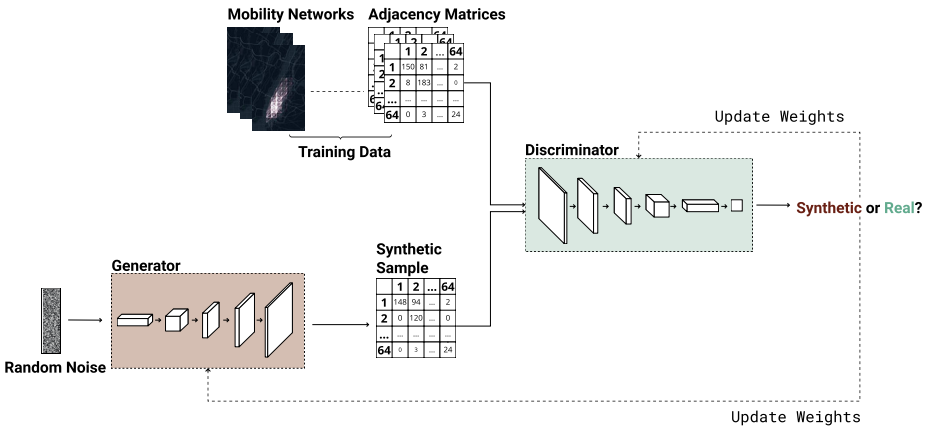

Figure 1 Architecture of MoGAN. The generator (a Convolutional Neural Network or CNN) performs transposed convolution operations that upsample the input random noise vector, transforming it into a 64 × 64 adjacency matrix representing a mobility network. The discriminator (a CNN) takes as input both 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's and generator's weights are then backpropagated

4 Baseline models

We compare MoGAN with the Gravity and the Radiation models, two classical approaches for mobility flows' generation [4, 34, 54, 55], using the implementations provided in library scikit-mobility [46].

The singly-constrained Gravity model [4, 8, 30, 65] 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:

$$\bar{y}(l_i, l_j) = O_i p_{ij} = O_i \frac{m_j^{\beta_1} f(r_{ij})}{\sum_k m_k^{\beta_1} f(r_{ik})}, \tag{1}$$

where O_i is the number of people leaving location l_i, m_j is the population of location l_j (estimated as O_j), p_{ij} is the probability to observe a trip (unit flow) from location l_i to location l_j, β_1 is a parameter and $f(r_{ij})$ is the deterrence function, which is a function of the distance r_{ij} between two locations. We model the deterrence function as a power-law function, $f(r) = r^{\alpha}$, where α is another parameter. These parameters can be fitted from a subset of available flows. We report the value of α and β_1 resulting from the fitting of the model in Supplementary Note 3.

The Radiation model [4, 55] 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:

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

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_{ij}.

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 β_1 and β_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 [46], which provides methods to fit parameters and generate flows from locations' relevance and outflow.

For a further analysis, we compare MoGAN with a Random Weighted (RW) model that creates a mobility network where the weight of each edge is randomly chosen from the distribution of weights for that edge in the training set. In other words, given an edge $e = (i, j)$ connecting node i to node j in the mobility network, the edge weight $\hat{w}(e)$ is a number picked at random from $\{w_1(e), w_2(e), \dots w_n(e)\}$, i.e., the distribution of the weights of e in the training set.

In terms of computational time required to generate a new mobility network, MoGAN is way faster (<1 second) than the Gravity model (about one minute) and the Random Weighted model (10-20 seconds). However, MoGAN needs a training phase that requires from 1 up to 3 hours depending on the dataset. In out experiments, we train MoGAN on a server with a GPU Tesla P100 with 16 GB of VRAM, 13 GB of RAM and a 2-core Intel Xeon CPU.

5 Experimental setup

5.1 Datasets

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 [11] and the Divvy Bike Dataset for Chicago [14]. 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 [11, 14]. 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 [9]. 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

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

Dataset	Rides	Locations	#bikes/taxis
CHI bikes [14]	350,503	198	6293
NYC bikes [11]	29,294,326	509	19,514
CHI taxis [9]	11,050,936	96	5668
NYC taxis [41]	157,485,483	68	N.D.

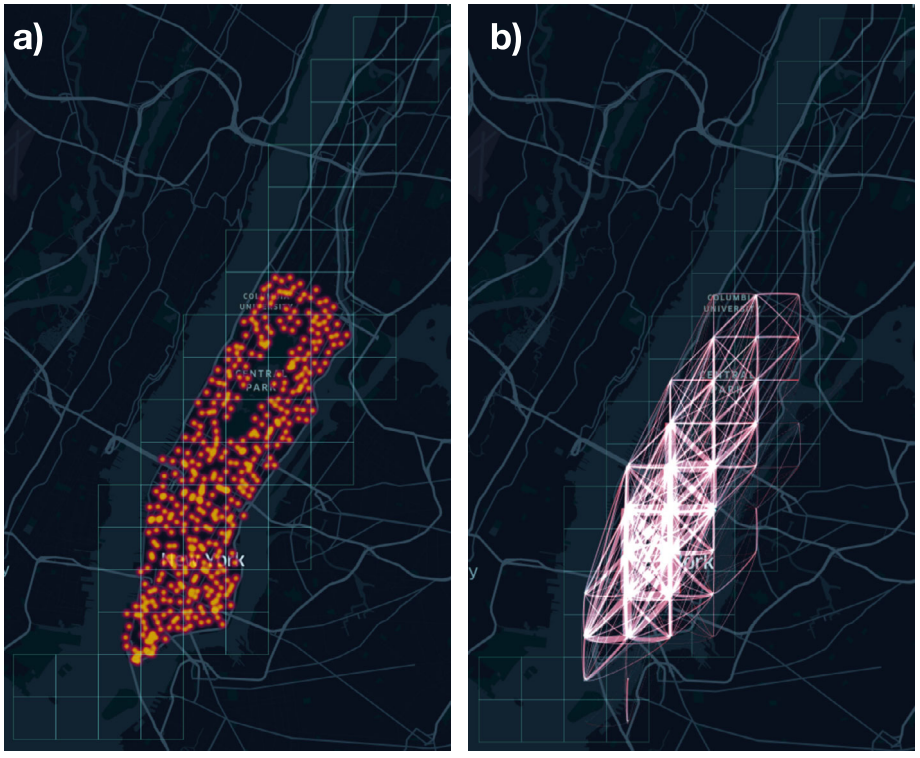

Figure 2 Examples of a real mobility network. (**a**) Position of bike stations in Manhattan. (**b**) A daily mobility network in Manhattan, where the size of each edge is proportional to the flow they represent

tiles to obtain an origin-destination matrix representing the daily mobility network. We obtain, for each dataset, a representation of the daily flows in the city, which is divided into 64 equally spaced tiles. The mobility networks represent the flow of people moving, daily, across these zones. We remind that, since MoGAN is based on DCGANs which are designed to work with images (matrices) of size 64×64, we are constrained to tessellate the city into 64 equally-sized tiles. This means that different cities have different tile size, depending on the city size.

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 datasets used in our study. As an example, Fig. 2 visualizes where bike stations concentrate and a mobility network representing daily flows in Manhattan, New York City.

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.

The idea behind this validation methodology is that we need to verify whether MoGAN is capable to reproduce the variability of mobility networks in the training set. If the distribution of differences among the networks in the synthetic set is similar to that of networks in the test set, MoGAN can approximate well the variability of mobility networks in the training set. The use of the mixed set further tests MoGAN's ability to reproduce the variability in the training set: by verifying that the distribution of differences between networks in the synthetic set and those in the test set is similar to the distribution of network distances within the test set and the synthetic set separately, we argue that MoGAN can reproduce the variability of networks in the training 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) [56]. 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: *(i)* Normalized Root Mean Square Error (NRMSE), *(ii)* Common Part of Commuters (CPC), and *(iii)* Cut Distance (CD). The Root Mean Square Error (RMSE) [34, 54] is defined as:

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

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 (64×64). Note that RMSE is substantially equivalent to the Frobenious norm (see Supplementary Note 5). The NRMSE is a min-max normalization of the RMSE, defined as:

$$\mathrm{NRMSE} = \frac{\mathrm{RMSE}(A,B)}{\max(A,B)-\min(A,B)}.$$

The Common Part of Commuters (CPC), also known as Sørensen-Dice index [4, 31, 34, 54], a well-established measure to compute the similarity between real and generated matrices, is defined as:

$$\mathrm{CPC}(A,B) = \frac{2\sum_{i,j=1}^{n}\min(a_{ij},b_{ij})}{\sum_{i,j=1}^{n}a_{ij}+\sum_{i,j=1}^{n}b_{ij}}.$$

CPC is a widely used metric in human mobility studies [30, 34] and it ranges between 0 and 1. A CPC of 1 indicates a perfect match between the generated flows and the ground truth. On the other hand, 0 highlights a bad performance with no overlap. In other terms, CPC can be seen and interpreted as a metric of accuracy.

The Cut Distance (CD) [32] is based on the notion of cut weight, widely used in network theory [56], 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:

$$\mathrm{CD}(A,B) = \max_{S\in V} \frac{1}{|V|}\left|e_A\left(S,S^C\right) - e_B\left(S,S^C\right)\right|$$

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 $S^C = V \setminus S$. [1]. Maximizing this quantity is a computationally heavy problem, so we use the Semidefinite Program (SDP) approximation proposed by Chan and Sun [42]. For calculating CD, we use the python implementation available in the library cutnorm [10].

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 $\epsilon = 0.01$ to the denominator just 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 [17, 45]:

$$\mathrm{JS}\left(P||Q\right) = \frac{1}{2}\mathrm{KL}\left(P||M\right) + \frac{1}{2}\mathrm{KL}\left(Q||M\right),$$

where P and Q are two density distributions, $M = \frac{1}{2}(P+Q)$, and KL is the Kullback–Leibler divergence (KL) [27, 58], defined as:

$$\mathrm{KL}\left(P||Q\right) = \sum_{x\in X} P(x)\log\left(\frac{P(x)}{Q(x)}\right).$$

An alternative to the usage of divergence metrics may consist in using kernels to measure similarities between graphs [40, 59]. However, while kernel methods compare networks' representation in a latent space, in this paper we aim to capture the mobility network's

[1] The geographic distance between two nodes is calculated as the distance between the centroids of the tile that represents that node.

topological macro-scale features (e.g., degree distribution, clustering coefficient). For the sake of completeness, in Supplementary Note 10, we provide a comparison between topological properties of the generated and real mobility networks such as the clustering coefficient and the weighted degree distribution.

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 6). Similar results hold for the other metrics: MoGAN typically outperforms the baselines regarding CPC (Fig. 4) and RMSE (Supplementary Note 7). Table 2 shows, for each model, the JS-divergence between *(i)* the CPC distribution of the mixed and test sets and *(ii)* the CPC distribution of the synthetic and test sets.

To compute the improvement in performance of MoGAN with respect to the baseline 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, Radiation or Random Weighted) and the test set.

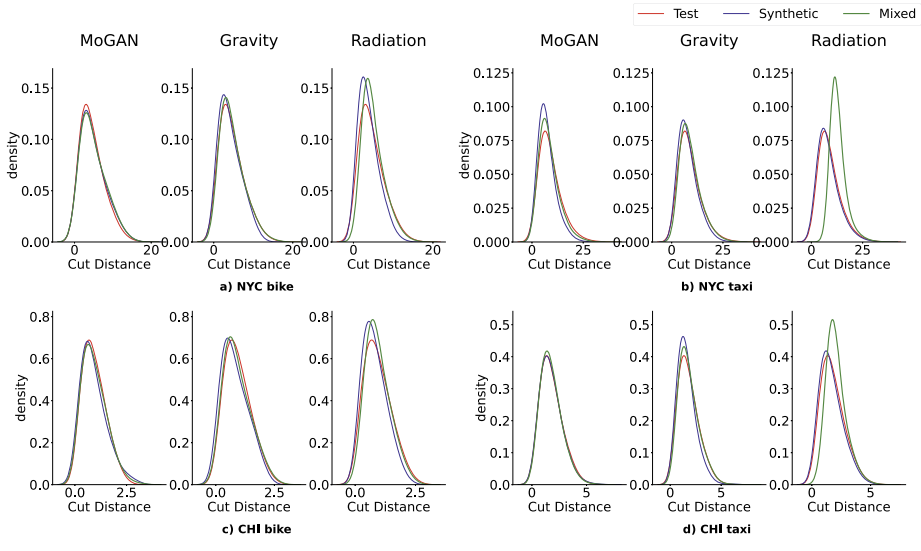

Figure 3 Results for the Cut Distance. Distributions of the pairwise cut distances between mobility networks in the test set (red), synthetic set (blue), and mixed set (green), for the four datasets. For each dataset, we compare the overlap among the distributions of MoGAN and the two baselines (Gravity and Radiation). The Radiation model's mixed and synthetic sets distributions significantly differ from the test set for all datasets. In contrast, the Gravity model clearly outperforms the Radiation model for all datasets

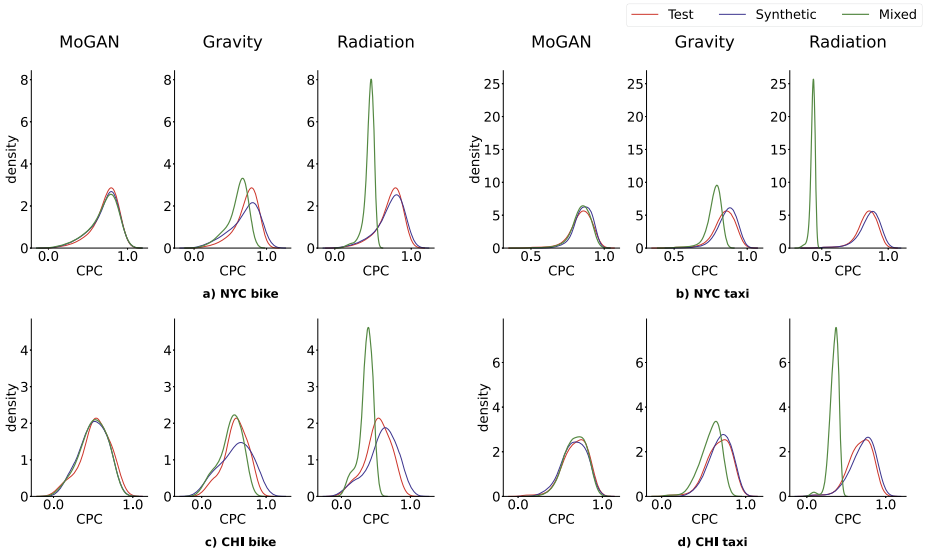

Figure 4 Results for the CPC. Distributions of the pairwise CPC distances between mobility networks in the test set (red), synthetic set (blue), and mixed set (green), for the four datasets. For each dataset, we compare the overlap of the distributions of MoGAN and the two baselines (Gravity and Radiation). For both the Gravity model and the Radiation model, the three distributions are significantly different, especially for the latter

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 test set (column JS_s). The last four $\Delta_{x,Z}$-like columns represent the improvement of MoGAN compared to the Gravity model on the mixed and the synthetic sets (columns $\Delta_{m,G}$ and $\Delta_{s,G}$) and the improvement of MoGAN compared to the Radiation model on the mixed and synthetic sets (columns $\Delta_{m,R}$ and $\Delta_{s,R}$)

Data	MoGAN		Gravity		Radiation		Rel. Improvement			
	JS_m	JS_s	JS_m	JS_s	JS_m	JS_s	$\Delta_{m,G}$	$\Delta_{s,G}$	$\Delta_{m,R}$	$\Delta_{s,R}$
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 shows that, according to the CPC, MoGAN outperforms the Gravity and Radiation 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 of the comparison with the Gravity and Radiation models for RMSE, CD, weights distribution and weight-distances distribution in Supplementary Notes 7, 8 and 9.

MoGAN also outperforms the Random Weighted model for all proposed metrics. Figure 5 compares the performance of MoGAN and the Random Weighted model according to CPC. For each dataset, MoGAN's test, synthetic and mixed set distributions are more overlapping than the ones of the Random Weighted model. We report the results of the comparison with Random Weighted model for RMSE, CD, weights distribution and weight-distances distribution in Supplementary Notes 11-14. Table 3 shows that, according to CPC, MoGAN outperforms the Random Weighted model for all datasets.

Table 3 JS divergences of the distributions of the CPC scores with the Random Weighted model. 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 test set (column JS_s). The last two $\Delta_{x,Z}$-like columns represent the improvement of MoGAN compared to the Random Weighted model on the mixed and the synthetic sets (columns $\Delta_{m,\mathrm{RW}}$ and $\Delta_{s,\mathrm{RW}}$)

Data	MoGAN		Random Weighted		Rel. Improvement	
	JS_m	JS_s	JS_m	JS_s	Δ_m, RW	Δ_s, RW
$\mathrm{NYC_{bike}}$	**0.06**	**0.08**	0.45	0.63	86%	88%
$\mathrm{NYC_{taxi}}$	**0.09**	**0.11**	0.37	0.59	76%	82%
$\mathrm{CHI_{bike}}$	**0.14**	**0.16**	0.4	0.56	64%	71%
$\mathrm{CHI_{taxi}}$	**0.08**	**0.09**	0.37	0.55	79%	84%

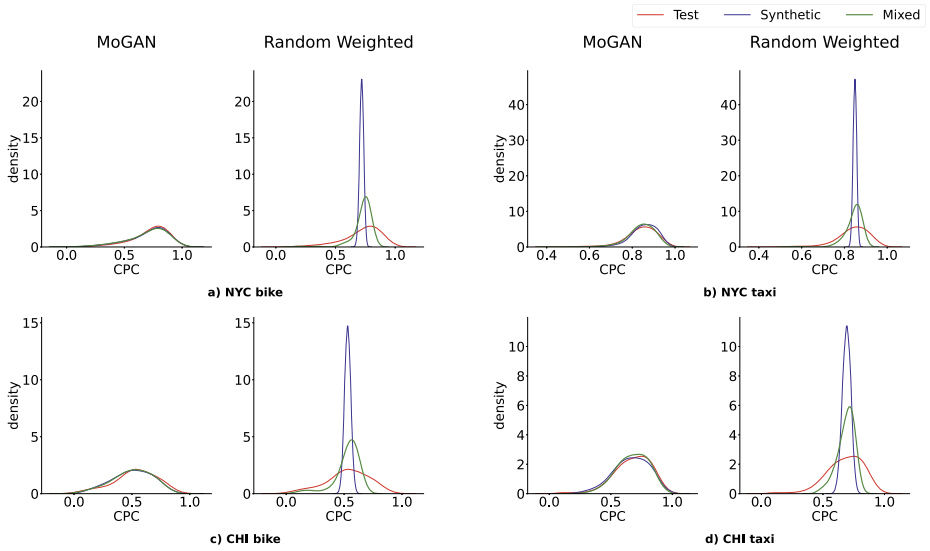

Figure 5 Results for the CPC with Random Weighted Model. Distributions of the pairwise CPC distances between mobility networks in the test set (red), synthetic set (blue), and mixed set (green), for the four datasets. For each dataset, we compare the overlap of the distributions of MoGAN and the Random Weighted model. MoGAN distributions are perfectly overlapping, while the Random Weighted ones show significant differences

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 6 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).

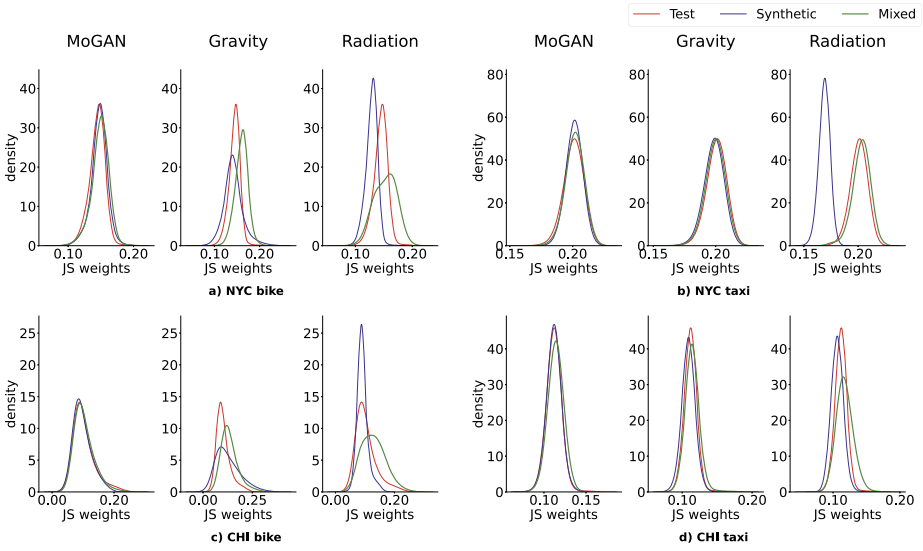

Figure 6 Results for weights distribution. Distributions of the pairwise JS distance between the distribution of weights of the mobility networks in the test set (red), synthetic set (blue), and mixed set (green), for the four different datasets. For each dataset, we compare the overlap of the distributions 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

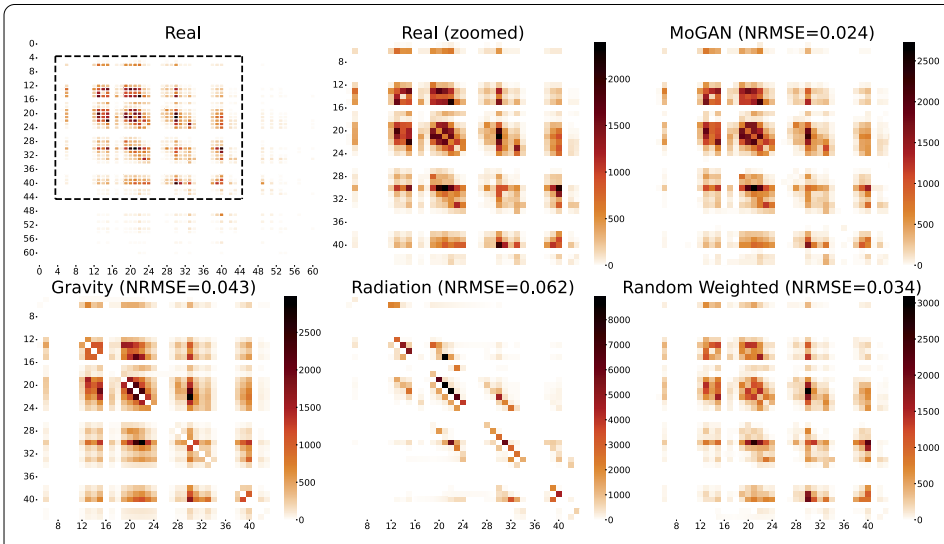

Figure 7 Visual comparison of the adjacency matrices of the Mobility Networks. Visualization of the more dense part of the mobility networks of NYC Bikes having the maximum sum of flows observed in the Test Set (Real Zoomed) and of the Mobility Networks having the maximum sum of flows observed in the fake sets produced by all of the other models. Per each generated matrix, we reported the RMSE with respect to the Real matrix. In the top left panel, we show the full 64×64 mobility network and highlight the most dense zones, on which we focus in the other plots of the figure

In Fig. 7, we compare a subset of the entries in the adjacency matrices representing the generated mobility networks with the adjacency matrix of a real mobility network. We compared only this part of the matrices for visualization reasons: the external part of

them is, in fact, made up of 0 entries. For each model, we visualize the generated mobility network with the maximum sum of flows. We observe that MoGAN's adjacency matrix is way more similar to the real one than the other models. The Gravity model produces an adjacency matrix that looks quite similar to the real one, but it lacks the self-loops (the elements on the diagonal). The Random Weighted model's matrix resembles the real one but the magnitude of flows differ in several parts of the network. The Radiation model's adjacency matrix is way different to the real one.

Figure 7 shows that MoGAN is way better than the Gravity model at predicting flows between close tiles. In contrast, the two models reach a similar performance for flows regarding tiles that are very distant to each other. In Supplementary Note 15, we report the correlation between the error and the distance between flows' tiles for the BikeNYC dataset, for both MoGAN and the Gravity model.

7 Conclusion

This paper introduces 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 classic models such as the Gravity and the Radiation models.

Although MoGAN's performance is encouraging, it also has some limitations. Being based on DCGAN [49], 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], which would better capture the network dependencies and include other location-related information (e.g., population or relevance), and the use of the Wasserstein loss [3], which improves the performance of GANs in several contexts [21, 62]. It would also be interesting to test MoGAN's effectiveness on cities of different sizes and shapes and regarding the generation of individual mobility trajectories, which represent the aggregated movements of single individuals among a city's locations [6, 50, 53]. Finally, we plan to design a version of MoGAN capable of generating a network describing the mobility network of a weekday, a weekend day, or a specific day of the week.

An important aspect to investigate as future work is also to what extent MoGAN is geographically transferable [34], i.e., it can be trained on a specific city and then used to generate mobility networks in a different city effectively. Geographic transferability can be crucial when there is a scarcity or even an absence of mobility data for a city.

In this study, we use data from Chicago and New York City, which differ considerably by size, population, and shape, as well as by socio-demographic factors, POIs distribution, land use, etc. So, it does not make sense to transfer Chicago's MoGAN to New York City and vice versa. We leave experiments about the geographic transferability of MoGAN among cities to future works.

As MoGAN leverages the architecture of DCGAN, it only works with 64×64 matrices. While representing geographic areas with less than 64×64 zones is not an issue (using, e.g., padding techniques [19]), in its current version, MoGAN cannot work with areas split into more than 64×64 zones. Future extensions of MoGAN may consider using GAN architectures that deal with matrices larger than 64×64 [61]. Adapting such models to

deal with temporal and spatial aspects would allow us to design a new GAN for mobility flows to deal with larger geographic areas.

Another promising future direction is developing a GAN to generate a realistic mobility network for a specific condition (e.g., a rainy day or a day with some public events in the city). Having a so-called conditional GAN [38] may represent a unique opportunity for policymakers to generate realistic scenarios for specific circumstances. Finally, an exciting open challenge consists in interpreting which rules or well-known mobility laws (e.g., the gravity law) generative models are learning.

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. MoGAN can synthesize aggregated movements within a city into a realistic generator, which can be used for data augmentation, 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 the presence of pandemic-driven mobility restrictions, events, and natural disasters.

Acknowledgements
We thank Ramon Ferrer-i-Cancho, Matteo Böhm, Giuliano Cornacchia, and Vasiliki Voukelatou for the useful suggestions. We thank Daniele Fadda and Eleonora Cappuccio for the visualization suggestions. We thank CINI Lab for recognizing a price to the ideas that guided the development of MoGAN. We also thank René Ferretti and Dante Milonga for the inspiration.

Funding
Luca Pappalardo and Giovanni Mauro have been supported by EU projects: 1) SoBigData++ grant agreement #871042 and 2) NextGenerationEU - National Recovery and Resilience Plan (Piano Nazionale di Ripresa e Resilienza, PNRR), project "SoBigData.it - Strengthening the Italian RI for Social Mining and Big Data Analytics", prot. IR0000013, avviso n. 3264 on 28/12/2021.

Availability of data and materials
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/jonpappalord/GAN-flow.

Declarations

Competing interests
The authors declare that they have no competing interests.

Author contribution
GM: study conceptualization, data preprocessing and analysis, experiment running, code implementation, interpretation of results, writing, plots and images. ML: study conceptualization, interpretation of results. AL: study conceptualization, interpretation of results, writing. BL: interpretation of results, writing, study direction. LP: study conceptualization, experiment design, interpretation of results, writing, study direction and management. All authors read and approved the final manuscript.

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

Publisher's Note

Received: 22 February 2022 Accepted: 16 November 2022 Published online: 05 December 2022

References

1. Alon N, Naor A (2004) Approximating the cut-norm via Grothendieck's inequality. In: Proceedings of the thirty-sixth annual ACM symposium on theory of computing, pp 72–80

2. Andrienko G, Andrienko N, Boldrini C, Caldarelli G, Cintia P, Cresci S, Facchini A, Giannotti F, Gionis A, Guidotti R, Mathioudakis M, Muntean CI, Pappalardo L, Pedreschi D, Pournaras E, Pratesi F, Tesconi M, Trasarti R (2020) (so) big data and the transformation of the city. Int J Data Sci Anal
3. Arjovsky M, Chintala S, Bottou L (2017) Wasserstein generative adversarial networks. In: International conference on machine learning, pp 214–223. PMLR
4. Barbosa H, Barthelemy M, Ghoshal G, James CR, Lenormand M, Louail T, Menezes R, Ramasco JJ, Simini F, Tomasini M (2018) Human mobility: models and applications. Phys Rep 734:1–74. https://doi.org/10.1016/j.physrep.2018.01.001
5. Batty M (2013) The new science of cities. MIT Press, Cambridge
6. Berke A, Doorley R, Larson K, Moro E (2022) Generating synthetic mobility data for a realistic population with rnns to improve utility and privacy. CoRR, 2201.01139. arXiv:2201.01139
7. Böhm M, Nanni M, Pappalardo L (2022) Gross polluters and vehicle emissions reduction. Nat Sustain. https://doi.org/10.1038/s41893-022-00903-x
8. Carey HC (1867) Principles of social science. JB Lippincott & Company
9. chicago.gov: TLC Trip Record Data (2013–). https://data.cityofchicago.org/Transportation/Taxi-Trips/wrvz-psew/data
10. Chiu P-K (2018) cutnorm package. https://pypi.org/project/cutnorm/
11. citibikenyc.com: CitiBike System Data (2013–). https://www.citibikenyc.com/system-data
12. Cornacchia G, Pappalardo L (2021) A mechanistic data-driven approach to synthesize human mobility considering the spatial, temporal, and social dimensions together. ISPRS Intl J Geo-Inf 10(9). https://doi.org/10.3390/ijgi10090599
13. Creswell A, White T, Dumoulin V, Arulkumaran K, Sengupta B, Bharath AA (2018) Generative adversarial networks: an overview. IEEE Signal Process Mag 35(1):53–65
14. divvybikes.com: Divvy System Data (2016–). https://www.divvybikes.com/system-data
15. Erlander S, Stewart NF (1990) The gravity model in transportation analysis: theory and extensions. Vsp
16. Feng J, Yang Z, Xu F, Yu H, Wang M, Li Y (2020) Learning to simulate human mobility. In: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. KDD'20. Association for Computing Machinery, New York, pp 3426–3433. https://doi.org/10.1145/3394486.3412862
17. Fuglede B, Topsoe F (2004) Jensen-Shannon divergence and Hilbert space embedding. In: International symposium onInformation theory, 2004. ISIT 2004. Proceedings. IEEE, p 31
18. Goodfellow I (2016) Nips 2016 tutorial: Generative adversarial networks. arXiv preprint. arXiv:1701.00160
19. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge
20. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Proceedings of the 27th international conference on neural information processing systems, pp 2672–2680
21. Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville A (2017) Improved training of Wasserstein gans. In: Proceedings of the 31st international conference on neural information processing systems, pp 5769–5779
22. Huang D, Song X, Fan Z, Jiang R, Shibasaki R, Zhang Y, Wang H, Kato Y (2019) A variational autoencoder based generative model of urban human mobility. In: 2019 IEEE conference on multimedia information processing and retrieval (MIPR), pp 425–430
23. Jiang S, Yang Y, Gupta S, Veneziano D, Athavale S, Gonzalez MC (2016) The timegeo modeling framework for urban mobility without travel surveys. Proc Natl Acad Sci 113:201524261
24. 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
25. Kroll C, Warchold A, Pradhan P (2019) Sustainable development goals (sdgs): are we successful in turning trade-offs into synergies? Palgrave Communications 5(1):1–11
26. Kulkarni V, Tagasovska N, Vatter T, Garbinato B (2018) Generative models for simulating mobility trajectories. arXiv preprint. arXiv:1811.12801
27. Kullback S (1997) Information theory and statistics. Courier Corporation
28. Lai S, Farnham A, Ruktanonchai NW, Tatem AJ (2019) Measuring mobility, disease connectivity and individual risk: a review of using mobile phone data and health for travel medicine. J Travel Med 26(3)
29. Le Blanc D (2015) Towards integration at last? The sustainable development goals as a network of targets. Sustain Dev 23(3):176–187
30. Lenormand M, Bassolas A, Ramasco JJ (2016) Systematic comparison of trip distribution laws and models. J Transp Geogr 51:158–169
31. Lenormand M, Bassolas A, Ramasco JJ (2016) Systematic comparison of trip distribution laws and models. J Transp Geogr 51:158–169. https://doi.org/10.1016/j.jtrangeo.2015.12.008
32. Liu Q, Dong Z, Wang E (2018) Cut based method for comparing complex networks. Sci Rep 8(1):1–11
33. Liu X, Chen H, Andris C (2018) Trajgans: using generative adversarial networks for geo-privacy protection of trajectory data (vision paper). In: Location privacy and security workshop, pp 1–7
34. Luca M, Barlacchi G, Lepri B, Pappalardo L (2021) A survey on deep learning for human mobility. ACM Comput. Surv. 55(1):1–44
35. Luca M, Lepri B, Frias-Martinez E, Lutu A (2022) Modeling international mobility using roaming cell phone traces during Covid-19 pandemic. EPJ Data Sci 11(1):22
36. Lucchini L, Centellegher S, Pappalardo L, Gallotti R, Privitera F, Lepri B, De Nadai M (2021) Living in a pandemic: changes in mobility routines, social activity and adherence to Covid-19 protective measures. Sci Rep 11(1):24452. https://doi.org/10.1038/s41598-021-04139-1
37. Masucci AP, Serras J, Johansson A, Batty M (2013) Gravity versus radiation models: on the importance of scale and heterogeneity in commuting flows. Phys Rev E 88(2):022812
38. Mirza M, Osindero S (2014) Conditional generative adversarial nets. Preprint. Available at arXiv:1411.1784
39. Moreira-Matias L, Gama J, Ferreira M, Mendes-Moreira J, Damas L (2013) Predicting taxi–passenger demand using streaming data. IEEE Trans Intell Transp Syst 14(3):1393–1402
40. Nikolentzos G, Siglidis G, Vazirgiannis M (2021) Graph kernels: a survey. J Artif Intell Res 72:943–1027
41. nyc.gov: TLC Trip Record Data (2009–). https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
42. O'Donnell R, Wu Y (2008) 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

43. Oliver N, Lepri B, Sterly H, Lambiotte R, Deletaille S, De Nadai M, Letouzé E, Salah AA, Benjamins R, Cattuto C et al (2020) Mobile phone data for informing public health actions across the COVID-19 pandemic life cycle
44. Ouyang K, Shokri R, Rosenblum DS, Yang W (2018) A non-parametric generative model for human trajectories. In: IJCAI, pp 3812–3817
45. Pappalardo L, Simini F (2018) Data-driven generation of spatio-temporal routines in human mobility. Data Min Knowl Discov 32(3):787–829
46. Pappalardo L, Simini F, Barlacchi G, Pellungrini R (2022) Scikit-mobility: a python library for the analysis, generation, and risk assessment of mobility data. J Stat Softw 103(4):1–38. https://doi.org/10.18637/jss.v103.i04
47. Pepe E, Bajardi P, Gauvin L, Privitera F, Lake B, Cattuto C, Tizzoni M (2020) Covid-19 outbreak response, a dataset to assess mobility changes in Italy following national lockdown. Sci Data 7(1):1–7
48. Prieto Curiel R, Pappalardo L, Gabrielli L, Bishop SR (2018) Gravity and scaling laws of city to city migration. PLoS ONE 13(7):1–19. https://doi.org/10.1371/journal.pone.0199892
49. Radford A, Metz L, Chintala S (2016) Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. arXiv:1511.06434
50. Rinzivillo S, Gabrielli L, Nanni M, Pappalardo L, Pedreschi D, Giannotti F (2014) The purpose of motion: learning activities from individual mobility networks. In: 2014 international conference on data science and advanced analytics (DSAA), pp 312–318
51. Ruktanonchai NW, Floyd JR, Lai S, Ruktanonchai CW, Sadilek A, Rente-Lourenco P, Ben X, Carioli A, Gwinn J, Steele JE, Prosper O, Schneider A, Oplinger A, Eastham P, Tatem AJ (2020) Assessing the impact of coordinated Covid-19 exit strategies across Europe. Science 369(6510):1465–1470
52. Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G (2008) The graph neural network model. IEEE Trans Neural Netw 20(1):61–80
53. Schneider CM, Belik V, Couronné T, Smoreda Z, González MC (2013) Unravelling daily human mobility motifs. J R Soc Interface 10(84):20130246
54. Simini F, Barlacchi G, Luca M, Pappalardo L (2021) A deep gravity model for mobility flows generation. Nat Commun 12(1):1–13
55. Simini F, González MC, Maritan A, Barabási A-L (2012) A universal model for mobility and migration patterns. Nature 484(7392):96–100
56. Tantardini M, Ieva F, Tajoli L, Piccardi C (2019) Comparing methods for comparing networks. Sci Rep 9(1):1–19
57. United Nations General Assembly: Transforming our world: the 2030 Agenda for Sustainable Development. https://sdgs.un.org/2030agenda. Accessed: 2021-02-23 (2015)
58. Van Erven T, Harremos P (2014) Rényi divergence and Kullback-Leibler divergence. IEEE Trans Inf Theory 60(7):3797–3820
59. Vishwanathan SVN, Schraudolph NN, Kondor R, Borgwardt KM (2010) Graph kernels. J Mach Learn Res 11:1201–1242
60. Wang J, Kong X, Xia F, Sun L (2019) Urban human mobility: data-driven modeling and prediction. In: ACM SIGKDD explorations newsletter, pp 1–19
61. Wang Z, She Q, Ward TE (2021) Generative adversarial networks in computer vision: a survey and taxonomy. ACM Comput Surv 54(2):1–38
62. Weng L (2019) From gan to wgan. arXiv preprint. arXiv:1904.08994
63. Yan X-Y, Wang W-X, Gao Z-Y, Lai Y-C (2017) Universal model of individual and population mobility on diverse spatial scales. Nat Commun 8(1):1639. https://doi.org/10.1038/s41467-017-01892-8
64. Yin D, Yang Q (2018) Gans based density distribution privacy-preservation on mobility data. Secur Commun Netw 2018
65. Zipf GK (1946) The p 1 p 2/d hypothesis: on the intercity movement of persons. Am Sociol Rev 11(6):677–686