

User Mobility Synthesis based on Generative Adversarial Networks: A Survey

Seungjae Shin, Hongseok Jeon, Chunglae Cho, Seunghyun Yoon and Taeyeon Kim

Network Research Division

Electronics and Telecommunications Research Institute

218 Gajeong-ro, Yuseong-gu, Daejeon, 34129, Republic of Korea

{sjshin0505, jeonhs, clcho, shpyoon, tykim}@etri.re.kr

Abstract—User mobility characteristics are considered to significantly affect the performance and UX of modern IT systems such as mobile communications, context-aware services, location-based applications, smart mobility, and so on. Due to privacy concerns, legal issues, and expensive measurement costs for real user mobility traces, synthetic mobility traces are widely used for research and development purposes. Until now, most of the user mobility synthesis approaches have been categorized into two major paradigms: stochastic modeling and simulation. However, they naturally have a limitation in mimicking actual user movements because it is impossible to capture all temporal, spatial, and behavioral characteristics that are implicitly melted in real user mobility. Along with the recent advancements of deep learning technologies, there have been several challenging proposals that exploit the power of the deep learning, especially generative models, in mobility synthesis. This paper reviews and summarizes recently proposed user mobility synthesis schemes based on generative adversarial networks that have been one of the most leading deep learning technologies for the last few years.

Index Terms—user mobility, mobility synthesis, mobility generation, mobility trace synthesis, mobility trace generation, generative adversarial networks, GAN

I. INTRODUCTION

Lasting improvements in mobile communications, *Internet-of-Things* (IoT) and data-driven technologies have continuously been introduced to information technology (IT) systems, especially a variety of novel features such as future prediction, context awareness, location-based services, personalized applications, and ubiquitous access. These features sometimes exploit user mobility traces as important data for enhancing the processing performance and the quality of *user experience* (UX) [1], [2]. However, it is not trivial to use real user mobility traces because of privacy concerns and expensive measurement costs. Therefore, synthetic mobility traces are widely used instead of real ones in most of *research & development* (R&D) processes.

The majority of existing mobility synthesis schemes are mainly categorized into two approaches: One is a *stochastic modeling* approach [3]–[7], and the other is a *simulation-based* approach [8]–[12]. The stochastic modeling approach synthesizes user mobility traces by generating random variables that follow a set of probability distribution functions that reflect the characteristics of real user mobility samples through distribution fitting [2], [13]. The simulation-based ap-

proach generates mobility traces by simulating user movement behaviors in various scenarios and environments like vehicles and road networks.

While the above mobility synthesis schemes have shown considerable achievements, they still have natural limitations in capturing all the characteristics of user mobility including intentions of visits that lead to the movements and causal relationships between the random variables that comprise the mobility traces.

Nowadays, deep learning brings new breakthroughs to a variety of domains such as image processing [14], computer vision [15], [16], autonomous vehicles [17], unmanned aerial vehicles (UAV) [18], games [19], [20], and so on. Especially, *generative adversarial network* (GAN) is recognized as a new innovative way for generation and synthesis purposes across diverse data types including text, image, voice, video, and even more complicated ones like natural languages [21]–[27].

This trend also appears in user mobility synthesis. Recently, several studies have been conducted to exploit the GAN as a new stochastic generator instead of the conventional stochastic models [28]–[32]. These works show the potential of deep generative learning-based mobility synthesis schemes. In this paper, we present a survey on recent research efforts that exploit the GAN for synthesizing user mobility traces.

The remaining parts of this paper are organized as follows: In Section II, we briefly review the GAN. Then, we introduce and summarize the recent research efforts on mobility synthesis based on the GAN in Section III. In Section IV, we conclude the paper.

II. GENERATIVE ADVERSARIAL NETWORKS: AN OVERVIEW

GAN is a type of generative machine learning paradigm that exploits *fake-versus-discriminate* game between two neural networks so as to eventually obtain a good quality generation, in other words, the generated data samples are likely to be indistinguishable from the real ones. Since its proposal in 2014 [21], the GAN has been extensively explored and shown many promising results in both the literature and industry.

Fig. 1 depicts the basic concept of a GAN. The concept adopts the adversarial game process in the training of two neural networks: the generator G and the discriminator D . For each iteration of the training process, G generates a

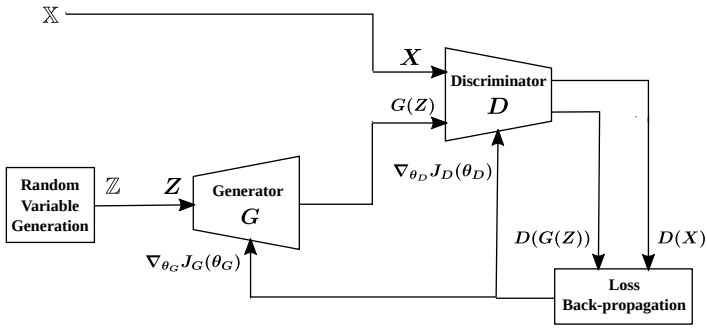


Fig. 1. Basic concept of a GAN

set of fake data samples $G(\mathbb{Z})$ for the given latent random variables $\mathbb{Z} = \{z^{(0)}, z^{(1)}, z^{(2)}, \dots, z^{(m)}\}$. And, D determines the probability of real data for each sample in $\mathbb{X} \cup G(\mathbb{Z})$, where $\mathbb{X} = \{x^{(0)}, x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$. For example, if D produces 0.75 for the given data sample, it means that the data sample is expected to be real with a probability of 75% and fake with a probability of 25%. After D determines the probability for every sample in $\mathbb{X} \cup G(\mathbb{Z})$, G and D change their weights θ_G and θ_D by back-propagating some amounts of the losses¹ that evaluate how they did well with respect to their corresponding goals. By iterating this adversarial fake-versus-discriminate competition, G and D progressively evolve their weights so as to obtain a much better generation and discrimination capability. When $\mathbb{E}[D(G(\mathbb{Z}))]$ is greater than some threshold, the training finishes, and the generated data samples are expected to be realistic enough.

The goals of G and D are opposite to each other: The goal of G is to fool D by generating fake data samples as similar as possible to the real ones so as to make D incapable of discriminating between the real and fake ones. In contrast, D tries to distinguish real and fake data sample as accurately as possible. Such adversarial goals are expressed as the following min-max game model:

$$\min_G \max_D V(D, G) = \mathbb{E}_{X \sim p_{data}(x)} [\log D(X)] + \mathbb{E}_{Z \sim p_z(z)} [\log(1 - D(G(Z)))] \quad (1)$$

where $X (\in \mathbb{X})$ denotes a real data sample, and $Z (\in \mathbb{Z})$ denotes a latent random variable used to generate a fake data sample. X and Z follow the *probability density function* (pdf) $p_{data}(x)$ and $p_z(z)$, respectively.

In Eq. (1), D should likely produce 1 for each real data sample X to maximize the first component $\mathbb{E}_{X \sim p_{data}(x)} [\log D(X)]$, and to produce 0 for each fake data input $G(Z)$ to maximize the second component $\mathbb{E}_{Z \sim p_z(z)} [\log(1 - D(G(Z)))]$. On the other hand, G should try to fool D into producing a higher value for each $G(Z)$ by generating it as similar as possible to X in order to minimize the second component $\mathbb{E}_{Z \sim p_z(z)} [\log(1 - D(G(Z)))]$.

¹The term *loss* is also called the *error* or *cost*.

Therefore, the loss functions of G and D are denoted by $J_G(\theta_G)$ and $J_D(\theta_D)$, respectively, as

$$J_D(\theta_D) = -\frac{1}{m} \sum_{i=1}^m \left[\log D(x^{(i)}) + \log(1 - D(G(z^{(i)}))) \right]$$

$$J_G(\theta_G) = \frac{1}{m} \sum_{i=1}^m \log(1 - D(G(z^{(i)}))). \quad (2)$$

Note that the $\pm$ sign of $J_D(\theta_D)$ is reversed from Eq. (1) to express the loss for maximization as the loss for minimization.

Since its proposal in [21], GAN has been extensively explored and improved up to now [25], [26]. Among those advancements, we list some variants that are discussed in the next section:

- **SGAN** (Semi-supervised GAN) [33] is a variant of GAN that provides not only data generation but also regression, classification, or prediction.
- **WGAN** (Wasserstein GAN) [34] introduces the *Wasserstein* distance as an improved loss function for better learning stability. It was further improved to WGAN-GP (WGAN with Gradient Penalty) [35] by adding the *gradient penalty* as a regularization term.
- **DC-GAN** (Deep Convolutional GAN) [36] refers to the GAN that uses deep *convolutional neural networks* (CNN) [37] as the internal part of the generator and discriminator. Compared to using conventional fully-connected neural networks, the DC-GAN is known to be much more appropriate to synthesize multi-dimensional matrix data including image and video.
- **CGAN** (Conditional GAN) [38] takes conditioning inputs in addition to original input data to generate synthetic data that follows conditional probability. With the CGAN, we can synthesize data for much more detailed or specific scenarios.
- **RGAN** (Recurrent GAN) [39] refers to a GAN that uses *recurrent neural network* (RNN) [17] as the internal part of the generator and discriminator. Compared to using other neural approximators, RGAN is known to be much more proper to synthesize sequential or time-series data like text, language and trajectory.
- **SeqGAN** (Sequential GAN) [40] is designed for synthesizing sequential data, thus, it is a similar variant to the RGAN. Instead of using sequential neural approximators like the RNN, it designs the generator as a *reinforcement learning* agent [41] that learns a strategic *observe-action-reward* process for more similarity in synthetic data.

III. USER MOBILITY SYNTHESIS SCHEMES BASED ON GAN

To the best of our knowledge, there have been only five studies on the GAN based mobility synthesis. In this section, we review each of them from the perspective of how GAN is used for generating user mobility data.

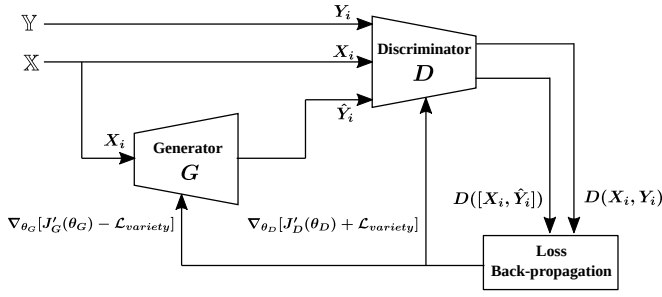


Fig. 2. Social GAN: overview

A. Social GAN

Social GAN [28] can be explained as a *meter-scale* microscopic mobility trace generator that synthesizes realistic and *socially acceptable* walking trajectories of pedestrians. The term “socially acceptable” means that mobility synthesis should take into account social norms like *yielding right-of-way* or *respecting personal space* in a crowded place. The generated trajectories can be used as *a-few-seconds-future predictions* for the given input trajectory and as such must be useful to improve the motion intelligence of self-driving cars or social robots.

Fig. 2 shows the system overview of Social GAN. In the figure, the generator G takes as inputs a set of user mobility trace samples $\mathbb{X} = \{X_1, X_2, \dots, X_n\}$ where $X_i = (x_i^t, y_i^t)$ denotes the 2-dimensional coordinates of user i from the time steps $t = 1, \dots, t_{obs}$. And then, G generates $\hat{\mathbb{Y}} = \{\hat{Y}_1, \hat{Y}_2, \dots, \hat{Y}_n\}$ where $\hat{Y}_i = (\hat{x}_i^t, \hat{y}_i^t)$ denotes the fake coordinates of user i from the time steps $t = t_{obs} + 1, \dots, t_{pred}$. In short, $\hat{\mathbb{Y}}$ is the fake future mobility predictions for $\mathbb{X}$ from $t_{obs} + 1$ to t_{pred} . The discriminator D takes $\mathbb{X}$, $\mathbb{Y}$ and $\hat{\mathbb{Y}}$ as inputs and then, determines $D([X_i, Y_i])$ and $D([X_i, \hat{Y}_i])$ for each user i where $Y_i = (y_i^t, x_i^t) \in \mathbb{Y} = \{Y_1, Y_2, \dots, Y_n\}$ denotes a real user mobility sample of user i from the time steps $t = t_{obs} + 1, \dots, t_{pred}$. In this arrangement, G is likely to fool D into producing a higher $D([X_i, \hat{Y}_i])$. And, D tries to produce a higher $D([X_i, Y_i])$ and lower $D([X_i, \hat{Y}_i])$. In addition, because fake mobility traces are used as future trajectory predictions, D and G also make efforts to predict $\hat{Y}_i$ as accurately as possible for the given (X_i, Y_i) pair. Therefore, the loss functions of G and D include $L2$ norms that are widely used in supervised learning for regressions as follows:

$$\begin{aligned} J_D(\theta_D) &= J'_D(\theta_D) - \mathcal{L}_{variety}, \\ J_G(\theta_G) &= J'_G(\theta_G) + \mathcal{L}_{variety} \end{aligned} \quad (3)$$

where $J'_D(\theta_D) = -\mathbb{E}[\log D([X_i, Y_i]) + \log(1 - D([X_i, \hat{Y}_i]))]$ and $J'_G(\theta_G) = \mathbb{E}[\log(1 - D([X_i, \hat{Y}_i]))]$. And, $\mathcal{L}_{variety} = \min_k \|Y_i - \hat{Y}_i^{(k)}\|_2$ where k is the factor of the fake sampling. For example, if $k = 3$, $\mathcal{L}_{variety}$ is computed from one real sample and three fake ones. For the predictive capability devised by the introduction of $\mathcal{L}_{variety}$, Social GAN belongs to the SGAN mentioned in Section II.

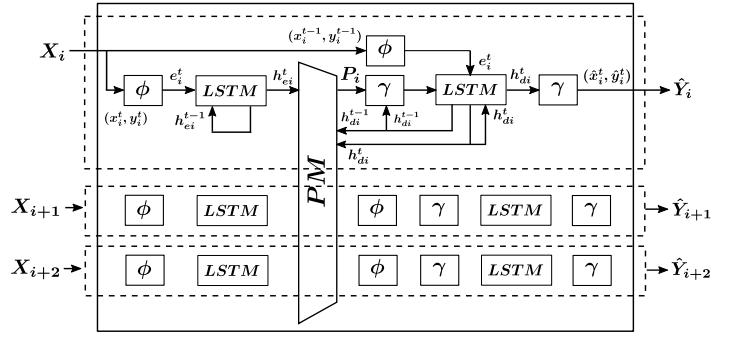


Fig. 3. Internal part of the generator of the Social GAN

Fig. 3 shows the internal part of G that adopts *long short-term memory* (LSTM) [42] to accumulate sequential and historic patterns of trajectories in the memory matrix. This makes the Social GAN also belongs to RGAN [39]. For $i \in \{1, 2, \dots, n\}$, G processes all X_i simultaneously.² For each coordinate (x_i^t, y_i^t) , G produces an embedding vector e_i^t and encoding vector h_{ei}^t by applying the following functions:

$$\begin{aligned} e_i^t &= \phi(x_i^t, y_i^t; W_{ee}) \\ h_{ei}^t &= LSTM(h_{ei}^{t-1}, e_i^t; W_{encoder}) \end{aligned} \quad (4)$$

where $\phi(\cdot)$ is an embedding function with a *rectified linear unit* (ReLU) [43] non-linearity. W_{ee} and $W_{encoder}$ are weight matrices of $\phi(\cdot)$ and $LSTM(\cdot)$, respectively. Note that all users in $\mathbb{X}$ share the same $W_{encoder}$.

After t_{obs} , G produces $\hat{Y}_i = (\hat{x}_i^t, \hat{y}_i^t)$ by repeating recurrent applications of the following functions:

$$\begin{aligned} e_i^t &= \phi(x_i^{t-1}, y_i^{t-1}; W_{ed}) \\ P_i &= PM(h_{di}^{t-1}, \dots, h_{dn}^{t-1}) \\ h_{di}^t &= LSTM(\gamma(P_i, h_{di}^{t-1}), e_i^t; W_{decoder}) \\ (\hat{x}_i^t, \hat{y}_i^t) &= \gamma(h_{di}^t) \end{aligned} \quad (5)$$

where $\phi(\cdot)$ is an embedding function with ReLU non-linearity. γ is a *multi-layer perceptron* (MLP). W_{ed} and $W_{decoder}$ are weight matrices of $\phi(\cdot)$ and $LSTM(\cdot)$, respectively. And, at $t = t_{obs}$, h_{di}^t is initialized as

$$\begin{aligned} c_i^t &= \gamma(P_i, h_{ei}^t; W_c), \\ h_{di}^t &= [c_i^t, z] \end{aligned} \quad (6)$$

where γ is a MLP with ReLU non-linearity and weights W_c . Note that a latent random variable z is used here.

Pooling module (PM) is a pooling function which produces pooled tensor P_i that holds the correlation between the trajectory of user i and those of all other users. Semantically, P_i represents the correlations among trajectories that imply the social behaviors making people prefer not to interfere in the paths of others for the sake of social aspects such as *manners*, *kindness*, *law*, and so on. We illustrate the internal part of PM in Fig. 4. For more details about the PM, please refer [28].

²For convenience, we drew the computational flow only for the thread of $(X_i, \hat{Y}_i)$ in Fig. 3.

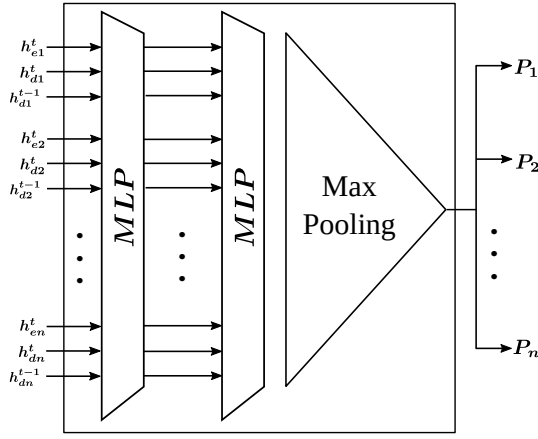


Fig. 4. Internal of the pooling module of the Social GAN

The discriminator D consists of encoders that separately evaluates $D([X_i, Y_i])$ and $D([X_i, \hat{Y}_i])$ for each user i . The encoders are implemented in the same manner as Eq. (4).

In the experiments in [28], the Social GAN reduced, on average, 12.86% of the *average displacement error* (ADE) and 20.39% of the *final displacement error* (FDE)³, from up-to-date LSTM based methods [44], for a set of microscopic human walking trajectories where each trajectory is 12-consecutive locations for every 0.4s from five datasets which covers 1536 pedestrians in crowded places at a university, hotel and stores [45], [46]. Additionally, the experimental results also show that the Social GAN effectively mimics the human walking behaviors caused by social norms as follows:

- A person changes her direction or speed to avoid a collision with another when they look like to collide at the same time.
- When the trajectories of two walkers are merging, they implicitly negotiate their order in walking. One overtakes the other in front, and the other follows behind the front walker to yield the right-of-way.
- When a group of persons are walking together, they are likely to take the same changes in direction and speed to avoiding collision with other groups, in order to keep their alignment in the walking.

B. Non-Parametric Trajectory Generator

Ouyang et. al proposed a non-parametric trajectory generator [29]. It combines WGAN-GP [35], DC-GAN [36], and *location-major mobility conversion* to generate synthetic mobility traces that mimic not only geographically but also *semantically* realistic visiting behaviors of humans. The term “semantically realistic” means that the synthetic mobility data have similar semantic features (e.g. *visiting popularity*) to those of real data. And, the term “non-parametric” means

³ADE is the average L_2 distance between the real trajectory and its fake counterpart over all the predicted time steps, and FDE is the L_2 distance between the final destination of the real trajectory and that of its fake counterpart. The reduction ratios are computed by comparing the results of LSTM and SGAN-20VP-20 in Table 1 in [28].

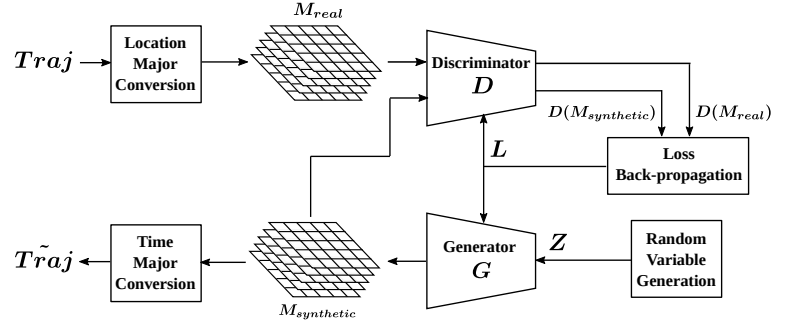


Fig. 5. Non-parametric trajectory generator: overview

that the generator does not assume any explicit parametric distribution function for mobility traces. This is an enjoyable advantage that saves us from looking for the appropriate parametric function for input traces.⁴ In [29], the authors mention that the non-parametric trajectory generator preserves both of the aggregate and individual statistics of human trajectories, such as inter/intra-similarity across trajectories.

We show a system overview of the non-parametric trajectory generator in Fig. 5. In the figure, $Traj = \{(x, y)_{t_1}, \dots, (x, y)_T\}$ denotes a set of user mobility trace samples, where $(x, y)_{t_i}$ is a visit to location (x, y) at time t_i ($i = 1, \dots, T$). In [29], the authors insist that $Traj$ is a *time-major representation* that is not appropriate for identifying semantic behaviors such as the visit frequency and stay duration. Therefore, they propose to convert $Traj$ to the multi-dimensional matrix M_{real} that is called the *location-major representation*. The location-major conversion consists of the following steps:

- The region of trajectories are discretized into a 2-dimensional $(N_1 \times N_2)$ -grid. By this, some different locations that are close to each other may be mapped to a same grid block. So, N_1 and N_2 should be set with a tolerable precision. We still use (x, y) to denote the discretized grid block for convenience.
- Timely consecutive visits to a same block are grouped into a *stay* $r_{t_i:t_j} = \{(x, y)_{t_i}, \dots, (x, y)_{t_j}\}$, where $t_i < t_j$. Now, we can get a visit time t and duration d for each stay $r_{t_i:t_j}$ where $t = t_i$ and $d = t_j - t_i$.
- M_{real} is constructed as the $(N_1 \times N_2 \times K)$ -matrix by setting $M_{x,y,k} = (t, d)$ for the k -th stay, where $k \in \{1, \dots, K\}$. Compared to $Traj$, M_{real} is more suitable to capture and visualize the semantic features of mobility much more directly. For example, suppose that we plot a heat-map of d in the 3-dimensional space, where each axis corresponds to x , y and k , respectively. If we use a brighter color for a higher value of d , we can easily identify frequently visited and long-stayed locations.

G generates $M_{synthesis}$ that is, of course, the location-major representation. D determines whether if $D(M_{real})$ or

⁴In fact, gratefully, all the other works in Section III are also the non-parametric method. During training, the GAN self-configures its weights as implicit parameters of the target distribution function.

$D(M_{synthesis})$ is from real mobility trace samples. During the training, G and D compete with each other in an adversarial game while improving their fake and discrimination capabilities by performing back-propagation with an improved loss function as follows:

$$L = \mathbb{E}_{\hat{X} \sim p_{M_{synthesis}}(x)} [D(\hat{X})] - \mathbb{E}_{X \sim p_{M_{real}}(x)} [D(X)] + \lambda \mathbb{E}_{\tilde{X} \sim p_{\tilde{x}}(x)} [(\|\nabla_{\tilde{X}} D(\tilde{X})\|)^2] \quad (7)$$

where $p_{\tilde{x}}(x)$ is drawn from a convex combination of $p_{M_{real}}(x)$ and $p_{M_{synthesis}}(x)$. This loss variant is referred to WGAN-GP [35] mentioned in Section II. The terms $\mathbb{E}_{\hat{X} \sim p_{M_{synthesis}}(x)} [D(\hat{X})] - \mathbb{E}_{X \sim p_{M_{real}}(x)} [D(X)]$ and $\mathbb{E}_{\tilde{X} \sim p_{\tilde{x}}(x)} [(\|\nabla_{\tilde{X}} D(\tilde{X})\|)^2]$ are referred to the *Wasserstein distance* and *gradient penalty*, respectively. Eq. (7) is known to provide more stable gradients than those of Eq. (2). For more detail, please refer to [29] and [35].

As mentioned earlier, the non-parametric trajectory generator also belongs to the DC-GAN. The internals of G and D consist of the CNN [37] that consists of convolution layers. In each layer, rectangular filters extract sorts of visualized patterns that are used to recognize and distinguish the visit-stay matrices converted from different trajectories.

After generation, $M_{synthesis}$ is converted to a set of synthetic mobility trace samples $Traj$ which is the time-major representation like $Traj$. The conversion is the reverse process of the location-major conversion described above.

In [29], the authors present several experimental results about the performance of the non-parametric trajectory generator on the Nokia Lausanne location trajectories dataset [47]. In the results, the non-parametric trajectory generator reduces 70.8%, 72.98%, 75.88% and 82.89% of the *Jansen-Shannon divergence* (JSD) of the *visiting probability* from four existing methods: *first order Markov Chain* (MC) [48], *time-dependent MC* [49], *hidden Markov model* (HMM) [50], and *LSTM maximum likelihood estimation* (MLE) [51], respectively. This shows that the non-parametric generator has a much better *geographic similarity* than the existing works. In addition, its generated trajectories have considerable similarity to real ones in the metrics about the semantic reality such as the *top-N visited locations* and *pair-wise semantic distance*. For more detail, please refer to [29] and [52].

C. GAN based Location Density Matrix Generator

Because some personal information of a user can be sometimes exposed from the trajectory (e.g. home address, path for commute), we may use *location density matrix* as aggregate mobility data instead of individual trajectory [53]. The location density matrix is somewhat similar to the location-major representation explained in Section III-B. Fig. 6 illustrates a conversion of the location density matrix from a set of trajectories. In the figure, for example, we can see that block (2, 2) has a density 4 at time slot t_3 because 4 users are in the block at that time slot. Note that one of the trajectories (purple colored) begins at block (3, 2) at time slot t_2 . For

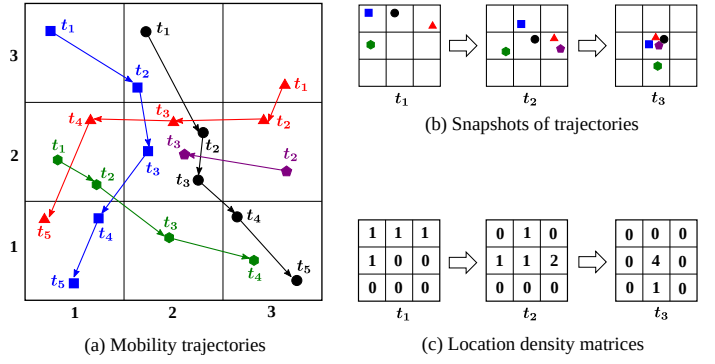


Fig. 6. Example of the location density matrix

the given location density matrix, we can mix some statistical noises to provide a reasonable level of *differential privacy* which means that an addition (deletion) of a trajectory to (from) the input trajectory set does not significantly affect the resulting location density matrix to make it difficult to attack for inferring a personal trajectory [53], [54]. However, this privacy preservation approach has an inherent *privacy-reality trade-off* which implies that one should compromise the reality⁵ for better privacy and vice versa.

In [30], Yin et. al suggested that the GAN can be exploited to generate a synthetic location density matrix with a better privacy-reality trade-off than that of the existing noise mixing approaches for differential privacy. Considering that the synthetic data from the GAN introduces inherent noise incurred from the latent parameter Z of the generator G , they propose to use the synthetic location density matrix generated from the GAN based generator as privacy preserving data, instead of explicitly mixing noise into the real density matrix. In this approach, the reality increases as the training proceeds because the training improves the fake capability of G . On the other hand, the privacy decreases because a higher reality implies a lower noise. Therefore, one should determine the appropriate number for the training iteration to balance this privacy-reality trade-off.

Fig. 7 depicts the system organization of the GAN based location density matrix generator. In the figure, the discriminator D takes as inputs a set of location density matrices $P = \{P_1, P_2, \dots, P_K\}$ converted from the raw trajectory set $Traj$, where P_k is the k -th $(N \times M)$ -matrix which contains N blocks, i.e., $\{b_1, b_2, \dots, b_N\}$, during M time slots, i.e., $\{t_1, t_2, \dots, t_M\}$. The generator G takes as inputs a set of latent random variable matrix $Z = \{Z_1, Z_2, \dots, Z_K\}$, and then, generates a set of synthetic location density matrices $\hat{P} = \{\hat{P}_1, \hat{P}_2, \dots, \hat{P}_K\}$, where $\hat{P}_k$ is the synthetic $(N \times M)$ -matrix corresponding to P_k and Z_k is the $(N \times M)$ -matrix containing random variables. In the training phase, G and D repeat playing a competitive fake-verses-discriminate game process. In each iteration, G synthesizes $\hat{P}$ from Z , and D

⁵For consistency of terminologies throughout this paper, we use the term “reality” instead of “utility” which is the term originally used in [30].

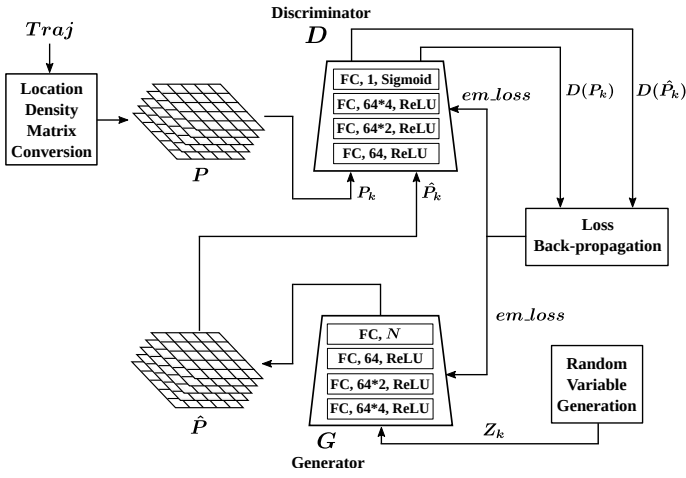


Fig. 7. GAN based location density matrix generator

tries to determine whether if each $P_i \in (P \cup \hat{P})$ is from $Traj$. For the error back-propagation, em_loss is adopted as

$$em_loss(Y, \hat{Y}) = \frac{1}{2K} \sum_{i=1}^{2K} \sum_{j=1}^{2K} y_i \cdot \hat{y}_j \quad (8)$$

where $Y = \{y_1, y_2, \dots, y_{2K}\}$ is the set of the ground truth decision values determining whether each $P_i \in (P \cup \hat{P})$ is real. And, $\hat{Y} = \{\hat{y}_1, \hat{y}_2, \dots, \hat{y}_{2K}\}$ is the set of the decision value made by D for P_i . Note that $|P \cup \hat{P}| = 2K$ from the definitions of P and $\hat{P}$. y_i in Eq. (8) is set differently upon D and G . For D , it is set to 1 only if $P_i \in P$. Conversely, for G , we should set y_i to 1 only if $P_i \in \hat{P}$ to cheat D .

The internal of D consists of three ReLU layers and a single *sigmoid* layer. The first three layers learn distinguishable features from the input density matrix P_i , and the last one computes the decision value $\hat{y}_i$. The internal of G consists of three ReLU layers and a single identity layer. Considering that D and G are opponents to each other, the authors reverse the first three layers in D as the first part of G .

In [30], the author present the experimental results about the performance of the GAN based location density matrix generator on two mobility datasets: MoMo mobile APP (MoMo) [55] and San Francisco cabs (SFC) [56]. In the results, it succeeds to defend against a *trajectory reconstruction attack* [57] with at least 7km of reconstruction error on the MoMo and SFC dataset, and 2km on SFC dataset. That means a much higher level of privacy preservation than that of the existing noise mixing approaches which provide on average 4.32km and 1km of reconstruction error for the MoMo and SFC dataset, respectively. In addition, the GAN based location density matrix generator also achieves a much better privacy-reality trade-off. For example, it reduces at least 66% of the *mean square error* (MSE) between real and synthetic density matrices from the noise mixing methods when we set the target reconstruction error to 2.8km for the SFC dataset.

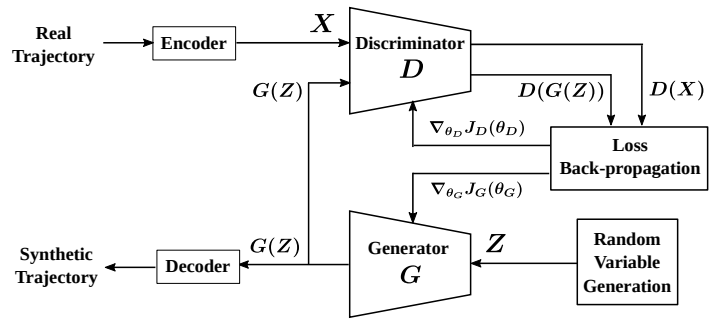


Fig. 8. trajGANs: overview

D. trajGANs

trajGANs [31] is a conceptual design for synthesizing mobility trace samples on road segments and place visit trajectories. The authors claim that, by adopting GAN, one can obtain high-quality synthetic user mobility trace samples that preserve the statistical features of the real samples while managing the geo-privacy of users.

Fig. 8 shows the system overview of trajGANs. The organization is almost similar to the non-parametric trajectory generator presented in Section III-B. In this figure, X denotes the preprocessed data converted from the real trajectory. And, the synthetic mobility trace samples are produced by the decoder from $G(Z)$ that is generated by the generator G . The loss functions are the same as in Eq. (2). A concern of the trajGANs is that they are only conceptual design and not have been implemented or experimented in studies. In [31], the authors just present several suggestions and discussions for the sub-components of trajGANs as follows:

- Two types of user mobility trace samples are considered: One is *road segment*-based traces that present a trajectory as a sequence of road segments and time stamps, i.e. $\langle s_1, t_1 \rangle \rightarrow \langle s_2, t_2 \rangle \rightarrow \dots \rightarrow \langle s_n, t_n \rangle$, where s_i and t_i denote the road segment and time, respectively. The other type is *place*-based traces that are denoted as $\langle p_1, t_1 \rangle \rightarrow \langle p_2, t_2 \rangle \rightarrow \dots \rightarrow \langle p_n, t_n \rangle$, where p_i refers to a place or *point-of-interest* (PoI). If raw mobility trace samples are not in these types, you can convert them into the considered types by some preprocessing.
- trajGANs may be used for both individual and aggregated trajectories. When we input the mobility trace samples from an individual (or group of people), trajGANs will produce the synthetic mobility samples with similar statistical features to those of real samples from an individual (or group).
- LSTM and *gated recurrent unit* (GRU) [58] would be the ideal structure for the encoder and decoder to encode the real trajectory to X or decode $G(Z)$ to the synthetic trajectory. RNN based embedding also can be adopted to improve learning performance.
- CGAN [38] may be required to generate synthetic mobility traces that are conditioned on certain locations such as residential and commercial areas.

TABLE I
SUMMARY OF RESEARCHES ON GAN BASED MOBILITY SYNTHESIS

Research	Type of GAN	Purpose and Organization
Social GAN [28]	SGAN [33], RGAN [39]	• <i>Meter-scale</i> microscopic mobility trace generator • Generates realistic and <i>socially acceptable</i> walking trajectories • Predicts <i>a-few-second-future</i> trajectory for given input trajectory • Organizes the internal of the generator as a combination of MLPs, LSTMs and pooling module • Organizes the internal of the discriminator as a combination of MLPs and LSTMs • Adopts the pooling module to capture the human walking behaviors caused by <i>social norms</i> • Organizes the pooling module by connecting 2-layer MLP and max pooling
Non-Parametric Trajectory Generator [29]	WGAN-GP [35], DC-GAN [36]	• <i>Non-parametric</i> trajectory generator • generates synthetic mobility traces which mimic not only geographically but also <i>semantically realistic</i> visiting behaviors of humans • Converts raw mobility trajectories to <i>location-major representation</i> that is appropriate for identifying the semantic behaviors such as the visit frequency and stay duration • Uses <i>Wasserstein</i> loss function with <i>gradient penalty</i> and CNN in the generator and discriminator networks
Location Density Matrix Generator [30]	WGAN [34]	• Generates a synthetic <i>location density matrix</i> instead of raw mobility trajectories due to privacy concerns • Exploits noise incurred from the latent parameters to defend against trajectory inference attacks • Adopts <i>Wasserstein</i> loss function • Organizes the discriminator by connecting a 4-layer MLP with three ReLU layers and a single sigmoid layer • Organizes the generators by connecting a 4-layer MLP with three ReLU layers and a single identity layer • Reverses the first three ReLU layers in the discriminator as the first three ReLU layers of the generator
trajGANs [31]	CGAN [38]	• GAN based mobility trace synthesizer for road segment and place visit trajectories • <i>Provides only conceptual design</i>, neither of implementation or experiment • Suggests using the LSTM and GRU for encoder/decoder structures • Suggests adopting the RNN embedding to improve learning performance • Suggest using the CGAN to generate synthetic mobility trace samples conditioned on certain locations such as residential and commercial areas
Empirical Evaluation [32]	RGAN [39], SeqGAN [33]	• <i>Empirical research</i> that benchmarks the performances of seven generative machine learning models for user mobility synthesis • Compares 4 types of RNNs (Char-RNN, LSTM, RHN, PSMM), 2 types of GANs (RGAN, SeqGAN) and copula based approach in terms of the <i>geographic/semantic similarity</i>, <i>statistical similarity</i>, <i>long-range dependency</i>, <i>privacy</i> and <i>computational efficiency</i> • Suggests the copula as the most preferred approach for the overall performance • Suggests the RGAN as the second most preferred approach

E. Empirical Evaluation of Generative Model-based Mobility Synthesis

Kulkarni et al [32] conducted an empirical study that benchmarked the performances of generative machine learning models for user mobility synthesis. They evaluated following seven generative models:

- Char-RNN (Character RNN) [59]
- LSTM
- RHN (Recurrent Highway Network) [60]
- PSMM (Pointer Sentinel Mixture Model) [61]
- RGAN
- SeqGAN⁶
- Copula [62]

The Char-RNN, LSTM, RHN, and PSMM belong to the RNN. And, the SeqGAN and RGAN are included in the GAN. The *copula* is referred to as a multivariate cumulative distribution function that describes the dependence between the random variables of a joint distribution function. For given data samples from a unknown probability distribution, if we know its copula, we can generate synthetic samples through an

⁶We use the term “SeqGAN” instead of “SGAN” that is the term originally used in [40] because we already use the term “SGAN” to refer the semi-supervised GAN.

inverse transform method [63] without knowing the original probability distribution. Since it is likely to be easy to estimate copulas instead of target probability distributions, the copula may be a competitive approach for synthetic data generation for complex datasets. The authors adopted kernel-based non-parametric copulas [62].

The experiments were performed using the Nokia mobile dataset [64] that includes a set of trajectories, such that, $T_u = \langle (l_1, t_1), (l_2, t_2) \dots (l_n, t_n) \rangle$, where $l_i = (lat_i, lon_i)$, $0 \leq i \leq n$, the latitude-longitude coordinate pair, and t , the timestamp such that $t_{i+1} > t_i$. The raw trajectories are transformed onto a uniform grid for dimensionality reduction using the so called *geo-hash* technique, thus, translating T_u into a hashed trace $S(t) = \langle (c_1, t_1), (c_2, t_2) \dots (c_n, t_n) \rangle$, where c_i is the geo-hash of the transformed grid block.

On the translated trajectories, each of the seven models performed its training for about 70 million trace samples and then, generate synthetic samples. Their performances are evaluated on the following metrics:

- Distribution of the top-50 visited locations
- Mean maximum discrepancy (MMD) [65]
- Mutual information decay [66]
- Attack accuracy

- Training and synthesize time

Each of the above metrics is used to assess the *geographic/semantic similarity*, *statistical similarity*, *long-range dependency*, *privacy*, and *computational efficiency*, respectively. The long-range dependency means that, in human trajectories, the current location is affected by the locations, not only in the near past, but also in the far past. The attack accuracy quantifies the robustness against two attacks on the location privacy: *location-sequence attack* [49] and *membership interference attack* [67]. We summarize the experimental results of [32] as follows:

- Char-RNN, RGAN and copulas have the closest fit to the true distribution for the top-50 visited locations.
- Copula achieves the most similar results to those of the real samples in terms of MMD. SeqGAN and RGAN produce the second and third most similar results, respectively. Other schemes have much more degraded similarities.
- Copula shows the most similar trends to the true samples in terms of mutual information decay. RGAN achieves the second most similar results. The others show considerable difference from the ground truth.
- PSMM and copula are the most robust against the location-sequence attack and membership inference attack, respectively. Char-RNN and SeqGAN are more vulnerable to security attacks than others.
- Copula requires 6.5s and 0.76s for the training and synthesis, respectively. Other approaches require from 9000s to 12700s for training, and from 10s to 15s for synthesis.

The above empirical evaluation suggests the copula as the most preferred approach for overall performance. RGAN looks like the second most preferred one.

F. A Summary of Researches

Table I show a brief summary of researches described in this section. For each research, we present what is its purpose and how it is organized. We also give what types of GANs are adopted.

IV. CONCLUSION

In this paper, we reviewed and summarized recent researches on user mobility synthesis that employ GAN. These studies show that GAN based mobility synthesis brings to us (a) the intelligence to capture not only the geographic but also the semantic characteristics of user mobility, (b) a better geographic/semantic reality and security and (c) convenience for non-parametric synthesis. Even with the above promising aspects and some challenging achievements we have seen in Section III-A, III-B, III-C and III-D, it looks like much more research is still required to mature the technical stability and practicality as shown in Section III-E. In other words, this research domain seems to be still in the early phase. We expect to see much more number of creative and innovative researches in this area.

ACKNOWLEDGMENT

This work was supported by ICT R&D program of MSICT/IITP. [2017-0-00045. Hyper-Connected Intelligent Infra Structure Technology Development]

REFERENCES

- [1] R. Jiang, X. Song, Z. Fan, T. Xia, Q. Chen, Q. Chen, and R. Shibasaki, "Deep ROI-based modeling for urban human mobility prediction," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 2, no. 1, pp. 14:1–14:29, Mar. 2018.
- [2] J. Wang, X. Kong, F. Xia, and L. Sun, "Urban human mobility: Data-driven modeling and prediction," *SIGKDD Explor. Newsl.*, vol. 21, no. 1, pp. 1–19, May 2019.
- [3] V. A. Davies, "Evaluating Mobility Models Within An Ad Hoc Network," Master's thesis, Colorado School of Mines, 1500 Illinois St, Golden, CO 80401, 2000.
- [4] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Commun. Mobile Comput.*, vol. 2, no. 5, pp. 483–502, 2002.
- [5] S. Mousavi, H. Rabiee, M. Moshref, and A. Dabirmoghaddam, "MobiSim: A Framework for Simulation of Mobility Models in Mobile Ad-Hoc Networks," in *Third IEEE Int'l Conf. on Wireless and Mobile Comput. Netw. and Commun.*, Oct. 2007, pp. 82–82.
- [6] J. Harri, F. Filali, and C. Bonnet, "Mobility models for vehicular ad hoc networks: a survey and taxonomy," *IEEE Commun. Surveys Tuts.*, vol. 11, no. 4, pp. 19–41, Fourth 2009.
- [7] M. Baratchi, N. Meratnia, P. J. M. Havinga, A. K. Skidmore, and B. A. K. G. Toxopeus, "A hierarchical hidden semi-markov model for modeling mobility data," in *Proc. ACM UbiComp'14*, Sept. 2014, pp. 401–412.
- [8] G. Gidofalvi and T. B. Pedersen, "ST-ACTS: A spatio-temporal activity simulator," in *Proc. ACM GIS'06*, 2006, pp. 155–162.
- [9] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo - simulation of urban mobility: An overview," in *Proc. The Third Int'l Conf. Adv. Syst. Simul.*, 2011, pp. 63–68.
- [10] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO - simulation of urban mobility," *Int'l J. Adv. Syst. Measurements*, vol. 5, no. 3 & 4, pp. 128–138, 2012.
- [11] F. J. Ros, J. A. Martinez, and P. M. Ruiz, "A survey on modeling and simulation of vehicular networks: communications, mobility, and tools," *Comput. Commun.*, vol. 43, pp. 1–15, 2014.
- [12] N. Pelekis, S. Sideridis, P. Tampakis, and Y. Theodoridis, "Hermoupolis: A semantic trajectory generator in the data science era," *SIGSPATIAL Special*, vol. 7, no. 1, pp. 19–26, May 2015.
- [13] A. Hess, K. A. Hummel, W. N. Gansterer, and G. Haring, "Data-driven human mobility modeling: a survey and engineering guidance for mobile networking," *ACM Comput. Surv.*, vol. 48, no. 3, pp. 38:1–38:39, Dec. 2015.
- [14] P. N. Druzhkov and V. D. Kustikova, "A survey of deep learning methods and software tools for image classification and object detection," *Pattern Recog. Image Anal.*, vol. 26, no. 1, pp. 9–15, Jan. 2016.
- [15] A. Ioannidou, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris, "Deep learning advances in computer vision with 3d data: a survey," *ACM Comput. Surv.*, vol. 50, no. 2, pp. 20:1–20:38, Apr. 2017.
- [16] K. Yun, Y. Kwon, S. Oh, J. Moon, and J. Park, "Vision-based garbage dumping action detection for real-world surveillance platform," *ETRI Journal*, vol. 41, no. 4, pp. 494–505, July 2019.
- [17] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M.-L. Shyu, S.-C. Chen, and S. S. Iyengar, "A survey on deep learning: algorithms, techniques, and applications," *ACM Comput. Surv.*, vol. 51, no. 5, pp. 92:1–92:36, Sep. 2018.
- [18] Y. Zhao, Z. Zheng, and Y. Liu, "Survey on computational-intelligence-based UAV path planning," *Knowledge-Based Syst.*, vol. 158, pp. 54–64, 2018.
- [19] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," in *NIPS Deep Learning Workshop*, Dec. 2013.
- [20] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of Go without human knowledge," *Nature*, vol. 550, pp. 354–359, Oct. 2017.

- [21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. NIPS* 27, 2014, pp. 2672–2680.
- [22] J. Briot, G. Hadjeres, and F. Pachet, "Deep learning techniques for music generation - a survey," *CoRR*, vol. abs/1709.01620, 2017.
- [23] X. Wu, K. Xu, and P. Hall, "A survey of image synthesis and editing with generative adversarial networks," *Tsinghua Science and Technology*, vol. 22, no. 6, pp. 660–674, Dec. 2017.
- [24] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, and F. Wang, "Generative adversarial networks: introduction and outlook," *IEEE/CAA J. Automatica Sinica*, vol. 4, no. 4, pp. 588–598, Oct. 2017.
- [25] Z. Pan, W. Yu, X. Yi, A. Khan, F. Yuan, and Y. Zheng, "Recent progress on generative adversarial networks (GANs): A survey," *IEEE Access*, vol. 7, pp. 36 322–36 333, 2019.
- [26] Z. Wang, Q. She, and T. E. Ward, "Generative adversarial networks: A survey and taxonomy," *CoRR*, vol. abs/1906.01529, 2019.
- [27] D.-W. Kim, J.-R. Chung, J. Kim, D. Y. Lee, S. Y. Jeong, and S.-W. Jung, "Constrained adversarial loss for generative adversarial network-based faithful image restoration," *ETRI Journal*, vol. 41, no. 4, pp. 415–425, May 2019.
- [28] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially acceptable trajectories with generative adversarial networks," in *Proc. IEEE CVPR*, June 2018, pp. 2255–2264.
- [29] K. Ouyang, R. Shokri, D. S. Rosenblum, and W. Yang, "A non-parametric generative model for human trajectories," in *Proc. IJCAI*, July 2018, pp. 3812–3817.
- [30] D. Yin and Q. Yang, "GANs based density distribution privacy-preservation on mobility data," *Security and Communication Networks*, Dec. 2018.
- [31] X. Liu, H. Chen, and C. Andris, "trajGANs: Using generative adversarial networks for geo-privacy protection of trajectory data," in *Location Privacy and Security Workshop*, Aug. 2018.
- [32] V. Kulkarni, N. Tagasovska, T. Vatter, and B. Garbinato, "Generative models for simulating mobility trajectories," *CoRR*, vol. abs/1811.12801, 2018.
- [33] A. Odena, "Semi-supervised learning with generative adversarial networks," *ArXiv*, vol. abs/1606.01583, 2016.
- [34] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. ICML*, vol. 70, 06–11 Aug 2017, pp. 214–223.
- [35] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Proc. NIPS* 30, 2017, pp. 5767–5777.
- [36] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015.
- [37] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
- [38] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *CoRR*, vol. abs/1411.1784, 2014.
- [39] C. Esteban, S. L. Hyland, and G. Rätsch, "Real-valued (medical) time series generation with recurrent conditional GANs," *CoRR*, vol. abs/1706.02633, 2017.
- [40] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence generative adversarial nets with policy gradient," in *Proc. the Thirty-First AAAI Conf. Artificial Intelligence*, 2017, pp. 2852–2858.
- [41] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
- [42] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.
- [43] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proc. IEEE ICASSP*, May 2013, pp. 8609–8613.
- [44] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE CVPR*, June 2016, pp. 961–971.
- [45] S. Pellegrini, A. Ess, and L. Van Gool, "Improving data association by joint modeling of pedestrian trajectories and groupings," in *Computer Vision – ECCV 2010*, 2010, pp. 452–465.
- [46] L. Leal-Taixé, M. Fenzi, A. Kuznetsova, B. Rosenhahn, and S. Savarese, "Learning an image-based motion context for multiple people tracking," in *Proc. IEEE CVPR*, June 2014, pp. 3542–3549.
- [47] N. Kiukkonen, B. J., O. Dousse, D. Gatica-Perez, and J. K. Laurila, "Towards rich mobile phone datasets: Lausanne data collection campaign," in *Proc. Int. Conf. on Pervasive Services*, 7 2010.
- [48] L. Song, D. Kotz, R. Jain, and Xiaoning He, "Evaluating location predictors with extensive Wi-Fi mobility data," in *Proc. IEEE INFOCOM*, vol. 2, March 2004, pp. 1414–1424 vol.2.
- [49] R. Shokri, G. Theodorakopoulos, J. Le Boudec, and J. Hubaux, "Quantifying location privacy," in *Proc. IEEE S&P*, May 2011, pp. 247–262.
- [50] J. Krumm and E. Horvitz, "LOCADIO: inferring motion and location from Wi-Fi signal strengths," in *Proc. The First Annual Int'l Conf. Mobile and Ubiquitous Syst.: Netw. and Services*, Aug 2004, pp. 4–13.
- [51] X. Song, H. Kanasugi, and R. Shibasaki, "Deeptransport: Prediction and simulation of human mobility and transportation mode at a citywide level," in *Proc. IJCAI*, 2016, pp. 2618–2624.
- [52] V. Bindshaedler and R. Shokri, "Synthesizing plausible privacy-preserving location traces," in *IEEE S&P*, May 2016, pp. 546–563.
- [53] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: differential privacy for location-based systems," in *Proc. ACM CCS'13*, 2013, pp. 901–914.
- [54] C. Dwork, "Differential privacy: A survey of results," in *Proc. the 5th Int'l Conf. Theory and Appl. Models of Comput.*, 2008, pp. 1–19.
- [55] T. Chen, M. A. Kaafar, and R. Boreli, "The where and when of finding new friends: Analysis of a location-based social discovery network," in *Proc. the Seventh Int'l AAAI Conf. Weblogs and Social Media*, July 2013, pp. 61–70.
- [56] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser, "CRAW-DAD dataset epfl/mobility (v. 2009-02-24)," Downloaded from <https://crawdad.org/epfl/mobility/20090224>, Feb. 2009.
- [57] F. Xu, Z. Tu, Y. Li, P. Zhang, X. Fu, and D. Jin, "Trajectory recovery from ash: User privacy is not preserved in aggregated mobility data," in *Proc. WWW'17*, 2017, pp. 1241–1250.
- [58] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014.
- [59] A. Karpathy, "char-rnn," <https://github.com/karpathy/char-rnn>, 2016.
- [60] J. G. Zilly, R. K. Srivastava, J. Koutník, and J. Schmidhuber, "Recurrent highway networks," in *Proc. ICML*, vol. 70, International Convention Centre, Sydney, Australia, 06–11 Aug. 2017, pp. 4189–4198.
- [61] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," *CoRR*, vol. abs/1609.07843, 2016.
- [62] G. Geenens, A. Charpentier, and D. Paindaveine, "Probit transformation for nonparametric kernel estimation of the copula density," *Bernoulli*, vol. 23, no. 3, pp. 1848–1873, Aug. 2017.
- [63] S. M. Loss, *Introduction to Probability Models*. Academic Press, March 2003, ch. 11.2.1, pp. 644–645.
- [64] J. K. Laurila, D. Gatica-Perez, I. Aad, B. J., O. Bornet, T.-M.-T. Do, O. Dousse, J. Eberle, and M. Miettinen, "The mobile data challenge: Big data for mobile computing research," 2012. [Online]. Available: <http://infoscience.epfl.ch/record/192489>
- [65] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Machine Learning Res.*, vol. 13, no. 1, pp. 723–773, Mar. 2012.
- [66] A. Mahalunkar and J. D. Kelleher, "Understanding recurrent neural architectures by analyzing and synthesizing long distance dependencies in benchmark sequential datasets," *CoRR*, vol. abs/1810.02966, 2018.
- [67] A. Pyrgelis, C. Troncoso, and E. D. Cristofaro, "Knock knock, who's there? membership inference on aggregate location data," *CoRR*, vol. abs/1708.06145, 2017.



Seungjae Shin received the BS degree in electrical and computer engineering from Chung-Nam National University, Daejeon, Korea, in 2007, the MS and Ph.D. degrees in computer science from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2009, and 2017, respectively. He is currently with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea. His research interests include computer networks, cloud computing, and reinforcement learning.



Hongseok Jeon received the B.S. degree in Industrial Engineering from Sungkyunkwan University, Seoul, Korea, in 2002. He received the M.S. degree in Engineering from Information and Communications University (ICU), Daejeon, Korea, in 2004. He joined Electronics and Telecommunications Research Institute (ETRI) in 2004. He is interested in the Software-Defined Networking (SDN) and Network Functions Virtualization (NFV).



Chunglae Cho received the B.S. and M.S. degrees in computer science from Pusan National University, Korea, in 1994 and 1996, respectively, and the Ph.D. degree in Computer and Information Science and Engineering Department at the University of Florida, Gainesville, in 2011. He is currently with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea. His research interests include computer networks, software-defined networking, cloud computing, edge computing, and machine learning.



Seunghyun Yoon received the BS, MS and PhD degree in Industrial Engineering from Sungkyunkwan University, Seoul, Korea, in 1991, 1993 and 1997, respectively. He is currently with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea. He is interested in the computer networks, cloud computing and optimization in network and computer.



Taeyeon Kim received Ph.D. degree in computer science from ChungBuk National University, Korea, in 2007. He also received B.S. and M.S. degrees from Chung-Ang University, Seoul, in 1990 and 1992, respectively. He joined Electronics and Telecommunications Research Institute (ETRI) in 1992. His research includes network & computing convergence platform, AI based self-driving networks and edge intelligence.