GenDT: Mobile Network Drive Testing Made Efficient with Generative Modeling

Citation for published version:

Digital Object Identifier (DOI):
10.1145/3555050.3569124

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Proceedings of The 18th International Conference on emerging Networking EXperiments and Technologies

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and/or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
GenDT: Mobile Network Drive Testing Made Efficient with Generative Modeling

Chuanhao Sun†, Kai Xu†, Mahesh K. Marina†, Howard Benn‡
The University of Edinburgh‡, Samsung‡

ABSTRACT

Drive testing continues to play a key role in mobile network optimization for operators but its high cost is a big concern. Alternative approaches like virtual drive testing (VDT) target device testing in the lab whereas MDT or crowdsourcing based approaches are limited by the incentives users have to participate and contribute measurements. With the aim of augmenting drive testing and significantly reducing its cost, we propose GenDT, a novel deep generative model that synthesizes high-fidelity time series of key radio network key performance indicators (KPIs). The training of GenDT relies on a relatively small amount of real-world measurement data along with corresponding and easily accessible network and environment context data. Through this, GenDT learns the relationship between context and radio network KPIs as they vary over time, and therefore trained GenDT model can subsequently be relied on to generate time series for different KPIs for new drive test routes (trajectories) without having to collect field measurements. GenDT represents an initial attempt at enabling efficient drive testing via generative modeling. Evaluations with real-world mobile network drive testing measurement datasets from two countries demonstrate that GenDT can synthesize significantly more dependable data than a range of baselines. We further show that GenDT has the potential to significantly reduce the drive testing related measurement effort, and that GenDT-generated data yields similar results to that with real data in the context of two downstream use cases – QoE prediction and handover analysis.

CCS CONCEPTS
• Networks → Mobile networks; • Computing methodologies → Neural networks.

KEYWORDS
Mobile Network Drive Test Measurement Data, Synthetic Data Generation, Deep Generative Modeling, Conditional GANs

Permission to make digital or hard copy of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM Reference Format:

1 INTRODUCTION

Drive testing has traditionally been an integral part of operating mobile networks [11, 17, 48]. A key aim of drive testing is measurement based assessment and optimization of mobile network coverage, capacity and quality of service (QoS). It involves collecting field measurements in a controlled manner by driving or walking in a target scenario. Several measurement tools are available to perform drive or walk testing [2, 27, 29, 52]. The principal concern with traditional drive testing is that it requires manual effort to obtain measurements and so is costly and time-consuming.

There exist broadly two alternative approaches to reduce drive testing cost. One approach, generally referred to as Virtual Drive Testing (VDT) [6], is aimed at enabling device or infrastructure equipment testing in the lab under realistic conditions. The idea is to initially obtain a set of field measurements, as in traditional drive testing, and then recreate the field environment in the lab by replaying drive test scenarios and replicating field-measured channel conditions through a hardware channel emulator. Keysight VDT toolset [35] and Spirent Live2Lab [37] represent this approach. This approach is obviously limited to device/equipment testing and so does not cater to the needs of optimizing operational mobile networks – the latter is our focus in this paper.

The other existing approach seeks to leverage measurements from real end-user devices. From a network/operator perspective, 3GPP has introduced minimization of drive tests (MDT) feature in Release 10 to obtain measurements from actual user devices and enhanced it since [1, 28]. While this is an appealing approach and has been the focus of some industry solutions and trials (e.g., [22, 39, 65]), users’ consent is needed for their devices to participate in the MDT framework, especially to provide device side context information (e.g., location) to annotate measurements. This in turn causes the issue of sparse or skewed measurement data with MDT [54]. On the other hand, inferring device locations on the network side suffers from inaccuracies along with the additional concern due to device diversity [57].

Alternatively, device side measurements can also be collected in a crowdsourced manner via dedicated measurement apps or SDKs (from third-party mobile analytics companies) installed on user devices (e.g., OpenSignal [36], Tutela [38]). The scope and granularity of measurements that can be gathered with such crowdsourced solutions are limited by device OS APIs (e.g., Android Telephony API [25]) and so they are mostly limited to coverage mapping based on signal strength measurements [3, 19]. Crucially, the effectiveness of both MDT and crowdsourcing based measurement approaches are limited by the ability to provide incentives for users to participate and to safeguard their privacy.

In this paper, we introduce a new approach, termed GenDT, that is powered by deep generative modeling for making drive testing efficient. Unlike the VDT approach [6, 35, 37], we design GenDT with
measurement and optimization of operational mobile networks in mind. The essential idea behind our approach in GenDT is to develop a deep generative model that effectively mimics drive testing. Traditional drive testing results in a time-series of measurements for different radio network KPIs (e.g., RSRP, RSRQ) over a specified measurement trajectory. Similarly, GenDT takes a trajectory as an input and generates the time-series data for multiple radio network KPIs corresponding to that trajectory (see Figure 5 for an illustration). Note that trajectory here means a sequence of (location, timestamp) tuples so the user/device mobility is implicitly captured by this notion of trajectory. As our aim is to reduce the number of measurements required with drive testing, we use readily available network and environment ‘context’ as an aid, and train GenDT to learn the relationship between the relevant context around a measurement trajectory and the corresponding radio KPI time-series data. For the network context, we use cell site location and configuration information that an operator would hold. Points of interest (PoIs) and types of land use around the device location make up our environment context.

Given the above, the core technical problem we target with GenDT is conditional multivariate time-series data generation, where the drive testing trajectory and its context make up the condition (input) to the model to steer the data generation process, and the output is the time-series data for multiple variables (i.e., radio KPIs of interest). For training the GenDT model, we leverage a small number of controlled radio network measurements for different measurement scenarios (highway, city center, etc.) collected as with traditional drive testing. Each of these measurements is annotated with the device location and the corresponding contextual information. The GenDT model once trained as above can then be relied on to generate radio network KPI time-series data for a new unseen drive test trajectory without having to collect field measurements, by simply providing the trajectory and its surrounding context as input to the model.

Realizing the GenDT approach as outlined above poses a significant challenge. On one hand, GenDT should be able to generate high-fidelity (dependable) KPI time-series data for new unseen trajectories (i.e., generalize well). On the other hand, GenDT should rely on minimal amount of measurement data for training. Addressing this challenge entails tackling a number of issues in turn: (i) Dynamic context input: the relevant context keeps changing as the device moves along the drive testing trajectory. This includes not only the immediate environment but also the number and the actual set of potential serving cells around the device location; (ii) Long and complex scenarios: drive testing trajectories can be arbitrarily long which means the model should be able to generate correspondingly long time series of radio KPIs without loss of fidelity. Moreover, real-world drive testing trajectories can be complex spanning several different measurement scenarios (highway, city center, etc.); (iii) Stochasticity: radio network KPIs are inherently stochastic and so the generated data should preserve this characteristic by having the distribution of synthesized data aligning with real measurement data; (iv) Minimal training data: the model should provide insights to optimize the amount of training data needed while ensuring high fidelity so as to strike the right balance between dependability and measurement efficiency.

In GenDT, we address (i) via a tailored Graph Neural Network (GNN) [5] based LSTM network component, where a node level network is used to map the time-varying cell information context into a high-dimensional graph; this then feeds into another aggregation network to learn the graph level information and output a multichannel time-series output, where each channel of the output represents a different radio network KPI. We tackle (ii) with a batch generation mechanism – the training and generation is done at a smaller batch level to preserve temporal patterns and improved training efficiency. We address (iii) by introducing a stochastic layer in the LSTM network and adversarial training for effectively modeling the stochastic nature of radio KPIs. Finally to address (iv), we incorporate a residual generation component in the model whose parameters give hints on model versus data uncertainty, thereby help achieve high fidelity with minimal training data.

We evaluate the GenDT with respect to a range of baseline approaches, using two real-world drive testing measurement datasets from two different countries. We not only assess the fidelity of the data generated with GenDT relative to baselines but also highlight its ability to achieve high fidelity with minimal amount of training data – the latter translates to greater measurement efficiency to benefit drive testing. All our evaluations are over the testing subset of each of the datasets that is non-overlapping with the part used for training. As such, we demonstrate the ability of GenDT to generalize to new unseen trajectories. We also present evaluations showing the effectiveness of GenDT in supporting downstream use cases as well as an ablation study to evaluate design choices underlying GenDT. In summary, we make the following key contributions:

- (§3) We first present an analysis of drive testing measurement data characteristics that motivate our model design.
- (§4) We propose a novel conditional deep generative model, GenDT, featuring several new innovations. To the best of our knowledge, GenDT is the first method for synthesizing dependable radio KPI time series data and as such the first step towards enabling efficient drive testing via generative modeling.
- (§6.1) Using real-world drive testing measurement datasets from two countries, we show that GenDT synthesizes realistic time series for multiple key radio network KPIs for new unseen trajectories and generally outperforms all baselines.
- (§6.2) Crucially, we demonstrate the potential of GenDT to reduce the measurement effort with drive testing by leveraging the model uncertainty measure within GenDT - it maintains high fidelity for long and complex trajectories using as little as 10% of the available data, or equivalently yield 90% measurement efficiency.
- (§6.3) Moreover, we demonstrate the utility of GenDT for downstream applications through two distinct use cases, showing that using data generated by GenDT yields results comparable to those obtained using real drive test measurements.

2 BACKGROUND

2.1 Related Work

Our work is positioned in the context of mobile network drive testing and is aimed at reducing its cost associated with measurement data collection. As stated at the outset, the VDT approach [6, 35, 37] is limited to device/equipment testing and so is unsuitable for this
purpose. The other alternative approaches involving user device based measurement collection via MDT [1, 28, 54] or crowdsourcing [3, 19, 36, 38] are hindered by insufficient incentives and privacy concerns. To the best of our knowledge, our work is the first to explore the generative modeling approach towards making drive testing efficient and cost effective.

Broadly related are the works focusing on coverage mapping and pathloss prediction, which can be seen as a subset of drive testing use cases. In contrast to traditional methods including ray-tracing [43], recent work (e.g., [3, 19, 57, 61]) has adopted statistical and machine learning approaches for measurement or computational efficiency. Alimpertis et al. [3] propose a random forests based model for prediction of signal strength (RSRP) map, whereas Thrane et al. [57] present a convolutional neural network (CNN) based supervised spatial regression model that maps satellite images of a target region to signal quality parameters like RSRP and RSRQ in that region. On the other hand, [61] focuses on pathloss prediction using multi-layer perceptron (MLP) based neural network model. The above mentioned works cannot mimic measurements with drive testing as they do not have a notion of user trajectory or temporal variations. They also make a simplifying but inaccurate assumption that serving cell at each location is fixed and known. Moreover, the model in [57] due to being trained with satellite images for a specific region does not generalize beyond that region. In contrast, our proposed GenDT approach overcomes the above limitations through a tailored and novel deep generative model.

Our design of GenDT leverages graph neural networks (GNNs) [5] to effectively handle varying network context around a drive testing trajectory. While there have been some recent works employing GNNs for time-series prediction problems (e.g., [32, 58]), to our knowledge, ours is the first work on GNN based time-series data ‘generation’. As noted in prior work [62], data generation is a much harder task than prediction. We comparatively evaluate our model with the LSTM-GNN model [58].

Using deep generative models, especially generative adversarial networks (GANs) and variational autoencoders (VAEs), for data synthesis is of prime interest currently [34]. Such models are being used to generate data for machine learning, in finance, healthcare and other domains. Within the mobile networking domain, there have been few recent works proposing deep generative models for various types of network and wireless data. The potential for GANs to generate physical layer channel response samples for MIMO channels has been discussed in [63]. SpectraGAN [62] is another broadly related work in this domain that targets the generation of spatiotemporal mobile traffic data. Unlike our setting, mobile traffic data has certain unique properties such as ‘recurring’ patterns that are exploited in SpectraGAN for effective data generation.

Works on multivariate time-series synthesis in general are related to our problem involves generating time-series data for multiple radio network KPIs. Existing work [10, 30, 31] however, targets very different problems from ours. For instance, in [30], an unconditional GAN based multivariate time-series synthesis model is introduced to generate data for resource utilization measurement of CDN caches whereas we target a conditional data generation problem. As another example, Chen et al. [10] focus on mitigating the severe class imbalance in the data for predicting rare events (e.g., solar flares).

Among these works, DoppelGANger (DG) [31] is a more closely related work that is aimed at unconditional GAN based generation of multivariate time-series data for networks and systems (e.g., Wikipedia article views over time, network monitoring data over time, resource usage in compute clusters). In §B, we provide a detailed discussion on the suitability of DG design to our drive testing data generation problem, along with its limitations with respect to GenDT. In our evaluations, we compare GenDT with the original DG design and an optimized variant, as elaborated in §5.2.

2.2 Representative Radio Network KPIs

Drive testing involves measuring a number of different radio network KPIs. Here we outline a representative set of key LTE radio network KPIs [53] that we target in GenDT.

**Reference Signal Received Power (RSRP)** is the average power received from a single reference signal. It typically ranges between -44 dBm (good) and -140 dBm (bad). RSRP is related to another KPI called Received Signal Strength Indicator (RSSI), which represents the total received power from the serving cell, co-channel cells and other sources of noise:

\[
\text{RSRP}(\text{dBm}) = \text{RSSI}(\text{dBm}) - 10 \times \log(12N_{RB})
\]

where \(N_{RB}\) is the number of resource blocks.

**Reference Signal Received Quality (RSRQ)** indicates the quality of the received signal and typically ranges from -19.5dB (bad) to -3dB (good). RSRQ is related to the above mentioned KPIs, as follows:

\[
\text{RSRQ}(\text{dB}) = N_{RB} \left( \frac{\text{RSRP}(\text{dBm})}{\text{RSSI}(\text{dBm})} \right)
\]

Based on the above, given any two of RSRP, RSRQ and RSSI, we can obtain the third. We focus on RSRP and RSRQ given their central role in influencing handover decisions for mobility management [51].

**Signal to Interference plus Noise Ratio (SINR)** is a key determinant of the received data rate. It is related to the transmit power, pathloss and interference.

**Channel Quality Indicator (CQI)** is a key KPI that is related to SINR, and is used for downlink resource scheduling and link adaptation, including the choice of modulation and coding scheme [12]. It takes discrete values between 1 and 15.

Although the above set of KPIs are a subset of KPIs considered for drive testing measurements [1], they are an essential subset as discussed above and so are sufficient to highlight the potential of the proposed GenDT approach. We leave the extension of GenDT to cover additional KPIs for future work.

2.3 Measurement and Context Data

For our analysis and evaluation, we use two real-world mobile network measurement datasets from two different countries, both obtained through a drive testing like process. We also compile corresponding network and environment context data from public sources.

2.3.1 **Dataset A.** We collected this dataset through first-hand measurements using Nemo Handy [26], a commercial drive testing tool, mostly in and around a city center area in country A. The Nemo Handy tool allows measurement of a comprehensive set of radio network KPIs at a consistent and fine time granularity of 1s. These measurements were obtained using a custom Samsung S20 device.
Table 1: Statistics of Dataset A for different scenarios.

<table>
<thead>
<tr>
<th>City Driving 1</th>
<th>City Driving 2</th>
<th>Highway 1</th>
<th>Highway 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Granularity</td>
<td>3.8</td>
<td>3.5</td>
<td>2.1</td>
</tr>
<tr>
<td>Avg. Velocity (m/s)</td>
<td>9.1</td>
<td>9.5</td>
<td>22.5</td>
</tr>
<tr>
<td>Avg. Duration at each Serving Cell (s)</td>
<td>31.4</td>
<td>27.3</td>
<td>22.0</td>
</tr>
<tr>
<td>Avg. RSRP (dBm)</td>
<td>-84.6</td>
<td>-85.0</td>
<td>-86.5</td>
</tr>
<tr>
<td>Std. RSRP (dBm)</td>
<td>9.6</td>
<td>7.1</td>
<td>10.5</td>
</tr>
<tr>
<td>Std. RSRQ (dB)</td>
<td>0.95</td>
<td>0.83</td>
<td>1.11</td>
</tr>
<tr>
<td>Avg. RSRQ (dB)</td>
<td>-7.3</td>
<td>-10.0</td>
<td>-8.7</td>
</tr>
<tr>
<td>Std. RSRQ (dB)</td>
<td>0.10</td>
<td>2.6</td>
<td>2.2</td>
</tr>
<tr>
<td>EDR RSRQ (dB)</td>
<td>0.36</td>
<td>0.41</td>
<td>0.38</td>
</tr>
<tr>
<td>Sample Size</td>
<td>2.1 × 10^7</td>
<td>2.3 × 10^7</td>
<td>5.9 × 10^7</td>
</tr>
</tbody>
</table>

Table 2: Statistics of Dataset B for different scenarios.

with Nemo Handy installed. There are other studies in the literature that have reported measurements obtained using this tool (e.g., [16, 47]). Table 1 provides a summary of this dataset.

2.3.2 Dataset B. This is a publicly available measurement dataset provided by the authors of [55, 56]. It covers a much wider geographical region than Dataset A. Specifically it is centered around the city of Dortmund in Germany and spans to nearby cities, including Bonn, Cologne and Hamm. It consists of measurements taken at campus, suburban, urban, and highway areas. This dataset was collected using a custom Android app [23] accessing the Telephony API [25] on commodity Android phones. It is known that with this API the measurement granularity is coarser around 5s and varies across chipsets. We focus on measurements collected using One Plus 8 devices as they cover the largest area. Table 2 provides a summary of this dataset. Here ROC refers to “rate of change”, i.e., the first-order derivative of the corresponding KPI.

2.3.3 Network Context: Cell Information. For each measurement location in the above two datasets, we treat the corresponding cell deployment information as the network context. Specifically, we consider the cell site location, estimated transmit power and cell orientation for each cell within a certain range around the device measurement location, as such cells are seen as potential serving cells. See Figure 3 for an illustration. We discuss the setting of this range around the device in the next section. We obtain the cell site location and configuration information from CellMapper [8], a non-profit crowd sourced cell information dataset.

2.3.4 Environment Context. The radio network KPI data characteristics are not only dependent on the network context described above but also on the environment around the device (terrain, obstacles, etc.). So we additionally consider the environment context, which in our case is represented by a set of 26 attributes (see Table 11 in Appendix A.1). These attributes are obtained from public sources and broadly fall into two categories: (1) land use type from

\[\text{Distance to Serving Cell}\] From Figure 16, we observe that the distribution of distance to primary serving cell are as per intuition – slow mobility (e.g., walking) or inner city (e.g., city center cases in Dataset B) have serving cells that are relatively close. Yet, there is considerable degree of variation in distance to serving cells within and across scenarios. A direct implication of this observation for our purpose of generating radio KPI time series data conditioned
As stated at the outset, we aim at faithfully mimicking drive testing through a data generation model to reduce the need for collection of field measurements. This goal translates to generating time-series data for different radio KPIs corresponding to an input drive testing trajectory, as would be the case with traditional drive testing. Figure 5 illustrates the problem we target and our proposed approach to resolve it through the GenDT model. Note that this schematic depicts the operational process once the GenDT model is trained; we discuss the model training aspect shortly. The process starts with providing an input trajectory (Figure 5: Input), which is a timestamped sequence of locations for the user device (UE) format). Then the network context (Figure 5: 2 as described in §2.3.3) and environment context (Figure 5: 3 as described in §2.3.4) corresponding to each timestamp in the input trajectory are consolidated into a series of context snapshots (Figure 5: 4), each including the user device (UE) location at the snapshot’s timestamp. This context annotated trajectory together with noise makes up the input to the trained GenDT data generation model ((Figure 5: 5), which outputs time-series data for different radio KPIs. Here context (Figure 5: 5) serves as conditioning input to the generator, whereas noise represents factors unaccounted for in the context for the data generation process such as cell load as well as statistical variation. In the training phase that precedes the generation/operational phase outlined above, the model is trained using a small set of real drive testing measurement data. The training follows the same pipeline as in Figure 5 except that the model is updated based on the divergence between real and generated data.

Resolving the above outlined problem for high-fidelity and generalizable radio KPI time-series data synthesis with minimal training data is a significant challenge. A number of issues have to be addressed as part of tackling this challenge: (1) context input varies over time with device location; (2) drive testing trajectories can be arbitrarily long and complex spanning multiple different scenarios (city center, highway, etc.); (3) considering the inherent stochasticity of the radio KPI data, generated KPI data should match the distribution of the real data; (4) all of the above needs to be achieved with minimal amount of training data to achieve our intended goal of efficient drive testing.

4 GENDT

4.1 Problem Statement

As stated at the outset, we aim at faithfully mimicking drive testing through a data generation model to reduce the need for collection of field measurements. This goal translates to generating time-series data for different radio KPIs corresponding to an input drive testing trajectory, as would be the case with traditional drive testing. Figure 5 illustrates the problem we target and our proposed approach to resolve it through the GenDT model. Note that this schematic depicts the operational process once the GenDT model is trained; we discuss the model training aspect shortly. The process starts with providing an input trajectory (Figure 5: Input), which is a timestamped sequence of locations for the user device (referred to in [Latitude, Longitude] format). Then the network context (Figure 5: 3 as described in §2.3.3) and environment context (Figure 5: 4 as described in §2.3.4) corresponding to each timestamp in the input trajectory are consolidated into a series of context snapshots (Figure 5: 5), each including the user device (UE) location at the snapshot’s timestamp. This context annotated trajectory together with noise makes up the input to the trained GenDT data generation model ((Figure 5: 6), which outputs time-series data for different radio KPIs. Here context (Figure 5: 6) serves as conditioning input to the generator, whereas noise represents factors unaccounted for in the context for the data generation process such as cell load as well as statistical variation. In the training phase that precedes the generation/operational phase outlined above, the model is trained using a small set of real drive testing measurement data. The training follows the same pipeline as in Figure 5 except that the model is updated based on the divergence between real and generated data.

Resolving the above outlined problem for high-fidelity and generalizable radio KPI time-series data synthesis with minimal training data is a significant challenge. A number of issues have to be addressed as part of tackling this challenge: (1) context input varies over time with device location; (2) drive testing trajectories can be arbitrarily long and complex spanning multiple different scenarios (city center, highway, etc.); (3) considering the inherent stochasticity of the radio KPI data, generated KPI data should match the distribution of the real data; (4) all of the above needs to be achieved with minimal amount of training data to achieve our intended goal of efficient drive testing.

4.2 Overview of Proposed Solution

Motivated by the above, we propose an original conditional deep generative model, GenDT, that addresses the aforementioned challenges and issues. Specifically, the issue (1) is addressed via a tailored GNN based time-series model, together with customized data processing, training method, and hyper-parameter tuning, as elaborated in this and the next subsection. Broadly speaking, the generation of time series data for different radio KPIs in GenDT is done in two steps, as elaborated in §4.3.1. The first step generation is conditioned on the network context (cell information). Then the environment effect is added on through a residual generator component (§4.3.2). We address (2) through batch training and generation (§4.3.3) that enables effective long time-series generation and training efficiency. We tackle (3) through a combination of mechanisms: noise in the input, adding stochastic layers in the different neural network components of the generator (§4.3.4) and adversarial training (à la GANs). To address issue (4), we leverage the learned parameters of the residual generator model, whose variation offers insight on the extent to which additional training data will help improve model fidelity.

Formally, the target output of our generation model is to generate time-series data for \( N_{ch} \) different radio KPIs (e.g., RSRP, RSRQ) over a given time period \( T: x_{1:T,i}^{c} = [x_{i,1}^{c}, \ldots, x_{i,T}^{c}], i \in [1, \ldots, N_{ch}] \). Here \( N_{ch} \) can be viewed as different ‘channels’ of the model output. The generated series \( x_{1:T,i}^{c} \) should exhibit high fidelity with respect to the corresponding true series: \( x_{1:T,i} = [x_{1}, \ldots, x_{T}], i \in [1, \ldots, N_{ch}] \). The whole multivariate time series data \( x_{1:T,i}^{c} \) can be generated in one shot but at the risk of compromising fidelity, especially when \( T \) is long. So we employ generation in smaller batches, each of length \( L \). As such, the generated series can be seen as a sequence of \( \left\lfloor \frac{T}{L} \right\rfloor \) batches.

The above data generation is conditioned on context \( c \). As such, \( c \) serves as an input to the model. As noted earlier, overall context \( c \) is made up of network and environment context. The network context in each batch \( b \) is dependent on the set of potential serving (visible) cells over the course of the batch’s duration (i.e., \( L \)). As per the analysis in §3, we consider cells within a certain distance \( d_{b} \) of the user location as the relevant network context. The value of \( d_{b} \) is...
Figure 6: Schematic of GenDT generator architecture, labeled with input and output dimensions for each component.

Figure 7: Illustration of ResGen network architecture and the generation of distribution parameters (FC: Fully Connected Layer, LeakyReLU: Leaky Rectified Linear Unit).

4.3 Detailed Model Design

4.3.1 Generator. As illustrated in Figure 6, our conditional neural sampler $p_θ$ has three main neural network components: 1) a GNN node network $G_θ^n$ that does convolution operation over network context (cell level information) time series; 2) an aggregation network $G_θ^a$ to process the temporal graph after the convolution; 3) a residual generator (ResGen) $G_θ^r$ that accounts for the environmental effects to model the ‘residual’ and adds it to the output of the aggregation network. All these three components operate at the batch level.

- $G_θ^n: \mathbb{R}^L \times (N_c + N_b) \times 1 \rightarrow \mathbb{R}^{L \times H \times N_b}$, where $N_{b}$ is the dimension of the input noise and $N_{c}$ is number of target KPIs. We use a multi-channel LSTM for generation of multiple KPI time series, all together. To make sure the GNN node LSTM network does not have a bottleneck effect, we set the hidden dimension size $H >> N_c$. Based on our empirical insights, we set $H = 100$, which we find to achieve the right balance between convergence efficiency and training performance. The additive input noise $z_0$ on the GNN-node network is not for introducing statistical variation but rather to help the model learn a de-noise behavior and avoid over-fitting [60]; this eases the training process and makes it robust.

- $G_θ^a: \mathbb{R}^{L \times H \times N_a} \rightarrow \mathbb{R}^{L \times 1 \times N_a}$. The input $h_{avg}$ to $G_θ^a$ is the high dimensional representation of the input graph. We take the average of the hidden representation of all cells as the input graph level representation, i.e., $h_{avg} = \frac{1}{N_c} \sum_{i=1}^{N_c} h_i$. The aggregation network has the similar structure as the GNN-node network. Both are based on LSTM and only differ in dimensions and number of input-output channels.

- $G_θ^r: \mathbb{R}^L \times (N_b + N_s) \rightarrow \mathbb{R}^{L \times 1 \times N_s}$, where $N_s$ is the dimension of the input noise. The $N_s$ environment context attributes are concatenated with the noise as input. The output of $G_θ^r$ has the same dimensions as $G_θ^n$ as they are added together to produce the generator’s final output. This component is elaborated further in §4.3.2.

4.3.2 ResGen. The network context driving the first two components GenDT generator architecture (Figure 6) helps model the effect of cell deployment and configuration on radio network KPI dynamics but that by itself is insufficient. Environment (terrain, obstacles, etc.) has an equally important effect on radio KPI behavior. Crucially, the complexity of the environment determines the cost of drive testing (required number of measurements) in practice, as previously noted in [57]. So we design the third component of GenDT generator $G_θ^r$ termed ResGen (Figure 7) to model...
the environment effect, and crucially also to get cues on the need for additional training data. ResGen complements the other two components in that its output (referred to as ‘residual’) is added to the output of the aggregation network to generate the final output time-series data for the target radio KPIs.

In ResGen, we model the residual for each timestamp with a parametric Gaussian distribution, conditioned on the environment context \( \epsilon_{\text{env}, t} \in \mathbb{R}^{1 \times N_y} \), noise \( z_1 \) and the recent values of radio KPI time-series data. The latter is real (generated) data during training (generation) phase of GenDT, and importantly makes ResGen an auto-regressive model with temporal pattern learning capability [14]. The noise input is sampled from a standard Gaussian distribution to represent the unaccounted contextual information and also for capturing statistical variation. We observe that simply using a noise input is insufficient to model the required variation on the output. Hence, we use a dropout layer [21] before the final layer of ResGen. Once trained, we sample the Gaussian distribution \( N(\mu_{\theta, t}, \sigma_{\theta, t}) \) to obtain the residual, where mean \( \mu_{\theta} \) and standard deviation \( \sigma_{\theta} \) are the learned distribution parameters.

Characteristics of the parameters \( [\mu_{\theta}, \sigma_{\theta}] \) can be leveraged to guide the training process. They allow distinguishing between ‘model uncertainty’ and ‘data uncertainty’. If the parameters \( [\mu_{\theta}, \sigma_{\theta}] \) themselves exhibit a high degree of variation during the training process, then that suggests model uncertainty and the need for more training data to stabilize these parameters. On the other hand, if the \( \sigma_{\theta} \) has a stable but large value then that indicates that the underlying data being modeled itself has a high degree of variation and so model is not the limitation. Our target is to reduce the model uncertainty using minimal amount of training data and accordingly we leverage the above insight to that end.

4.3.3 Batch Training and Generation. In GenDT, instead of handling the whole radio KPI time series from training input or target output all in one shot, we do that in small steps called batches. We employ such a batch based training and generation approach for the following reasons:

- **Long series generation**: The time series of radio KPI measurements with drive testing can be quite long. We thus need to be able to generate similarly long time series but doing that in one shot risks fidelity. It is known that learning to generate long time series data at high fidelity with recurrent neural networks (RNNs), including its widely used LSTM variant, is hard [31]. So we turn the learning task of synthesizing arbitrary length series into two sub-tasks that are easier to handle with a LSTM-based architecture: 1) learning short-term temporal correlations within each batch; 2) capturing long-term temporal correlations across batches.

- **Training efficiency**: With conditional generative models, operating at the batch level has a weight-sharing effect among batches and so enhances learning efficiency.

- **Computational efficiency**: With batch training and generation, we only need to consider context input at the batch level, which makes the processing of input more efficient compared to treating the whole time series at once.

Concretely, we view the whole training input and target output time series for each KPI as a sequence of batches, each of length \( L \):

\[
x_{1:T} \longrightarrow \{ x_{1:L}; x_{1+L}; x_{1+2L}; \ldots; x_{1+\frac{T}{L}L}; \ldots \}
\]

Figure 8: (a) Training with overlapping batches; (b) Stochastic layers of LSTM in GNN-Node Network and Aggregation Network components of GenDT.

where \( \Delta t \) is the step length of the sliding window, which allows different forms of batching. During the training phase, we allow the batches to be overlapping (as illustrated in Figure 8a) to additionally optimize the training efficiency. On the other hand, for generation, we use non-overlapping batches (i.e., \( \Delta t = L \)) to ensure that there are no smoothing artifacts introduced in the output and that the desired statistical variation is not compromised.

4.3.4 Stochastic Layers. The inherently and highly stochastic nature of radio KPI data (even at the same location) needs special attention to model this characteristic, especially in the generator part driven by the dynamic network context. We find that straightforward approaches to introducing noise such as injecting noise directly in the input or using a FiLM layer [42] are ineffective in our setting. So we employ a variant of the Stochastic RNN (SRNN) method [20] to efficiently propagate uncertainty in a latent state representation with RNNs. Specifically, we use stochastic layers in the LSTM structures of both GNN-node and aggregation networks. As illustrated in Figure 8b, we introduce noise to memories (\( c_t \)) and hidden states (\( h_t \)), where the noise is added just before each iteration. The noise modulated versions of hidden state and memory are respectively \( h'_t = f_h(h_t, n_{t, h}, a_{h}) \) and \( c'_t = f_c(c_t, n_{t, c}, a_{c}) \), where \( f_h \) is a function to control the intensity of noise input, and the noise of intensity of noise added to hidden state \( h \) and \( c \) are controlled by \( a_h \) and \( a_c \) respectively. We assume that the noise has an uniform distribution between \([0, h_t]\) and \([0, c_t]\), where \( h_t \) and \( c_t \) represent the average value of \( h_t \) and \( c_t \) of all hidden dimensions, so that the noise adapts to the hidden state values. Unlike the variational inference based learning used in [20], we use an adversarial training method with a discriminator. See Appendix A.2 for further details.

4.3.5 Training. Following the standard GAN formulations [13], we train the model by minimizing Jensen-Shannon divergence, \( D_{JS} [p_T || p_0] \), and with the aid of discriminator as in the GAN framework. We denote such discriminator as \( D \) due to their role as density ratio estimators [59]. Specifically, for given training input measurement data time series and context batch \((x_t, c_t)\), the corresponding adversarial loss between the data \( p_T(x_t, c_t) \) and the model \( p_0(x_t, c_t) \) distribution is defined as:

\[
\mathcal{L}_{JS}^R(p_T, p_0) = E_{p_T}[\log D(x_t, c_t)] + E_{p_0}[\log (1 - R(x_t, c_t))].
\]

In our case, we consider one discriminator, named as \( R_0 \), the context input into discriminator is the high dimensional representation of \( c \), which is \( h_{\text{aux}} \). The discriminator is a single layer LSTM network.

We additionally use the standard mean squared error loss:

\[\text{Here we show for one radio KPI (channel) case but the same applies for all channels.}\]
\[ L^M(x, x') = \frac{1}{T} \sum_{t=1}^{L} (x_t - x'_t)^2 \]

Since the batch length \( L \) is constant during training, this loss has an equivalent effect to using \( L^2 \) loss.

Overall, together with the adversarial (GAN) loss, the loss function to fit \( \theta \) is:

\[ L = L^M + \lambda L_{FS} \]

where the \( \lambda \) is a weight to balance the effect of adversarial loss, which in our case is set as \( \lambda = 0.1 \) by default. Appendix A.3 elaborates on the setting of hyper-parameters.

## 5 EVALUATION METHODOLOGY

Broadly speaking, we evaluate GenDT in two ways. First, we assess the fidelity of the GenDT generated radio KPI time series data with respect to real measurement data using multiple different metrics described in §5.1 and in comparison with various baseline approaches outlined in §5.2. Second, we evaluate GenDT through two different downstream use cases and show that GenDT generated data is a dependable substitute for real drive testing measurement data to support such use cases.

### 5.1 Metrics

Mean Absolute Error (MAE) for any given KPI between its real measurement data time series (\( x : \{x_1, x_2, \ldots, x_T\} \)) and generated time series (\( y : \{y_1, y_2, \ldots, y_T\} \)) is calculated as: \( MAE = \frac{1}{T} \sum_{t=1}^{T} |y_t - x_t|/n \). As such, it is a natural measure for evaluating fidelity of GenDT and alternative approaches.

Dynamic Time Warping (DTW) [7] is an alternative metric to MAE for assessing the similarity between two time series (real and generated in our setting). The main feature of this distance measure is that it allows to recognize similar shapes between two time-series signals, even if they need signal transformations such as shifting and/or scaling. As such, it provides a more robust similarity measure. Events like accessing a specific cell or going around the same location have a similar effect on the temporal pattern of KPIs across different measurement trajectories, though with slight time shift due to differences in user device path and velocity each time. DTW is better at identifying such similarity, as the other distance metrics are too sensitive to temporal shifts. Hence, the DTW is very useful in capturing real world performance, especially when used in conjunction with MAE, as we do.

Histogram Wasserstein Distance (HWD). Besides having the generated time series of different radio KPIs matching with their corresponding ground-truth time series (as quantified by the MAE and DTW metrics), we would also want the generated data for any target KPI to have the same distribution (histogram) as the real data. Rather than limiting the comparison of histograms of real and generated data to just visualization, we quantify the similarity between these histograms by computing their Wasserstein Distance (WD) [49] and call this metric as the Histogram Wasserstein Distance (HWD).

Measurement Efficiency. While fidelity of the generated data along different aspects as quantified by the above metrics is important, the required amount of training data to achieve that fidelity is equally important. Lower the training data needed the better as it demonstrates the cost reduction and efficiency improvement that GenDT can provide, aligned with the motivation behind its design. As different scenarios involve different movement speeds, lengths of trajectories included in the training data in terms of distance are not representative. We therefore factor in speed in trajectories and consider data used for training in terms of time (~distance/speed). Specifically, we evaluate the percentage of the available data in a dataset that is used for training as our measurement efficiency metric.

### 5.2 Baselines

We are unaware of any other work in the literature adopting a generative modeling approach like ours for efficient mobile network drive testing. So we consider a range of alternative approaches from other domains as baselines.

Fit Distribution and Sample (FDAs). FDAs [15, 40] is another simple minded baseline that focuses on modeling the distribution (histogram) of the data for any given radio KPI. Specifically, it fits a distribution based on the real KPI data (ignoring the time dimension) using maximum likelihood estimation, and samples from it afterwards to generate the data for that KPI. While this baseline can be effective with respect to the HWD metric, it can be quite poor in terms of the other fidelity metrics as it does not consider relationship with context nor the temporal relationships in the data.

Multilayer Perceptron (MLP) is a simple minded baseline that infers the data for each radio KPI independently at each time step through regression over the context input. Clearly, this baseline does not account for the temporal relationships within the real KPI time series data. Moreover, as it focuses solely on the relationship between context and KPI data, it does not model stochasticity of the latter either.

LSTM-GNN [58], a variant of [24], is a state-of-the-art model architecture for GNN based time-series prediction. We use it as a baseline as an alternative approach especially with respect to the first two neural network components of GenDT generator (§4.3.1), and highlight the benefit of GenDT’s handling of dynamic context input, batch based generation and use of stochastic layers.

DoppelGANger (DG) [31] and Variant. As mentioned in §2.1, DG is a state-of-the-art multivariate time series data generation model and so is a natural baseline approach to compare with. The original DG model (depicted in Figure 17a) generates the context in its first stage. In our problem setting, however, this context data is readily accessible to the operator and can be directly used without having to learn to generate it. So we additionally consider an optimized variant of DG called ‘Real Context DG’ in which we bypass the context generation stage and directly input real context to the second stage time-series data generator in DG, as depicted in Figure 17b.

## 6 EVALUATION RESULTS

Here in §6.1 we first evaluate GenDT on the fidelity metrics from §5.1 and benchmark it against the baselines outlined in §5.2. Then we demonstrate that the uncertainty measure within GenDT can be used to optimize measurement efficiency (§6.2). In §6.3, we demonstrate the value of GenDT-generated data for two downstream use cases. Finally, we carry out an ablation study of GenDT to examine the effect of its underlying design choices (§C.1).

### 6.1 Fidelity and Generalization

Setup. To assess the generalization capability of GenDT to new unseen trajectories, we split each of our datasets into two non-overlapping parts: training and testing. We further make sure to
worse than MLP and LSTM-GNN. This is because it is limited by focusing on generation of time series for RSRP, RSRQ, SINR and CQI KPIs. We notice that relative to significant improvements seen across all scenarios is reported in Table 6, also considering the versatility of GenDT (and other baselines) in different scenarios separately to highlight variation at all. We attribute this to two reasons: (1) LSTM-GNN is a prediction model not a generative one; and (2) it does not have and insufficient mechanisms to capture stochasticity, latter clearly reflected in the poor HWD performance relative to GenDT. The shortcoming of real context DG relative to GenDT with respect to the former context handling issue and the effectiveness of GNN structure in GenDT to that end is illustrated in the generated RSRP series with these methods in Figure 18 (in Appendix C).

As before, we first consider RSRP and report performance at the per-scenario level in Table 5. Again, we observe that the big performance improvements seen with GenDT above continue to hold with the exception of CQI performance, where benefits are somewhat marginal. We attribute this to the fact that, unlike other KPIs, CQI generation is a classification problem involving a choice of one among discrete values from 1 to 15. Overall, we observe that the overlapping batches based training on top of batch generation and handling time-varying relevant context input plays a key role in the superior performance of GenDT, so does the SRNN structure in the generator (§4.3.4) which helps in better modeling the data distribution.

### Table 4: Average performance of GenDT and baselines across all scenarios in Dataset A for RSRP, RSRQ, SINR, and CQI time series generation.

<table>
<thead>
<tr>
<th>Method</th>
<th>RSRP</th>
<th>RSRQ</th>
<th>SINR</th>
<th>CQI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td>DTW</td>
<td>HWD</td>
<td></td>
</tr>
<tr>
<td>GenDT</td>
<td>10.1</td>
<td>15.8</td>
<td>19.5</td>
<td>8.5</td>
</tr>
<tr>
<td>FDaS</td>
<td>10.2</td>
<td>16.7</td>
<td>19.1</td>
<td>9.0</td>
</tr>
<tr>
<td>MLP</td>
<td>9.5</td>
<td>15.6</td>
<td>12.7</td>
<td>10.2</td>
</tr>
<tr>
<td>LSTM-GNN</td>
<td>9.2</td>
<td>16.0</td>
<td>16.0</td>
<td>9.0</td>
</tr>
<tr>
<td>Orig. DG</td>
<td>9.2</td>
<td>16.0</td>
<td>16.0</td>
<td>9.0</td>
</tr>
<tr>
<td>Real Cont. DG</td>
<td>9.2</td>
<td>16.0</td>
<td>16.0</td>
<td>9.0</td>
</tr>
</tbody>
</table>

### Table 5: Generated RSRP time series fidelity with GenDT and baselines for different scenarios in Dataset B.

<table>
<thead>
<tr>
<th>Method</th>
<th>RSRP</th>
<th>RSRQ</th>
<th>SINR</th>
<th>CQI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td>DTW</td>
<td>HWD</td>
<td></td>
</tr>
<tr>
<td>GenDT</td>
<td>10.1</td>
<td>15.8</td>
<td>19.5</td>
<td>8.5</td>
</tr>
<tr>
<td>FDaS</td>
<td>10.2</td>
<td>16.7</td>
<td>19.1</td>
<td>9.0</td>
</tr>
<tr>
<td>MLP</td>
<td>9.5</td>
<td>15.6</td>
<td>12.7</td>
<td>10.2</td>
</tr>
<tr>
<td>LSTM-GNN</td>
<td>9.2</td>
<td>16.0</td>
<td>16.0</td>
<td>9.0</td>
</tr>
<tr>
<td>Orig. DG</td>
<td>9.2</td>
<td>16.0</td>
<td>16.0</td>
<td>9.0</td>
</tr>
<tr>
<td>Real Cont. DG</td>
<td>9.2</td>
<td>16.0</td>
<td>16.0</td>
<td>9.0</td>
</tr>
</tbody>
</table>

### Table 3: Generated RSRP time series fidelity with GenDT and baselines for different datasets.

<table>
<thead>
<tr>
<th>Method</th>
<th>RSRP</th>
<th>RSRQ</th>
<th>SINR</th>
<th>CQI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td>DTW</td>
<td>HWD</td>
<td></td>
</tr>
<tr>
<td>GenDT</td>
<td>10.1</td>
<td>15.8</td>
<td>19.5</td>
<td>8.5</td>
</tr>
<tr>
<td>FDaS</td>
<td>10.2</td>
<td>16.7</td>
<td>19.1</td>
<td>9.0</td>
</tr>
<tr>
<td>MLP</td>
<td>9.5</td>
<td>15.6</td>
<td>12.7</td>
<td>10.2</td>
</tr>
<tr>
<td>LSTM-GNN</td>
<td>9.2</td>
<td>16.0</td>
<td>16.0</td>
<td>9.0</td>
</tr>
<tr>
<td>Orig. DG</td>
<td>9.2</td>
<td>16.0</td>
<td>16.0</td>
<td>9.0</td>
</tr>
<tr>
<td>Real Cont. DG</td>
<td>9.2</td>
<td>16.0</td>
<td>16.0</td>
<td>9.0</td>
</tr>
</tbody>
</table>

(DG) is free from this limitation and better reflects the performance of data generator in DG. Still, it yields only intermediate performance due to its inability to handle dynamic network context input and insufficient mechanisms to capture stochasticity, latter clearly reflected in the poor HWD performance relative to GenDT. The shortcoming of real context DG relative to GenDT with respect to the former context handling issue and the effectiveness of GNN structure in GenDT to that end is illustrated in the generated RSRP series with these methods in Figure 18 (in Appendix C).

Considering all the considered KPIs including RSRP, the average performance across all scenarios is reported in Table 4. We observe that the big performance improvements seen with GenDT above continue to hold with the exception of CQI performance, where benefits are somewhat marginal. We attribute this to the fact that, unlike other KPIs, CQI generation is a classification problem involving a choice of one among discrete values from 1 to 15. Overall, we observe that the overlapping batches based training on top of batch generation and handling time-varying relevant context input plays a key role in the superior performance of GenDT, so does the SRNN structure in the generator (§4.3.4) which helps in better modeling the data distribution.

### Dataset B

We now consider Dataset B which consists of longer and more complex movement trajectories over a wider geographical region. This dataset, however, lets us evaluate with respect to generation of time series for only RSRP and RSRQ KPIs as it lacks the other KPIs.

As before, we first consider RSRP and report performance at the per-scenario level in Table 5. Again, we observe that GenDT generally yields the best performance and FDaS doing marginally better in terms of HWD as expected. The average performance across all scenarios is reported in Table 6, also considering the RSRP KPI. We notice that relative to significant improvements seen with GenDT in the case of RSRP, gains for RSRQ are less striking. We find that this is because the RSRQ values in the test scenarios are fairly stable and also vary in a much smaller range than RSRP, thereby limiting the room for improvement.
6.1.3 Long and Complex Scenarios. We now consider a long continuous trajectory lasting 2230s (~40mins) as the testing set to evaluate GenDT and baselines for generation of long series of radio KPI data over a complex scenario. The considered trajectory spans three cities in Dataset B (Wuppertal, Hamm, and Koln), including inner city driving and highway driving between them. The total length of the trajectory is about 40km. We make sure that this test trajectory does not overlap nor has significant proximity to trajectories in the training set. Moreover, the training set does not include data from any of the three cities or routes between them.

We first show qualitative results in Figure 9, where we can see that the generated RSRP series with GenDT varies in a range that tightly covers the ground truth (Figure 9a), and also shows good match with ground truth in terms of RSRP data distribution (Figure 9b). Note that the upper/lower bounds shown in Figure 9a are not themselves generated time series with GenDT. Rather, they represent min/max statistics of the generated samples for each time instant. We then summarize the quantitative results in Table 7 that show the overall performance of GenDT compared to baselines. We see that GenDT consistently and significantly outperforms on all metrics for both RSRP and RSRQ. These results particularly highlight the benefit of batch generation given the length of the target trajectory with only Real Context DG coming close to the performance of GenDT. The additional measures in GenDT to aid in effective long series generation (autoregressive ResGen) and beyond (GNN structure and stochastic layers) explain its superior performance. These results also highlight the pitfall of FDaS as data distribution of the complex target trajectory is not captured by the training set and so FDaS yields poor performance even in terms of HWD.

### Table 6: Average performance of GenDT and baselines across all scenarios in Dataset B for RSRP and RSRQ generation.

<table>
<thead>
<tr>
<th>Method</th>
<th>RSRP</th>
<th>RSRQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>GenDT</td>
<td>0.78</td>
<td>1.40</td>
</tr>
<tr>
<td>LSTM-GNN</td>
<td>1.25</td>
<td>2.60</td>
</tr>
<tr>
<td>Orig. DG</td>
<td>1.06</td>
<td>2.40</td>
</tr>
<tr>
<td>ResGen</td>
<td>0.86</td>
<td>1.78</td>
</tr>
<tr>
<td>MLP</td>
<td>1.43</td>
<td>2.53</td>
</tr>
<tr>
<td>Real Cont. DG</td>
<td>0.95</td>
<td>2.00</td>
</tr>
</tbody>
</table>

### Table 7: Overall performance of GenDT and baselines for long and complex trajectory case in Dataset B.

<table>
<thead>
<tr>
<th>Method</th>
<th>RSRP</th>
<th>RSRQ</th>
<th>RSRQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>GenDT</td>
<td>0.78</td>
<td>1.40</td>
<td>0.70</td>
</tr>
<tr>
<td>LSTM-GNN</td>
<td>1.25</td>
<td>2.60</td>
<td>1.60</td>
</tr>
<tr>
<td>Orig. DG</td>
<td>1.06</td>
<td>2.40</td>
<td>1.30</td>
</tr>
<tr>
<td>ResGen</td>
<td>0.86</td>
<td>1.78</td>
<td>0.90</td>
</tr>
<tr>
<td>MLP</td>
<td>1.43</td>
<td>2.53</td>
<td>1.20</td>
</tr>
<tr>
<td>Real Cont. DG</td>
<td>0.95</td>
<td>2.00</td>
<td>0.70</td>
</tr>
</tbody>
</table>

### Table 8: GenDT performance compared with short trajectory generation for long trajectory case in Dataset B.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAE</th>
<th>MAE</th>
<th>MAE</th>
<th>MAE</th>
<th>MAE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>GenDT</td>
<td>0.78</td>
<td>1.40</td>
<td>0.70</td>
<td>1.60</td>
<td>1.30</td>
<td>0.90</td>
</tr>
<tr>
<td>LSTM-GNN</td>
<td>1.25</td>
<td>2.60</td>
<td>1.60</td>
<td>2.53</td>
<td>1.30</td>
<td>0.90</td>
</tr>
<tr>
<td>Orig. DG</td>
<td>1.06</td>
<td>2.40</td>
<td>1.30</td>
<td>2.53</td>
<td>1.20</td>
<td>0.70</td>
</tr>
<tr>
<td>ResGen</td>
<td>0.86</td>
<td>1.78</td>
<td>0.90</td>
<td>1.20</td>
<td>0.70</td>
<td>0.30</td>
</tr>
</tbody>
</table>

6.2 Measurement Efficiency

6.2.1 Model Uncertainty. Data uncertainty is irreducible due to the nature of the data while model uncertainty can be reduced by training on more data and actively selecting new training points [21]. The design of GenDT naturally decouples data and model uncertainty: the data uncertainty is reflected by the actual value of the standard deviation in the learned Gaussian distribution from ResGen while the model uncertainty is determined by the variation of the Gaussian parameters. We use MC dropout [21] to obtain the model uncertainty of GenDT, i.e., the dropout is turned on during generation time to obtain multiple outputs of the model. As the parameters of observation model (mean and standard deviation of the parametric Gaussian) are the (direct) output of the neural network of ResGen, we use the standard deviation of them averaged over time as the model uncertainty. Specifically, the model uncertainty is defined as:

$$U(G_t) = \frac{1}{T} \sum_{t=1}^{T} \text{std}(\sigma_t) + \text{std}(\mu_t)$$

where $T$ is the length of target series and $\text{std}$ is the standard deviation computed by empirical samples with dropout turned on.

6.2.2 Uncertainty Driven Measurement. We evaluate the usefulness of the model uncertainty in an active learning setup on Dataset B, mimicking a real-world uncertainty driven drive test measurement data collection process.

Here we take the long trajectory in §6.1.3 as the testing set (named as $S_t$). We remove the testing set from Dataset B, and split the rest of the data into 23 subsets with no overlap in geographical region between them. We initialize start with just one small subset
Dataset A

We focus on which they engage in drive test measurement data collection. Application layer throughput is a key QoE metric of interest that in turn further use cases that depend on drive testing measurement data, and evaluate GenDT cases that use GenDT-generated RSRP/RSRQ data when applied to QoE (throughput and PER) prediction use case, relative to baselines.

Nemo Handy [26] but also corresponding downlink throughput and PER measurements obtained with iperf3 [18].

For QoE prediction, we leverage a recent work [56] that examined machine learning based prediction of application QoE metrics like throughput based on drive testing based radio KPI measurement data, including RSRP and RSRQ. In particular, we use the MLP based regression model for QoE metric prediction from [56] that uses RSRP, RSRQ, device location, etc. as features. We first confirm that RSRP and RSRQ KPIs are critical for accurate QoE prediction with this model by dropping these two KPIs from the model and observing the significant divergence between real (measured) and predicted throughput (see Figure 12a and second row in Table 9). In contrast, including measured RSRP and RSRQ KPI data greatly improves the throughput prediction (see Figure 12b and first row in Table 9).

To assess the usefulness of GenDT for this use case, we now evaluate the effect of using GenDT-generated RSRP and RSRQ time series data. Quantitative results are shown in Table 9 when using data generated with GenDT and baselines. Note that we use the same fidelity metrics of MAE, DTW and HWD as before, except that these results evaluate the fidelity of predicted throughput and PER time series with respect to their real (measured) series. We observe that GenDT-generated RSRP/RSRQ data yields QoE predictions very similar to that of using corresponding real data, and much superior to using data generated with baselines.

### 6.3 Downstream Use Cases

In this section, we assess how well our GenDT approach can support drive testing use cases. The general idea here is to consider use cases that depend on drive testing measurement data, and evaluate the effect of using GenDT-generated data for those use cases in comparison with using actual measurement data. The choice of the use cases highlighted is constrained by the access to ground-truth radio KPI measurement data to conduct such an evaluation. In the following, we present results for two distinct use cases, each relying on data for a different set of radio KPIs. In Appendix C.2, we discuss further use cases that GenDT can support.

#### 6.3.1 Mobile Service Quality of Experience (QoE) Prediction

User QoE assessment is a key focus of mobile network operators for which they engage in drive test measurement data collection. Application layer throughput is a key QoE metric of interest that in turn depends on lower layer radio KPIs such as RSRP and RSRQ [44, 45]. We also consider Packet Error Rate (PER) as another key QoE metric. We focus on Dataset A that not only includes drive/walk testing based measurement data for multiple radio KPIs collected with

![Figure 11: Assessing selection of new training data based on GenDT uncertainty measure relative to random selection.](image)

We evaluate the GenDT model performance on $S_2$ at each step to assess the benefit with the above uncertainty guided training data selection. As shown in Figure 11, just after two steps (with 10% of the available data used), the performance on $S_2$ no longer shows clear improvement on both DTW and HWD. We omit MAE results for brevity as they are similar to DTW.

As an alternative approach, we perform random selection with the same starting subset of the selected 10 subsets. In other words, we follow the same process as above but at each step randomly selecting the training point to add instead of relying on the uncertainty measure. Results in Figure 11 shows that for the same number of selected subsets, the random selection always shows lower training efficiency compared to the uncertainty based method. Furthermore, the random selection never goes into a case where its performance is better than uncertainty based selection, which means that the uncertainty based method does provide an optimal path to add the most informative data. Overall, with uncertainty guided (random) training data selection, 10% (20%) of the available data (23 subsets) is sufficient to achieve the most generalization that can be evaluated for Dataset B. We could equivalently view this as achieving 90% (80%) measurement efficiency compared to traditional drive testing. Indeed, this efficiency could be higher as the model can generate many more trajectories for which ground truth may not be available.
### 7 DISCUSSION

#### 7.1 Using GenDT in Practice

The typical and intended key user of GenDT is a mobile network operator. They would naturally possess network (cell information) context for their deployed network. Environment context information can be easily accessed through public sources, along the lines of what we did in this work. As illustrated in Figure 14, the operator can build a pre-trained GenDT model using historical drive test measurement data and the corresponding context information. This pretrained model can be readily used for generating multi-KPI time series data like that obtained with traditional drive testing by inputting a context annotated test trajectory (see Figure 5) and noise. This is depicted as the ‘Generation Phase’ in Figure 14 and can be imagined as a desktop tool for the operator to support use cases like in §6.3 and Appendix C.2 that would otherwise require a drive test measurement campaign.

GenDT design has built-in support to allow an initial trained model to be efficiently updated for high-fidelity data generation in new unseen scenarios, shown under the ‘Training Phase’ in Figure 14. Although GenDT design is region agnostic, its use in new regions offers a natural opportunity for potential model retraining. This is best explained through an example. Suppose the new previously unseen target region is a city like New York City. We bootstrap the model retraining step with existing pretrained model – Figure 14:① – along with coarse-grained measurement of target region (e.g., drive test measurements in a randomly selected street within each district of NYC) and corresponding contextual information – Figure 14:②. This can start the cyclical process of further measurement data collection guided by the model uncertainty metric (see §4.3.2 and §6.2) and model retraining – Figure 14:③. The outcome of this model retraining process is an updated version of the model used in the Generation Phase.

#### 7.2 Limitations of GenDT

Here we discuss some limitations of our work, which provide opportunities for future research on efficient drive testing.

- **7.2.1 Region independence:** GenDT’s design has built-in support to allow an initial trained model to be efficiently updated for high-fidelity data generation in new unseen scenarios, shown under the ‘Training Phase’ in Figure 14. Although GenDT design is region agnostic, its use in new regions offers a natural opportunity for potential model retraining. This is best explained through an example. Suppose the new previously unseen target region is a city like New York City. We bootstrap the model retraining step with existing pretrained model – Figure 14:① – along with coarse-grained measurement of target region (e.g., drive test measurements in a randomly selected street within each district of NYC) and corresponding contextual information – Figure 14:②. This can start the cyclical process of further measurement data collection guided by the model uncertainty metric (see §4.3.2 and §6.2) and model retraining – Figure 14:③. The outcome of this model retraining process is an updated version of the model used in the Generation Phase.

- **7.2.2 Limitations of GenDT:** This limits the generalizability of our approach. Extending GenDT to a closed-loop design aided by network side information is a significant issue for future work.

Our analysis of drive test measurement data characteristics has revealed that radio KPIs exhibit significant inherent stochasticity. Similarly, our measurement efficiency evaluation in §6.2 shows that some parts of measurement data carry significantly more information than others that can be exploited to reduce the model uncertainty. Digging deeper into the root causes of both these aspects is an issue for future work.

Our evaluations assessing the fidelity of GenDT generated data do not include comparison with alternative approaches for efficient drive testing, namely MDT or crowdsourced based measurement approaches. Addressing this issue, however, would depend on having access to sufficiently large and representative MDT and crowdsourcing measurement datasets, which is a challenge in itself. As such, it is a topic for future work.

### 8 CONCLUSIONS

We have presented GenDT, a new conditional deep generative model. GenDT is the first data generation method for radio KPI time series data, aimed at reducing the measurement effort with drive testing. It embeds a number of innovative aspects, including the use of stochastic layers on top of a GNN and LSTM based network to process dynamic input network context and to model stochasticity, and batch based training and generation for high fidelity long series generation. We evaluate GenDT with real drive test measurement data from two different countries, covering a wide range of scenarios. Our results show that GenDT generally outperforms a range of baselines, and by a big margin. We also show that GenDT can generate radio KPI time series over long and complex trajectories with high fidelity. Moreover, GenDT is being able to tell apart data uncertainty from model uncertainty. The knowledge of model uncertainty in turn enables selection of the most informative measurement data for model training, which can significantly reduce the measurement overhead – our results show the potential to optimize measurement efficiency by up to 90% while not compromising data fidelity. We also demonstrate that the efficacy of GenDT-generated data to support downstream drive test measurement use cases is comparable to that of real data.
A DATA ANALYSIS AND MODEL DETAILS

A.1 Visualization of Environment Context Attributes

![Attribute Visualization](image-url)

Figure 15: Spatial distribution of 3 selected environment context attributes in DATASET B.

<table>
<thead>
<tr>
<th>Environment Context Attribute</th>
<th>Land Use Type</th>
<th>Poli</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous Urban</td>
<td>Tourism</td>
<td></td>
</tr>
<tr>
<td>High Dense Urban</td>
<td>Cafe</td>
<td></td>
</tr>
<tr>
<td>Medium Dense Urban</td>
<td>Parking</td>
<td></td>
</tr>
<tr>
<td>Low Dense Urban</td>
<td>Restaurant</td>
<td></td>
</tr>
<tr>
<td>Very-Low Dense Urban</td>
<td>Post/Police</td>
<td></td>
</tr>
<tr>
<td>Isolated Structures</td>
<td>Traffic Signal</td>
<td></td>
</tr>
<tr>
<td>Green Urban</td>
<td>Office</td>
<td></td>
</tr>
<tr>
<td>Industrial/Commercial</td>
<td>Public Transport</td>
<td></td>
</tr>
<tr>
<td>Air/Sea Ports</td>
<td>Shop</td>
<td></td>
</tr>
<tr>
<td>Leisure Facilities</td>
<td>Primary Roads</td>
<td></td>
</tr>
<tr>
<td>Barren Lands</td>
<td>Secondary Roads</td>
<td></td>
</tr>
<tr>
<td>Sea</td>
<td>Motorways</td>
<td></td>
</tr>
<tr>
<td>Railway Stations</td>
<td>Tram Stops</td>
<td></td>
</tr>
</tbody>
</table>

Table 11: List of environment context attributes considered.

See examples in Figure 15

A.2 Details of Stochastic Layers

The intensity of noise is controlled by a function. When we add noise, we do not want to change the total value of hidden state of all hidden dimensions, so we have:

\[ h'_i = (h_i + \alpha h_i n_i) \frac{\sum_{i=1}^{H} h_{i'}}{\sum_{i=1}^{H} (h_{i'} + \alpha h_i n_i h_{i'})}, h_t = \{h_{t,1}, \ldots, h_{t,H}\} \]

\[ c'_t = (c_t + \alpha c_t n_t c_t) \frac{\sum_{i=1}^{H} c'_{t,i}}{\sum_{i=1}^{H} (c_{t,i} + \alpha c_t n_t h_{i})}, c_t = \{c_{t,1}, \ldots, c_{t,H}\} \]

Where \( H \) is the dimension of hidden state \( h_t \) and \( c_t \). Using different \( \alpha_h \) and \( \alpha_c \), we can control the relative intensity of noise to the hidden states, and thus control the uncertainty level during training.

We use a different training method compared with [20], where the learning was done by variational inference with an inference network introduced to use the backward-recurrent state to approximate the nonlinear dependence of \( h'_t \) with future observation \( x_{t:T} \) and states \( h_{t:T} \). Instead, in our case effective training of SRNN is realized via adversarial training with a discriminator. A LSTM based discriminator provides extra training signal on top of the L2 norm loss function to make the model converge with nonlinear dependence of \( h'_t \).

A.3 Hyper Parameters

We use single layer LSTM network for both GNN-Node and aggregation networks in the GenDT generator. Hidden layer dimensions for both GNN-Node and aggregation networks are set to 100.

We use 50 for the batch length by default and the default step length is set to 5. Note that, in our experiments, we found that any step length between 1 and 15 gives similar result.
Noise intensity \(a_h, a_c\) are chosen in the range of \([1, 3]\) with the best fit of histogram – larger intensity means more significant variation in output series but needs to be fine-tuned per scenario. In general, \(a_h = a_c = 2\) gives good results for most of the cases.

**B DISCUSSION ON DOPPELGANGER**

As DoppelGANger (DG) seeks to provide a generic data generation architecture across different types of time series data and use cases as well as allow hiding sensitive context (called metadata in DG), it adopts a two stage generation process. In the first stage, context is generated from noise through an unconditional GAN model. The generated context then is used to condition (control) the generation of target time-series network/system data in the second stage via a conditional GAN model.

From the perspective of our mobile network drive testing data generation problem and our proposed GenDT method, DG has four key limitations:

- The DG model architecture cannot handle dynamic network context input. GenDT overcomes this issue through a tailored GNN based generation model.
- There is very limited support for modeling stochasticity in DG via naive direct injection of noise as input to the model. GenDT, on the other hand, comprehensively and effectively deals with this issue through stochastic layers in the model as well as noise input through its residual generator.
- DG adopts a batch generation approach for long time series generation, while GenDT builds on this and optimizes it much further through its autoregressive residual generator and training with overlapping batches.
- DG lacks any mechanism to minimise training data required, whereas GenDT has the built-in residual generator to provide cues on the need for more training data.

It is worth noting that the motivation behind DG (and even SpectraGAN) is to overcome the barrier to accessing real data stemming from commercial sensitivity or privacy concerns, whereas the high cost of measurement data collection with drive testing motivates our design of GenDT.

**Figure 17: Schematic of original DoppelGANger (DG) and its optimized variant.**

**Figure 18: Sample of generated RSRP time series with GenDT and real context DG in DATASET A for the Walk scenario.**

**Table 12: GenDT ablation test on DATASET B considering RSRP and RSRQ.**

<table>
<thead>
<tr>
<th>Method</th>
<th>RSRP MAE</th>
<th>RSRQ MAE</th>
<th>RSRP DTW</th>
<th>RSRQ DTW</th>
<th>RSRP HWD</th>
<th>RSRQ HWD</th>
</tr>
</thead>
<tbody>
<tr>
<td>GenDT</td>
<td>8.10</td>
<td>6.60</td>
<td>7.67</td>
<td>1.7</td>
<td>1.34</td>
<td>1.65</td>
</tr>
<tr>
<td>No ResGen</td>
<td>7.99</td>
<td>6.60</td>
<td>13.7</td>
<td>1.6</td>
<td>2.3</td>
<td>9.7</td>
</tr>
<tr>
<td>No SRNN</td>
<td>11.53</td>
<td>8.89</td>
<td>10.4</td>
<td>2.4</td>
<td>1.99</td>
<td>4.8</td>
</tr>
<tr>
<td>No GAN loss</td>
<td>14.66</td>
<td>12.45</td>
<td>15.3</td>
<td>3.8</td>
<td>3.6</td>
<td>6.9</td>
</tr>
<tr>
<td>No batch</td>
<td>12.9</td>
<td>9.60</td>
<td>10.5</td>
<td>2.6</td>
<td>2.3</td>
<td>3.7</td>
</tr>
</tbody>
</table>

**C ADDITIONAL EVALUATION AND USE CASES**

**C.1 Ablation Study**

Comparison with baselines earlier in §6.1 has already highlighted the limitations of alternative designs. Here we examine the benefit from some of the key design choices underlying GenDT through an ablation test. For this, we consider RSRP and RSRQ KPIs, common to both datasets, and report results with DATASET B.

From the results in Table 12, we see that ResGen plays a critical role in effectively introducing noise to help model stochasticity. Without ResGen, GenDT degrades considerably in terms of the HWD metric. An interesting related observation is that environment context input through ResGen in GenDT does not always help in improving the fidelity on other metrics (MAE, DTW), maybe because KPI dynamics can be high for the same input environment context. In contrast, the use of stochastic layers (SRNN) consistently improves all metrics, including HWD targeted by this mechanism.

Ablation results indicate that the adversarial training (i.e., use of discriminator) is key to GenDT performance. Dropping ‘GAN loss’ from the loss function results in the most performance degradation on all metrics compared to all other design choices. The adversarial network of GenDT is trained to learn to play a similar role as the Inference Network in [20], and thus it is critical for effective model training. As expected, the use of batch generation and training with overlapping batches has a beneficial effect on MAE and DTW fidelity metrics but also improves HWD. The batching related mechanisms are particularly effective when generating data for long trajectories, as previously highlighted in §6.1.3.

**C.2 Further Use Cases**

GenDT is intended to support the use cases that rely on traditional drive testing. We evaluated GenDT for two such cases in §6.3. Here we outline several more example use cases. While GenDT can be readily applied to these use cases listed below without reliance on drive test measurements, evaluating its effectiveness requires access to relevant KPI measurement data as well as ground-truth for use case specific metrics.

- **Video Streaming QoE Prediction.** Depending on the QoE metric, measurement of multiple radio KPIs are required to infer...
the video streaming QoE [33]. GenDT can support this use case along the lines of throughput and PER prediction use case we highlighted in §6.3.

- Cell Load Estimation. In [9, 46], the authors proposed using RSRQ and SINR to estimate the cell load under different scenarios. Since we do not have the ground truth cell load information, we are not able to verify the accuracy of these methods. But these prior works offer a way to infer cell load through drive test measurements, which can be efficiently supported with GenDT.

- Link Bandwidth Prediction. In [64], the authors identify five KPIs has significant correlation with link bandwidth (namely, RSRP, RSRQ, CQI, Handover, and BLER) and proposed a method to infer the link bandwidth with these five KPIs. As we have considered several of these KPIs, it would be straightforward to support this use case with GenDT and evaluate it when real link bandwidth measurement data is accessible.

- Uplink Network Jitter Prediction. KPIs such as RSSI, Cell ID, device location, RSRQ, RSRP and, importantly, the average transport block (TB) size, enable prediction of uplink jitter [50]. This use case can be supported by GenDT via generation of data for these aforementioned KPIs.

What-If Analysis Studies. Over and beyond the type of radio KPI based use cases mentioned above, the context driven design of GenDT naturally lends itself to what-if analysis studies. An example of such a study is to examine the impact of deploying new cells in the operator’s network on radio KPIs, prior to deployment. Another example is to easily study the effect of recent/potential changes in the environment context of a target region (e.g., construction of new highways or big buildings) on radio KPIs without needing to conduct drive test measurements.