This is the author's accepted version of the article. The final version published by IEEE is C. Sun, K. Xu, M. Fiore, M.K. Marina, Y. Wang, C. Ziemlicki, "AppShot: A Conditional Deep Generative Model for Synthesizing Service-Level Mobile Traffic Snapshots at City Scale," IEEE Transactions on Network and Service Management, doi: 10.1109/TNSM.2022.3199458.

AppShot: A Conditional Deep Generative Model for Synthesizing Service-Level Mobile Traffic Snapshots at City Scale

Chuanhao Sun*, Kai Xu*, Marco Fiore†, Mahesh K. Marina*, Yue Wang‡, and Cezary Ziemlicki§
*The University of Edinburgh †IMDEA Networks Institute ‡Samsung §Orange

***Abstract*—Service-level mobile traffic data enables research studies and innovative applications with a potential to shape future service-oriented communication systems and beyond. However, real-world datasets reporting measurements at the individual service level are hard to access as such data is deemed commercially sensitive by operators. APPSHOT is a model for generating synthetic high-fidelity city-scale snapshots of service level mobile traffic. It can operate in any geographical region and relies solely on easily available spatial context information such as population density, thus allowing the generation of new and open traffic datasets for the research community. The design of APPSHOT is informed by an original characterization of service-level mobile traffic data. APPSHOT is a novel conditional GAN design instantiated by a convolutional neural network generator and two discriminators. The model features several other innovative mechanisms including multi-channel and overlapping patch based generation to address the unique challenges involved in generating mobile service traffic snapshots. Experiments with ground-truth data collected by a major European operator in multiple metropolitan areas show that APPSHOT can produce realistic network loads at the service level for areas where it has no prior traffic knowledge, and that such data can reliably support service-oriented networking studies.**

***Index Terms*—Mobile network traffic data, mobile services, synthetic data generation, deep generative models, generative adversarial networks, mobile traffic analysis, mobile network resource management, network slicing.**

I. INTRODUCTION

Large-scale data about the demands generated by individual mobile services is a key enabler for research and innovation in networking and beyond, leveraging advances in artificial intelligence (AI) and machine learning (ML). Indeed, 5G has determined a shift towards a flexible and shared network architecture that can effectively support a diverse array of services as virtualized instances or network slices [1]. In this paper, the service-level data consists of the traffic produced by 10 top popular mobile application, namely WhatsApp, Facebook, Snapshot, Instagram, YouTube, Google Play, Netflix, Twitter, iTunes, Apple Store, which make up 95% of the total mobile traffic in major European cities [2]. Here, service-level mobile traffic data is paramount to steer the development of slicing mechanisms and to assess their efficiency in terms of resource sharing [3], [4]. Equally, such data is crucial in planning and configuration of edge computing and storage infrastructure to optimize latency for certain services and benefit others by computation offloading to the edge cloud [5], [6]. Furthermore, service-level traffic data is key to other service oriented studies within networking, including the design of traffic classification techniques [7] and energy efficiency optimization [8], and beyond, such as for data plan analysis [9], [10] or to reveal links between apps consumption and urbanization levels or socioeconomic status [11], [12].

Despite the importance and breadth of these potential use cases, service-level mobile traffic data in practice is rarely accessible to researchers, as this data is seen as commercially sensitive by mobile operators. In fact, it is way harder to access service-level data compared to aggregate mobile traffic data (e.g., [13]). The aggregate traffic data here refers to the spatial distribution (map) of total traffic volume across a given region (e.g., city) for a given period (e.g., hour, day) as illustrated in the top map in Fig. 1. Our focus is on such data but at the granularity of individual mobile services (e.g., YouTube, FaceBook) instead of just total traffic volume [2], [11]. In other words, for each location (pixel) on the map, in our case we have a vector of traffic volumes with each element of the vector corresponding to a different mobile service. The few groups who do have access to such data are limited by restrictive NDAs on the data use and distribution. *In this paper, we seek to overcome this data access barrier through a synthetic mobile service traffic data generation model. Specifically, we aim to design a model that takes in publicly available contextual attributes (e.g., population and land use distribution) for a geographical region (e.g., a city) as input, and generates the snapshot of traffic for different services in that region for a given time period.* By relying solely on publicly available context data, such a model can enable researchers to generate service-level mobile traffic data for regions of interest to them and use it to drive studies that require such data.

Designing a data synthesis model that can generate high-fidelity service-level mobile traffic snapshots and generalizes well to new regions is challenging due to a number of reasons. First, the publicly available context data for a target region may not fully determine the mobile services traffic for that region and in general cannot capture the stochasticity inherent to mobile traffic. Second, mobile traffic is known to have complex spatiotemporal correlations both overall and at service level [14], [15], which need to be captured by the model. Third, locations and times with high traffic intensity (which we refer to as *hotspots* in this paper) are particularly important for downstream use cases on research management and beyond (e.g., [9]), and need to be faithfully modeled. Fourth, the

model should correctly capture correlations between traffic for different services and their relative contribution to overall traffic. Finally, the model should be flexible in accommodating the fact that the target regions for traffic generation may differ widely in their geographical dimensions as well as contextual attributes and traffic characteristics.

As a key contribution of this paper, we propose APPSHOT*, a conditional deep generative model that addresses the aforementioned challenges to synthesize city-scale*[1] *and service-level mobile traffic snapshots.* APPSHOT builds upon and substantially extends CartaGenie, a previous model we proposed for generating snapshots of aggregate city-scale mobile traffic [16]. Similarly to CartaGenie, APPSHOT leverages generative adversarial networks (GANs) and convolutional neural networks (CNNs) towards high-fidelity traffic data generation capturing spatial correlations for the given time. More specifically, it is a conditional model that takes contextual attributes for a target city as multi-channel image input to control the output traffic data from the model to match with that city while capturing stochasticity in traffic data.

APPSHOT significantly extends CartaGenie in multiple ways (§V) for generation of *service-level* mobile traffic snapshots. First, it faithfully models correlations among services and their relative proportions (with respect to overall traffic) via joint generation of traffic for all services as different channels of an image and using adversarial training with two discriminators. Moreover, it operates at a smaller 'patch' level to allow generating service snapshots for regions of different dimensions. Finally, APPSHOT employs overlapping patch based learning with slide-by-1-pixel method to ensure correct modeling of hotspots, better generalization to new cities and absence of artefacts when sewing up patches into a city-scale traffic map. Overall, these enhancements allow producing dependable service-level traffic snapshots that cannot be synthesized via the original CartaGenie.

To train and test APPSHOT, we use an operator-provided mobile traffic dataset, spanning the ten most significant mobile services for ten cities in a major European country, and we augment it with corresponding context data from public sources (§III). Our evaluation results (§VII) considering five different fidelity metrics show that APPSHOT generalizes to unseen cities and yields service-level traffic snapshots for different peak periods that are significantly superior to a range of baseline approaches, including CartaGenie. We additionally evaluate APPSHOT through a representative use case (§VIII) on resource sharing efficiency with network slicing [3] and show that it provides similar results to those obtained with real (ground-truth) traffic data.

In summary, our key contributions are as follows:

(i) We propose a novel conditional deep generative model, APPSHOT, which to the best of our knowledge is the first method for synthesizing dependable city-scale and service-level mobile traffic data in the literature (discussed in §II). Upon publication of this work, we will make a synthetic service-level mobile traffic dataset generated using APPSHOT available to the research community at https://github.com/netsys-edinburgh/AppShot/.

(ii) We present a novel analysis of service-level mobile traffic data across multiple cities and derive insights that inform our design of APPSHOT (§IV).

(iii) Using an operator provided multi-city and multi-service mobile traffic measurement dataset, we show that APPSHOT synthesizes realistic service-level traffic snapshots solely using contextual input, outperforms all baseline approaches, and effectively generalizes to cities not seen during training. We also demonstrate the utility of APPSHOT for downstream applications through a use case.

The remainder of this paper is structured as follows. The next section discusses related work from the networking and computer vision domains. In §III, we elaborate on the mobile traffic and context data relevant to our work. Then in §IV, we conduct an analysis of the aforementioned data, including service-level traffic characteristics and correlation between traffic and context. The proposed generative model APPSHOT is described in detail in §V. §VI introduces the methods we use to evaluate the proposed APPSHOT, including the metrics and baselines. Evaluation results are presented and discussed in §VII, followed by the use of APPSHOT for a downstream service-level traffic dependent application in §VIII. Finally, §IX concludes the paper.

II. RELATED WORK

Traditional network traffic generation focuses on creating different packet-level workloads. There are a number of tools that exist for this purpose (*e.g.*, iPerf, MGEN, Ostinato) and are also embedded in popular network simulators (*e.g.*, ns-3). Some of these tools like D-ITG [17], [18] support modeling different applications through parameterized probability distributions for packet sizes, their inter-arrival times, etc. This form of traffic generation does not have a spatial dimension. In contrast, our focus is on generating snapshots of application/service level mobile traffic volumes (aggregated across multiple users and flows) at different locations of a target region (*e.g.*, a city).

We are unaware of any prior work for generation of service-level mobile network traffic data. The few related works that exist in the mobile networking context [16], [19]–[21] focus on overall traffic across *all* services. Di Francesco *et al*. [19] propose an approach for assembling a cellular dataset for a given region by integrating multiple sources of data, including census data for population distribution, base station locations and estimation of data demand per subscriber. For the data demand, they simply model this as a probability distribution based on operator provided data on overall mobile traffic across all services and then sample from it. We consider this approach as a baseline in our evaluations and highlight its limitations in handling traffic correlations. In another mobile traffic related work, Bo *et al*. [20] target generation of mobile traffic patterns for a region focusing on hotspots through geotagged Twitter data for that region. Here again, only total traffic volume across all services is considered and not at the individual service level like we do. Moreover, access to

[1]For convenience, we use the terms 'city' and 'region' interchangeably, but our model can be applied to any sized region bigger/smaller than a city.

Twitter data is no more easier than accessing mobile traffic data whereas we base our generation on context data for the target region that can be easily obtained from public sources.

SpectraGAN [21] and CartaGenie [16] are recent proposals that can be viewed as the state of the art on mobile traffic data generation. As in our work, SpectraGAN and CartaGenie take a conditional deep generative modeling approach but focus on generation of spatial or spatiotemporal data for *total* traffic volumes in a city. We target a different and orthogonal dimension, *i.e.*, on the *individual service level* contributions that make up the total traffic. As we show in our comparative evaluation, applying SpectraGAN or CartaGenie for our purpose yields poor quality generation due to its inability to model inter-service correlations and their relation to total traffic. DoppelGANger [22] is another broadly related work that employs deep generative modeling in the networking context but solely focuses on network time-series data generation with no spatial dimension.

As we represent city-scale mobile traffic snapshots as images, their generation at service level can be viewed as a multi-channel image generation problem. Furthermore, since we aim at conditional generation using contextual attributes as a multi-channel image input, image translation works from the computer vision domain are particularly relevant. Pix2Pix [23] is a representative prior work on conditional image translation. When applied to our problem setting, this work has several key limitations as we show in our evaluation: (i) it does not take particular care to capture correlations among channels (services in our problem); (ii) it fails to model variation in the data from using just dropout for stochasticity; (iii) it can also result in undesirable edge effects and artefacts when generating traffic maps for arbitrary sized regions. These limitations also apply to other related works from the computer vision literature (e.g., Style-GAN [24], Cycle-GAN [25]), which are essentially rooted in the fact that they do not cater to the unique requirements of mobile service traffic map generation. The works from the transportation domain, exemplified by Traffic-GAN [26], for road vehicle traffic generation are also broadly related. However, these works do not differentiate between different vehicle types (individual mobile services in our case) and also make a strong assumption of knowing correlations among traffic on different roads for the target region, which is unrealistic.

Besides generation of multi-service mobile traffic maps, our work also includes an analysis of mobile network traffic across different services and cities. This part is novel compared to prior service-oriented mobile traffic analysis works (e.g., [2], [11]) by focusing on the key characteristics that need to be kept in mind when generating service-level mobile traffic data. In particular, unlike [11], we analyze the correlation between traffic of different mobile services as well as with a wide range of contextual attributes beyond urbanization. Compared to [2], we study the similarities and differences in mobile traffic across cities, with a focus on peak periods, traffic stochasticity, hotspot density and distribution.

III. Mobile Traffic and Context Data

For the purpose of modeling, analysis and evaluation in this work, we make use of a real-world mobile traffic dataset collected in the production network of a major mobile network operator in Europe. We also gather data for a variety of contextual attributes for the target regions from public sources.

Mobile Traffic Data. Our traffic dataset spans 10 major cities in a European country (referred henceforth as City **1**-City **10**), where it covers the mobile demands of the whole subscriber base of the operator, amounting to around 30% of the local user population. This data was obtained by monitoring individual IP data flow sessions in the operator's network over the General Packet Radio Service (GPRS) Tunneling Protocol User plane (GTP-U). To infer the services corresponding to the traffic flows, the operator employs a combination of proprietary and commercial traffic classification tools on top of Deep Packet Inspection (DPI) probes, which allows identifying a very wide range of mobile services with a high degree of accuracy [11]. Note that the data was aggregated geographically (per antenna sector) and temporally by the operator, so as to make the data non-personal and to preserve user privacy; all operations were carried out within the operator premises, under control of the local Data Privacy Officer (DPO), and in compliance with applicable regulations, according to GDPR (General Data Protection Regulation) regulations [27]. The data was aggregated over all users in space and time in secure servers at the operators' premises, and we only accessed de-personalized aggregates.

Each city is represented in the data as a regular grid tesselation over space with each grid cell (*i.e.* pixel) covering $250 \times 250\ m^2$. Unsurprisingly, different cities have different geographical sizes in terms of number of pixels in each dimension, and range from 33×33 to 97×123 pixels. Traffic data per pixel consists of overall mobile traffic volume *for each service* across uplink and downlink directions in bits/s, over time. The dataset covers a continuous period of 6 weeks. In this dataset, we consider the top 10 popular services that contribute to more than 80% of the total traffic volume, namely: YouTube (YT), Instagram (INS), SnapChat (SC), WhatsApp (WA), Netflix (NF), Apple Store (AS), iTunes, Facebook (FB), Twitter (TW), and Google Play (GP). As such, the effective total mobile traffic in our study is the sum of traffic due to these top-10 services.

Context Data. Our conditional generation model takes advantage of contextual attributes to produce credible synthetic traffic. We gather a wide range of context data from easily accessible public sources, so that the method is applicable as widely as possible. All attributes for each city are mapped to the corresponding regular grid tessellation used to represent mobile traffic data, examples under each attribute are illustrated in Figure 1. In all, we consider 27 different contextual attributes, as outlined below.

Population. The number of inhabitants residing in each grid cell, as reported in the relevant national census.

Land Use. The different uses of the land within each grid cell, obtained from the Copernicus Urban Atlas repository [28]. We only retain land use types that exhibit non-

negligible correlation with mobile traffic (as per Spearman's correlation coefficient (SCC) [29]). Ultimately, 12 land use attributes are considered, listed in Table I.

Points of Interest (PoIs). The number of landmarks of a specific class within each grid cell, extracted from the OpenStreetMap (OSM) repository [30]. We use a similar correlation analysis with traffic as above, and retain the 14 significant PoI categories (Table I).

Contextual Attribute	Avg. SCC
Population	0.639
Continuous Urban	0.220
High Dense Urban	0.180
Medium Dense Urban	0.128
Low Dense Urban	0.254
Very-Low Dense Urban	0.102
Isolated Structures	0.051
Green Urban	0.325
Industrial/Commercial	0.252
Air/Sea Ports	0.321
Leisure Facilities	0.322
Barren Lands	0.067
Sea	0.072
Tourism	0.135
Cafe	0.002
Parking	0.2110
Restaurant	0.1797
Post/Police	0.118
Traffic Signal	0.430
Office	0.343
Public Transport	0.080
Shop	-0.018
Primary Roads	-0.074
Secondary Roads	-0.009
Motorways	0.254
Railway Stations	0.371
Tram Stops	0.158

TABLE I: List of contextual attributes considered.

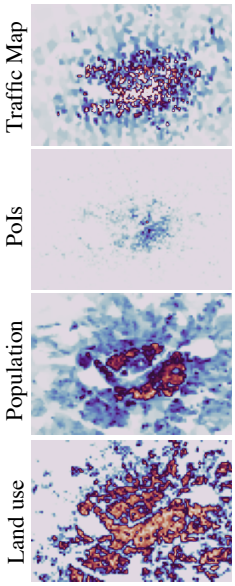
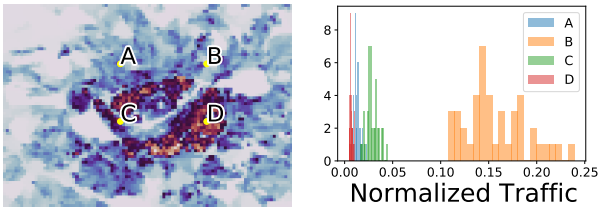


Fig. 1: Spatial distribution of total traffic in CITY 1 and 3 selected context attributes.



(a) Four locations in CITY 1 (b) Corresponding traffic

Fig. 2: Illustration of inherent variation in traffic at each location over time, considering CITY 1 as an example.

IV. Analysis of Mobile Traffic Characteristics across Services and Cities

In order to better inform the design of our generator, we first investigate the properties of mobile network traffic at the service level, across a number of different dimensions.

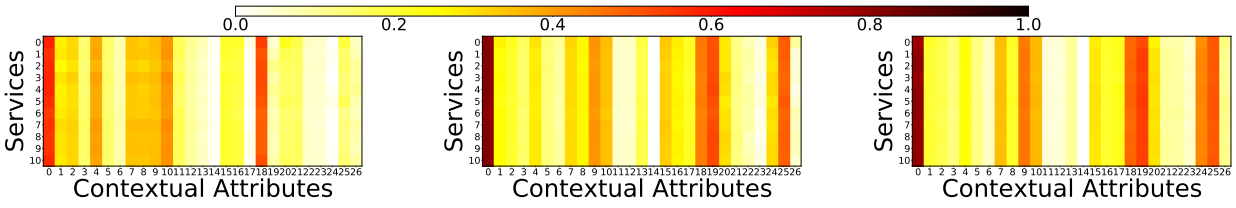
Relationship between Context and Traffic. We start by investigating how the traffic across the 10 target services relates to contextual information. A first important observation concerns the inherent stochasticity of mobile traffic: Figure 2b shows the distribution of total traffic observed over time at four different pixels in CITY 1, whose locations are shown in Figure 2a, and the traffic is normalized by the maximum pixel scale traffic (maximum value of traffic map over all dates) as displayed in the X-axis of Figure 2b. Note that mobile traffic can exhibit substantial variation at a location even though the corresponding context remains the same: this is, *e.g.*, the case of the population density illustrated in Figure 2a. In addition, the correlation between mobile traffic and contextual attributes for any given region is non-trivial. This is as exemplified in Figure 1, where three sample contextual attributes do not show any obvious visual correlation with the mobile traffic.

Takeaway message. The generation process must capture the stochastic nature of mobile traffic, by correctly modeling the relationship between static context information and spatial traffic demand at different time periods. Also, the lack of simple correlations between individual contexts and traffic indicates that a naive univariate statistical model based on any one attribute is not an effective generator, thus motivate the more complex multivariate designs we consider.

Correlations with Service Level Traffic. The above analysis considers aggregate traffic. As we are interested in service-level generation, we now examine the dependence of the demand for individual services on the various contextual attributes. Figure 3 shows correlation between the traffic snapshots of different services and the contextual attributes in three cities. We observe that, for a given city, the correlation between different services and any single attribute is close – each column generally has a similar color, but the service-context correlation varies across attributes (columns). This hints that the spatial distribution of traffic is consistent across services in a same city. A more detailed analysis of cross-service traffic similarity further corroborates this observation: in Figure 4, we use structural similarity (SSIM) [31] measure to compute the spatial similarity between the spatial demand of pairs of services, for different daily peak hours in the morning, midday and evening. Note that SSIM is a classical image fidelity metric, which allows comparing individual pixels between a pair of images (here traffic maps of a pair of services) while also accounting for the differences in the whole spatial construct across the compared images. As shown in the plots, the spatial variations between different services stay relatively consistent at all times. Yet, not all service-level demand pairs display the same level of similarity, as SSIM between different service pairs ranges from 0.55 to 0.95 for any given time period.

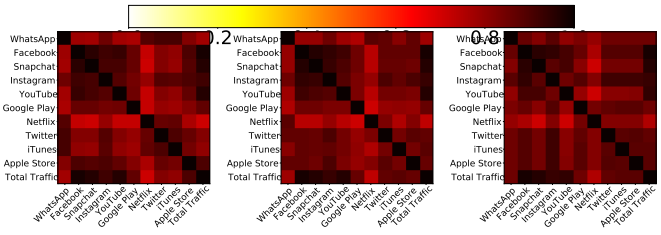
Takeaway message. The diverse correlations among services indicate that naive transformations (*e.g.*, scaling) are insufficient to generate traffic snapshots for one service from the snapshots of a different service. However, more complex transformations may still take advantage of the significant but varying degree of similarity among the traffic of individual services. This suggests a model design that natively performs a joint synthesis of all per-service snapshots.

Traffic Characteristics in Different Cities. Figure 3 also suggests that the relationship between service-level traffic and contextual attributes is different across different cities. The heterogeneity among cities also appears in terms of average daily traffic volume, depicted separately for weekdays and weekends in Figure 5. Population, city size, and user preference, all contribute to such heterogeneity. For instance, CITY 1 has significantly higher traffic volume, about six to



(a) CITY **1** (b) CITY **2** (c) CITY **3**

Fig. 3: Correlation between the traffic of mobile services and contextual attributes in three different cities.



(a) *Morning Peak* (b) *Midday Peak* (c) *Evening Peak*

Fig. 4: Pairwise similarity between traffic snapshots of different mobile services (as per SSIM) in CITY **3**.

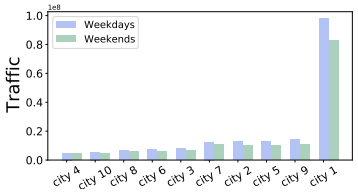


Fig. 5: Average daily traffic.

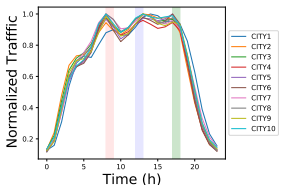


Fig. 6: Total traffic.

twenty times that of other cities. The traffic generation model must be able to capture such traffic heterogeneity across different regions. We also notice that traffic demand during weekdays is around 20% higher than weekends for all cities. Our evaluations therefore highlight weekday traffic generation but relative performance results across different methods are similar for weekends.

In contrast, Figure 6 shows that such differences do not emerge at the level of aggregate normalized daily traffic, which is very consistent across all cities. Specifically, we identify the same three peak hours for all cities: in the morning (8-9am), around midday (12-1pm), and early in the evening (5-6pm).

Takeaway message. Generalizing the traffic generation task across cities is a significant challenge, as context-traffic correlations are highly diverse between cities. So the model must be designed so as to facilitate such generalization, which shall also be a key element of the performance assessment. Also, in our evaluation we will focus on the three peak hours identified above, as they are consistent across cities and especially important for, *e.g.*, network planning or network resource management purposes.

Per Service Traffic Proportions. Mobile services generate highly diverse amount of traffic, even for the top-10 services in our analysis, as shown in Figure 7. This figure also illustrate how service demand contributions to total traffic vary in time.

Takeaway message. The generation shall capture the diverse and time-varying demands of individual services and how they

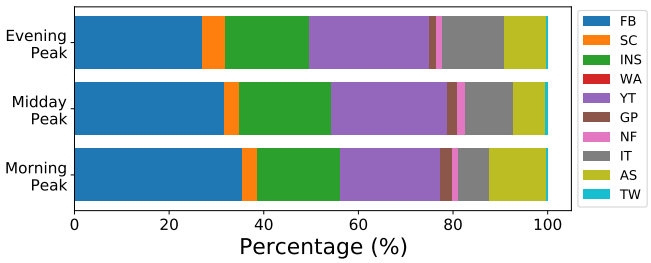
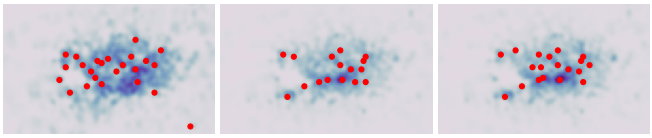


Fig. 7: Per-service traffic proportions at different peak hours across all cities.WA: WhatsApp, FB: Facebook, SC: Snapchat, INS: Instagram, YT: YouTube, GP: Google Play, NF: Netflix, TW: Twitter, AS: Apple Store, IT: iTunes.



(a) Morning Peak (b) Midday Peak (c) Evening Peak

Fig. 8: Hotspots (dots) in CITY **1** at different times.

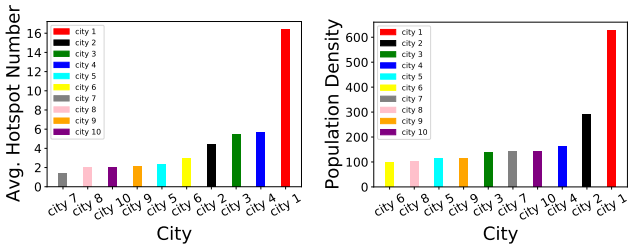
compose, again heterogeneously over time, into total traffic.

Traffic Hotspots. Locations with high traffic activity, which we refer to as 'traffic hotspots' are especially important for many downstream applications based on mobile traffic data. We identify traffic hotspots in a city for a given time period following the approach taken previously in [32]. A traffic hotspot has to meet two requirements: (1) the traffic volume of a hotspot pixel should be higher than a threshold (**High Traffic Volume**); and (2) should not be lower than neighboring pixels (**Non-Negative Gradient**). We find that the position and number of hotspots so detected vary over time even within a city, as shown in Figure 8. Also, the hotspot density varies widely across cities, as shown in Figure 9; interestingly, the relationship between the hotspot and population density is not obvious, as shown by the rankings of cities based on these two metrics.

Takeaway message. A dependable traffic generation model shall correctly synthesize city-specific and time-varying service-level traffic hotspots. This is challenging, as correlations with context (*e.g.*, population) are complex.

V. APPSHOT

Based on the insights from §IV, the generation of high-quality multi-service traffic snapshots faces the following major challenges: 1) synthesizing high-fidelity traffic snapshot from context input with significant statistical variation;



(a) Number of Hotspots (b) Population density

Fig. 9: Cities ranked by hotspot and population density.

2) preserving correlations among multiple services, both in terms of structural similarity and percentage contribution to total traffic; 3) allowing traffic generation for target cities of arbitrary spatial sizes; and 4) accommodating diverse traffic characteristics and context data ranges across cities.

With AppShot, we tackle challenges 1) and 2) by designing a tailored conditional deep generative model (§V-C), and solve challenges 3) and 4) via customized data processing and training methods (§V-B) and hyper-parameter tuning.

A. Problem Statement

Let $\chi = \{X^1, X^2, \cdots, X^N\}$ be a real-world mobile network traffic dataset that contains sets of observations of mobile traffic, such that each set is collected in a different geographical region, *i.e.*, city. The data for each city $n \in \{1, \cdots, N\}$, includes observations over a given span of time T^n, hence $X^n = \{x_1^n, \cdots, x_{T^n}^n\}$. The observation at each time slot is composed of traffic due to S different services, *i.e.*, $x_t^n = \{x_{t,1}^n, \cdots, x_{t,S}^n\}$. For time slot $t \in 1, \cdots, T^n$ and service $s \in 1, \cdots, S$, we represent the mobile traffic observation $x_{t,s}^n \in \Re^{H^n \times W^n}$ as a single channel image, whose pixels map to the spatial units over which network traffic is recorded. Then, each pixel value corresponds to the network traffic value recorded for service s at a specific geographic location; and, $H^n \times W^n$ are the height and width of the city n's dimensions in pixels, respectively; the dimensions may differ between cities.

In addition, each observation $x_{t,s}^n$ is associated with a set of K conditions, *i.e.*, publicly available contextual attributes that may explain the volume of traffic generated by mobile users (*e.g.*, as Figure 1 illustrates, population distribution in the region, land use characteristics, presence of points of interest, *etc.*). We denote the set of conditions for each city as its *context*, and represent it as the set $C^n = \{c_1^n, \cdots, c_K^n\}$. For each condition $k \in \{1, \cdots, K\}$ in city n, we have $c_k^n \in \Re^{H^n \times W^n}$, and thus $C^n \in \Re^{K \times H^n \times W^n}$, which is a multi-channel image with one channel per attribute. Note that the static, spatial contextual attributes alone are typically insufficient to fully explain the corresponding network traffic, as illustrated earlier in Figure 2b.

Our goal is to synthesize network traffic data $f_{t,s}^m \in \Re^{H^m \times W^m}$ for an unseen region m at a particular time t and given context C^m in a way that the synthetic $f_{t,s}^m$ samples exhibit similar data characteristics to the real training data χ and are compatible with the provided C.

B. Patch based Learning Methods

In order to optimize the learning process, the mobile traffic data and contextual attributes need to be carefully formatted, as presented next.

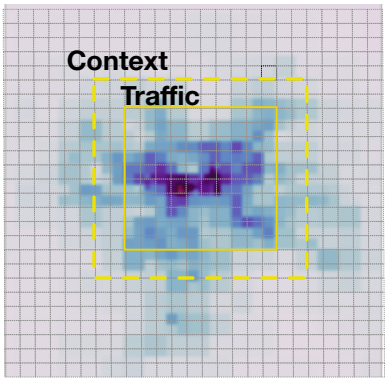
1) Patching and Formatting: To create the training samples, we divide the $H^n \times W^n$ traffic map $x_{t,s}^n$ of each city n and service s in time slot t into smaller patches $x_{t,s}^{n,l}, l \in \{1, \cdots, L\}$, where L is the total number of patches. This formatting has two advantages. Firstly, cities vary in their geographical span and so their traffic maps have varied dimensions, hindering the design of a single model that can handle different sized cities: here, employing fixed smaller sized traffic patches allows using the same generator model architecture regardless of the city dimensions considered for training or generation. Secondly, it allows using diverse traffic patches from different snapshots together to enable a more efficient training via stochastic gradient optimization. Moreover, different local sub-regions of a same city can have similar relationship between the context and traffic. So training at the patch level can be seen as a form of *weight-sharing* – a type of *regularization* technique – to let the model learn the actual casual relationship between context and traffic instead of memorizing the mapping.

The output synthetic traffic generated at the patch level (denoted as $f_{t,s}^{m,l}$) can be of a high quality for each individual patch but may leave artefacts at the boundary of patches when sewing the patch level outputs to a city-level traffic map. To overcome this issue, we associate with each traffic patch $x_{t,s}^{n,l}$ a *trimmed* context patch $c_k^{n,l}$ (identical across all attributes) that includes a margin around the traffic patch (see Figure 10a). This is considering that only a portion of the city-wide context that is in the geographical vicinity of the traffic patch stays relevant to the learning process. Crucially, the additional margin ensures that the border pixels of a traffic patch $x_{t,s}^{n,l}$ have sufficient context during the learning process. Clearly, the number of context patches is the same as that of traffic patches, and we denote by $c_k^{n,l}$ the complete context patch corresponding to $x_{t,s}^{n,l}$.

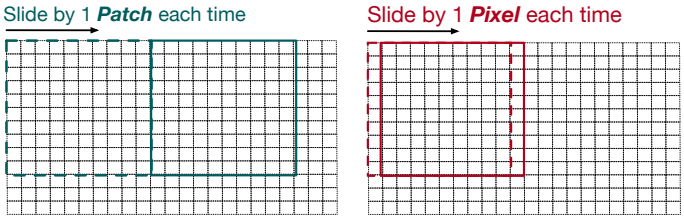
As an additional measure towards artefact-free synthetic traffic maps, we consider overlapping traffic patches as shown in Figure 10c and slide across them by 1 pixel each time during training and generation. Compared to the straightforward 'no overlap' case illustrated in Figure 10b, each pixel in the output traffic map benefits from being part of multiple traffic patches. This not only helps with avoiding edge effects but also serves as a data augmentation method. For example, with a city map of 20×20 and traffic patch size of 10×10 with dimensions in pixels for both, no overlap case yields just 4 patches. Overlapping case, on the other hand, results in 121 patches[2] for the same example. More effective data for training aids in capturing key spatial traffic features at high fidelity but also helps with better generalization across diverse cities.

2) Normalization: Given that different services may have vastly different traffic volumes, the training and generation for services with relatively lower volume can become an issue

[2]We get 121 patches for this example by considering traffic patches with their top-left most pixel falling at each of the pixels in the $[1, 1]$ to $[11, 11]$ square region.



(a) Patch of context and traffic map



(b) No overlap (c) Overlap

Fig. 10: (a) Traffic patch and corresponding context patch; (b) non-overlapping and (c) overlapping traffic patch cases.

if not handled properly, especially in a multi-channel CNN model where the weights involved in the generation of distinct services are broadly shared. To guard against this issue, we employ service-level traffic normalization as a pre-processing step. Specifically, we normalize the traffic values, x_s, of each service s by dividing them with $x_{s,\max}$, *i.e.*, the *global* per-pixel maximum traffic volume value observed for s. This results in traffic values for each service to independently fall between 0 and 1. As part of this normalization step, we also add a small ϵ value to handle cases where no traffic is recorded for a service. The above normalization step can be easily reversed during post-processing on the output synthetic traffic map via a rescaling step.

C. Detailed Model Design

1) Generator: The generator in APPSHOT is responsible for generating traffic map of all services, each represented as a different channel in the multi-channel image output.

The generator, denoted as G_θ with θ representing the weights of the neural network, is a conditional latent variable model instantiated by a CNN based architecture. Formally, we use the latent variable z to model stochasticity and unobserved conditions, then the probability of observing an actual mobile traffic x given conditions c is modeled as $p_\theta(x|c) = \int p_\theta(x|c,z)p(z)dz$, where θ represents the parameters of the conditional probability. Figure 11 shows a schematic of the generator's neural network architecture in APPSHOT. An important remark is that context c is spatial, whereas z is non-spatial. The non-spatial input z in APPSHOT is processed via a specialized FiLM conditioning layer [33], which effectively creates two convolutional entry branches in the initial stages of the generator network. This design avoids the risks of a naive conditioning on the latent variable

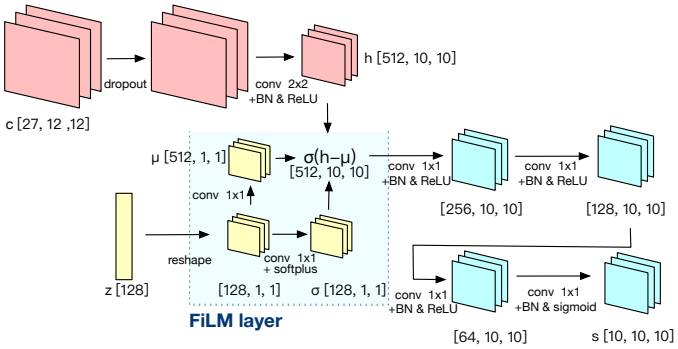


Fig. 11: Schematic of APPSHOT generator architecture, labeled with input and output at each layer.

z (*e.g.*, simply concatenating it with c) that can lead to the network completely ignoring stochasticity. Instead, the FiLM ensures that the latent variable is duly accounted for in the generation process. The result of the separate convolutional branches are then merged via an affine transformation into a hidden representation whose spatial dimension is same as the output traffic map. This representation is then processed by stacked convolution layers with size-1 kernels to produce the final sample s.

In Figure 11, we label the dimensions of input/output at each layer using the format of $[channels, size_x, size_y]$. For example, the $c[27, 12, 12]$ on the input side means the context input is a multi-channel image with 12×12 dimensions and 27 channels; each channel here represents a particular condition (*i.e.*, contextual attribute). To reduce over-fitting to the conditions, we add a channel-wise dropout layer to the input conditions (with a dropout rate of 0.02). The first convolutional layer has a kernel size of $N_c - N_x + 1$. Here N_x refers to the dimension of the output traffic patch size (empirically set to 10) whereas N_c is the context patch dimension (empirically set to 12 to provide the best average output quality N_x = 10). The rest of convolutions are of size 1. As per the number of channels: $N_c \to 8F \to 4F \to 2F \to F \to 1$, where the base number of features F are 64. For the FiLM layer process, the latent variable z is a N_z dimensional noise vector with N_z set to $16F$. All intermediate activations are ReLU [34], following a batch normalization (BN) layer [35]; the final activation is Sigmoid. For σ from the FiLM layer, we use the softplus activation, $F(x) = \log(1 + \exp(x))$, to ensure it is positive.

2) Training: To learn θ, we train the model by optimize the loss function, as elaborated below. The generator is trained in an adversarial manner with two discriminators to reflect correlation among services and their contribution to total traffic. For this purpose, we define a conditional probability distribution $p_\mathcal{D}$ based on real data (ground truth traffic) and corresponding context for cities 1 to N in the training data (*i.e.*, $\{(x^1, C^1), ..., (x^N, C^N)\}$). We then find the model weights θ^* that minimize a divergence criterion between the data distribution $p_\mathcal{D}$ and the model p_θ. Specifically, following standard GAN formulations [24], we train the model by minimizing the Jensen-Shannon (JS) divergence, *i.e.*, $\theta^* = \text{argmin}_\theta \, JS[p_D||p_\theta]$. One of the discriminators called **individual quality discriminator** (D_1) is designed to evaluate

the overall fidelity of the multi-service traffic map. For this discriminator, the adversarial loss is defined as:

$$\mathcal{L}_{JS}^{D_1}(p_{\mathcal{D},p_\theta}) = \mathbf{E}_{p_\mathcal{D}}[\log D_1(x,c)] + \mathbf{E}_{p_\theta}[\log(1 - D_1(\widetilde{x},c))].$$

where the $\widetilde{x}$ is the synthetic traffic map of x.

Unlike conventional works on image generation that treat different channels independently, we need to capture the correlation between different channels. We also need to minimize the divergence between the sum of output channels and real total traffic at the pixel level. By training each channel to target generation of synthetic traffic map for a different service does not ensure the correct sum of all traffic maps from different channels, so providing extra regularization is helpful in our case. To constrain the sum of traffic from different services and encourage the model to learn the correct composition of total traffic, we introduce a second discriminator called **sum quality discriminator** (D_2) with its adversarial loss defined as:

$$\mathcal{L}_{JS}^{D_2}(p_{\mathcal{D},p_\theta}) = \mathbf{E}_{p_\mathcal{D}}[\log D_2(\textstyle\sum_{s=1}^{S} x_s, c)] + \mathbf{E}_{p_\theta}[\log(1 - D_2(\sum_{s=1}^{S} \widetilde{x}_s, c))],$$

where S is the total number of services under consideration, x_s refers to the traffic map of service s within S. $\widetilde{x}_s$ means the synthetic traffic corresponds to ground truth x_s.

Training solely with adversarial training as described above is insufficient, which generally leads to higher training instability and lower fidelity output. So in APPSHOT, besides adversarial training, we make the training process more stable and controllable by adding the L1 loss. Specifically, we use the **L1 norm** of the synthetic multi-channel traffic map (with respect to its real counterpart) as part of the loss function. L1 loss function is shown to be empirically effective in prior work (e.g., [36]).

As the overall loss function of the generator, we take the sum of the above two adversarial losses and the weighted supervised learning loss (L1 norm):

$$\mathcal{L} = \mathcal{L}_{JS}^{D_1}(p_\mathcal{D}, p_\theta) + \mathcal{L}_{JS}^{D_2}(p_\mathcal{D}, p_\theta) + \lambda \mathbf{E}_c\{||\mathbf{E}_{x\sim p_\mathcal{D}}[x], \mathbf{E}_{x\sim p_\theta}[\widetilde{x}]||_1\}.$$

This final loss $\mathcal{L}$ is used to update the discriminators and generator in turn. Here λ is a tuneable parameter to adjust the weight of L1 loss; we set $\lambda = 0.5$ by default in our tests.

VI. EVALUATION METHODOLOGY

We now present the various fidelity metrics and multiple baseline approaches considered for APPSHOT evaluation.

A. Fidelity Metrics

Weighted Error (WErr). This metric quantifies the composition of a synthesized multi-service mobile traffic dataset relative to the corresponding real (ground-truth) data. Suppose in a real dataset made up of traffic from multiple services, the actual percentage of traffic due to a service s among S services with respect to total traffic is r_s and its traffic volume is t_s. If the traffic volume of the same service in the corresponding synthetically generated dataset is $\widetilde{t}_s$, then the Weighted Error (**WErr**) is defined as:

$$\text{WErr} = \sum_{s=1}^{S} r_s \frac{|t_s - \widetilde{t}_s|}{t_s}.$$

In other words, it is the relative estimation error in traffic volume per service weighted by each service's actual percentage, averaged over all services. Smaller WErr means more accurate service composition in the synthetic dataset.

Normalized EMD (NEMD). Earth Mover's Distance (EMD), also known as Wasserstein Distance [37], is a distance function defined between two probability distributions over a given metric space (*e.g.*, 1D, 2D). It has been used in similar settings as ours, *e.g.*, to assess the quality of GAN models [38], or to compare two spatial distributions [39].

For our particular purpose of comparing real and synthetic service-level traffic maps, EMD is sufficient when we focus on a particular city. But that is not true for comparison over a set of cities due to their widely different sizes. To address this issue, we normalize the EMD between real and synthetic maps by the EMD between real map and uniform (2D) distribution. Let us denote the uniform traffic map as ϕ, the real map in simplex space as μ, and the synthetic map in simplex as v; then, the normalized EMD (NEMD) is defined as:

$$\text{NEMD} = \frac{EMD(\mu, v)}{EMD(\mu, \phi)},$$

where $EMD(a, b)$ is the EMD between 2D distributions a and b. It is worth noting that with EMD, the images will be converted to simplex space, and thus the information of the original data range is lost. This calls for use of complementary metrics such as those outlined next.

SSIM and PSNR. We also consider Structural Similarity Index Metric (SSIM) and Peak Signal-To-Noise Ratio (PSNR) – the two commonly used image quality assessment metrics [40] – to respectively evaluate the structural and pixel-level fidelity.

Hotspot Histogram EMD (HEMD). As noted earlier, hotspots are a key spatial feature of interest with mobile traffic data. To quantify the extent to which different methods faithfully capture this feature, we use the EMD between 1D distribution (histogram) of hotspots in synthetic and real data.

Besides the above quantitative fidelity metrics, we also consider qualitative measures including traffic histograms at city and pixel level as well as for number of hotspots to visualize the quality of the synthesized data with different methods, especially to gauge their ability to model underlying data variations (stochasticity).

B. Baselines

We consider a wide range of baseline methods to comparatively evaluate APPSHOT in terms of the metrics above.

CNN based Regression. A simple-minded approach for our multi-service traffic map generation task is to train a deep neural network (DNN) that takes conditions c as input and predicts a multi-channel image output x. Given the spatial nature of the output, CNN based regression is a good choice. This approach clearly fails to model stochasticity, a key characteristic of mobile traffic data. We implement this baseline via CNN on U-net architecture [36], and perform patch learning with non-overlapping fixed size patches with patch dimensions same as in APPSHOT.

	AppShot			Data	CartaGenie	SpectraGAN	Pix2Pix	CNN	FDaS
	Morning	Midday	Evening	All Peaks	Morning	Morning	Morning	Morning	Morning
WErr ↓	17.93%	19.09%	18.98%	15.10%	24.85%	17.74%	38.81%	42.07%	63.67%
NEMD ↓	0.35	0.36	0.35	0.23	0.44	0.57	0.52	0.58	0.99
SSIM ↑	0.84	0.86	0.86	0.96	0.85	0.79	0.78	0.76	0.44
PSNR ↑	34.88	32.61	32.47	39.24	36.15	35.01	34.07	34.88	29.33
HEMD ↓	2.09	1.99	2.24	0.6	3.90	2.93	4.31	5.79	9.20

TABLE II: Fidelity performance of AppShot at different peak periods (left) and of baselines for morning peak period (right).

Pix2Pix [23]. This model has been successfully used for image transformation tasks in computer vision. It is both conditional and stochastic like conditional GANs, but makes use of a tailored DNN architecture for image-to-image translation. Pix2Pix has several limitations compared to our AppShot approach, as discussed earlier in §II. In our implementation of this baseline, we perform non-overlapping patch based learning as above.

Fit Distribution and Sample (FDaS) [19]. As discussed in §II, a prior approach for mobile traffic data generation essentially involves fitting an empirical distribution to model the traffic data using maximum likelihood estimation of parameters and then sampling it afterwards to generate synthetic traffic [19]. While only total traffic demand across all services is considered in [19], we apply their approach separately for each service to allow comparison. Like in [19], we find log-normal distribution best fits the data but with different parameters across distributions, as expected. This approach has the inherent limitation of not being able to capture traffic correlations in space or time.

CartaGenie [16] and **SpectraGAN** [21]. These are the state of the art mobile traffic data generation methods that also employ conditional deep generative modeling as in AppShot. They, however, target generation of spatial snapshot or spatiotemporal data for *total* traffic, as with the FDaS baseline above. To apply them to the multi-service traffic generation case studied in this paper and have them as baselines, we train and use multiple separate instances of CartaGenie and SpectraGAN models, one per each service.

Data. In addition to the above baselines, we also consider an ideal case for reference, which we refer to as "Data". Metrics for this case are computed by splitting the real dataset (with 30 weekdays) into two distinct subsets (15 weekdays each part), and comparing these subsets of real data against each other. This captures the variability of the dataset within itself, which is a proxy for the 'upper bound' fidelity performance a synthetic data generation model like AppShot can achieve.

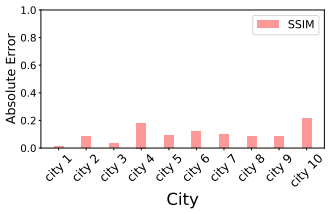
VII. Results

A. Fidelity and Generalization

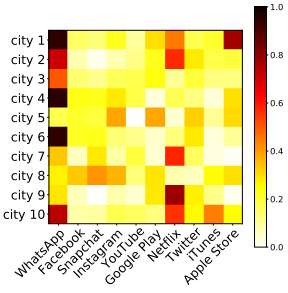
Throughout this section, we consider a leave-one-city-out evaluation. Specifically, each of the 10 cities in our dataset is taken as a test city in turn while using the data for the remaining 9 cities as the training set. This type of evaluation lets us assess the ability of AppShot and various baselines to generalize to unseen cities as well as their ability to handle different sized cities and their differences in traffic/context data value ranges. Following the earlier analysis in §IV, our evaluations focus on weekdays and morning/midday/evening peak hours. For brevity, we mainly show results for the morning peak hour period, unless otherwise specified; but similar conclusions apply for other periods.

Correlation between services. As shown in §IV, traffic for different services exhibit strong mutual correlations. So it is important for the generated traffic data to preserve this feature. To assess AppShot on this aspect, for each test city in the dataset, we compute the average of SSIM between traffic snapshots of every pair of services in the real ground-truth data, and similarly in the data synthesized with AppShot. We then compute the absolute error in the average pairwise SSIM computed over synthetic data with respect to that on real data. Results shown in in Figure 12a indicate that AppShot yields a small error, within 14% of the real data on average.

Composition of different services for different cities. Besides maintaining inherent correlations between traffic for different services, it is also important to ensure that their proportions relative to total traffic are preserved in the synthesized data. Figure 12b shows the error on this measure with AppShot relative to real data for different services with each test city. We observe that AppShot yields a low error within 20% of real in most cases. WhatsApp case is the only exception but this is an artefact due to traffic for this service making up a very small percentage (0.5%) so small absolute errors appear as big relative errors.



(a) Error in mutual correlation between services



(b) Traffic proportion error for each service and city

Fig. 12: AppShot service-level performance across cities.

Performance relative to baselines. The results of the comparative evaluation for the morning peak hour period are summarized in Table II. We observe that AppShot, in the comparison with the baseline methods (CartaGenie, SpectraGAN, Pix2Pix, CNN and FDaS), yields the best performance on two of the metrics (NEMD and HEMD) while being close to the best result on the other three metrics. Overall, AppShot provides the performance closest to the ideal 'Data' reference

across all metrics. Among the baselines, FDaS is clearly the worst performer on all metrics, showing the limitations of this approach in handling correlations in traffic and ensuring fidelity of the service-level snapshots.

Two other baselines – Pix2Pix and CNN based regression – have somewhat similar performance on all metrics, but considerably worse on most metrics relative to AppShot. This highlights their inability to accurately capture the spatial distribution of traffic relative to ground truth, which particularly harms the way certain key characteristics in the generated data (*e.g.*, the position and number of hotspots). This is particularly reflected in the HEMD performance which is more than double (twice as worse) than AppShot. Since Pix2Pix is marginally better than CNN based regression on all metrics, we only consider the former in the rest of our evaluations.

SpectraGAN exhibits slightly better performance than AppShot with respect to two metrics (WErr and PSNR) but substantially worse on the remaining three metrics. CartaGenie is similar in that it does slightly better than AppShot with respect to SSIM and PSNR but has substantially worse performance on the other three metrics. Broadly speaking, this overall relatively poor performance of SpectraGAN and CartaGenie compared to AppShot can be attributed to their inability to exploit inter-service correlations due to independent generation of per-service traffic and insufficient measures to correctly model hotspots (reflected in their significantly worse performance in terms of HEMD).

The shortcomings of these two baselines with respect to AppShot are apparent in the visualizations of synthetic traffic maps they generate as shown in Figure 13. We see that CartaGenie and SpectraGAN respectively yield unacceptable synthetic traffic maps for Instagram and YouTube for City **1**, the most challenging target city given its vastly bigger size, population density, traffic volume and hotspots compared to other cities in our dataset (see §IV). In §VII-B, we will further explore the performance of CartaGenie and SpectraGAN relative to AppShot. In Figure 13, also note that Pix2Pix fails to provide meaningful traffic maps for any service and exhibits severe artefacts, consistent with its poor performance in terms of quantitative fidelity metrics as seen above.

Performance at different peak periods. We now consider how well AppShot generates service level traffic snapshots at different peak periods. Results for different fidelity metrics averaged across all test cities are summarized in the left panel of Table II. We observe that AppShot provides consistent performance for all periods close to the ideal 'Data' reference, with WErr under 20% and near-ideal results for SSIM and PSNR.

Capturing statistical variations. It is important for a synthetic mobile traffic data generation model to model inherent stochasticity in such data. This reflects the model's ability to learn traffic distributions conditioned on the contextual input, rather than simply outputting a deterministic transformation (as CNN based regression would do). We examine this aspect considering histograms of city-level and pixel-level total traffic volume across all test cities, and the number of hotspots. Results shown in Figure 14 for AppShot clearly demonstrate that it achieves this intended goal. Note

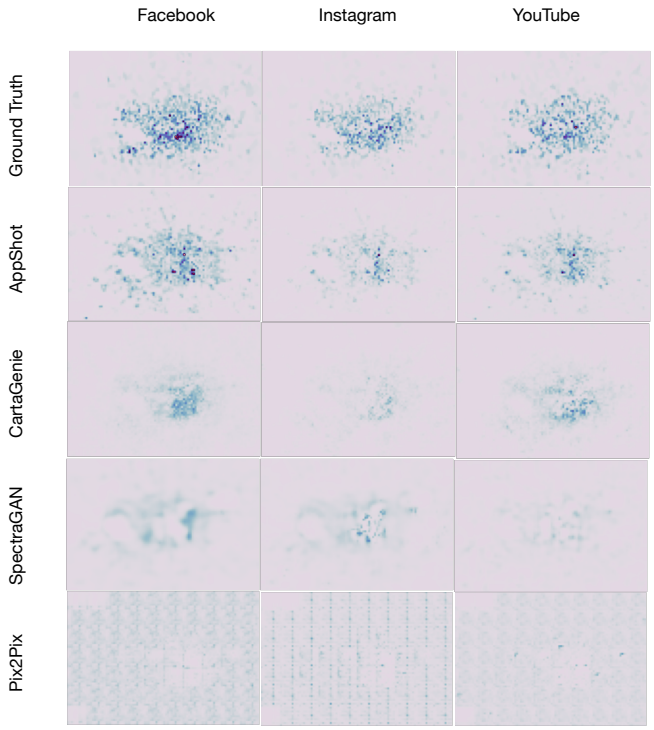
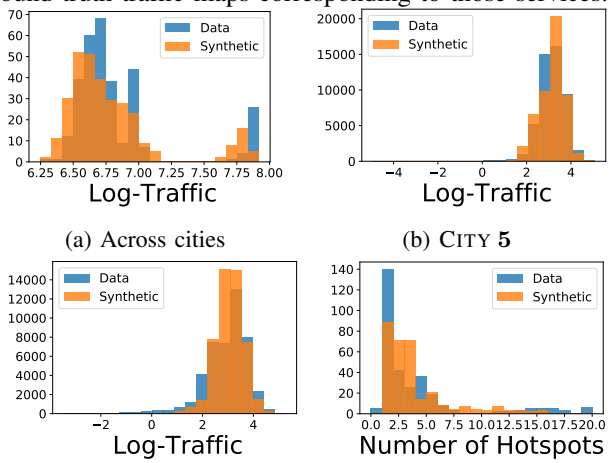


Fig. 13: Synthetic traffic maps for select services in City **1** generated with different methods compared against the ground truth traffic maps corresponding to those services.



(a) Across cities (b) City **5**

(c) City **9** (d) #Hotspots

Fig. 14: Statistical features of AppShot-generated data.

that we include pixel-level histograms for only two arbitrarily selected test cities for brevity.

B. Detailed Comparisons with CartaGenie and SpectraGAN

1) CartaGenie: Earlier in this section, we have already highlighted the benefit of AppShot as a whole relative to the alternative of using multiple separate per-service instantiations of the CartaGenie model. Here we dissect AppShot to examine the benefit due to some of its underlying design choices and contrast with those underlying CartaGenie.

In Table III, the 'L1+D1+D2' represents the AppShot design, using adversarial training with two discriminators as well as use of overlapping patches and sliding across them one pixel at a time (see Figure 10c). The 'No Overlap' case

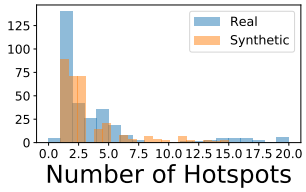


Fig. 15: Per-hour NEMD histogram for AppShot.

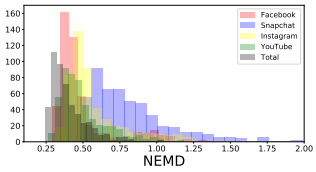


Fig. 16: Per-hour NEMD histogram for SpectraGAN.

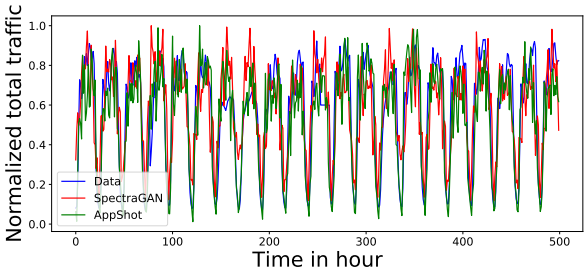


Fig. 17: City scale total traffic time series of YouTube in City 1, normalized by the maximum hourly traffic.

is different from 'L1+D1+D2' in that the former uses non-overlapping patch based training (see Figure 10b) as in the CartaGenie design. Clearly, non-overlapping patches worsens performance on all metrics, significantly so for several of the metrics (WErr, NEMD and HEMD).

The other two alternative designs – 'L1 Only' and 'L1+D1' – shown in Table III use overlapping patches as in AppShot but differ in their loss functions. Here 'L1 Only' represents the case where only L1 loss is used for the loss function as done in CartaGenie. We see that doing so results in overall worse performance compared to AppShot (*i.e.*, 'L1+D1+D2'). In particular, using L1 loss alone is clearly insufficient to accurately model traffic composition (as measured by WErr) and capturing hotspot distribution (HEMD). Addition of a discriminator (via adversarial training as in GAN), shown as L1+D1, helps on both fronts. Yet another discriminator (L1+D1+D2) to ensure correct traffic composition, as we do in AppShot, provides the best performance overall.

	No Overlap	L1 Only	L1+D1	L1+D1+D2
WErr ↓	32.04%	34.68%	22.68%	17.93%
NEMD ↓	0.47	0.33	0.36	0.35
SSIM ↑	0.79	0.81	0.85	0.84
PSNR ↑	32.6	37.3	35.2	34.88
HEMD ↓	3.70	5.59	2.19	2.09

TABLE III: AppShot design (shown under L1+D1+D2 in the table) compared against alternative design choices. The 'No Overlap' and 'L1 Only' represent the design choices underlying the CartaGenie model.

2) SpectraGAN: Different from other baselines, SpectraGAN is designed to capture the spatiotemporal features of mobile traffic. We extend SpectraGAN to service-level generation by training it on each service independently. To evaluate AppShot in the time domain and show that it generalizes to different periods, we train AppShot to generate service level snapshots for each hour of the day (*i.e.*, the same granularity as SpectraGAN) separately, and obtain synthetic service-level traffic over time by stitching the hourly snapshots. Specifically, we train 24 models with AppShot that correspond to different hours of a day. The time series of city-scale total traffic of YouTube after stitching is illustrated in Figure 17.

Spatial-domain performance. The histogram of per-hour NEMD over a period of 3 weeks is shown in Figure 15 and Figure 16 for AppShot and SpectraGAN, respectively; there we consider total traffic and the four popular services. AppShot yields consistently good performance for individual services as well as for total traffic. SpectraGAN, on the other hand, performs significantly worse for some services, and is unstable over time. These results are in line with worse spatial fidelity (in terms of NEMD and SSIM) seen previously with SpectraGAN in Table II. A key reason for this is its inability to exploit inter-service correlations, unlike AppShot.

Time domain performance. We employ the L1 distance of autocorrelation between synthetic and real data (AC-L_1), also considered in previous work [21], [22], to comparatively evaluate the temporal fidelity of the synthetic data between SpectraGAN and AppShot. Specifically, we compute this metric by taking the average value of L_1 norm between the corresponding points of the auto-correlations of real and synthetic time-series data, at the pixel level. Lower values thus imply better performance.

	Instagram	Snapchat	Facebook	YouTube	Total (10)
AppShot	47.3	60.1	69.8	46.8	62.0
SpectraGAN	75.6	94.6	75.9	71.2	65.3

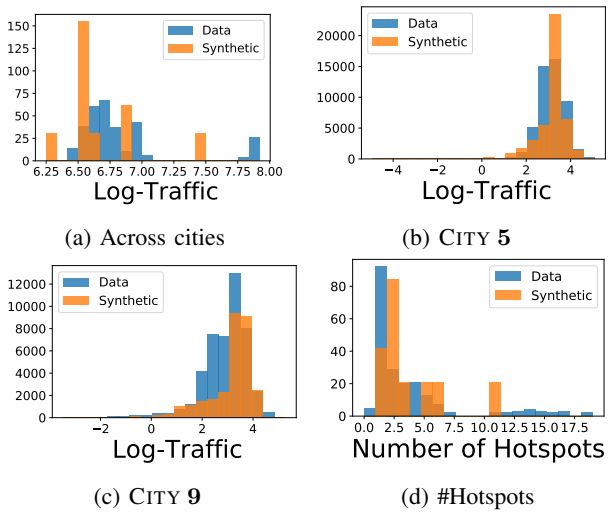
TABLE IV: Time domain performance comparison between AppShot and SpectraGAN in terms of AC-L_1 (lower is better).

Table IV shows results comparing the performance AppShot with SpectraGAN in terms of median AC-L_1, considering the top four popular services and total traffic of all ten services available. While the two methods achieve similar performance for total traffic, AppShot has substantially better performance at the individual service level. Figure 17 highlights the particular case of YouTube traffic by way of explaining these performance differences. In Figure 17, we observe that SpectraGAN tends to largely overestimate the actual traffic relative to ground truth, especially during idle periods (*e.g.*, in hours 180 or 335), while AppShot correctly models such situations.

C. Benefit from Other Design Choices and Parameter Tuning

We now present further results supporting design choices in AppShot, and discuss the tuning of its key hyper-parameters.

Noise Input Effect with FiLM Layer. As discussed in §V-C, naive conditioning on the noise input to the generator by simply concatenating it with (spatial) context input can cause the model to ignore the noise input altogether and prevent it from modeling stochasticity in the data. We avoid this issue by using FiLM layer [33] to provide the noise input separately



(a) Across cities (b) CITY 5

(c) CITY 9 (d) #Hotspots

Fig. 18: Statistical features of APPSHOT-generated data *without* the FiLM layer.

through it. The benefit from this choice is highlighted by comparing the results in Figure 14 with that of Figure 18 where in the latter case APPSHOT uses naive conditioning on noise input without the FiLM layer.

Hyper-parameters. We have determined the best settings for various hyper-parameters empirically. These resulted in the use of 12×12 as input context patch size (for 10×10 output traffic patch size) and 3×3 kernel size at the first convolutional layer. Among the various hyper-parameters of the APPSHOT neural network model, through experiments and analysis, we find that the kernel size of initial convolutional layers in APPSHOT plays a critical role in determining the fidelity of output synthetic traffic maps (as illustrated in Table V and Figure 19). We choose 3×3 kernel size as the setting for the first convolutional layer that generally works well.

VIII. RADIO NETWORK SLICING USE CASE

As part of our performance evaluation, we assess the utility of APPSHOT through a downstream application use case of multi-service mobile traffic data. Specifically, we employ synthetic data generated by APPSHOT to effectively feed a recent model for the estimation of the multiplexing efficiency with radio network slicing [3]. The actual size of each pixel in our dataset is $250 \times 250m^2$, which is close to the coverage range of a small cell. We thus assume that the radio network is composed of small cells each matching one pixel. We further assume that each service is associated with an individual slice, *i.e.*, a dedicated and customized set of network resources and functions that allows achieving strong quality of service (QoS) guarantees to the service providers. The need to isolate resources to each slice (*i.e.*, service) is at the root of a reduced multiplexing efficiency: resources need to be allocated for each slice, and cannot be multiplexed as in legacy networks that cannot provide strong QoS [3]. In our case, the sliced resources are at the radio access level (*e.g.*, spectrum or baseband processing resources), hence must accommodate the per-service traffic generated in each pixel separately.

Formally, suppose there are N cells in the target region, and let us denote by $r_{i,s}(p,t)$ the minimal resource to serve the traffic demand of slice (*i.e.*, service) s in cell i for a fraction of time p over a reconfiguration period t. The value of $r_{i,s}(p,t)$ can be derived from multi-service mobile traffic data generated with APPSHOT and using the model in [3]. The (minimum) amount of resources needed to serve the overall traffic in absence of slicing (*i.e.*, when multiplexing across services is possible) is $R_i(p,t)$. Then the network slicing efficiency is:

$$E(p) = \frac{\sum_{t=1}^{T}\sum_{i=1}^{N} R_i(p,t)}{\sum_{t=1}^{T}\sum_{i=1}^{N}\sum_{s=1}^{S} r_{i,s}(p,t)}.$$

We compute the accuracy of estimating multiplexing efficiency with APPSHOT-generated data relative to using real data. We consider low and high coverage cases, respectively corresponding to covering 95% and 99% of demand in each reconfiguration period, *i.e.*, $p = \{0.95, 0.99\}$. In other words, more than 95% or 99% of the demand must be accommodated in each slice during a reconfiguration period. We consider a wide range of reconfiguration periods from 2h to 36h. As seen from the results in Table VI, the APPSHOT-generated data only introduces about 5% error in estimating the multiplexing efficiency compared with the real traffic data for short reconfiguration periods. The estimation error with APPSHOT data slightly increases with increasing reconfiguration period as well as lowered coverage probability but it always is within 10% relative to using real data.

	2h	4h	8h	12h	16h	20h	24h	36h
95% Coverage	4.7%	4.6%	5.6%	4.9%	5.5%	5.5%	8.1%	8.4%
99% Coverage	4.7%	4.6%	5.5%	4.9%	5.4%	5.5%	6.0%	6.8%

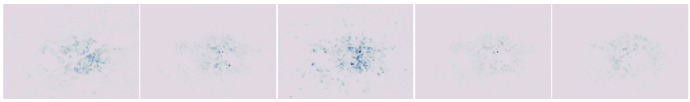
TABLE VI: Error in multiplexing efficiency estimation with APPSHOT-generated data for different coverage probabilities and reconfiguration periods.

IX. CONCLUSIONS

We have presented APPSHOT, a novel conditional deep generative model for synthesizing high-fidelity multi-service network traffic data that needs only publicly available context information of target regions. We have used real-world service-level mobile traffic data for multiple cities for our evaluation and show that APPSHOT not only outperforms a range of baseline approaches in terms of fidelity and also generalizes well to unseen regions. Our patch-based learning approach and corresponding operations have proved to be effective in generating traffic for cities with different sizes. Also, data augmentation with overlapping patches significantly enhances the performance with respect to handling traffic hotspots and diverse traffic ranges. The APPSHOT neural network architecture and the service level constraints it incorporates significantly enhance the accuracy of service compositions in synthetic traffic, while preserving a strong structural correlation between services. Furthermore, APPSHOT is shown to capture realistic statistical variations on both city-wide traffic demand and structural characteristics (e.g., number of hotspots). Finally, we have demonstrated the utility of APPSHOT-generated data through a use case on radio network slicing.

Kernel Size	CITY 1	CITY 2	CITY 3	CITY 4	CITY 5	CITY 6	CITY 7	CITY 8	CITY 9	CITY 10	Average
1	0.32	0.52	0.41	0.34	0.46	0.45	0.40	0.44	0.36	0.28	0.38
2	0.31	0.47	0.42	0.37	0.45	0.50	0.34	0.38	0.35	0.33	0.39
3	0.27	0.40	0.40	0.35	0.38	0.41	0.35	0.38	0.36	0.26	0.35
6	0.30	0.53	0.55	0.43	0.49	0.47	0.55	0.53	0.35	0.26	0.44
11	0.45	0.54	0.44	0.37	0.51	0.46	0.54	0.44	0.35	0.25	0.44

TABLE V: APPSHOT performance in terms of NEMD with different kernel sizes for the first convolutional layer.



(a) K=1 (b) K=2 (c) K=3 (d) K=6 (e) K=11

Fig. 19: APPSHOT-generated traffic maps for Instagram in CITY 1 with different kernel sizes (K) for the first convolutional layer.

REFERENCES

[1] "Description of network slicing concept," https://www.ngmn.org/wp-content/uploads/Publications/2016/161010_NGMN_Network_Slicing_framework_v1.0.8.pdf, NGMN Alliance.

[2] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, C. Ziemlicki, and Z. Smoreda, "Not all apps are created equal: Analysis of spatiotemporal heterogeneity in nationwide mobile service usage," in *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2017, pp. 180–186.

[3] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Perez, "How Should I Slice My Network?: A Multi-Service Empirical Evaluation of Resource Sharing Efficiency," in *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2018, pp. 191–206.

[4] O. U. Akgul, I. Malanchini, and A. Capone, "Slice-Aware Capacity Expansion Strategies in Multi-Tenant Networks," in *IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[5] G. Klas, "Edge Computing and the Role of Cellular Networks," *IEEE Computer*, vol. 50, no. 10, pp. 40–49, 2017.

[6] G. S. Paschos, G. Iosifidis, M. Tao, D. Towsley, and G. Caire, "The Role of Caching in Future Communication Systems and Networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 6, pp. 1111–1125, 2018.

[7] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Multi-classification approaches for classifying mobile app traffic," *Journal of Network and Computer Applications*, vol. 103, pp. 131–145, 2018.

[8] H. Rocha, G. Delsart, A. Andersson, A. Bousselmi, A. Conte, A. Gati, A. M. Masucci, C. Grangeat, C. Cavdar, D. Marquet *et al.*, "Soogreen: Service-oriented optimization of green mobile networks," in *2017 15th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*. IEEE, 2017, pp. 1–8.

[9] C. Joe-Wong, S. Ha, S. Sen, and M. Chiang, "Do Mobile Data Plans Affect Usage? Results from a Pricing Trial with ISP Customers," in *Passive and Active Measurement (PAM) Conference*, 2015.

[10] Y. Zhao, H. Wang, H. Su, L. Zhang, R. Zhang, D. Wang, and K. Xu, "Understand Love of Variety in Wireless Data Market Under Sponsored Data Plans," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 4, pp. 766–781, 2020.

[11] R. Singh *et al.*, "Urban Vibes and Rural Charms: Analysis of Geographic Diversity in Mobile Service Usage at National Scale," in *The ACM Web Conference (WWW)*, 2019, pp. 1724–1734.

[12] I. Ucar, M. Gramaglia, M. Fiore, Z. Smoreda, and E. Moro, "News or social media? socio-economic divide of mobile service consumption," *Journal of The Royal Society Interface*, vol. 18, no. 185, p. 20210350, 2021. [Online]. Available: https://royalsocietypublishing.org/doi/abs/10.1098/rsif.2021.0350

[13] G. Barlacchi *et al.*, "A multi-source dataset of urban life in the city of Milan and the Province of Trentino," *Nature Scientific Data*, vol. 2, no. 150055, 2015.

[14] U. Paul, A. P. Subramanian, M. M. Buddhikot, and S. R. Das, "Understanding traffic dynamics in cellular data networks," in *Proceedings IEEE INFOCOM*, 2011, pp. 882–890.

[15] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, and J. Wang, "Characterizing geospatial dynamics of application usage in a 3G cellular data network," in *Proceedings of IEEE INFOCOM*, 2012, pp. 1341–1349.

[16] K. Xu, R. Singh, H. Bilen, M. Fiore, M. K. Marina, and Y. Wang, "CartaGenie: Context-Driven Synthesis of City-Scale Mobile Network Traffic Snapshots," in *2022 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE Computer Society, Mar 2022, pp. 119–129. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/PerCom53586.2022.9762395

[17] "D-ITG homepage," http://traffic.comics.unina.it/software/ITG/, accessed: 2021-10-19.

[18] A. Botta, A. Dainotti, and A. Pescapé, "A tool for the generation of realistic network workload for emerging networking scenarios," *Computer Networks*, vol. 56, no. 15, pp. 3531–3547, 2012.

[19] P. Di Francesco, F. Malandrino, and L. A. DaSilva, "Assembling and using a cellular dataset for mobile network analysis and planning," *IEEE Transactions on Big Data*, vol. 4, no. 4, pp. 614–620, 2017.

[20] B. Ma, B. Yang, Z. Zhang, and J. Zhang, "Modelling mobile traffic patterns using generative adversarial neural networks," in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, pp. 1–7.

[21] K. Xu, R. Singh, M. Fiore, M. K. Marina, H. Bilen, M. Usama, H. Benn, and C. Ziemlicki, "SpectraGAN: Spectrum Based Generation of City Scale Spatiotemporal Mobile Network Traffic Data," in *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies (CoNEXT'21)*. ACM, 2021, p. 243–258. [Online]. Available: https://doi.org/10.1145/3485983.3494844

[22] Z. Lin, A. Jain, C. Wang, G. Fanti, and V. Sekar, "Using GANs for Sharing Networked Time Series Data: Challenges, Initial Promise, and Open Questions," in *Proceedings of the ACM Internet Measurement Conference*, ser. IMC '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 464–483. [Online]. Available: https://doi.org/10.1145/3419394.3423643

[23] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1125–1134.

[24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[25] Y. Zhao, R. Wu, and H. Dong, "Unpaired image-to-image translation using adversarial consistency loss," in *European Conference on Computer Vision (ECCV)*. Springer, 2020, pp. 800–815.

[26] Y. Zhang, Y. Li, X. Zhou, X. Kong, and J. Luo, "TrafficGAN: Off-Deployment Traffic Estimation with Traffic Generative Adversarial Networks," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 1474–1479.

[27] "General Data Protection Regulation (GDPR)," https://gdpr-info.eu/.

[28] "Copernicus Urban Atlas," https://land.copernicus.eu/local/urban-atlas/urban-atlas-2012, accessed: 2021-10-19.

[29] J. L. Myers, A. D. Well, and R. F. Lorch Jr, *Research design and statistical analysis*. Routledge, 2013.

[30] "OpenStreetMap Overpass API," http://overpass-turbo.eu/, accessed: 2021-10-19.

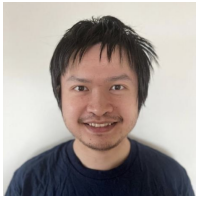
[31] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[32] E. Mededovic, V. G. Douros, and P. Mähönen, "Node centrality metrics for hotspots analysis in telecom big data," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2019, pp. 417–422.

[33] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "FiLM: Visual reasoning with a general conditioning layer," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[34] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. Madison, WI, USA: Omnipress, 2010, p. 807–814.
[35] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*. PMLR, 2015, pp. 448–456.
[36] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
[37] L. Rüschendorf, "The Wasserstein distance and approximation theorems," *Probability Theory and Related Fields*, vol. 70, no. 1, pp. 117–129, 1985.
[38] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of Wasserstein GANs," *arXiv preprint arXiv:1704.00028*, 2017.
[39] S. Isaacman, R. Becker, R. Cáceres, M. Martonosi, J. Rowland, A. Varshavsky, and W. Willinger, "Human mobility modeling at metropolitan scales," in *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 239–252. [Online]. Available: https://doi.org/10.1145/2307636.2307659
[40] A. Hore and D. Ziou, "Image quality metrics: PSNR vs. SSIM," in *20th International Conference on Pattern Recognition*. IEEE, 2010, pp. 2366–2369.

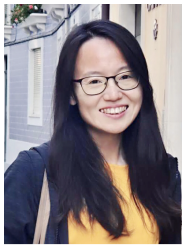
Chuanhao Sun is a PhD student at the School of Informatics, University of Edinburgh, under the supervision of Prof. Mahesh Marina. He received MSc degree from Wireless Signal Processing and Networking (WSPN) Lab, Beijing University of Post and Telecommunication, China (2019). He did internship with Intel Lab China, Beijing, China (2018 and 2019), on 5G NR protocol simulation, and Microsoft Azure of Operators (AFO) Research, Cambridge, U.K. (2022), on enhancing user equipment energy efficiency via NR networking mechanism. His research mainly focus on mobile networking measurement & management, and the application of machine learning in mobile networking.



Kai Xu is a Research Scientist at Amazon working on generative models for creative designs. Kai obtained his PhD with Charles Sutton at the University of Edinburgh and his MPhil with Zoubin Ghahramani at the University of Cambridge. Kai's research has been focusing on approximate inference methods, deep generative models and probabilistic programming, with papers published at top tier machine learning venues including NeurIPS, ICML, ICLR and AISTATS. He co-developed the Turing probabilistic programming language in Julia with Hong Ge at Cambridge Machine Learning Group.

Marco Fiore is Research Associate Professor at IMDEA Networks Institute and CTO at Net AI Tech Ltd. He received MSc degrees from University of Illinois at Chicago, IL, USA (2003), and Politecnico di Torino, Italy (2004), a PhD degree from Politecnico di Torino (2008), Italy, and a Habilitation a Diriger des Recherches (HDR) from Universite de Lyon, France (2014). He held tenured positions as Maitre de Conferences (Associate Professor) at Institut National des Sciences Appliquees (INSA) de Lyon, France (2009-2013), and Researcher at Consiglio Nazionale delle Ricerche (CNR), Italy (2013-2019). He has been a visiting researcher at Rice University, TX, USA (2006-2007), Universitat Politecnica de Catalunya (UPC), Spain (2008), and University College London (UCL), UK (2016-2018). Dr. Fiore is a senior member of IEEE, and a member of ACM. He was a recipient of a European Union Marie Curie fellowship and a Royal Society International Exchange fellowship. He leads the Networks Data Science group at IMDEA Networks Institute, which focuses on research at the interface of computer networks, data analysis and machine learning.

Mahesh Marina is a Professor in the School of Informatics at the University of Edinburgh and a Turing Fellow at the Alan Turing Institute in London. Before joining Edinburgh, he had a two-year postdoctoral stint at the UCLA Computer Science Department. He earned his PhD in Computer Science in 2004 from the State University of New York at Stony Brook. He has previously held visiting researcher positions at ETH Zurich and Ofcom London. He is a Distinguished Member of the ACM and a Senior Member of the IEEE. More information about him and his research can be found at http://homepages.inf.ed.ac.uk/mmarina/.



Yue Wang is a Senior Manager at Samsung UK. She leads a team working on the cutting-edge research for beyond 5G and 6G Networks, and delivers advanced technologies, standard contributions, and proof-of-concepts for AI applications to Networks. At Samsung, she has participated in multiple SDOs, including ETSI ENI and O-RAN. From 2017-2019, she has been the Secretary and Rapporteur of ETSI ISG ENI (Experiential Networked Intelligence). From 2015-2017, she led the H2020 mmMAGIC project with 19 industry and academia partners, on the design and development of the mmwave radio access networks. She is currently the Industry Advisory Board member of King's College of London, Loughborough University, and University of Sussex. Yue has over 40 publications and over 30 patents. Prior to joining Samsung, Yue held various roles in the US and the UK, all on wireless communications research, development, and standards. She is the recipient of ETSI ENI Award for her 'extraordinary contributions' in 2019, and Outstanding Woman Engineer of the Year of Cambridge Wireless Technology and Engineering Award for in 2020.

Cezary Ziemlicki is a R&D engineer at Orange Innovation, in the laboratory Sociology & Economics of Networks and Services, Chatillon, France. A graduate of the Warsaw University of Technology in automation of industrial processes, Cezary is a research engineer and joined Orange Innovation in 2000. His work at Orange is to develop methodologies for Big Data analysis of telco operator data for use in human and economics sciences.