GenDT: Mobile Network Drive Testing Made Efficient with Generative Modeling

Citation for published version:
Sun, C, Xu, K, Marina, MK & Benn, H 2022, GenDT: Mobile Network Drive Testing Made Efficient with Generative Modeling. in *Proceedings of The 18th International Conference on emerging Networking EXperiments and Technologies.* Association for Computing Machinery (ACM), pp. 43-58, The 18th International Conference on emerging Networking EXperiments and Technologies, 2022, Rome, Italy, 6/12/22. https://doi.org/10.1145/3555050.3569124

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

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

ACM Reference Format:
Chuanhao Sun[†], Kai Xu[†], Mahesh K. Marina[†], Howard Benn[‡]. 2022. GenDT: Mobile Network Drive Testing Made Efficient with Generative Modeling. In *The 18th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '22), December 6–9, 2022, Roma, Italy*. ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3555050.3569124

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 inaccuracy 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

CoNEXT '22, December 6–9, 2022, Roma, Italy
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9508-3/22/12…$15.00
https://doi.org/10.1145/3555050.3569124

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 given 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:

$$RSRP(dBm) = RSSI(dBm) - 10 \times \log(12^{N_{RB}})$$

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:

$$RSRQ(dB) = N_{RB}\Big(RSRP(dBm) \div RSSI(dBm)\Big)$$

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

	Walk	Bus	Tram
Time Granularity	1s	1s	1s
Avg. Velocity (m/s)	1.4	5.6	11.5
Avg. Duration at each Serving Cell (s)	80.5	49.5	43.42
Avg. RSRP (dBm)	-86.6	-87.3	-85.6
Std. RSRP (dBm)	9.9	10.7	10.0
Avg. RSRQ (dB)	-14.4	-12.9	-13.3
Std. RSRQ (dB)	2.1	2.2	2.1
Measurement Samples (s)	15245	13890	14198

Table 1: Statistics of Dataset A for different scenarios.

	City Driving 1	City Driving 2	Highway 1	Highway 2
Time Granularity	3.8	3.5	2.1	2.3
Avg. Velocity (m/s)	9.1	9.8	26.7	31.1
Avg. Duration at each Serving Cell (s)	31.4	27.3	22.0	22.2
Avg. RSRP (dBm)	-84.6	-85.0	-86.5	-84.1
Std. RSRP (dBm)	8.8	7.1	10.5	10.2
ROC RSRP (dBm)	0.95	0.83	1.11	1.03
Avg. RSRQ (dB)	-9.5	-10.6	-8.7	-8.5
Std. RSRQ (dB)	2.0	2.5	2.2	1.9
ROC RSRQ (dB)	0.36	0.41	0.38	0.31
Sample Num.	2.1×10^4	2.3×10^4	3.9×10^4	4.6×10^4

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[1]. 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].

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

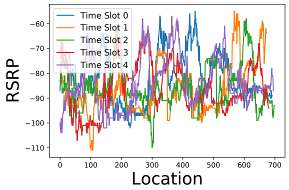


Figure 1: RSRP over the same trajectory with locations aligned.

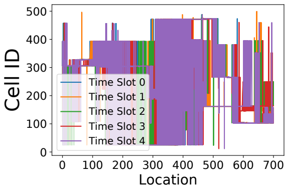


Figure 2: Serving Cell ID changes aligned with the RSRP in Figure 1.

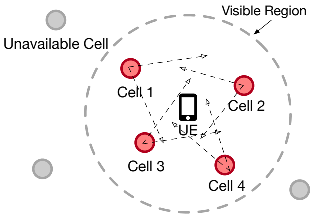


Figure 3: Cells in sight of a device location.

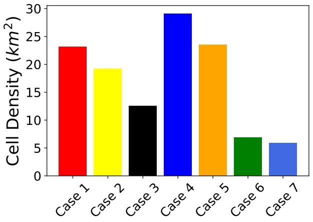


Figure 4: Cells density in Km^2 of different cases.

Copernicus Urban Atlas repository [4]; and (2) points of interest (PoIs) from the OpenStreetMap (OSM) using the Overpass API [41]. Specifically, the value of all these attributes, centered at and within a small radius (set to $500m$ in this paper) of the device location, are taken together as the environment context. For the land use attributes, we use the percentage area of each land use type around the device as its value. For PoI attributes, we use the number of each PoI around the device as its value. Clearly, like the network context, the environment context also changes with the device location.

3 ANALYSIS OF DATA CHARACTERISTICS

Here we present a short analysis of drive test measurement data characteristics pertinent to our model design in §4.

Stochasticity of radio network KPI data. Figure 1 shows five measurements of RSRP time series taken over the same trajectory on the tram in Dataset A around the same time and on the same day. Measurement locations are aligned across the different time series. We see significant variations between the measurements at most locations. This shows that radio network KPI data is far from deterministic, which motivates the need for a generative model capable of modeling this stochasticity as opposed to using prediction/regression models. The high level of variation of a radio KPI (RSRP in this case) at any given location is partly due to serving cell changes. Figure 2 shows the serving cell ID corresponding to the measurement data in Figure 1. We observe that in locations with high degree of RSRP variations, there are also a wide range of serving cells. This suggests that assumption of serving cell at a given location is fixed and known made in prior work (e.g., [3, 57]) does not hold in practice.

Distance to Serving Cell. From Figure 16, we observe that distributions 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 closer. 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

[1] Here arrows indicate the sector and direction of each cell, i.e., each cell covers the direction between two arrows (< 180°). Dashed circle shows the furthest distance of a serving cell from the device. Cells within that range are shown in red circles. Unavailable cells beyond that range are shown as grey circles.

[2] Note that in practice, this information would be directly available to an operator employing our GenDT approach.

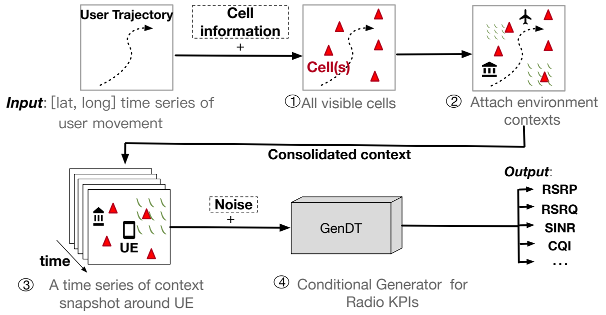

User Trajectory
Cell information
+
Cell(s)
Input: [lat, long] time series of user movement
①All visible cells
② Attach environment contexts
Consolidated context
Noise
+
GenDT
Output:
RSRP
RSRQ
SINR
CQI
…
UE
time
③ A time series of context snapshot around UE
④ Conditional Generator for Radio KPIs

Figure 5: Schematic of the GenDT approach for generation of drive testing data.

on relevant context is that the scope of the cell information context should reflect this wide diversity in order to be effective across different scenarios. For some of the scenarios, we also observe a substantial percentage of cells within almost zero distance from the device location. This reflects a common phenomenon in dense city center areas where users may pass by cells within a few meters distance and there may also be multiple cells that a user device could associate with. Also note that these plots show only the 2D distance between the device and cell site locations.

Figure 4 shows the cell density differences across different scenarios[3]. These results also follow intuition indicating that high mobility scenarios tend to experience lower density of cells compared to inner city and slow mobility cases. Significant diversity across scenarios highlighted in Figure 16 and Figure 4 emphasize the difficulty associated with generating radio KPI data for all scenarios using a single model. Though challenging, addressing this challenge is important to make drive testing efficient considering that trajectories of interest in practice span multiple different scenarios.

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 (represented in [Latitude, Longitude] format). Then the network context (Figure 5:① as described in §2.3.3) and environment context (Figure 5:② as described in §2.3.4) corresponding to each timestamp in the input trajectory are consolidated into a series of context snapshots (Figure 5:③), 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 5:④), which outputs time-series data for different radio KPIs. Here context (Figure 5:③) 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 challenge 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 through 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: $x'_{1:T,i} = [x'_1, \ldots, x'_T]_i \in \mathbb{R}^T, i \in [1, \cdots, N_{ch}]$. Here N_{ch} can be viewed as different 'channels' of the model output. The generated series $x'_{1:T,i}$ should exhibit high fidelity with respect to the corresponding true series: $x_{1:T,i} = [x_1, \ldots, x_T]_i \in \mathbb{R}^T, i \in [1, \cdots, N_{ch}]$. The whole multivariate time series data $x'_{1:T,i}$ 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 $\lfloor \frac{T}{L} \rfloor$ 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_s of the user location as the relevant network context. The value of d_s is

[3] Dataset A: Case 1 – Walking, Case 2 – Bus, Case 3 – Tram; Dataset B: Case 4 – City Center 1, Case 5 – City Center 2, Case 6 – Highway 1, Case 7 – Highway 2.

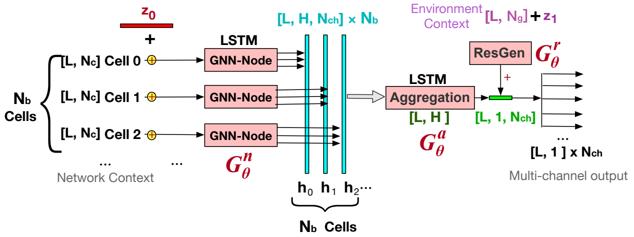


Figure 6: Schematic of GenDT generator architecture, labeled with input and output dimensions for each component.

dependent on the scenario. For example, in Dataset B, we find that serving cells are within 2 km's within the city and within 4 km's on highways. We note that empirically and conservatively setting d_s to a higher value is sufficient for GenDT, although an unnecessarily high value increases the computation time for training.

We use $C_{\text{cell},b}$ (N_b) to denote the set (number) of cells considered for the network context in a particular batch b. Note that by considering the set of potential serving cells instead of a specific one, we account for the fact that serving cells keep changing over time, as observed in §3. For each cell i in the set $C_{\text{cell},b}$, we consider N_c attributes. In this paper, we specifically consider N_c=5 attributes per cell: $c_{\text{cell},i,b} = [\text{lat}_i, \text{lon}_i, p_{\text{max},i}, \text{direction}_i, \text{distance}_{i,t}]$. Here the first four are as previously described in §2.3.3. Specifically, lat_i and lon_i refer to the location of cell i, whereas $p_{\text{max},i}$ and direction_i respectively refer to the max transmit power and direction of cell i. The $\text{distance}_{i,t}$ represents the distance to cell i from the user location in time stamp t. By using this distance attribute, we implicitly account for the time-varying device location. Based on the above, the network context information in batch b is $C_{\text{cell},b} = \{c_{\text{cell},i,b}\}, c_{\text{cell},i,b} \in \mathbb{R}^{L\times N_c}$ and $i = 1, \ldots, N_b$.

Besides the network context, we also consider the environment context as described earlier in §2.3.4. Specifically, we denote the environment context in batch b using $c_{\text{env},b} \in \mathbb{R}^{L\times N_g}$, where N_g (= 26 in our case) represents the number of attributes considered for the environment context. Based on the above, the overall input context to our model for each batch b is $c_b = \{C_{\text{cell},b}, c_{\text{env},b}\}$.

We take a data-driven approach, and accordingly design a parametric model $p_\theta(x_{1:T}|c)$ with parameter θ and fit the model on training data $\mathcal{D}$. Specifically, given training data consisting of ground-truth multi-KPI time series from M drive test measurements, i.e., $\mathcal{D} = [x_1^k, \ldots, x_T^k]_i \in \mathbb{R}^T, i \in [1, \cdots, N_{ch}], k \in [1, .., M]$, and corresponding context data c, we fit θ on $\mathcal{D}$ by finding θ^* that minimizes the divergence D between the data distribution $p_{\mathcal{D}}$ and the model p_θ, i.e., $\theta^* = \arg\min_\theta D(p_{\mathcal{D}}, p_\theta)$. Depending on the specific training methods, different divergence criteria (D) can be considered. Once trained, we can draw samples from the model p_θ for a new target trajectory n with context c^n as input to generate the data $x'^{n}_{1:T,i}$ for that trajectory, as illustrated in Figure 5. Note that the training and generation process in GenDT is actually done at the batch level as outlined above and elaborated later in §4.3.3. Also note that although real world scenario characteristics can be quite different from one another (e.g., cell density differences shown in §3) and a target trajectory may span multiple different scenarios, our model does not need to explicitly consider the myriad of possible scenarios. This allows us to use one single model for any scenario(s).

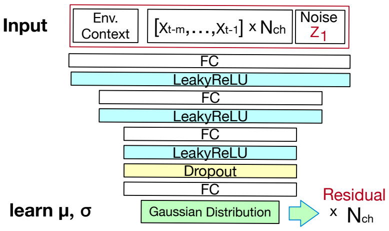


Figure 7: Illustration of ResGen network architecture and the generation of distribution parameters (FC: Fully Connected Layer, LeakyReLU: Leaky Rectified Linear Unit).

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.

- $G_\theta^n : \mathbb{R}^{L\times(N_c+N_{z_0})\times 1} \longrightarrow \mathbb{R}^{L\times H\times N_{ch}}$, where N_{z_0} is the dimension of the input noise and N_{ch} 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.
- $G_\theta^a : \mathbb{R}^{L\times H\times N_{ch}} \longrightarrow \mathbb{R}^{L\times 1\times N_{ch}}$. 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., $h_{avg} = \frac{\sum_{i=1}^{N_b} h_i}{N_b}$. 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.
- $G_\theta^r : \mathbb{R}^{L\times(N_g+N_{z_1})} \longrightarrow \mathbb{R}^{L\times 1\times N_{ch}}$, where N_{z_1} is the dimension of the input noise. The N_g environment context attributes are concatenated with the noise as input. The output of G_θ^r has the same dimensions as G_θ^a 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 ($c_{env,t} \in \mathbb{R}^{1\times N_g}$), 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 μ_θ and standard deviation σ_θ 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 σ_θ 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.

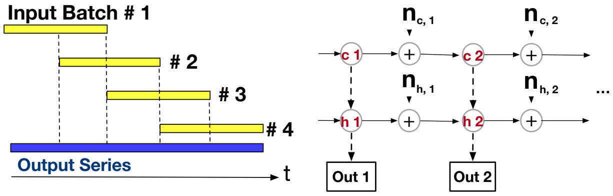
4.3.3 *Batch Training and Generation.*

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 be 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:

$$x_{1:T} \longrightarrow \{x_{1:1+L}, x_{1+\Delta t:1+\Delta t+L}, \ldots, x_{1+\lfloor \frac{T}{L} \rfloor \Delta t:T}\}$$



(a) (b)

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

where Δt 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[4], 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 $h'_t = f_n(h_t, n_{t,h}, a_h)$ and $c'_t = f_n(c_t, n_{t,c}, a_c)$, where f_n is a function to control the intensity of noise input, and the 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 $[0, \hat{h_t}]$ and $[0, \hat{c_t}]$, where $\hat{h_t}$ and $\hat{c_t}$ 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, *i.e.*, $\theta* = \arg\min_\theta JS[p_{\mathcal{D}}||p_\theta]$, and with the aid of discriminator as in the GAN framework. We denote such discriminator as R due to their role as density ratio estimators [59]. Specifically, for given training input measurement data time series and context batch $(\underline{x}, \underline{c})$, the corresponding adversarial loss between the data $p_{\mathcal{D}}(\underline{x}, \underline{c})$ distribution and the model $p_\theta(\underline{x}, \underline{c})$ distribution is defined as:

$$\mathcal{L}^R_{JS}(p_{\mathcal{D}}, p_\theta) = E_{p_{\mathcal{D}}}[\log R(\underline{x}, \underline{c})] + E_{p_\theta}[\log(1 - R(\underline{x'}, \underline{c}))].$$

In our case, we consider one discriminator, named as R_θ, the context input into discriminator is the high dimensional representation of c, which is h_{avg}. The discriminator is a single layer LSTM network.

We additionally use the standard mean squared error loss:

[4] Here we show for one radio KPI (channel) case but the same applies for all channels.

$$\mathcal{L}^M(\underline{x}, \underline{x'}|\underline{c}) = \frac{1}{L}\sum_{t=1}^{L}(\underline{x}_t - \underline{x}'_t)^2$$

Since the batch length L is constant during training, this loss has an equivalent effect to using $L2$ loss.

Overall, together with the adversarial (GAN) loss, the loss function to fit θ is: $\mathcal{L} = \mathcal{L}^M + \lambda\mathcal{L}^R_{JS}$

where the λ 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: $MAE = \sum_{i=1}^{T}|y_i - x_i|/n$. 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 use 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 the 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

Method	RSRP			RSRQ			SINR			CQI		
	MAE	DTW	HWD	MAE	DTW	HWD	MAE	DTW	HWD	MAE	DTW	HWD
GenDT	7.18	3.73	4.93	1.9	1.27	13.2	4.0	4.6	7.2	1.9	1.20	3.8
FDaS	13.63	17.23	4.80	2.8	1.80	10.1	8.2	6.2	5.9	3.1	1.90	3.8
MLP	10.83	9.3	12.20	2.4	1.70	11.0	7.6	5.9	9.0	2.7	1.33	6.1
LSTM-GNN	17.53	13.80	11.47	2.8	1.81	13.1	9.6	6.9	11.2	3.0	1.55	4.1
Orig. DG	12.93	14.17	4.98	2.9	1.86	11.9	8.8	5.9	6.5	3.2	1.60	3.8
Real Cont. DG	9.11	6.07	10.2	2.2	1.69	12.5	5.3	5.4	8.5	2.1	1.25	4.3

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

Method	MAE↓				DTW↓				HWD↓			
	City Center 1	City Center 2	Highway 1	Highway 2	City Center 1	City Center 2	Highway 1	Highway 2	City Center 1	City Center 2	Highway 1	Highway 2
GenDT	4.9	4.8	8.5	8.9	2.8	2.9	5.1	5.4	3.8	1.3	5.2	7.3
FDaS	9.8	11.7	16.7	10.8	6.5	8.8	14.8	10.1	3.4	3.1	7.9	6.4
MLP	8.5	3.2	14.5	16.9	5.6	3.1	11.9	15.2	4.1	2.8	18.7	14.0
LSTM-GNN	19.7	16.8	18.3	13.6	12.1	11.8	14.2	11.2	8.6	10.0	8.5	8.0
Orig. DG	15.6	14.3	17.1	14.6	11.5	10.1	10.4	9.8	5.0	3.2	9.5	9.2
Real Cont. DG	10.3	7.4	9.1	9.4	3.9	4.6	6.0	5.9	3.8	2.9	11.8	9.8

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

Method	MAE↓			DTW↓			HWD↓		
	Walk	Bus	Tram	Walk	Bus	Tram	Walk	Bus	Tram
GenDT	5.17	8.88	7.49	2.4	5.2	3.6	7.2	3.7	3.9
FDaS	11.2	15.6	13.1	15.5	19.0	17.2	6.9	4.2	3.3
MLP	9.9	11.5	12.1	9.0	11.1	7.8	13.8	9.9	12.9
LSTM-GNN	21.5	18.3	12.8	12.1	14.2	15.1	9.7	12.3	12.4
Orig. DG	10.8	14.3	12.7	11.9	16.1	14.5	11.9	13.4	10.1
Real Cont. DG	9.2	10.43	7.71	5.1	7.9	5.2	12.6	11.2	4.9

Table 3: Generated RSRP time series fidelity with GenDT and baselines for different scenarios in Dataset A.

avoid geographic proximity between training and testing measurement data locations. *We only report performance on the testing set throughout this whole section.* While we show results of GenDT (and other baselines) in different scenarios separately to highlight the versatility of GenDT, note that these are all generated using the same GenDT model.

6.1.1 *Dataset A.*

Here we present evaluation results with Dataset A focusing on generation of time series for RSRP, RSRQ, SINR and CQI KPIs. We first carry out the per scenario evaluation focusing on RSRP, before evaluating the average performance of GenDT for all KPIs across all scenarios.

By comparing the performance of different methods under multiple metrics in Table 3 for the generated RSRP KPI time series, we observe that the GenDT generally yields the best performance of each scenario for all metrics. Though FDaS expectedly can model the data distribution well (measured by HWD metric), its performance on other two metrics (particularly DTW) is the worst among all the alternatives compared. MLP performance is intermediate to worst on all metrics, especially in terms of HWD, as it does not model stochasticity and temporal behavior. The HWD performance of LSTM-GNN is similar to that of MLP due to the same underlying reason. Interestingly, it exhibits rather poor performance on MAE and DTW, even worse than MLP that does not model temporal 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 mechanism for effective long series generation.

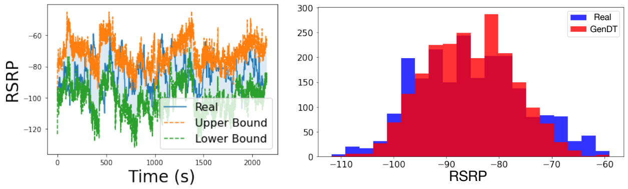
The original DG model, despite being a time-series data generation model, performs poorly across all metrics, about similar or worse than MLP and LSTM-GNN. This is because it is limited by the generated context. Real context DG (our optimized variant of DG) is free from this limitation and better reflects 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.

6.1.2 *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 RSRQ 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.



(a) RSRP time series

(b) RSRP data distribution

Figure 9: Visualization of GenDT performance evaluated over a long and complex trajectory in Dataset B.

Method	RSRP			RSRQ		
	MAE↓	DTW↓	HWD↓	MAE↓	DTW↓	HWD↓
GenDT	6.78	4.05	4.40	1.7	1.40	8.1
FDaS	12.25	10.05	5.20	2.9	1.98	10.8
MLP	10.63	8.95	9.90	2.6	1.81	8.5
LSTM-GNN	17.1	12.33	8.78	2.4	2.0	12.9
Orig. DG	17.93	9.17	11.80	2.9	1.86	12.9
Real Cont. DG	9.05	5.10	7.08	2.0	1.53	11.1

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

Method	RSRP			RSRQ		
	MAE↓	DTW↓	HWD↓	MAE↓	DTW↓	HWD↓
GenDT	11.69	7.18	10.4	3.9	2.40	2.1
FDaS	24.25	16.05	19.20	10.8	13.1	2.98
MLP	18.63	14.95	29.90	8.61	9.9	4.6
LSTM-GNN	18.1	13.80	30.78	10.45	9.9	4.9
Orig. DG	20.40	13.45	26.73	10.1	13.9	2.3
Real Cont. DG	15.05	10.80	27.08	5.08	7.1	3.0

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

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.

Method	MAE↓	DTW↓	HWD↓
GenDT	11.69	7.18	10.4
50s Trajectory	14.50	10.1	18.79
100s Trajectory	13.11	9.05	16.86

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

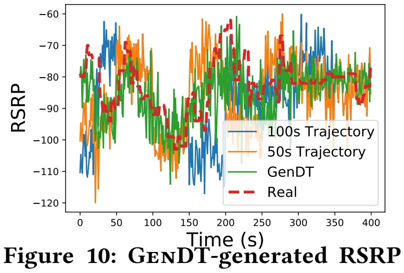


Figure 10: GenDT-generated RSRP time series compared with short independently generated trajectories.

Note that for high fidelity drive test data generation, it is essential to support long series generation. To illustrate this point, we compare GenDT with two cases, where the data for the long (2200+s) target trajectory considered in this subsection is instead obtained by stitching data from multiple independently generated short (50s and 100s) trajectories. Results shown in Table 8 clearly indicate that short trajectory generation does worse than GenDT, especially in terms of the data distribution (HWD metric). Visualization of RSRP series generated with these alternatives (GenDT and 50s/100s short independent trajectories) in Figure 10 clearly highlight the artifacts at the points successive short trajectories are stitched together, whereas GenDT-generated RSRP time series samples closely track the real measurement data. Note that in this figure, we zoom in on the last 400s of the long trajectory to allow the differences to be clearly seen. These results overall highlight the need to capture long-term temporal relations in the data to ensure high fidelity generation.

6.2 Measurement Efficiency

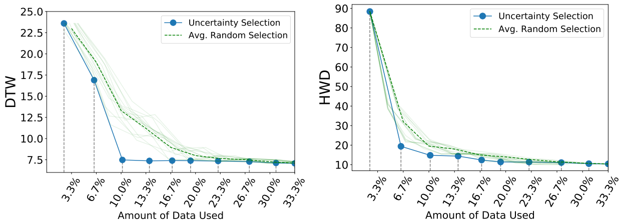
6.2.1 Model Uncertainty. Data uncertainty is irreducible due to the nature of the data while model uncertainty can be reduced by training on more data and actively selecting new training points [21]. The design of GenDT naturally decouples data and model uncertainty: the data uncertainty is reflected 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:

$$U(G_\theta) = \frac{1}{T} \sum_{t=1\cdots T} std(\sigma_\theta)_t + std(\mu_\theta)_t$$

where T is the length of target series and 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_L). 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 initially start with just one small subset



(a) DTW **(b) HWD**

Figure 11: Assessing selection of new training data based on GenDT uncertainty measure relative to random selection.

of data as the training set. At each step, we evaluate the trained model on each of the remaining subsets in the data to obtain the model uncertainty, and select the one with highest uncertainty as new training data to add to the current training set. Concurrently, we evaluate the GenDT model performance on S_L 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_L 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.

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.

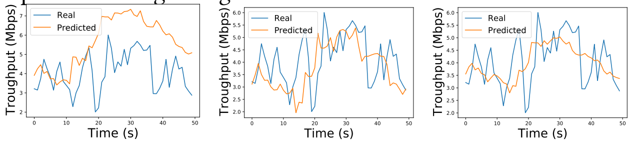
6.3.1 Mobile Service Quality of Experience (QoE) Prediction. User QoE assessment is a key focus of mobile network operators for which they engage in drive test measurement data collection. Application layer throughput is a key QoE metric of interest that in turn depends on lower layer radio KPIs such 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 Nemo Handy [26] but also corresponding downlink throughput and PER measurements obtained with iPerf3 [18].

Method	Throughput			PER		
	MAE↓	DTW↓	HWD↓	MAE↓	DTW↓	HWD↓
Real	6.7	4.0	1.2	0.22	0.18	1.9
RSRP & RSRQ Excluded	13.1	9.6	2.4	0.48	0.39	3.8
GenDT	5.9	4.6	1.4	0.24	0.23	2.7
FDaS	13.4	9.9	2.4	0.48	0.30	3.5
MLP	8.6	5.9	2.1	0.33	0.38	3.2
LSTM-GNN	14.0	9.4	2.5	0.35	0.39	3.4
Orig. DG	13.1	10.1	2.3	0.47	0.39	3.3
Real Cont. DG	7.9	5.1	1.2	0.28	0.31	2.8

Table 9: Performance with GenDT-generated RSRP and RSRQ data when applied to QoE (throughput and PER) prediction use case, relative to baselines.

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.



(a) Without RSRP and RSRQ **(b) Real RSRP and RSRQ** **(c) Generated RSRP and RSRQ**

Figure 12: (a, b) Throughput prediction performance with and without RSRP/RSRQ KPI measurement data, and (c) with GenDT-generated RSRP/RSRQ data.

6.3.2 Analysis of Handovers. Optimizing the handover frequency and performance is of key importance to mobile network operators as too many handovers can not only degrade user experience but also increase signalling overhead in the network. This is done in practice by tuning thresholds of multiple KPIs relevant for mobility management informed by drive testing measurement data on handovers [51].

To support this use case on inferring handovers for a given network deployment, we retrained GenDT to generate the time series of an additional KPI – the serving cell. Tracking serving cell changes essentially provides the information on time between handovers. Note that GenDT model itself remains unchanged from what is

Method	HWD↓
GenDT	2.4
FDaS	8.3
MLP	6.1
LSTM-GNN	5.3
Orig. DG	8.0
Real Cont. DG	3.0

Table 10: Inter-handover time distribution estimation with GenDT-generated serving cell data, relative to baselines.

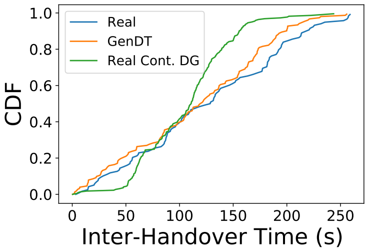


Figure 13: Inter-handover time distribution from GenDT-generated serving cell data compared to real distribution in Dataset B.

described in §4 to accomodate this new serving cell KPI. Quantitative results from Table 10 clearly show that GenDT-generated serving cell data provides inter-handover time distribution that closely match with real data compared to the baseline approaches. This is also apparent from the CDF of inter-handover times with GenDT shown in Figure 13 compared to their real counterpart from drive test measurements in Dataset B. In contrast, inter-handover times from DG-generated data are off from the real.

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.

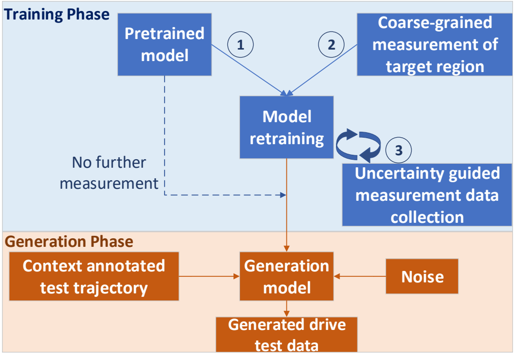

Training Phase
Pretrained model
1
2
Coarse-grained measurement of target region
Model retraining
3
No further measurement
Uncertainty guided measurement data collection
Generation Phase
Context annotated test trajectory
Generation model
Noise
Generated drive test data

Figure 14: Schematic illustrating GenDT use in practice.

Our approach towards efficient drive testing is to mimic drive testing process through a deep generative model. But we do this in an "open-loop" manner in that the effect of network side configuration, control mechanisms and traffic load are not accounted for. 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.

REFERENCES

[1] 3rd Generation Partnership Project (3GPP). 2017. *Radio measurement collection for Minimization of Drive Tests (MDT).* Technical Report. 3GPP, https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2602. TS37.320-v14.

[2] Accuver. 2022. XCAL. https://www.accuver.com/sub/products/view.php?idx=6&ckattempt=1.

[3] Emmanouil Alimpertis, Athina Markopoulou, Carter Butts, and Konstantinos Psounis. 2019. City-Wide Signal Strength Maps: Prediction with Random Forests. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 2536–2542. https://doi.org/10.1145/3308558.3313726

[4] Atlas. 2012. Copernicus Urban Atlas. https://land.copernicus.eu/local/urban-atlas/urban-atlas-2012.

[5] Sanchez B, Ling L, Reif E, Pearce A, and Wiltschko Alexander B. 2021. A Gentle Introduction to Graph Neural Networks. https://distill.pub/2021/gnn-intro/.

[6] Jue Cao, Di Kong, Michael Charitos, Denys Berkovskyy, Angelos A. Goulianos, Tom Mizutani, Fai Tila, Geoffrey Hilton, Angela Doufexi, and Andrew Nix. 2018. Design and Verification of a Virtual Drive Test Methodology for Vehicular LTE-A Applications. *IEEE Transactions on Vehicular Technology* 67, 5 (2018), 3791–3799. https://doi.org/10.1109/TVT.2018.2794263

[7] Carmelo Cassisi, Placido Montalto, Marco Aliotta, Andrea Cannata, and Alfredo Pulvirenti. 2012. Similarity measures and dimensionality reduction techniques for time series data mining. *Advances in data mining knowledge discovery and applications'(InTech, Rijeka, Croatia, 2012,* 1 (2012), 71–96.

[8] Cellmapper. 2021. Cellmapper. https://www.cellmapper.net/.

[9] Kwangrok Chang and Ragil Putro Wicaksono. 2017. Estimation of network load and downlink throughput using RF scanner data for LTE networks. In *2017 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*. 1–8. https://doi.org/10.23919/SPECTS.2017.8046779

[10] Yang Chen, Dustin J Kempton, Azim Ahmadzadeh, and Rafal A Angryk. 2021. Towards Synthetic Multivariate Time Series Generation for Flare Forecasting. In *International Conference on Artificial Intelligence and Soft Computing*. Springer, 296–307.

[11] Christopher Cox. 2014. *An Introduction to LTE* (2 ed.). Wiley, Chapter 17.

[12] Christopher Cox. 2014. *An Introduction to LTE* (2 ed.). Wiley.

[13] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. 2018. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine* 35, 1 (2018), 53–65.

[14] George Deodatis and M Shinozuka. 1988. Auto-regressive model for nonstationary stochastic processes. *Journal of engineering mechanics* 114, 11 (1988), 1995–2012.

[15] Paolo Di Francesco, Francesco Malandrino, and Luiz A DaSilva. 2017. Assembling and using a cellular dataset for mobile network analysis and planning. *IEEE Transactions on Big Data* 4, 4 (2017), 614–620.

[16] Lukas Eller, Vaclav Raida, Philipp Svoboda, and Markus Rupp. 2022. Localizing Basestations From End-User Timing Advance Measurements. *IEEE Access* 10 (2022), 5533–5544. https://doi.org/10.1109/ACCESS.2022.3140825

[17] Ayman Elnashar and Mohamed A El-Saidny. 2013. Looking at LTE in practice: A performance analysis of the LTE system based on field test results. *IEEE Vehicular Technology Magazine* 8, 3 (2013), 81–92.

[18] J. Dugan et al. 2022. iPerf – The TCP, UDP and SCTP network bandwidth measurement tool. https://iperf.fr/.

[19] Mah-Rukh Fida et al. 2017. ZipWeave: Towards efficient and reliable measurement based mobile coverage maps. In *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*. 1–9. https://doi.org/10.1109/INFOCOM.2017.8057098

[20] Marco Fraccaro, Søren Kaae Sø nderby, Ulrich Paquet, and Ole Winther. 2016. Sequential Neural Models with Stochastic Layers. In *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Eds.), Vol. 29. Curran Associates, Inc., Barcelona SPAIN. https://proceedings.neurips.cc/paper/2016/file/208e43f0e45c4c78cafadb83d2888cb6-Paper.pdf

[21] Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*. PMLR, 1050–1059.

[22] Groundhog. 2020. CovMo eVDT. https://www.ghtinc.com/the-future-of-drive-test-after-covid-19-virtual-is-here-to-stay/.

[23] CNI group. 2020. CNI Cell Tracker and Dataset. https://github.com/hendrikschippers/CNI-Cell-Tracker.

[24] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc., Long Beach, US. https://proceedings.neurips.cc/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf

[25] Google Inc. 2021. Telephony API of Android. https://developer.android.com/reference/android/provider/Telephony.

[26] Keysight Inc. 2021. Nemo Handy by Keysight. https://www.keysight.com/gb/en/product/NTH00000B/nemo-handy-handheld-measurement-solution.html?rd=1.

[27] Infovista. 2022. TEMS. https://www.infovista.com/tems.

[28] Johan Johansson, Wuri A. Hapsari, Sean Kelley, and Gyula Bodog. 2012. Minimization of drive tests in 3GPP release 11. *IEEE Communications Magazine* 50, 11 (2012), 36–43. https://doi.org/10.1109/MCOM.2012.6353680

[29] Keysight. 2022. Nemo Wireless Network Solutions. https://www.keysight.com/gb/en/products/nemo-wireless-network-solutions.html.

[30] Mark Leznik, Patrick Michalsky, Peter Willis, Benjamin Schanzel, Per-Olov Östberg, and Jörg Domaschka. 2021. Multivariate Time Series Synthesis Using Generative Adversarial Networks. In *Proceedings of the ACM/SPEC International Conference on Performance Engineering* (Virtual Event, France) *(ICPE '21)*. Association for Computing Machinery, New York, NY, USA, 43–50. https://doi.org/10.1145/3427921.3450257

[31] Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. 2020. Using GANs for Sharing Networked Time Series Data: Challenges, Initial Promise, and Open Questions. In *Proceedings of the ACM Internet Measurement Conference* (Virtual Event, USA) *(IMC '20)*. Association for Computing Machinery, New York, NY, USA, 464–483. https://doi.org/10.1145/3419394.3423643

[32] Zhilong Lu, Weifeng Lv, Yabin Cao, Zhipu Xie, Hao Peng, and Bowen Du. 2020. LSTM variants meet graph neural networks for road speed prediction. *Neurocomputing* 400 (2020), 34–45.

[33] Dimitar Minovski, Christer Åhlund, Karan Mitra, and Per Johansson. 2019. Analysis and Estimation of Video QoE in Wireless Cellular Networks using Machine Learning. In *2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX)*. 1–6. https://doi.org/10.1109/QoMEX.2019.8743281

[34] n.d. 2022. How do you generate synthetic data? https://www.statice.ai/post/how-generate-synthetic-data.

[35] n.d. 2022. Keysight VDT toolset. https://www.keysight.com/gb/en/assets/7018-06582/solution-briefs/5992-3870.pdf.

[36] n.d. 2022. Open Signal. https://www.opensignal.com/.

[37] n.d. 2022. Spirent Live2Lab. https://www.spirent.com/assets/u/virtual_drive_test.

[38] n.d. 2022. Tutela. https://www.tutela.com/.

[39] Nokia. 2020. Nokia and 3 Indonesia develop Zero Drive Test assessment solution to enhance network quality and user experience. https://www.nokia.com/about-us/news/releases/2020/08/19/nokia-and-3-indonesia-develop-zero-drive-test-assessment-solution-to-enhance-network-quality-and-user-experience/.

[40] Eduardo Mucelli Rezende Oliveira, Aline Carneiro Viana, Kolar Purushothama Naveen, and Carlos Sarraute. 2017. Mobile data traffic modeling: Revealing temporal facets. *Computer Networks* 112 (2017), 176–193.

[41] OSM. 2020. OpenStreetMap Overpass API. http://overpass-turbo.eu/.

[42] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[43] Caleb Phillips, Douglas Sicker, and Dirk Grunwald. 2013. A Survey of Wireless Path Loss Prediction and Coverage Mapping Methods. *IEEE Communications Surveys & Tutorials* 15, 1 (2013), 255–270. https://doi.org/10.1109/SURV.2012.022412.00172

[44] Darijo Raca, Ahmed H. Zahran, Cormac J. Sreenan, Rakesh K. Sinha, Emir Halepovic, Rittwik Jana, and Vijay Gopalakrishnan. 2017. Back to the Future: Throughput Prediction For Cellular Networks Using Radio KPIs. In *Proceedings of the 4th ACM Workshop on Hot Topics in Wireless* (Snowbird, Utah, USA) *(HotWireless '17)*. Association for Computing Machinery, New York, NY, USA, 37–41. https://doi.org/10.1145/3127882.3127892

[45] Darijo Raca, Ahmed H. Zahran, Cormac J. Sreenan, Rakesh K. Sinha, Emir Halepovic, Rittwik Jana, Vijay Gopalakrishnan, Balagangadhar Bathula, and Matteo Varvello. 2019. Empowering Video Players in Cellular: Throughput Prediction from Radio Network Measurements. In *Proceedings of the 10th ACM Multimedia Systems Conference* (Amherst, Massachusetts) *(MMSys '19)*. Association for Computing Machinery, New York, NY, USA, 201–212. https://doi.org/10.1145/3304109.3306233

[46] Vaclav Raida, Martin Lerch, Philipp Svoboda, and Markus Rupp. 2018. Deriving Cell Load from RSRQ Measurements. In *2018 Network Traffic Measurement and Analysis Conference (TMA)*. 1–6. https://doi.org/10.23919/TMA.2018.8506494

[47] Vaclav Raida, Philipp Svoboda, and Markus Rupp. 2020. Real World Performance of LTE Downlink in a Static Dense Urban Scenario - An Open Dataset. In *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*. 1–6. https://doi.org/10.1109/GLOBECOM42002.2020.9348204

[48] RantCell. 2022. What is the significance of drive test in telecom and drive test analysis? https://rantcell.com/RF-drive-test-analysis-significance.html.

[49] Ludger Rüschendorf. 1985. The Wasserstein distance and approximation theorems. *Probability Theory and Related Fields* 70, 1 (1985), 117–129.

[50] Megha Sahu, Snigdha Damle, and Arzad Alam Kherani. 2021. End-to-end uplink delay jitter in LTE systems. *Wireless Networks* 27, 3 (2021), 1783–1800.

[51] Martin Sauter. 2017. *From GSM to LTE-Advanced Pro and 5G* (3 ed.). Wiley, Chapter 4.

[52] Rohde & Schwarz. 2022. Mobile network testing. https://www.rohde-schwarz.com/us/solutions/test-and-measurement/mobile-network-testing/overview/mobile-network-testing_231692.html.

[53] Telecom Knowledge Share. 2016. LTE Drive Test Parameters. https://telecom-knowledge.blogspot.com/2016/09/lte-drive-test-parameters.html.

[54] Joel Shodamola, Haneya Qureshi, Usama Masood, and Ali Imran. 2021. Towards Addressing the Spatial Sparsity of MDT Reports to Enable Zero Touch Network Automation. In *2021 IEEE Global Communications Conference (GLOBECOM)*. 1–6. https://doi.org/10.1109/GLOBECOM46510.2021.9686011

[55] Benjamin Sliwa, Hendrik Schippers, and Christian Wietfeld. 2021. Machine Learning-Enabled Data Rate Prediction for 5G NSA Vehicle-to-Cloud Communications. In *2021 IEEE 4th 5G World Forum (5GWF)*. 299–304. https://doi.org/10.1109/5GWF52925.2021.00059

[56] Benjamin Sliwa and Christian Wietfeld. 2019. Data-driven network simulation for performance analysis of anticipatory vehicular communication systems. *IEEE Access* 7 (2019), 172638–172653.

[57] Jakob Thrane, Matteo Artuso, Darko Zibar, and Henrik L. Christiansen. 2018. Drive Test Minimization Using Deep Learning with Bayesian Approximation. In *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*. 1–5. https://doi.org/10.1109/VTCFall.2018.8690911

[58] Catherine Tong, Emma Rocheteau, Petar Veličković, Nicholas Lane, and Pietro Liò. 2021. Predicting patient outcomes with graph representation learning. In *International Workshop on Health Intelligence*. Springer, Springer, Berlin, Germany, 281–293.

[59] Masatoshi Uehara, Issei Sato, Masahiro Suzuki, Kotaro Nakayama, and Yutaka Matsuo. 2016. Generative adversarial nets from a density ratio estimation perspective. *arXiv preprint arXiv:1610.02920* 1, 1 (2016), 1–16.

[60] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research* 11, 12 (2010).

[61] Lina Wu, Danping He, Bo Ai, Jian Wang, Hang Qi, Ke Guan, and Zhangdui Zhong. 2020. Artificial neural network based path loss prediction for wireless communication network. *IEEE Access* 8 (2020), 199523–199538.

[62] Kai Xu et al. 2021. 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* (Virtual Event, Germany) *(CoNEXT '21)*. Association for Computing Machinery, New York, NY, USA, 243–258. https://doi.org/10.1145/3485983.3494844

[63] Yang Yang, Yang Li, Wuxiong Zhang, Fei Qin, Pengcheng Zhu, and Cheng-Xiang Wang. 2019. Generative-Adversarial-Network-Based Wireless Channel Modeling: Challenges and Opportunities. *IEEE Communications Magazine* 57, 3 (2019), 22–27. https://doi.org/10.1109/MCOM.2019.1800635

[64] Chaoqun Yue, Ruofan Jin, Kyoungwon Suh, Yanyuan Qin, Bing Wang, and Wei Wei. 2018. LinkForecast: Cellular Link Bandwidth Prediction in LTE Networks. *IEEE Transactions on Mobile Computing* 17, 7 (2018), 1582–1594. https://doi.org/10.1109/TMC.2017.2756937

[65] ZTE. 2019. One Person, One Car, One Terminal, ZTE WNG Automatic Drive Test Solution Dramatically Improves O&M Efficiency. https://www.mobileworldlive.com/zte-updates-2019-20/one-person-one-car-one-terminal-zte-wng-automatic-drive-test-solution-dramatically-improves-om-efficiency.

A DATA ANALYSIS AND MODEL DETAILS

A.1 Visualization of Environment Context Attributes

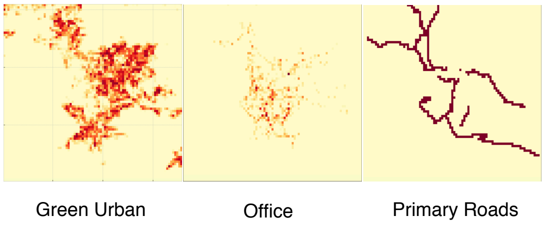
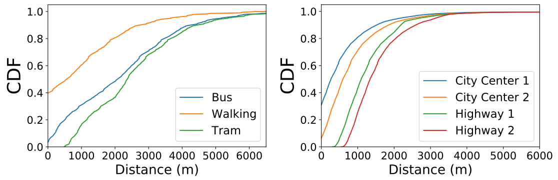


Figure 15: Spatial distribution of 3 selected environment context attributes in Dataset B.

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

Table 11: List of environment context attributes considered. See examples in Figure 15



(a) Dataset A (b) Dataset B

Figure 16: CDF of distance to serving cell in different scenarios.

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:

$$h'_t = (h_t + a_h n_{t,h}) \frac{\sum_{i=1:H} h_{t,i}}{\sum_{i=1:H}(h_{t,i} + a_h n_{t,h,i})}, h_t = \{h_{t,1}, \cdots, h_{t,H}\}$$

$$c'_t = (c_t + a_c n_{t,c}) \frac{\sum_{i=1:H} c_{t,i}}{\sum_{i=1:H}(c_{t,i} + a_c n_{t,h,i})}, c_t = \{c_{t,1}, \cdots, c_{t,H}\}$$

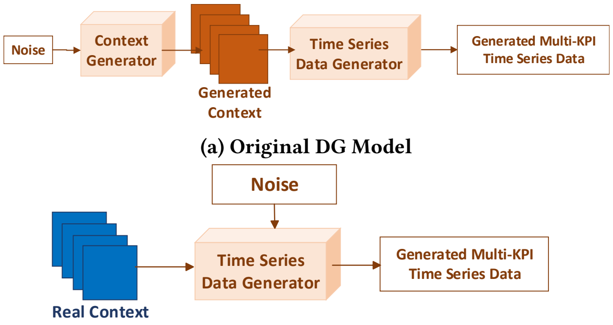
Where H is the dimension of hidden state h_t and c_t. Using different a_h and a_c, we can control the relative intensity of noise to the hidden states, and thus control the uncertainty level during training.

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
Context Generator
Generated Context
Time Series Data Generator
Generated Multi-KPI Time Series Data
Noise
Real Context
Time Series Data Generator
Generated Multi-KPI Time Series Data

(a) Original DG Model

(b) Real Context DG Model

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

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.

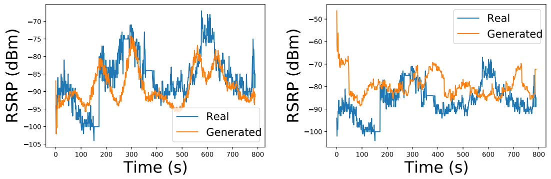
B DISCUSSION ON DOPPELGANGER

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.



(a) GenDT **(b) Real Context DG**

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

Method	RSRP			RSRQ		
	MAE↓	DTW↓	HWD↓	MAE↓	DTW↓	HWD↓
GenDT	8.10	5.89	7.67	1.7	1.34	1.65
No ResGen	7.99	6.60	13.7	1.6	2.3	9.7
No SRNN	11.53	8.89	10.4	2.4	1.99	4.8
No GAN loss	14.66	12.45	15.3	3.8	3.6	6.9
No batch	12.9	9.60	10.5	2.6	2.3	3.7

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

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.