UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

INVESTIGATING THE POTENTIAL OF SYNTHETIC DATA FOR ENABLING AI-BASED ZERO-TOUCH NETWORK AUTOMATION

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE IN ELECTRICAL AND COMPUTER ENGINEERING

BY

JOEL SHODAMOLA
Norman, Oklahoma
2021

INVESTIGATING THE POTENTIAL OF SYNTHETIC DATA FOR ENABLING AI-BASED ZERO-TOUCH NETWORK AUTOMATION

A THESIS APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY

Acknowledgments

My first appreciation goes to God who has been with me every step of this journey till date. I would like to give special thanks to Dr. Ali Imran for his support and his patience all through my Masters program. Dr. Ali Imran's clear vision and commitment motivated me all through my research and I am more than grateful to be under his tutelage. I would also like to thank the rest of the committee Dr. Hazem Refai and Dr. Samuel Cheng for taking time out to review the manuscript and being a part of my journey. I also appreciate all colleagues in AI4Networks Research Center who supported me one way or the other, I am blessed to be part of this intelligent team. Finally, I would like to give special thanks to my family who believed in me and supported all through the journey of my Masters program.

Table of Contents

List of Figures

List of Tables

Abstract

The essence and importance of rich and relevant data can not be overemphasized in the field of artificial intelligence. From machine learning to deep learning models, the performance of a model majorly dependent on the quantity and quality of data fed for training. However, in today's cellular network, this rich data is not easily accessible and attainable. For example, in Minimization of Drive Test (MDT) reports, manual operators realize only a fraction of the entire networks coverage map and this results into a suboptimal performance of the network ultimately leading to a poor performance in anticipatory Network automation. To enable seamless automation, synthetic data is leveraged to enrich and improve the quality as well as quantity of data for robust and intelligent networks. However, to generate synthetic that has close attribute with the ground is not a trivial task. In this thesis we present and evaluate a framework to address this challenge. We employ the use of generative models, specifically, Generative Adversarial Networks (GAN) and Variational Autoendocers to augment the sparse multi-dimensional attributes that exist in real datasets. Unlike image data where the quality of synthetic images produced by the generative models can be evaluated visually, establishing the authenticity of tabular synthetic data is a more complex problem.

We address this problem by leveraging a tripartite approach: 1) We use several statistical measures to quantify the resemblance of synthetic data with original data. 2) We compare the performance of an ensemble learning model trained on augmented data, with that of trained on original data only 3) We benchmark the performance of the generative models with several classical ML models. This analysis is carried out for varying levels of sparsity and reveals insights about robustness of generative models against training data sparsity as well as on suitability of various methods for evaluating the quality of the generated synthetic tabular data. Results show GAN performs considerably better compared to other approaches. The presented solution thus can be used to overcome the sparsity problem in MDT reports thereby enabling ML-based automation use cases.

CHAPTER 1

Introduction

1.1 Background and Motivation

In today's world of technology, the efficacy of data for seamless operations can not be overemphasized. Cutting across all domains, including but not limited to, transportation, banking and finance, medicine, online retail and telecommunications, relevant data is usually harnessed to either increase better services, improve quality of experience or predict future outcomes. In the cellular network domain, among many purposes, these data are specifically leveraged for optimization purposes. Where network operators tune certain network parameters to either maximize a key performance indicator (KPI) or concurrently optimize multiple KPIs. However, with an exponential increase in mobile devices, IOT devices, and general internet-dependent devices, the cellular network becomes ultra-densified with macro and small cells to meet user and service demands, subsequently leading to an innumerable amount of cellular network parameters (CNP) to be optimized. It is anticipated that the operational complexity of the network will be an albatross as this complexity is predicted to scale linearly with densification [1]. To overcome the limitation of manual operations, network automation has been proposed as a solution for autonomous configuration and optimization. Network automation on the other hand employs machine and deep learning architecture which requires a massive amount of data for training and subsequently perform predictive analysis. However, in today's realistic cellular network domain, real data is scarcely available for several reasons. For instance, in Minimization of Drive Test reports as specified by 3GPP [2], network coverage is usually estimated based on received power signal gathered from user equipment, yet, this coverage estimation is usually sparse in nature. The following are some of the factors that lead to the sparse nature of cellular network data:

1. Sparsity from small cells: In comparison with macro cells, user traffic under small cell coverage is expected to be less dense. Users reporting their signal will be fewer in number, thus leading to a scanty report.
2. User equipment incompatibility: While some user equipment are built with MDT report compatibility, others are not. Hence, when reports are gathered, the coverage map will consist of holes or white spaces.
3. Data Privacy: For privacy concerns, some regulations are enforced to prevent personal data of some users, hence creating gaps in reports when received by network operators. This reason is especially conversant in the medical domain.
4. Suboptimal planning process: Network operators avoid endangering quality of service, thus, they tend to only achieve suboptimal performance by exploring a subset of parameters. Not all possible combination of parameters are discovered, and this leads to a limited amount of data. In addition, modelling data from an offline simulator for either planning or optimization can be computationally inefficient and time consuming, considering the amount of time it takes to produce all combinations of parameters.

To maximize the full potential of autonomous driven operations that totally rely on data, synthetic data is ultimately harnessed for data augmentation. From machine learning to deep learning model reliability, a tremendous amount of data is needed to train the models to be able to predict coverage estimation (in terms of MDT reports) and further obtain complete optimization of network KPIs as would be the case when full ground truth is made available. Synthetic data involves the recreation of non-existent and representative data with properties of a real data. Although synthetic data generation has recently gotten much attention in several domain especially in the medical field, however, in cellular network studies, only a handful have considered its use in solving data sparsity challenges. Lately, there has been a growing reliance on deep learning generative models like generative adversarial network (GAN), variational autoencoders (VAE), Boltzmann machines to produce similar representative data from the real data. Some of these generative models perform better than others on different types of dataset [3]. While these model developments seem promising, a lot of application has been seen on image data, with recent transition to tabular data in other domains. This thesis focuses more on tabular synthetic data generation in cellular network field while examining their potentials for network automation. In today's industry, business and corporate world, the most common type of data that will be encountered is in tabular format having columns with descriptive properties. This type of data also stretches to the academia and research domain with relevant simulations

producing data that have close characteristics with that of the industries real data. The benefits that come with using synthetic data include but not limited to, testing new models or software for production and improvement purposes, research purpose and, use as substitute for privacy concerns with sensitive information.

1.2 Related Studies

Employing intelligence for network automation will require models that rely heavily for training however, in a realistic cellular network environment, there is not enough data due to the factors listed in section 1.1, hence the reliance on synthetic data generation. Recently, many literatures have proposed several solutions from interpolation techniques [4, 5, 6], sampling techniques [7] and most recently, deep learning generative model techniques [8, 9]. Most of these studies either address data imbalance, data corruption or privacy concerns. Most of the studies later addressed data imbalance data corruption or privacy concerns. In cellular network domain, synthetic data would be utilized for predictive network models as well as autonomous optimization. To address cell outage sparsity, authors in [10] propose a grey-prediction model that utilizes differential equation to predict reference signal gotten from the periodic updates from user equipment. In [11], authors integrated sampling, additive noise and variational autoencoder to generate synthetic data for localization of both indoor and outdoor environments and conclude that the augmentation techniques improve the accuracy of localization. As more technologies emerge, recent studies are delving into employing intelligent solutions from machine learning to deep learning. The authors in [12] declare that rather than collecting data from the entire network which requires an amount of effort, they use a machine learning technique together with sampling to intuitively select traffic data from a few base stations. Authors in [13], generate synthetic data as a way of support the 5G function, network data analytics functions (NWDAF), however they do not use any generative model for this. The use of transfer learning is proposed in [14] to optimize local edge caching given a sparse distribution of users in a small cell. Authors in [15], used hadoop to create a generator which was in turn employed to recreate synthetic data. They use this synthetic to create IoT related dataset for research purposes. [3, 16, 17, 18] have the closest relevance to this thesis paper, where synthetic data is generated from deep learning techniques and investigated to test for validity and productivity. While some of these works address privacy concerns, others address image or audio data, the rest utilize synthetic data for a different domain i.e medical, automobile. In this thesis, we use two deep learning generative techniques to generate synthetic data and as well evaluate if:

1. The synthetic data is representative, that is, having close or equal properties with the real data.
2. The synthetic data helps to reduce error or increase accuracy in a predictive model.
3. The generative models perform well with respect to increase or decrease in features in the data set.
4. The synthetic data has similar distribution with respect to several levels of sparse trained data.
5. The synthetic data plays any role in cellular network optimization.

1.3 Articles published/to be published under this thesis context

1. A machine learning based framework for KPI maximization in emerging networks using mobility parameters, 2020 Joel Shodamola, Usama Masood, Marvin Manalastas and Ali Imran

2. Towards Addressing Minimization of Drive Test Reports Spatial Sparsity to enable Zero Touch Automation, 2021 Joel Shodamola, Haneya Qureshi, Usama Masood and Ali Imran

1.4 Organization

The rest of this thesis is thus organized: chapter 2 discusses synthetic data generation in full details; the deep learning generative models like GAN and VAE especially for tabular data. Data simulations for the two dataset and evaluation methods are explained in chapter 3. In chapter 4, tuning of hyperparameters for both the generative and machine learning models is examined and how we reached conclusion of the parameters used. Chapter 5, results are basically discussed; the preceding part involves detailed evaluation of the synthetic data generated using a few statistical measures in comparison with the trained data, while the concluding part discusses the usefulness of synthetic data to aid prediction and data augmentation of cellular network. Chapter 6 brings the thesis to conclusion with some future recommendations.

CHAPTER 2

SYNTHETIC DATA GENERATION TECHNIQUES

To reproduce a replica of given data, the distribution, parameters, and underlying functions used to generate that data must be known. For example, given a gaussian distribution from a data, if the mean and standard deviation are known, as well as the minimum and maximum values, it is easy to make inference as well as regenerate a similar distribution. Another method would be sampling from the distribution. Learning the statistical properties of the original data is of utmost importance, however, given data with high dimensions or multiple columns (in our case, tabular data), column combinations will increase, thereby leading to increased complexity of the data. A robust model is therefore required to be able to learn intricate properties that exist in a specified data, hence, the heavy reliance on deep neural networks (DNN). Inspired by the working principle of the biological brain, neural networks consist of nodes, and neutrons that are interconnected with each other. With increased layers, this makes the network deeper with some layers having different type of activation functions depending on the type of data. These nodes and layers connected are able to learn distributions and functions from linear to non-linear, and subsequently generate synthetic data with same patterns that exist in original data. For this thesis, we will focus on the two main types of deep generative models namely; GANs and VAEs.

2.1 Generative Adversarial Network (GAN)

Invented by Ian Goodfellow et al in 2014 [19], GAN is a type of unsupervised learning that comprises of two neural networks competing against each other in what is known as a minimax game. These two networks, generator G and discriminator D, deep in architecture are operating in a way to outperform each other where the former is generating synthetic samples from a noise distribution z, and the latter acting as a critic to determine which data is real or fake (from the generator).

As seen from the function in the generator produces data $\hat{x}$ from a noise input and parameters $G(z;\theta)$. These parameters are derived from a Gaussian prior distribution p_z. The discriminator on the other hand takes in two types of data, the original data x and the generated (fake) data, $\hat{x}$ with output labels [0,1], where 0 represents fake data, and 1 representing real data where the generator has no access to the real data. This operation continues until the discriminator begins to see the synthetic data as the original and can no longer differentiate between both.

$$\min maxN\left(G, D\right) = \mathbb{E}_{x\ p_{ori(x)}}\left[\log D\left(x\right)\right] + \mathbb{E}_{z\ p_z}\left[\log(1 - D\left(G\left(z;\theta\right)\right))\right] \tag{2.1}$$

Where the loss of each network is:

$$\mathbb{L}_D = \left[\log D\left(x\right)\right] + \ \left[\log\left(1 - D\left(G\left(z;\theta\right)\right)\right)\right] \tag{2.2}$$

$$\mathbb{L}_G = \ \left[\log\left(1 - D\left(G\left(z;\theta\right)\right)\right)\right] \tag{2.3}$$

Where D(x) and D(G(z)) are the estimates of the discriminator probability the data is real and fake respectively. G(z) is the data produced from noise z. E_x and E_z are the expected values of the real data instance and synthetic data instance respectively. The challenge with using vanilla GAN architecture on our data were in two facets; vanishing gradients from the loss function and secondly mode collapse

2.1.1 Vanishing Gradient

In a typical deep neural network, weights of each nodes are usually updated through a method known as back propagation. As the gradient is calculated using the chain rule, this updates from end layers to initial layers, however, this gradient after some time tends to get smaller and eventually get to a point where the initial layers are not updated. In GAN architectures, when we have high performing discriminator, the generator performance tends to be poor and can subsequently fail to learn or produce meaningful data most times leading to NaN values (this occurred initially with the data used in this thesis). This is because of vanishing gradients.

2.1.2 Mode Collapse

Mode collapse is a common phenomenon that occurs when training GAN networks. This occurs when the discriminator reaches a local minimum and the generator tends to produce a subcategory of the distribution output. In essence, regardless of the noise input z, the generator repeatedly keeps producing a subset of the entire data, this is known as mode collapse. This means that the gradient of z tends to zero and the generator gets stuck. In training, partial mode collapse is usually common where total mode collapse are rare or don't occur at all. To overcome the aforementioned challenges, recent developments of GAN came up with a metric to calculate the loss called Wasserstein loss function which also developed into a variation of GAN called Wasserstein GAN.

2.1.3 Wasserstein GAN

Wasserstein GAN is a variant of the vanilla GAN in the exception of the replacement of the loss function with Wasserstein distance. Also known as the Earth mover's distance, the Wasserstein metric is used to measure the distance between two probability distributions. It can also be defined as the cost in the optimal movement involved in transforming the original distribution P_ori to the synthetic distribution P_syn. This is more described mathematically in the equation 2.4.

$$\mathbf{W}(\mathcal{P}_{ori}, \mathcal{P}_{syn}) = \underset{\gamma \in \pi(\mathcal{P}_{ori}, \mathcal{P}_{syn})}{inf} \mathbb{E}_{(a,b)\sim\gamma}[||a - b||] \tag{2.4}$$

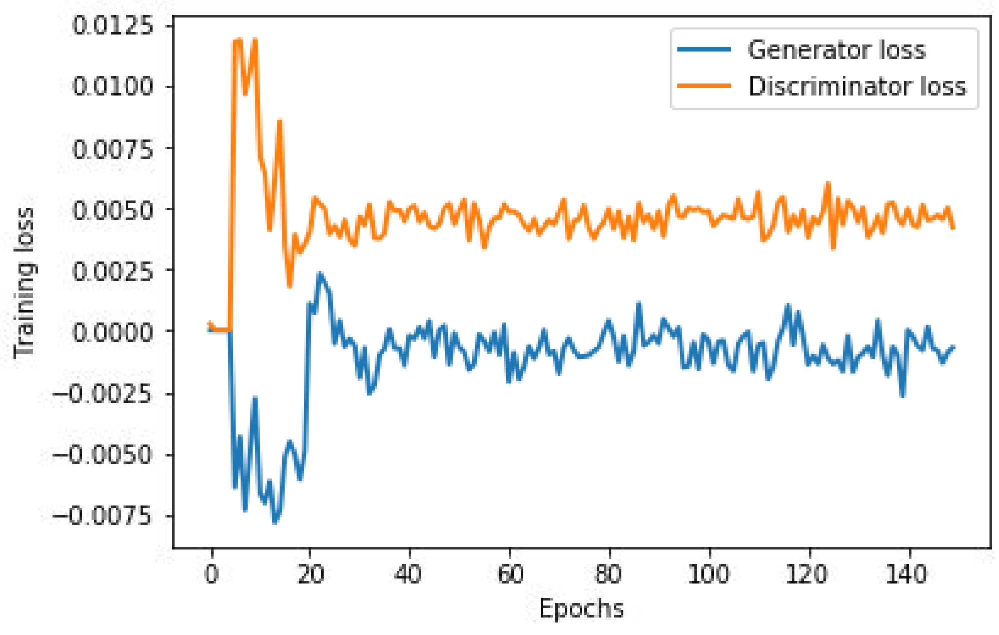

Fig. 2.1: WGAN Training loss

Where π (P_{ori}, P_{syn}) represents the joint distributions $\gamma(x, y)$ with marginals as P_{ori} and P_{syn}while Wasserstein distance has the advantage of continuity and differentiability at all points, it has the disadvantage of being intractable as all possible joint distributions need to be exhausted. To overcome this limitation, the Kantorovich- Rubeinstein duality was introduced simplifying equation 2.5 to:

$$\mathbf{W}(\mathcal{P}_{ori}, \mathcal{P}_{syn}) = \underset{||f||L \leq 1}{sup} \ \mathbb{E}_{x \ P_{ori}}[f(x)] - \mathbb{E}_{x \ P_{syn}}[f(x)] \tag{2.5}$$

With f being the Lipschitz function and sup the least upper bound. When formulating this loss, the weights are usually clipped during the training with the discriminator, acting as a critic here, being updated for about five times. The difference between the score of the vanilla GAN and WGAN is rather than a binary score of [0,1], the WGAN critic network measures how good the samples are with values closer to 1. With the above limitations mentioned in section 2.2.1 and 2.2.2 taken care of, the generator can produce synthetic data with distributions close to the real data irrespective of the performance of the critic network. As illustrated in figure 2.1, the discriminator and generator network converge at about 30 epochs and from this point forward, the generator begins to improve to have closer distribution.

2.1.4 TGAN (Tabular GAN)

The most popular applications of GAN are on image dataset, and it is important to mention that modelling and generating a similar tabular dataset is quite an herculean task given the different modes and types of data that could be present, i.e. numerical, textual, categorical including continuous and discrete. In cellular networks, the majority of data type will be in numerical format either in discrete or continuous values i.e. RSRP and SINR values. Hence, modelling a generative network that can accommodate the complications that come with tabular data is very essential. From figure 2.2, tabular GAN should be able to generate a similar tabular dataset with similar features an properties from the real training dataset, however, this tabular data generation comes with some challenges.

1. Missing Data: The efficacy of GAN to model a tabular dataset depends on its ability to effectively predict missing values in a particular column. For instace, MDT reports come with missing values in coverage maps, GAN has to be structured in such a way to identify and estimate the missing values in the dataset.
2. Multimodal-or Non-Gaussian Data: Traditional GAN works well on a uni-modal distribution, however, the challenge occurs when multi-modal or non-gaussian data is fed into the network. This challenge does not seem to occur in an image dataset, however, in a tabular dataset, it is often inevitable.
3. Data mixture: Unlike an image dataset, most tabular datasets comprises of discrete, continuous and categorical data. Where a sigmoid activation function would suffice for pixel data, tabular GAN networks would require *tanh* and *softmax* activation functions to generate data of these categories.
4. Data with very high dimensions: Given a heteregenous network with ultra high density, data gotten from this network will have high dimensions considering the superfluity

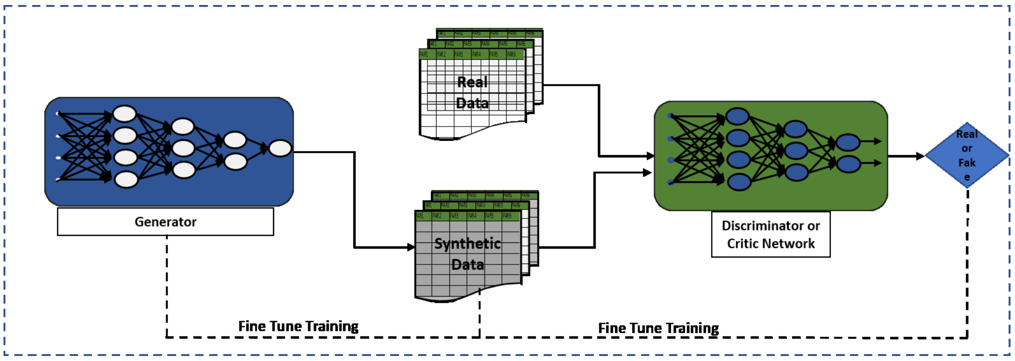

Fig. 2.2: GAN Network

of parameters to be optimized. Therefore, modelling data with these higher dimensions becomes more complex.

Lei Xu and Veeramachaneni in [20] came up with a GAN variant synthesizer that trains on a tabular dataset and produces similar dataset taking in to consideration the column data types. They address the multimodal distribution present in numerical dataset by applying a Gaussian Mixture Model (GMM) to firstly estimate and sample values from a multimodal distribution (common in a continuous variable). They subsequently train the GMM given k number of components with a weighted sum of k Gaussian distribution with mean $\mu_i^1, ...\mu_i^k$ and $\sigma_i^1, ...\sigma_i^k$. Then they computed the probability of the columns $(c_{i,j})$ in the data that is gotten from each of the k Gaussian distribution as a vector $u_{i,j}^1$ to $u_{i,j}^k$ where each vector is normalized over k Gaussian distributions. Following this, each column $C_{i,j}$ normalized to become $v_{i,j} = (c_{i,j} - \mu_i^j)/2\sigma_i^j$ where $j = argmax_j u_{i,j}^k$. This normalized vector is then clipped to a range [-0.99,0.99]. Each column is converted to a combination of $u_{i,j}$ and $v_{i,j}$. To address the differentiability of the model for categorical variables, they use softmax to produce the probability distribution and subsequently convert the categorical variables to a one-hot encoding format and add uniform noise.

The above techniques are preprocessing techniques before the data is fed into the neural network. For post-processing , the columns $(c_{i,j})$ are reconstructed from the cluster vector $u_{i,j}$ and $v_{i,j}$. The generator network consists of a Long-short term memory (LSTM) network and a feed forward multi-layer perceptron neural network. Besides its ability to store information from an internal memory network, the LSTM network generates each feature sequentially. Details of the tuning of generative and discriminator network parameters are further discussed in chapter 4.

2.1.5 CTGAN (Conditional Tabular GAN)

Lei Xu et al [9] also proposed Conditional Tabular GAN, which has a slight modification from its predecessor, TGAN, where they address tabular datasets that are not gaussian in nature and design a conditional generator with a synthesizer to address imbalanced discrete variables. The CTGAN follows the same pre-processing techniques as the TGAN and uses Wasserstein metric as the loss function with a critic network. Instead of the feed forward network, they use a fully connected neural network. In CTGAN, the objective is to have an effective sampling such that all categories in the discrete variables are evenly sampled and represented and this would also account for imbalance of the data. Firstly they introduce a conditional vector, then construct a generator loss and lastly perform a training by sampling technique. The preprocessing procedures take the following step:

1. For each jth column, create zero filled mask vectors of dimension N_n. Where the mask vector $m_j = \left[m_j^i\right]_{i=1\ldots|D|,\ \ where\ j=1,\ \ldots,\ N_n}$

2. Make a random selection of one discrete column from Nn with equal probability. We name the selected discrete column, j*.

3. Build a probability mass function on the range of values in the selected column j*. Where the probability mass

4. Select a sample i* from the probability mass function in step 3.

5. Select the i* the component of the j* to be one (i.e. $m_{j*}^i = 1$)

6. Concatenate the vectors to give the condition (i.e $m_1 \bigoplus \ldots .m_j * \bigoplus m_{Nn}.$) For example m_1 = [0,1,0] and m_2= [0,0,1], condition = [0,1,0,0,0,1]

The generator in this CTGAN takes in both the random noise and conditional vector as input utilizing cross entropy

2.1.6 Variational Autoencoder (VAE)

Variational autoencoder is another type of deep learning generative model that consists of a duo neural network each with an input and out data, namely the encoder and decoder. The encoder network takes in the data and convert it into a low-dimensional latent space z. The latent space has continuous distribution which allows for interpolation and random sampling. Also, this hidden latent distribution is the output of the encoder and as well the input of the decoder. As seen from figure 2.3 the decoder in turn reconstructs and restructures the vector from the latent distribution back to the original data distribution. Unlike the generator network which does not see the real dataset, the encoder has the real data passed through it as input. The general loss formulation of variational auto encoder is from maximizing an Evidence Lower Bound (ELBO) which comprises of the reconstruction loss and KL divergence. This is mathematical expressed in equation 2.6.

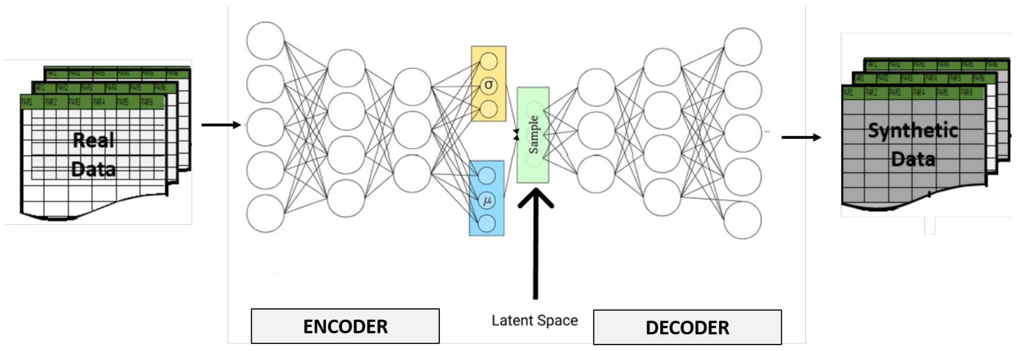

Fig. 2.3: TVAE Network

$$L = -\mathbb{E}_z \left[logp(x|z) \right] - KL(q(z|x)p(z)) \tag{2.6}$$

The first term, which is the reconstruction log-likelihood term puts a penalty on the decoder anytime an output different from the input data is generated. In addition, where p is the normal distribution, a penalty is added if the encoder output does not follow this normal distribution. The KL divergence term computes the information that is lost, basically measuring how close the q distribution is to the p distribution. To optimize the parameters of the distribution to be as close as the real data distribution, the KL divergence has to be minimized. As training begins, VAE uses gradient descent to optimize the loss term with respect to the encoder and decoder parameters.

2.2 Synthetic Minority Over-Sampling Technique for Regression with Gaussian Noise (SMOGN)

SMOGN was proposed by Paula Branco et al in 2017 [21] to produce synthetic dataset using an under-sampling and over-samping technique. This technique is majorly used for regression dataset to address data imbalance, however, in this thesis SMOGN is employed to produce synthetic data from a sparse real dataset. The authors propose two methods for generating the synthetic data, namely, SMOTER and Gaussian Noise, where the former is used with k-nearest neighbor to sample data points that have close distance and the latter for distant data samples. The procedures for SMOGN algorithm are as follows:

1. Partition the data with respect to the target variable based on some pre-defined threshold (i.e. xupper and xlower where the former connotes the relevant data that is less represented above the threshold and the latter represents the less relevant data.
2. Oversampling of the xupper using SMOTER or Gaussian Noise strategy; under-sampling of the xlower .
3. Using k-nearest neighbor, a safe and unsafe region is created. For example, using 5-nearest neighbor, data points within the safe region would use the SMOTER technique, while those in the unsafe region (farther distance) would require generating a new sample using Gaussian Noise.

CHAPTER 3

DATA GENERATION, SIMULATION AND EVALUATION METHODS

In this chapter, two types of datasets are presented to serve as ground truth for validation and training. Each dataset contains parameters and key performance indicators (KPIs) that are relevant in the cellular network domain. The first data, using a system-level environment, employs mobility parameters to determine coverage and throughput while the second data, using a link-level network, utilizes user location and a few hard parameters to determine only coverage (RSRP). Both datasets contain independent and dependent variables that would be passed through the generative and non-generative models. In addition, the simulation process and parameters used to simulate the data will also be expatiated. It is also worthy to note that the datasets contain both discrete and continuous variables.

3.1 KPI and Parameters Definition

3.1.1 Cell Individual Offset (CIO)

This is a measurement quantity used to determine cell association, regulate cell coverage and is usually incorporated in power measurement to control handover. As shown in figure 3.1 handover can be made earlier or later by changing the CIO values. CIO is usually given positive or negative values; with a positive value boosting the RSRP of the cell and a negative value diminishing the RSRP of target cells. Figure 3.1(a) and (b) also shows the effect of suboptimal CIO tuning in terms of SINR. Where a wrongly tuned CIO can cause early handover and the source cell is better than the target cell resulting to high interference and hence poor SINR.

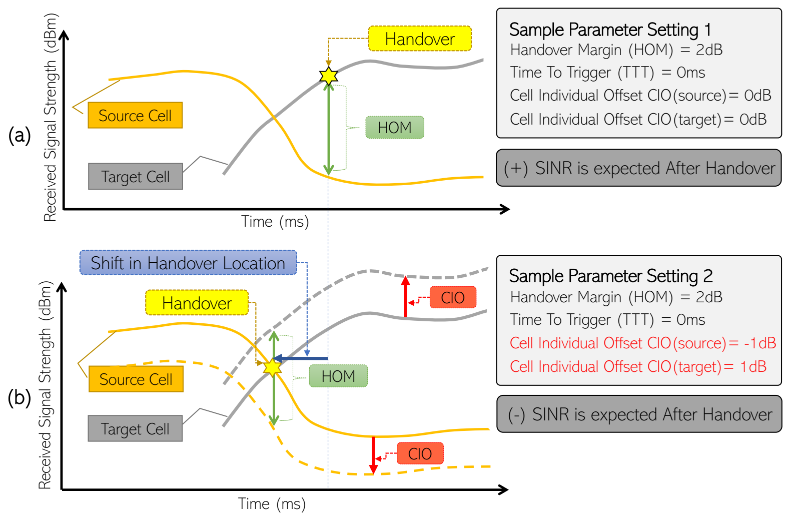
(a)
Received Signal Strength (dBm)
Handover
Source Cell
HOM
Target Cell
Time (ms)
Sample Parameter Setting 1
Handover Margin (HOM) = 2dB
Time To Trigger (TTT) = 0ms
Cell Individual Offset CIO(source)= 0dB
Cell Individual Offset CIO(target)= 0dB
(+) SINR is expected After Handover
(b)
Shift in Handover Location
Handover
CIO
Source Cell
HOM
Target Cell
CIO
Time (ms)
Sample Parameter Setting 2
Handover Margin (HOM) = 2dB
Time To Trigger (TTT) = 0ms
Cell Individual Offset CIO(source)= -1dB
Cell Individual Offset CIO(target)= 1dB
(-) SINR is expected After Handover

Fig. 3.1: Effect of CIO on Handover and SINR

3.1.2 Antenna Tilt

This is a key parameter standardized by 3GPP to determine coverage of a particular network. The antenna tilt adjustment can come in two ways, mechanical and electrical. In this work, we focus more on the mechanical aspect. The mechanical tilt refers to the physical adjustment of the tilt in angular positions. The electric tilting refers to the adjustment of phases of antenna lobes. Signal quality and reception, interference and coverage all depend on this parameter for optimal values.

3.1.3 Antenna Azimuth

This is also another hard parameter that involves rotating the antenna around a vertical axis in 360 degrees to desired position of reference. The ranges used in this work was between [0-3000] for the mobility dataset with optimal values falling in the upper quarter of the range.

3.1.4 Reference Signal Received Power (RSRP)

RSRP, a type of Received signal strength indicator usually measured in dBm, is the level of power signal that a UE detects from a cell when its in the coverage or when the UE moves from one cell to another. With this KPI, one can determine the coverage level or map of a network collectively. A good and strong signal lies between the range of -90dBm to -80dBm upward. This KPI is a fundamental indicator that describes the average power of resource elements.

3.1.5 Signal to Interference and Noise Ratio (SINR)

SINR measures the ratio of power of signal quality to the power of interfering cells and noise. Although not a standard measurement in 3GPP, UE equipment vendors still use it to determine the quality of received signal. Usually, it is not reported to the base station however, it is converted to a Channel Quality Indicator (CQI) to estimate the received signal. SINR is mostly used for radio link quality and throughput measurements. Where an increase in SINR means a proportional increase in capacity, this KPI can be used to determine capacity of a network. This is described in the analytical equation 3.1. As seen in figure 3.1, SINR is commonly used to model handover failure and determine parameters optimization, hence our use in the mobility dataset as a base KPI.

3.2 Mobility Dataset

Before discussing the data generation, we explain the mathematical relationship between the parameters used and the KPIs. In LTE networks, downlink Reference Signal Received Power (RSRP) is used to determine coverage of a cell. Sub optimal and wrong parameters tuning can lead to handover failures low throughput or a bad coverage. Similarly, downlink interference also contributes to low coverages and SINR values. We employ two hard parameters, antenna tilt an azimuth, as well as a soft parameter, Cell

Table 3.1: Mobility Tabular Dataset

CIO1	**Azimuth1**	**Tilt1**	**CIO2**	**Azimuth2**	**Tilt2**	**Mean-RSRP**	**Mean-SINR**
5	240	0	0	240	25	-85.19	14.622
0	120	40	10	120	25	-86.45	17.55
0	180	10	10	120	45	-79.72	16.16
0	300	30	5	300	30	-87.25	11.62
0	300	40	0	240	5	-86.79	14.83

Individual Offset (CIO) to quantify RSRP and SINR, which in turn are KPIS used to measure coverage and capacity. Table 3.1 shows the tabular dataset where the suffix number of each parameter represent the base station i.e CIO1 represent the CIO values for base station 1 and CIO2 represents the CIO values for base station 2.

3.2.1 Analytical Formulation

For basic understanding of the underlying function and the existing relationship between the KPIs and variables, we take the SINR γ_u^k, perceived by a single user with distance from a serving antenna of one sector which is given as:

$$\gamma_u^k \;=\; \frac{P^k G_u^k \;\left(d_u^k\right)^{\beta} \alpha \delta_u^k}{\sum_j P^j G_u^j \;\left(dj_u^j\right)^{\beta} \alpha \delta_u^j \;+\; n_0} \tag{3.1}$$

Where d_u^k is the distance of the user from the interfering cell antenna, P^k and P^j are the transmission powers of the serving cell and interfering cell respectively. δ_u^k and δ_u^j are the shadowing pathloss exponents of the associated cells. n_0 is the noise, α the pathloss constant where G_u^k and G_u^j are the antenna gains of the cells respectively. We model the antenna gain observed by each user as a function of tilt and azimuth as [22]:

$$G_u^k = 10^{-1.2\left(\lambda_v\left(\frac{\varphi_u^k-\varphi_{tilt}^k}{B_v}\right)^2+\lambda_h\left(\frac{\Phi_u^k-\Phi_{azim}^k}{B_h}\right)^2\right)} \tag{3.2}$$

Where φ_u^k is the vertical angle of the serving cell from the user, φ_{tilt}^k is the tilt angle, $\phi_u^k$is the horizontal angle with ϕ_{azim}^k as the antenna azimuth. With β_v and β_h as the vertical and horizontal widths of the beam of antenna , while λ_v and λ_h are the weights ascribed to the vertical and horizontal beam width of the transmitting antennas respectively. Other variables are rendered constants, since we are taking only the tilt, azimuth and CIO into considerations. Finally, we consider the mobility parameter CIO to support load balancing for adjacent cells as well as to complement the received power of the user. This is given as:

$$R_u = R_u^k + CIO \tag{3.3}$$

Where $R_u^k > R_{threshold}$. Here, R_u is the total received power of user with CIO and R_u^k is the actual received power from the serving cell antenna given the condition that the received power from the antenna is greater than a threshold. This condition is used to determine which antenna belongs to the serving cell. Here, R_u is the total received power of user with CIO and R_u^k is the actual received power from the serving cell antenna given the condition that the received power from the antenna is greater than a threshold. This condition is used to determine which antenna belongs to the serving cell.

Previously, mathematical models were used to generate synthetic data for several cellular networks, however, with the increasing network parameters, emerging technologies in 5G, and higher dimensions of parameters, modelling this environment with high precision will not be effective and accurate, hence, the recent reliance on data-driven approaches.

3.2.2 Simulation Setup

The simulation environment consists of a real geographical terrain comprising of 12 macro cells. Each macro cell has 3 sectors with fixed users on a Poisson distribution. Using a ray-tracing model to characterize the propagation and path loss, and as seen in figure 3.2, the simulation adapts a wrapped model to incorporate inference of neighboring cells with the first tier of interference being considered. Each user is served with a cell with the condition that the RSRP is greater than a threshold plus a CIO value. The antenna tilt, azimuth, CIO are within a given industrial standard range with a fixed transmission power as seen in the table 3.2. The range values make up a combinatorial dataset of 6 dimensions and 14400 samples for the two cells that were employed for configuration.

With respect to 3GPP specification, we defined the serving cell selection process in three ways; qualification, pre-selection and final selection.

1. Qualification: To determine if a cell is qualified to be a potential serving cell, the RSRP value received from the user equipment (UE) must be greater than or equal to a defined threshold value.

$$R_u^k \ \geq T_s^k \ + \ Max \ (0, \ R_{threshold}) \tag{3.4}$$

2. Pre-selection: Given the list of potential cells that meet the criteria in one above, the highest RSRP value in the list is pre-selected as the serving cell.

3. Final Selection: A supporting criteria is defined for all qualified cells with the condition that the total RSRP from the best serving cell be added to a CIO to augment the power value. This is explained in the equation.

$$R^j + \ CIO_j \ \geq \ R^k \ + CIO_k \tag{3.5}$$

3.3 Minimization of Drive Test Report (MDT) Dataset

3GPP [2], introduced MDT as a way of support network performance and coverage estimation at base station where reports are collected from different user equipment without the need for the previous method of drive tests. A typical MDT report consists of KPIS like received signal strength using local geographical information of the UE which is then reported to the base station and ultimately the network operators to

Table 3.2: Simulation Parameters

System Parameters	**Value**
No of Macro base stations	12
No of sectors/base station	3
Carrier frequency	2100MHZ
Transmission power	43 dBm
Minimum reference signal received power	-149dBm
Antenna gain	18.5 dBi
No of users in wrapped model	74
Main lobe antenna gain	18dBi
Pathloss model	Ray-tracing
BS height	30m
CIO range	0 dB – 10 dB
Tilt range	0°– 20°
Azimuth range	0°– 300°

produce coverage maps.

3.3.1 Simulation of Coverage Maps from MDT Reports

We utilize a commercial network planning tool with a sophisticated ray-tracing propagation model. The data obtained from this simulation contains RSRP reports of users distributed in a Poisson distribution. To capture and reflect realistic coverage measurements, realistic environment maps in the city of Brussel consisting of buildings, heights, clutters and terrains profiles were utilized. For this simulation, we consider one cell site location which is the middle site location in figure 3.2 with three sectors having the equal coordinates and coverage area divided into bin widths of 5m as shown in figure 3.3. Hence, given a user's point of reference (location), the assumption is that the user is within a circular area with radius r, with the user at the center. To model the effect of shadowing we use zero mean Gaussian distribution with standard deviation in dB. The antenna pattern is modelled based on a 3D antenna model that consists of both vertical and horizontal configurations where the antenna parameters are built in the simulator to create a more realistic model. As seen in table 3.2, a carrier of 2100 MHz

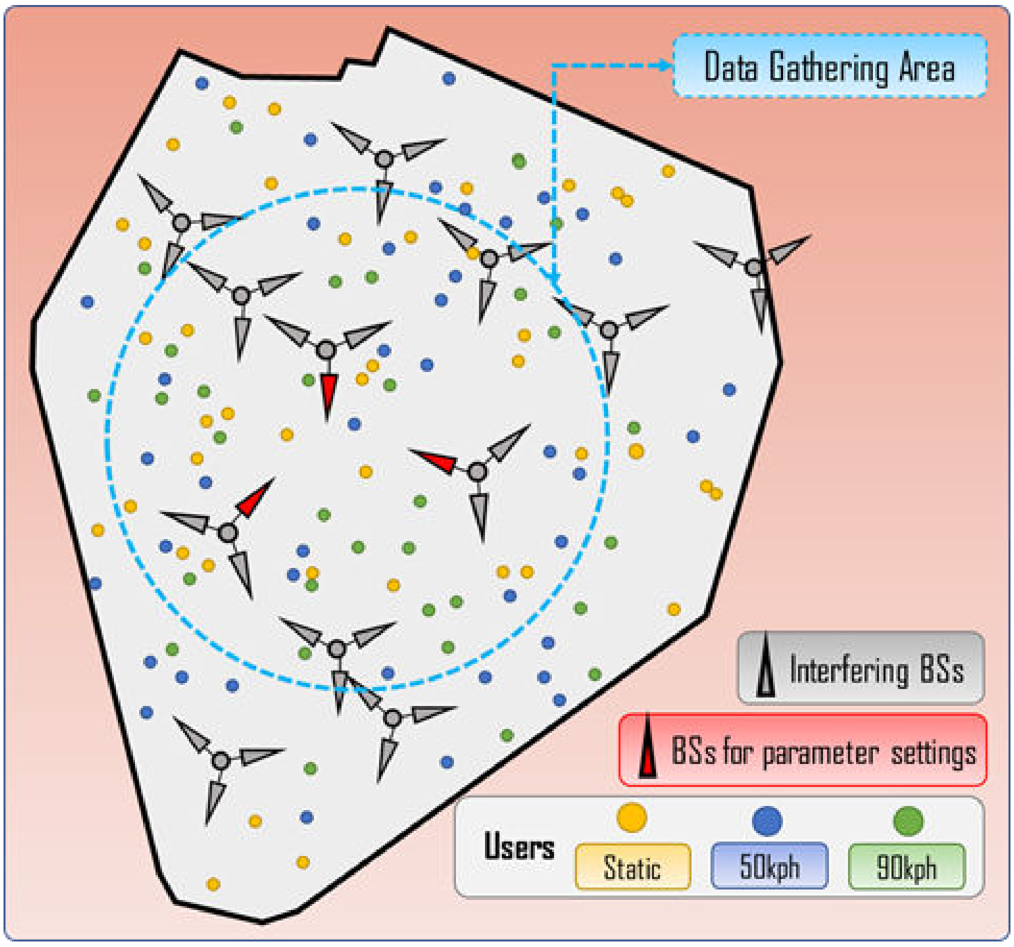

Fig. 3.2: Simulation topology

was employed as is the case in the industry. Given the range of 30m to 45m in typical realistic macro cell height, we incorporated a base station height of 30m stating that any overshoot beyond or below that range will lead to a less effective communication.

The coverage map exists in a tabular form as described in table 3.3, where the X and Y columns represent the coordinate position of each row sample. Each row sample represents a user/user equipment information or location. To visualize the tabular dataset into figure 3.3, we plot the X and Y coordinate of each user against their RSRP value. In addition, after finding the Euclidean distance of each user with the base station's coordinates as reference point, we took the logarithm of the distance value. This dataset is a link level daataset where the RSRP value is to be predicted using the location of the user, X and Y from the serving base station, the azimuth and tilt angle,

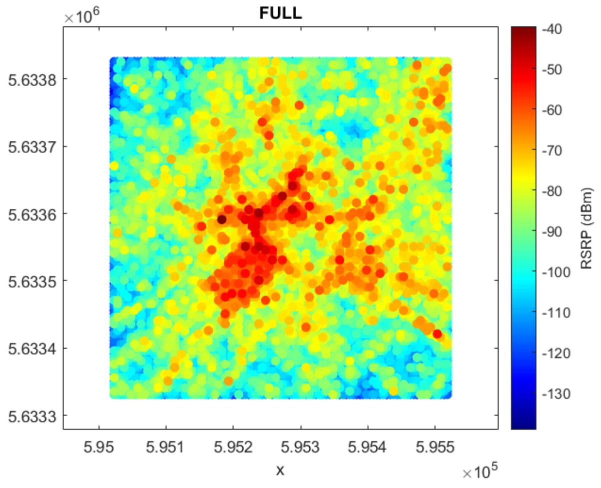

Fig. 3.3: Full Coverage map

all with respect to the antenna bore-sight of the serving base station.

3.3.2 Data Sparsity and Coverage Maps

To test for the potential of the generative models to reproduce similar synthetic dataset from sparse training dataset with complete statistical properties close to ground truth, we introduce different sparse levels into the full coverage map in figure 3.3. Here, the complete training data used to plot figure 3.3 contains 8000 samples, where one percent represents 80 samples, 50 percent represents 4000 data samples. As stated in chapter one, several factors contribute to why full ground truth data is not readily available. Manual operators receive the coverage maps in different sparse measures as seen from figure 3.4.

To formulate sparsity, we use random sampling to get the sparse matrix from the complete matrix using just the X , Y and RSRP values. We sample for 3 different percentages of empty area signifying no users report (white spaces in figure 3.4), this makes a total training data of 1%, 10% and 50%. To convey relevance in cellular network domain, we convert to connote user density in the sample space. For example (320,3200,9600

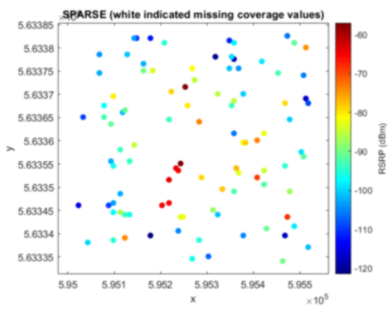

Fig 3.4a : 1% Training data

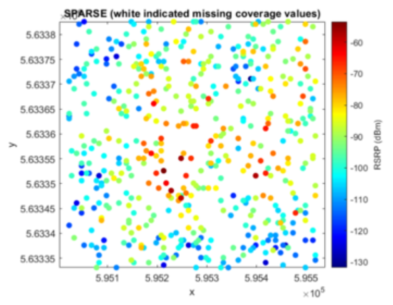

Fig 3.4b : 10% Training data

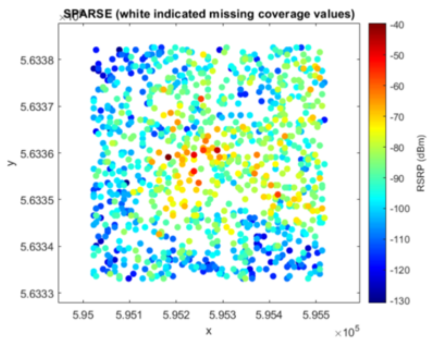

Fig 3.4c :50% Training data

Fig. 3.4: Sparse Coverage map with different percentage of training data

16000) users per square km all representing (1%,10%,50%,100%) of training data. We convey this data in terms of user density because network operators are unable to tell what percentage of data they have access to as compared to full ground-truth, however, with MDT reports they can observe user density information.

We proceed to use four classical machine learning models to interpolate the sparse matrix described in figure 3.4. They are Random Forest, k-nearest neighbors, support vector machine and linear regression. Using the regression-based tabular data, we split into train and test and feed into each ML model using repeated 10-fold cross validation for generalization. This prediction models are further used to predict the white regions in figure 3.4b using 10 percent training data. For visualization purposes, we show the predicted coverage map by converting the RSRP values into a matrix format using the X and Y location as the variables. Of all the models, we observe that k-nearest neighbor

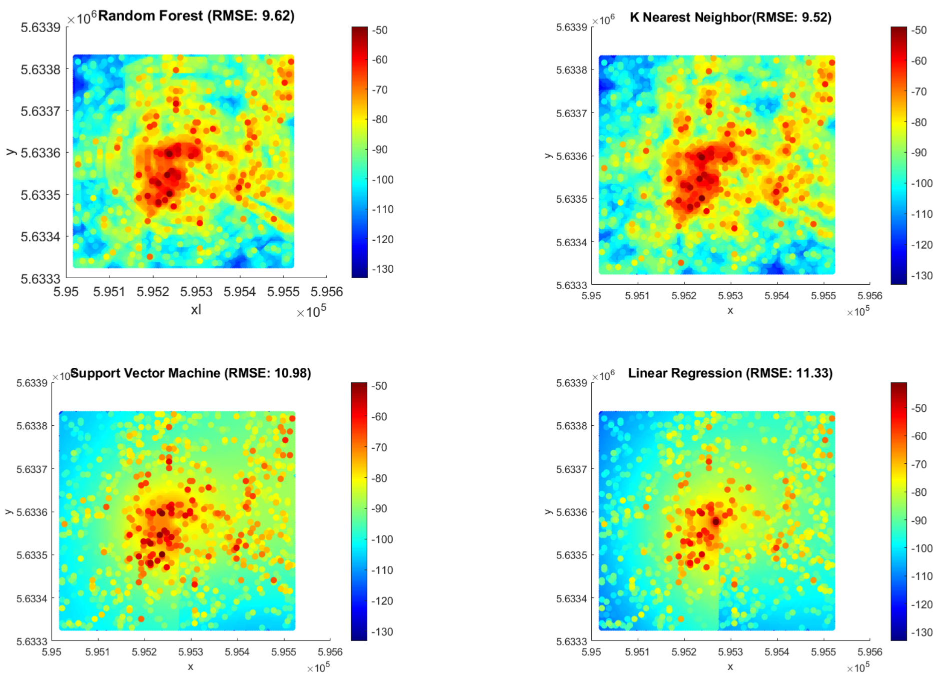

Fig. 3.5: Comparison of selected machine learning algorithms for coverage estimation.

performed the best with the closest map and lowest RMSE value. The k value used here was 7. As seen in figure 3.5, all models were able to predict the midpoint location of the base station with the highest values at the center.

3.3.3 Coverage Map Interpolation using GAN and VAE

Unlike classical model which predict the feature combination of the test data, GAN does not have access to the training data however it can intelligently tell that RSRP values are higher with smaller distance to the base station and vice versa. In addition, both deep models have close RSRP distribution with GAN having a closer spread as compared to VAE. This is seen in figures 3.6. In contrast with the classical machine to predicting the sparse coverage map, generative models like GAN and VAE reconstruct he entire map only by training the networks with the training data. For instance, in GAN networks, the generator does not have access to the training data but can train its network to produce an entire map with sparse relevant data. This explains the efficacy of

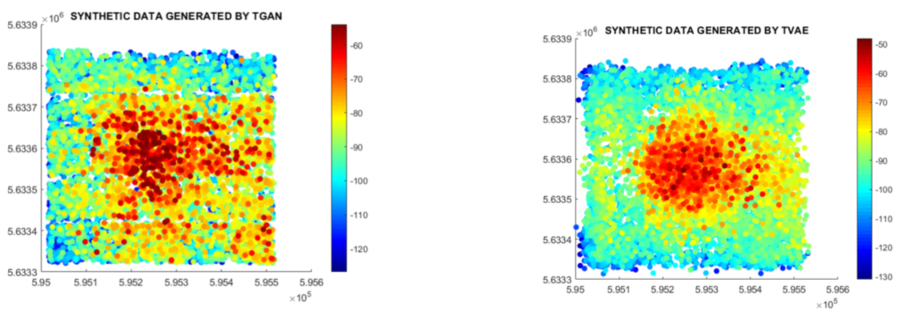

Fig. 3.6: GAN and VAE coverage map reconstruction

Table 3.3: MDT Tabular Datset

X	**Y**	**Distance**	**Azimuth**	**Tilt**	**RSRP**
595023	563331	2.543212	45.0	-4.3	-93.85
595023	5633336	2.538817	45.6	-4.4	-85.57
595023	5633346	2.530033	46.8	-4.5	-98.04
595023	5633351	2.52565	47.4	-4.5	-104.96
595023	5633366	2.512557	49.4	-4.7	-96.28
595023	5633381	2.499604	51.4	-4.8	-92.46

generative models compared to classical machine learning models. In reconstructing an absolute network environment, the take into consideration prediction of missing values from the training data as seen in figure 3.6.

3.4 Evaluation Methods

It is not just enough for the generative models to produce similar tabular dataset but representative dataset with close statistical properties to the real dataset. We divide the evaluation methods in two parts: Statistical measures and machine learning validation. For the statistical measure, we use Spearman's Correlation Coefficient (SCC), the joint distribution of selected parameters and Wasserstein Distance. The machine learning evaluation involves using shap analysis to compare the effect of parameters on the model, as well as RMSE (root mean square error) to evaluate the efficacy and effect of augmentation on the selected model.

3.4.1 Spearman's Correlation Coefficient (SCC)

Spearman's Correlation Coefficient is a metric used to measure the monotonicity if the relationship between two data or columns (variables). SCC can evaluate both linear and non-linear relationships that exists between selected variables, it can also capture distributions other than normal distributions and has less sensitivity to outliers. SCC scores are between the range [-1,+1], where +1 indicates direct proportionality, -1 indicates inverse proportionality and 0 showing no correlation. The formular below is used to compute the SCC score:

$$S_{(r)} \frac{\mathbf{6} \sum r_i^{\mathbf{2}}}{n(n^{\mathbf{2}} - \mathbf{1})} \tag{3.6}$$

with n is the sample size and r is the difference between the variable ranks of observation.

3.4.2 Wasserstein Distance Metric

As described in section 2, this is a metric used to measure the minimum cost it takes to transport a pile or dirt into another. Given two distributions, A and B within a space, we want to measure the cost it takes to make distribution A look like distribution B. Compared to KL divergence, the Wasserstein metric is symmetric in nature and does not demand the probability of both distributions be in the same space. Results using metric are described in section 4 of this thesis.

3.4.3 Machine learning evaluation

To enable zero touch network automation of cellular networks, there will be need for total dependence on artificial intelligence viz a viz machine learning for autonomous processes. Machine learning (ML) models also rely on data, and not just any data but relevant and representative data for their optimum performance. To evaluate the performance of these ML models, several metrics are used depending on the class of

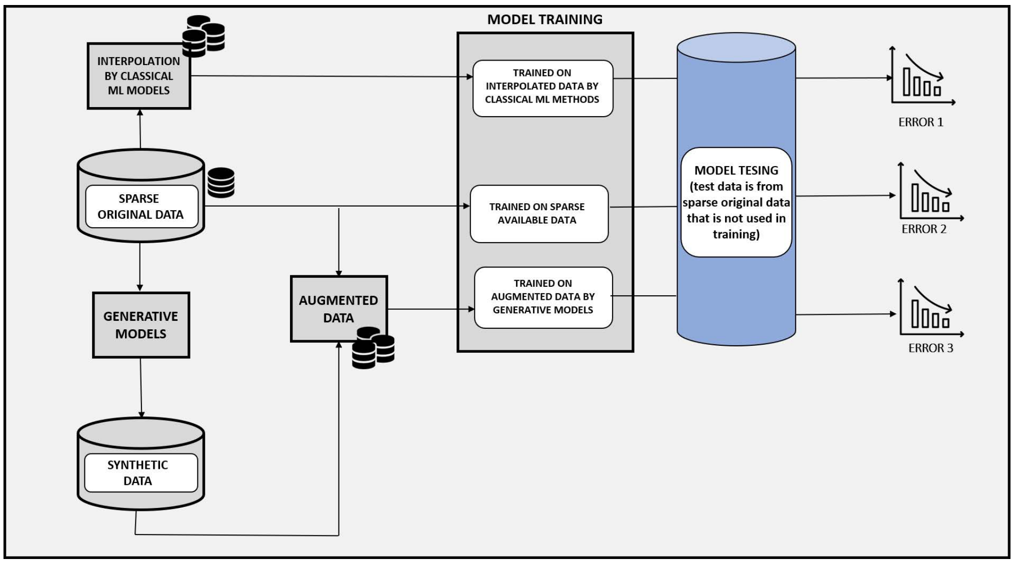
MODEL TRAINING
INTERPOLATION BY CLASSICAL ML MODELS
SPARSE ORIGINAL DATA
GENERATIVE MODELS
SYNTHETIC DATA
AUGMENTED DATA
TRAINED ON INTERPOLATED DATA BY CLASSICAL ML METHODS
TRAINED ON SPARSE AVAILABLE DATA
TRAINED ON AUGMENTED DATA BY GENERATIVE MODELS
MODEL TESING (test data is from sparse original data that is not used in training)
ERROR 1
ERROR 2
ERROR 3

Fig. 3.7: Machine Learning Evaluation Framework

data i.e. classification or regression. For classification data types, accuracy, precision, F1-score are examples of metrics used to evaluate ML performance. For regression, as in our case, we use R-square metrics and Root Mean Square Error (RMSE). Furthermore, we use Shapley Additive Explanations (SHAP) analysis to explain the prediction models performance. For example, if we feed the real training dataset into ML model A and observe the model response using SHAP, we want to vet that when we feed synthetic data into the same model, the same behavior or pattern is repeated. In this thesis, we firstly employ machine learning models to predict RSRP values given different sparse training samples. We visualize their RSRP values using locations as variables. Furthermore, and with respect to the framework illustrated in figure 3.7, we also use the values realized from these models to augment the sparse data. In addition, we employ ensemble learning model (i.e. xgboost) to validate the effect of synthetic data contribution in the area of data augmentation. Here we feed the ensemble model with original data and observe the error values using a separate test data from ground truth. We then go ahead to augment the original data with synthetic data and thereafter feed the same model to observe the error values if they reduce or increase.

3.4.4 Cross Validation (CV)

Additionally, we use cross validation (CV) techniques to ascertain the accuracy of the model and test how the model response can be generalized to different instances of the independent training sets. CV involves taking the average performance of a model when the model has been passed through several partitions of the data. In this case, a test dataset is usually held out for each partitioning case and used to evaluate the accuracy of the model. The basic method involves using a k-fold CV with the training dataset divided into k overlapping sets. Where the model is trained on the k-1 folds and tested on the remainder. In this thesis, we employ Repeated k-fold cross validation (RCV). RCV involves the same technique with k-fold CV except in this case, the procedure is repeated for desired multiple times. We employ this variant of CV because, with the different partitions, the score values can be different, resulting into a varying mean estimate of the model performance scores and this gets worse with noise in the data. For example, if we have 5-fold CV, we repeat the procedure for about 5 times, ultimately, this means that 25 datasets partitions were used to evaluate the model performance.

3.4.5 Root Mean Square Error (RMSE)

This is an accuracy score metric used to measure the difference between observed values y_i and the predicted values $\widehat{y_i}$. It is simply measuring the error or deviation of a model in predicting data and is mostly used for regression datasets. The formular is given below as:

$$RMSE = \sqrt{\sum_{j}^{N} \frac{(\widehat{y_i} - y_i)^2}{N}} \tag{3.7}$$

3.4.6 R-square Metric

This is also known as Coefficient of Determination and is a statistical tool used to measure how the variance of an dependent variable is explained by the variance in the independent variable. It basically explains the strength of the linear relationship that exists between the two variables. Its values usually lie between 0 and 1, with values closer to 1 stating that there is a very high linear relationship between the variance of the dependent and independent variables. Values tending towards negative means that the model has a terrible prediction.

$$R^2 = 1 - \frac{Rs}{Ts} \tag{3.8}$$

Where Ts represents the total sum of squares and Rs, the sum of residual squares.

3.4.7 Shapley Additive Explanations (SHAP Analysis)

SHAP is a concept derived from game theory which is used to explain predictive models based on the contribution of certain features to that model. As opposed to previous times, when machine learning models were considered as black box models, this concept, as the name implies explains the model characteristics using the contribution of a feature or a collection of features. The functions of this analysis cuts across three advantages:
1.Local interpretability: - the variables observed (i.e. features) have their own SHAP values and once can see how each react individually as opposed to the generic feature selection that cuts across all samples. Here we can see the importance of each feature.
2.Global Interpretability: This deals with the collective values of all features and their positive of negative contribution to the target variables.
3.SHAP analysis leverages on tree-based models where the values can be computed.
Mathematically, it is explained as

$$Y(x') = \theta_0 \ + \ \sum_{i=1}^{N} \theta_i x'_i \tag{3.9}$$

With x' being the coalition vector, N is the maximum possible coalition size and Y the explanation model. θ_i is feature characteristics for a variable i. A huge drawback of SHAP analysis is in the computational time it takes especially for models that are not tree-based. Results using this analysis are explained in section 5.

CHAPTER 4

MODELS HYPERPARAMETERS AND TUNING

In this chapter, the hyperparameters attained in giving the best output for both generative models and machine learning models as well as implementation details are discussed. The techniques employed are elaborated as well as the final tuned parameters for each of the generative networks. We also discuss the importance of tuning parameters with respect to data augmentation.

4.1 Machine learning models tuning

For every level of training data fed into each ML model used in this work, we tuned the parameters of the model to give the best score values. RMSE and R-squared were the metrics used to benchmark all model's performance and accuracy. The technique used in this thesis was the GridSearchCV (GSV). The optimal performance of a model depends on the nature of the data fed and most importantly the value of the hyperparameters of the model. In a case, where the data has been preprocessed and contains relevant information, more focus is spent on tuning the hyperparameters. Using a manual process especially for a parameter that has a wide range of values is time consuming, hence the reliance on GSV. We leverage on the library package from Scikit-learn for this purpose. GSV operates by looping through a set of predefined range of hyperparameters assigned to an estimator or model while looking for the best combination that produces the best output. These predefined hyperparameters values are listed in a dictionary and further automated for combination. Four types of ML models were employed in this thesis including a baseline model; ensemble learning models, Random Forest (RF), three other classical methods, K-Nearest Neighbor K-NN), Support Vector Machine (SVM) and the base line Linear Regression (LR). The best K values for the KNN model were between 5 and 9 for all training data depending on the sparsity level of the training data. SVM had radial basis function as the best kernel with gamma set to 'scale'. The ensemble learners had similar parameters but with different values. The parameters used for both RF and XGB were colsample-bytree, n-estimators, mininmum-samples, maximum-depth, gamma, learning rate and sub-sample. It is worthy to note that there was no one best value for each parameter because with different levels of sparsity and training data, the optimal values differed. However, with increase in training data, the maximum depth seemed to increase linearly. In addition, n-estimators had values between 100 and 150.

4.2 Generative Models Tuning

Parameter tuning for TGAN and TVAE is not a trivial task considering the amount of time it takes to train the real data for each combination of parameter. Using AdamOptimizer as the optimizer function, columns set to continuous, the use of brute force was employed to realizing the best optimal combination of parameters. The parameter values are displayed in table 4.1. Here several metrics were used to bench mark TGAN and TVAE performance with respect to the hyperparameters tuned. From Wasserstein distance to data visualization and machine learning performance using RMSE, all three objectives were utilized. It was observed that the poorer the performance of TGAN and TVAE training to produce authentic data, the higher the cost or Wasserstein distance. Additionally, increase in RMSE scores in comparison real data when passed through ML model signified a poor performance and vice versa.

Table 4.1: Table Showing the tested hyperparameters for TGAN

Hyperparameter	**Values**
Batch size	50,100,150,200,300
Maximum Epoch	100,300,500
Generator nodes	100,200,300,400,500
Discriminator nodes	100,200,300,400,500
Learning rate	0.0002,0.0005,0.001

Table 4.2: Table Showing the tested hyperparameters for TVAE

Hyperparameter	**Values**
Encoder dimension	(256,128), (256,512)
Decoder dimension	(256,128), (256,512)
L2scale	1e-5, 1e-4, 1e-3
Batch size	50,150,200
Epochs	100,300,500

CHAPTER 5

RESULTS

This chapter discusses the potential of synthetic data for network automation, model performance improvement, data augmentation and data interpolation. The authenticity of synthetic data is first validated using the statistical properties described in chapter 3, followed by machine learning validity. Following that, we check the potential of synthetic data in mitigating effects of sparse data, that is, how synthetic data can be augmented with sparse data to achieve expected result. This includes improving the accuracy or reducing the error of a selected machine learning. Lastly, the effect of hyperparameter tuning is discussed, where we observe model performance with and without tuning its parameter.

5.1 Synthetic Data evaluation

5.1.1 Spearman's Correlation Coefficient (SCC)

We start by using correlation coefficients to measure the relationship between variables in the real and synthetic dataset. Relevant works validate synthetic data by using Pearson Correlation Coefficient (PCC) to test for correlation. PCC measures the linear correlation or relationship between two datasets. The challenge with using this technique comes with a few limitations; firstly, PCC is limited to measuring variables that have linear relationships, secondly, it adapts only normal distribution. For this thesis, we move beyond PCC and use SCC to evaluate our synthetic data. As described in chapter 3, SCC measures the association of selected variables in a dataset, irrespective of their linearity. Scores close to +1 imply that the variables are directly proportional, while values close to -1 imply inverse proportionality. In table 5.1, we use the synthetic data gotten from GAN to compare with various training samples of ground truth.

Table 5.1: Spearman's Correlation Coefficient between RSRP and two variables (distance and tilt) of original and GAN synthetic data for MDT Dataset

Training size	Original (Distance)	GAN(Distance)	Original (Tilt)	GAN (Tilt)
1%	-0.44	-0.27	-0.44	-0.35
5%	-0.56	-0.57	-0.51	-0.56
10%	-0.46	-0.38	-0.40	-0.29
50%	-0.46	-0.41	-0.40	-0.43
100%	-0.47	-0.67	-0.41	-0.68

Negative values imply that increased distance away from the base station will yield a reduced RSRP value. GAN can reflect this relationship with both the distance and tilt feature with respect to RSRP.

5.1.2 Synthetic data distribution

In addition to the SCC values, we visualize the joint distribution and compare plots between original and synthetic data using VAE and GAN. With one percent of training data fed into GAN, the synthetic data can give a distribution and scatter plot representation as close as that of the full ground truth. The original plot mentioned in the figures 5.1 and 5.2 are the full ground training data and not the one percent training data used to train both networks. We state that this type of behavior can only occur if the sparse data fed into the network has sufficient information and characteristics between the parameters and KPIS.

For the mobility dataset, we compare the distribution of mean RSRP and mean SINR using Empirical Cumulative Distribution Function (ECDF) as seen in figure 5.3. In the exception of CTGAN, all other models had close distribution with the real data, TGAN and WGAN having the closest. We recall that WGAN had extra training technique, where each iteration, the discriminator was updated 5 times. These evaluations are

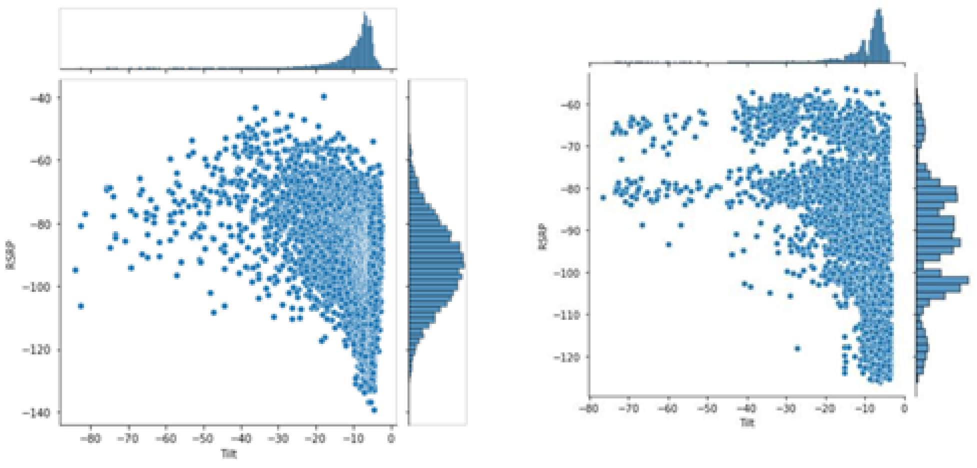

Fig. 5.1: Joint distribution between RSRP and tilt values.

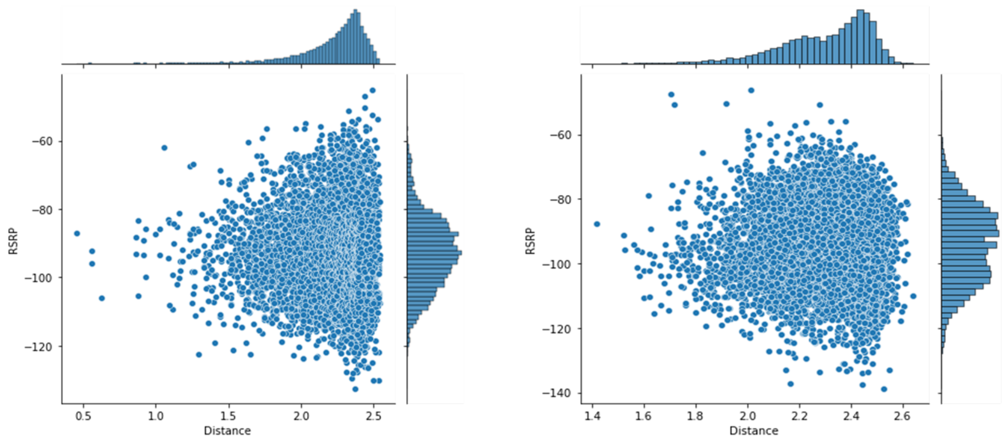

Fig. 5.2: Joint distribution between RSRP and distance values.

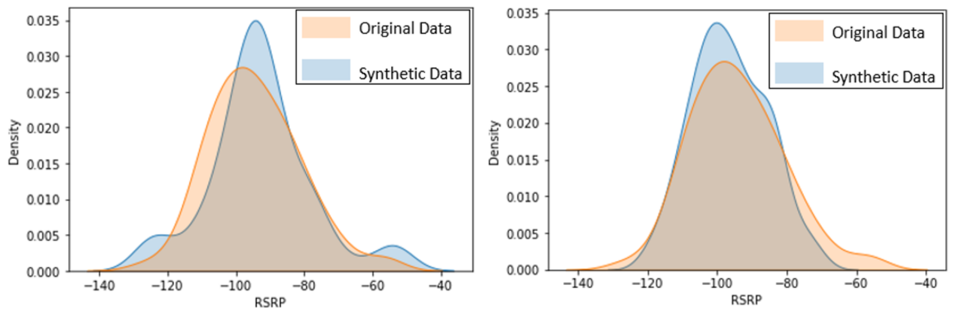

Fig. 5.3: Synthetic data RSRP distribution

necessary to vet the authenticity of synthetic before testing on ML models because, a misrepresented synthetic data will only lead to a degraded performance of an ML model.

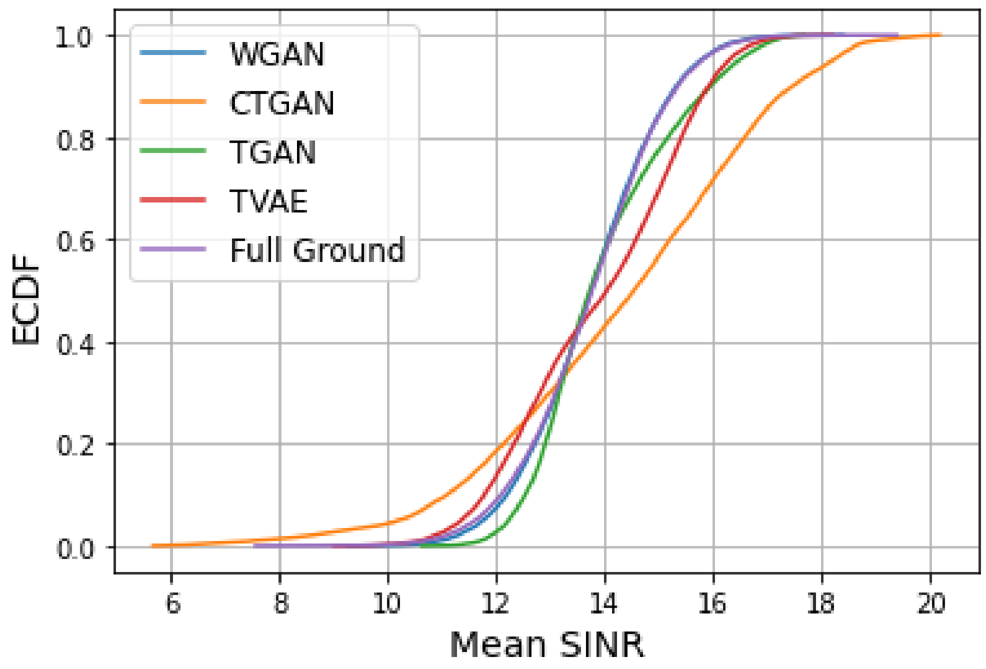

Fig. 5.4: Comparison of Mean SINR distribution of synthetic data from different generative method with ground truth distribution

5.1.3 Pairwise Correlation Difference (PCD)

We also verify and ascertain that the generative model can preserve the correlation that exist between features. As expressed in equation 5.1, we take the difference of the Pearson correlation of the matrices of the ground truth and synthetic data considering their vector or Frobenius norm. PCD computes the measure of correlation that each of the generative models can depict.

$$P(x_m, \hat{x}_m) = [||Corr(x_m) - Corr(\hat{x}_m)||]F \tag{5.1}$$

Where x_m and $\hat{x}_m$ connotes the feature matrix of the original data and synthetic data respectively. From figure 5.6, the closer the values are to zero from the legend, the closer the synthetic data to the real dataset in terms of linear correlation for all features represented. Generally, all models have close to zero correlation with CTGAN performing the best wit smaller feature values in terms of correlation. This is because CTGAN employs conditional probability to model features from the real dataset.

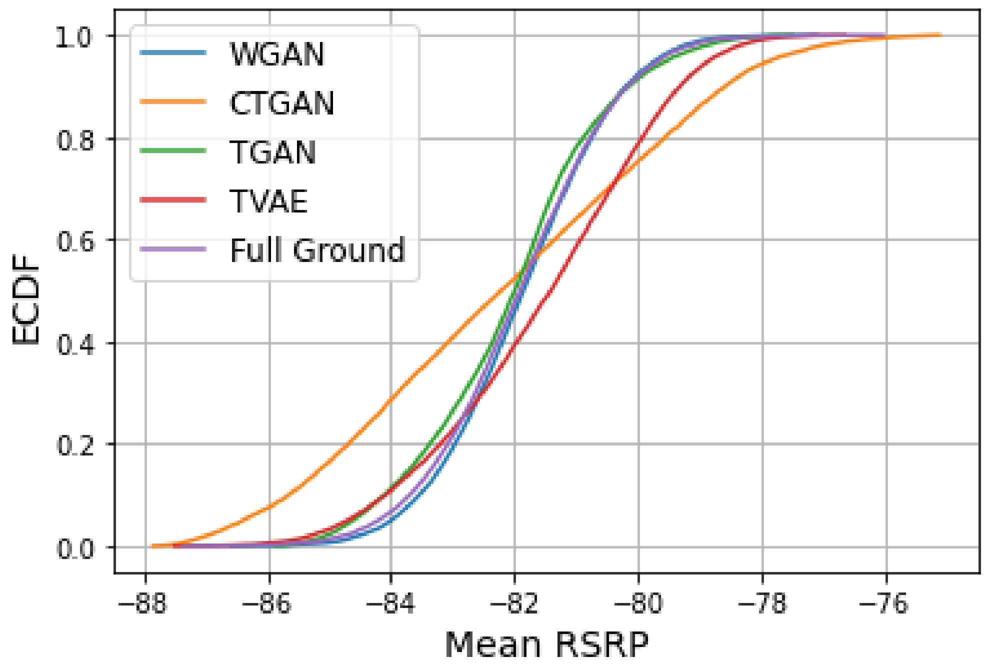

Fig. 5.5: Comparison of Mean RSRP distribution of synthetic data from different generative method with ground truth distribution

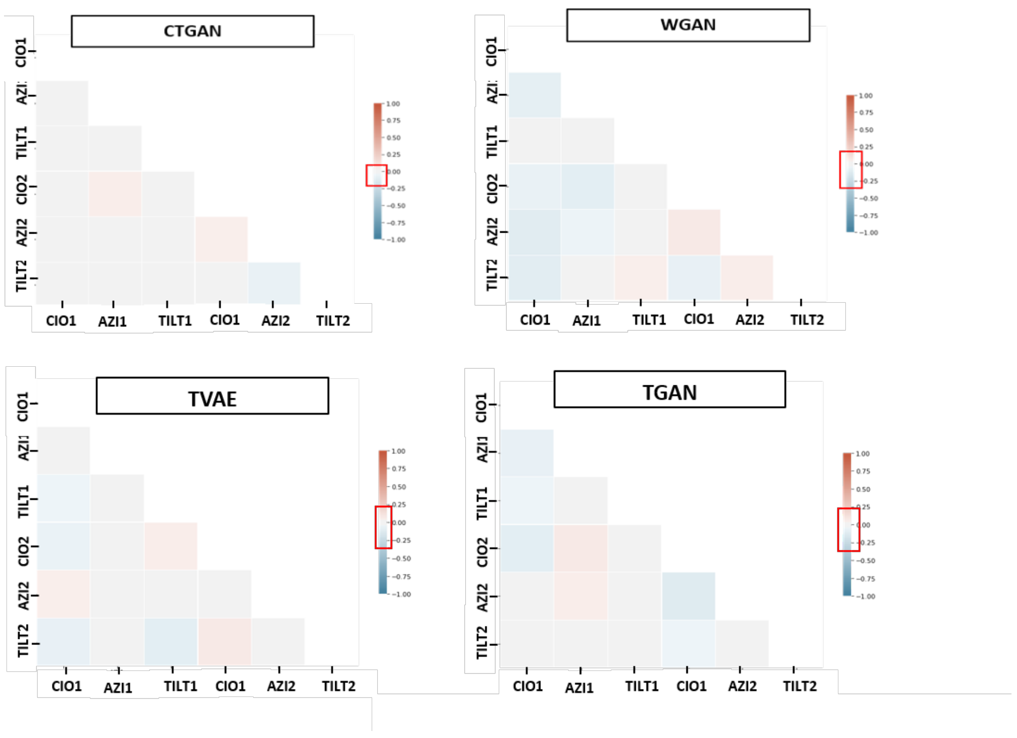

Fig. 5.6: Pairwise Correlation Difference of Synthetic gotten from all generative models for Mobility Dataset.

5.1.4 Feature importance comparison with SHAP

For operators to tune certain parameters to achieve desired KPI, it is essential for them to ascertain the effect of each parameter. This parameter to KPI relationship is still currently an albatross as manual operators avoid real-time adjustment of network parameters given the fragility of the network. Given the paradigm shift to data-driven models, AI will be utimatlety harnessed to associate the parameter to KPIs relationship. ML models are known for their ability to understand underlying functions that exist in certain data types. The interpretability of these models is key to understanding the hidden functions that may exist in the data. Using SHAP analysis we test for the marginal contribution of each feature used in the real mobility dataset according to the level of importance and verify if the same contribution exists in the dataset. We used an ensemble tree-based model for this regression task.

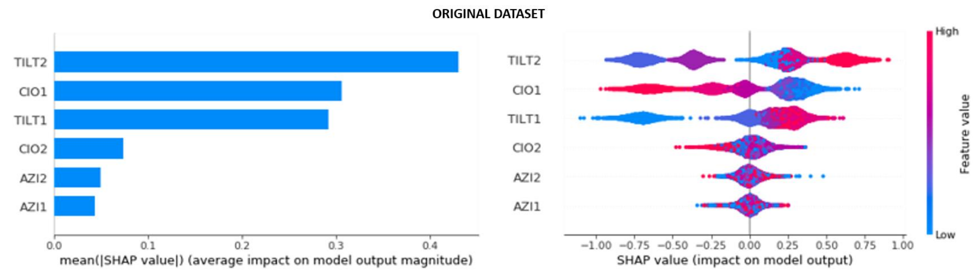

Fig. 5.7: SHAP analysis of real dataset

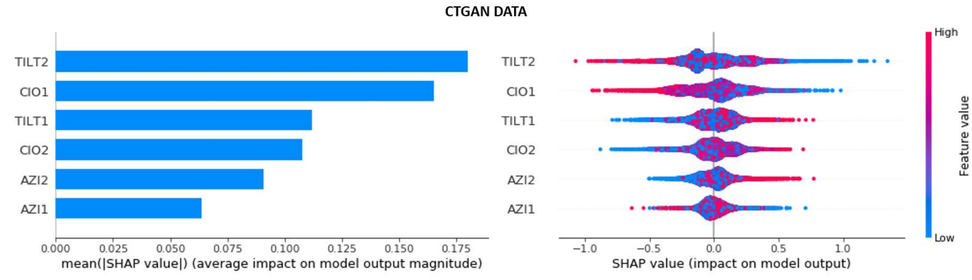

Fig. 5.8: SHAP analysis of CTGAN dataset

From figure 5.6, we observe that Tilt and CIO have the most contribution on the model performance on average RSRP and SINR prediction where the number succeeding each

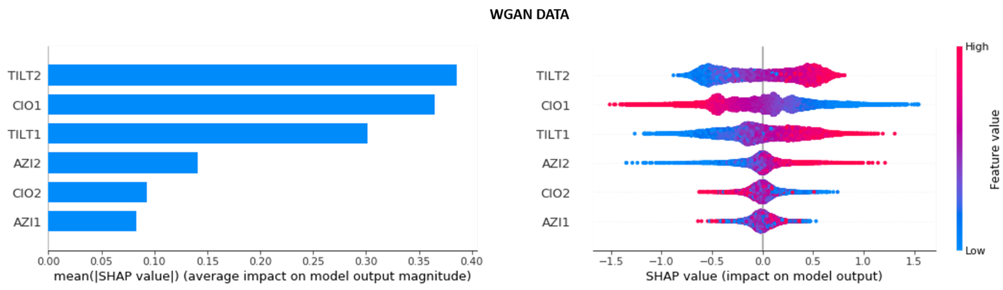

Fig. 5.9: SHAP analysis of WTGAN dataset

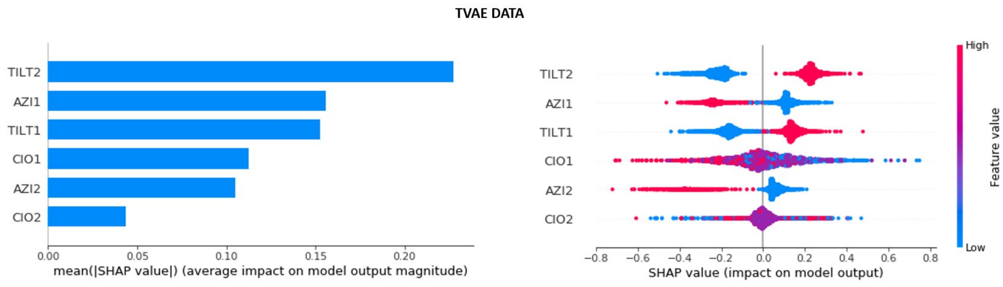

Fig. 5.10: SHAP analysis of TVAE dataset

parameter in the figure represents the base station. We feed the ensemble learning with the original dataset to train and then observe feature importance and impact on the model while doing the same for all generative models but this time feeding them with the same synthetic data. From figure 5.7 to 5.10, the left-hand diagram shows average individual feature impact on the model performance while the right-hand show the positive or negative impact on the model. For instance, using the original dataset, higher values of CIO have negative impact on the model while higher values of tilts have positive impact on the model. Using SHAP analysis will show if similar impacts occured with trained synthetic data. Synthetic data from CTGAN had the same feature importance in comparison with other models, however, WGAN is able to model correctly the positive or negative impact of each feature as described from the real dataset.

5.2 Data Augmentation

As mentioned in chapter 3, we convey training sample size in terms of user's density that is, user per square kilometer. We reiterate that network operators only observe the network in terms of user density rather than in training size.

We only incorporated different training size to observe their effect on ML models. We further evaluated the response of an ensemble learning by observing its performance with respect to errors when training with sparse data and with data augmented from the classical, deep learning and state-of-the-art models. In this augmentation process, we selected the best classical ML model, KNN, alongside SMOGN to compare with the deep learning generative models. As observed from figure 5.11, GAN achieved the lowest RMSE value for both the lowest and highest user density. This shows that with little but relevant, GAN is able to generate synthetic data that can improve performance of testing models.

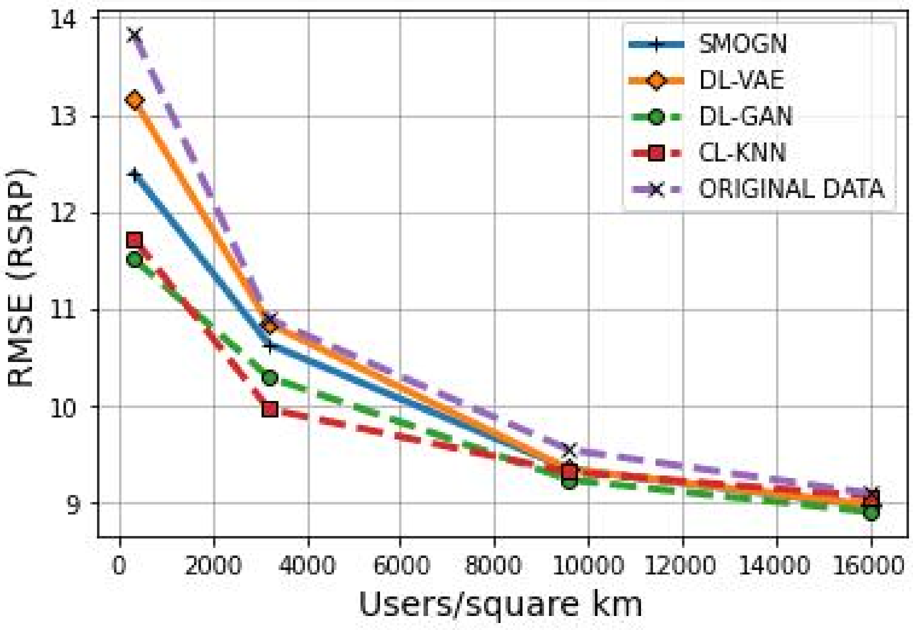

Fig. 5.11: Comparison of data augmentation techniques.

To observe the effect of parameter tuning in increasing performance, we compared the performance of XGboost with default parameters and with tuned parameters using the synthetic data from GAN. We utilized grid-search for each training sample to obtain the best parameter values. The hyperparameters tuned were the maximum, depth, estimators, learning rate and gamma values. From figure 5.12, it was observed that values

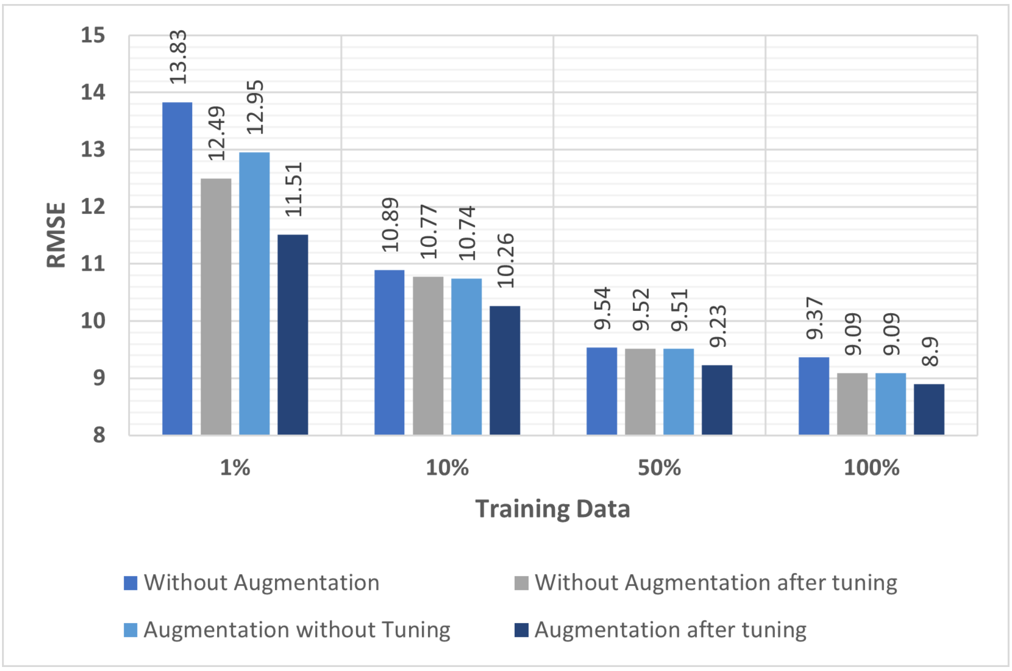

Fig. 5.12: Effect of model parameter tuning.

of maximum depth increased proportionally to the size of the data. ML model hyper-parameter optimization plays an important role in the field of artificial intelligence as different models respond differently to the quantity and quality of data fed into them. The importance of parameter tuning for each level signifies the ability of AI models to adjust their parameters accordingly to different datasets which can also be applied to time-varying data. For seamless operation of zero-touch network automation, intelligence in the network is required to autonomously configure and optimize its parameters given the network reaction to users mobility, interference, environmental conditions and newly integrated sites.

CHAPTER 6

Conclusion and Future Work

In this thesis, we investigated the application of synthetic data to support Network automation which consequently helps to barest minimum human intervention in optimization of network parameters. We incorporated deep learning generative models to produce synthetic data for two types of dataset; mobility and MDT report. We further validated the authenticity of the synthetic data generated by comparing distribution and similarity with real dataset (which we assume that we do not have access to). In addition to deep generative models, we also investigated synthetic data from classical machine learning methods and also from state of the art sampling methods like SMOGN. We validate by using several statistical approaches and metrics for validation.

After validation, we see the impact of synthetic data on Machine learning models in reducing the error of the models. To see the effect of deep generative models, we compare with classical machine learning methods and SMOGn and observed that deep learning generative models in particular, GAN, are capable of learning intrinsic properties of real training dataset with as little as 10 percent. This is only possible if the dataset have relevant information and properties between variables and KPIs.

Although, this work concentrated on static network environment, for future works, we will work on generating synthetic data in a dynamic environment with larger dimensions (that is more network parameters). Since real networks are have time-varying properties, future direction would be geared towards optimizing network parameters through the aid of synthetic data derived in a dynamic environment

Bibliography

[1] A. Imran, A. Zoha, and A. Abu-Dayya, “Challenges in 5G: how to empower SON with big data for enabling 5G,” *IEEE Network*, vol. 28, no. 6, pp. 27–33, 2014.

[2] V. . 3GPP, document TS 37.320, “Universal Terrestrial Radio Access (UTRA) and Evolved Universal Terrestrial Radio Access (E-UTRA); Radio measurement collection for Minimization of Drive Tests (MDT); Overall description; Stage 2 Release 10,” 2010.

[3] A. Goncalves, P. Ray, B. Soper, J. Stevens, L. Coyle, and A. P. Sales, “Generation and evaluation of synthetic patient data,” *BMC Medical Research Methodology*, vol. 20, pp. 1–40, 2020.

[4] J.-S. Ryu, M.-S. Kim, K.-J. Cha, T. H. Lee, and D.-H. Choi, “Kriging interpolation methods in geostatistics and dace model,” *KSME International Journal*, vol. 16, no. 5, pp. 619–632, 2002.

[5] J. A. Parker, R. V. Kenyon, and D. E. Troxel, “Comparison of interpolating methods for image resampling,” *IEEE Transactions on medical imaging*, vol. 2, no. 1, pp. 31–39, 1983.

[6] A. S. Agung Setianto and T. T. Tamia Triandini, “Comparison of kriging and inverse distance weighted (idw) interpolation methods in lineament extraction and analysis,” *Journal of Southeast Asian Applied Geology*, vol. 5, no. 1, pp. 21–29, 2013.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[8] Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, and L. Carin, “Variational autoencoder for deep learning of images, labels and captions,” in *Advances in neural information processing systems*, 2016, pp. 2352–2360.

[9] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, “Modeling tabular data using conditional gan,” in *Advances in Neural Information Processing Systems*, 2019, pp. 7335–7345.

[10] O. Onireti, A. Zoha, J. Moysen, A. Imran, L. Giupponi, M. A. Imran, and A. Abu-Dayya, "A cell outage management framework for dense heterogeneous networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 2097–2113, 2015.

[11] H. Rizk, A. Shokry, and M. Youssef, "Effectiveness of data augmentation in cellular-based localization using deep learning," in *2019 IEEE Wireless Communications and Networking Conference (WCNC).* IEEE, 2019, pp. 1–6.

[12] U. Paul, L. Ortiz, S. R. Das, G. Fusco, and M. M. Buddhikot, "Learning probabilistic models of cellular network traffic with applications to resource management," in *2014 IEEE International Symposium on Dynamic Spectrum Access Networks (DYSPAN).* IEEE, 2014, pp. 82–91.

[13] S. Sevgican, M. Turan, K. Gökarslan, H. B. Yilmaz, and T. Tugcu, "Intelligent network data analytics function in 5g cellular networks using machine learning," *Journal of Communications and Networks*, vol. 22, no. 3, pp. 269–280, 2020.

[14] W. Wang, Q. Liao, and Q. Zhang, "Cod: A cooperative cell outage detection architecture for self-organizing femtocell networks," *IEEE Transactions on Wireless Communications*, vol. 13, no. 11, pp. 6007–6014, 2014.

[15] R. Myung, S. Choi, W. Choi, H. Yu, D. Lee, and E. Lee, "Elaborate synthetic data generation for internet of things services at smart home environment," in *2016 International Conference on Computational Science and Computational Intelligence (CSCI).* IEEE, 2016, pp. 226–229.

[16] T. Zhang, K. Zhu, and D. Niyato, "A generative adversarial learning-based approach for cell outage detection in self-organizing cellular networks," *IEEE Wireless Communications Letters*, vol. 9, no. 2, pp. 171–174, 2019.

[17] Z. Wang, J. Hu, G. Min, Z. Zhao, and J. Wang, "Data augmentation based cellular traffic prediction in edge computing enabled smart city," *IEEE Transactions on Industrial Informatics*, 2020.

[18] B. Hughes, S. Bothe, H. Farooq, and A. Imran, "Generative adversarial learning for machine learning empowered self organizing 5G networks," in *2019 International Conference on Computing, Networking and Communications (ICNC).* IEEE, 2019, pp. 282–286.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair,

A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[20] L. Xu and K. Veeramachaneni, “Synthesizing tabular data using generative adversarial networks,” *arXiv preprint arXiv:1811.11264*, 2018.

[21] P. Branco, L. Torgo, and R. P. Ribeiro, “Smogn: a pre-processing approach for imbalanced regression,” in *First international workshop on learning with imbalanced domains: Theory and applications.* PMLR, 2017, pp. 36–50.

[22] F. T. T. v. M. . 3rd Generation Partnership Project, Sophia Antipolis, “Further advancements for E-UTRA physical layers aspects; Overall description; (Release 9),” 2010.