DEEP LEARNING BASED PREDICTION OF PATH LOSS VALUES AND DISTRIBUTIONS FROM SATELLITE IMAGES

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF

ENGINEERING AND NATURAL SCIENCES

OF ISTANBUL MEDIPOL UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

ELECTRICAL, ELECTRONICS ENGINEERING AND CYBER SYSTEMS

By

Ahmed Mohamed Nagib Ibrahim Marey

February, 2021

Deep Learning Based Prediction of Path Loss Values and Distributions from Satellite Images
By Ahmed Mohamed Nagib Ibrahim Marey
February, 2021

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Bahadır K. Güntürk (Advisor)

Prof. Dr. Cabir Vural

Prof. Dr. Hasan F. Ateş

Approved by the Graduate School of Engineering and Natural Sciences:

Assoc. Prof. Dr. Yasemin Yüksel Durmaz
Director of the Graduate School of Engineering and Natural Sciences

Foreword

This thesis marks the final chapter of my master study in Electrical-Electronics Engineering and Cyber Systems program at Istanbul Medipol University.

In this thesis, a new method was proposed for estimating excessive path loss in a point-wise manner using satellite images and height maps. The proposed method showed satisfactory performance to make it usable practice. Detailed comparisons were also made about using height maps and satellite images as input to convolutional neural networks.

This work would not have seen the light without the precious guidance and support of my advisor, Prof. Dr. Bahadır K. Güntürk. I would also like to send my gratitude to all of my classmates in IMU. Finally, I am thankful to my family which supported me during this journey.

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: AHMED MOHAMED NAGIB IBRAHIM MAREY

Signature :

Acknowledgement

The research in this thesis was supported by The Scientific and Technological Research Council of Turkey (TÜBİTAK) Grant No: 215E324.

I would like to express my sincere gratitude to my advisor Prof. Dr. Bahadır K. Güntürk for all the support he generously gave me during the research, and for his patience and motivation. Without his patience and commitment, this work would have never been completed. I learned a lot from him, and he was a great mentor to me.

I would like to give special thanks to Prof. Dr. Hasan F. Ateş, Ass. Prof. Dr. Tuncer Baykaş. My completion of this work could not have been accomplished without their support and guidance.

I am thankful for this time that has allowed me to become a more knowledgeable person who has overcome the different obstacles I have faced and made me who I am now.

Finally, I would like to take this chance to mention that without the support of my family, and without their encouragement and sincere love, nothing would have had a meaning at all.

Contents

List of Figures

List of Tables

List of Symbols

$\boldsymbol{\eta}$	Excessive path loss
$\boldsymbol{G(h_{te})}$	Transmitter altitude gain
$\boldsymbol{G(h_{re})}$	Receiver altitude gain
$\boldsymbol{G_{AREA}}$	Environmental gain
$\boldsymbol{f_c}$	Frequency (in MHz)
$\boldsymbol{h_{te}}$	Transmitter altitude
$\boldsymbol{h_{re}}$	Receiver altitude
$\boldsymbol{d}$	Distance between transmitter and receiver
$\boldsymbol{a(h_{re})}$	Correction factor
$\boldsymbol{L_F}$	Free space path loss
$\boldsymbol{PL_n}$	Power received
$\boldsymbol{P_T}$	Power transmitted
$\boldsymbol{FSPL_n}$	Free space path loss for the n^{th} receiver
$\boldsymbol{d_n}$	Distance between n^{th} receiver and the transmitter

Abbreviations

CNN Convolutional Neural Network. 2–4, 15, 23, 31, 57, 59, 60

GAN Generative Adversarial Network. viii, 3, 4, 13, 32, 33, 36, 59, 60

ILSVRC2014 ImageNet Large Scale Visual Recognition Challenge 2014. 22

MSE Mean Squared Error. 12, 30, 56–58

PDF Probability Distribution Function. 14, 15, 17, 23, 31, 39, 56

ÖZET

UYDU GÖRÜNTÜLERİNDEN YOL KAYBI DEĞERLERİNİN VE DAĞILIMLARININ DERİN ÖĞRENME TABANLI TAHMİNİ

Ahmed Mohamed Nagib Ibrahim Marey
Elektrik-Elektronik Mühendisliği ve Siber Sistemler, Yüksek Lisans
Tez Danışmanı: Prof. Dr. Bahadır K. Güntürk
Şubat, 2021

Açık alan iletişim sistemlerinin sağlam, etkili ve hızlı bir şekilde konuşlandırılmasının kritik olduğu günümüzde, pek çok kişi insansız hava araçları ve balonlar veya düşük yörüngeli uyduların havadan şebekelerinden kablosuz bağlantı hizmetleri sağlamaya çalışmaktadır. Bu amaç farklı araziler ve bina özellikleri için hızlı ve ayrıntılı analiz yapılmadan elde edilemez. Işın izleme yöntemleri ya da ampirik matematiksel modeller veya pratik ölçüm anketleri gibi mevcut yöntemler yeteri kadar genel ya da hızlı değildir. Bu tezde, kablosuz ağ planlamasında aşırı yol kaybını tahmin etmek için çevrim içi analize, hızlı kuruluma uygun ve yeterince doğru iki çözüm öneriyoruz. İlk olarak aşırı yol kaybının olasılık dağılımını tahmin etmek için VGG-16 gibi bir evrişimsel sinir ağı kullanılmasını öneriyor, uydu ve yükseklik haritasının girdi olarak kullanılmasıyla elde edilen sonuçları karşılaştırıyoruz. İkinci olarak, noktasal yol kaybının kestirimi için koşullu bir Üretici Çekişmeli Ağ geliştiriyoruz. Bir bölgenin yol kaybı haritasının kestirimi için üretici ve ayırtaç ağ eğitiminde çekişmeli kayıp fonksiyonu kullanmayı öneriyoruz. Önerilen yaklaşım için uydu görüntülerinin sonuçları gösterilmiş ve yükseklik haritasının girdi olarak kullanılmasıyla elde edilen sonuçlarla karşılaştırılmıştır.

Anahtar sözcükler: Derin öğrenme, yükseklik haritaları, kanal parametrelerinin kestrimi, regresyon, yol kaybı kuvveti, gölgeleme faktörü, aşırı yol kaybı, insansız hava aracı, havadan karaya iletişim sistemi.

ABSTRACT

DEEP LEARNING BASED PREDICTION OF PATH LOSS VALUES AND DISTRIBUTIONS FROM SATELLITE IMAGES

Ahmed Mohamed Nagib Ibrahim Marey
M.S. in Electrical, Electronics Engineering and Cyber Systems
Advisor: Prof. Dr. Bahadır K. Güntürk
February, 2021

It is critical to provide rapid deployment of robust and effective outdoor communication systems in today's world, where many seek to provide wireless connection services from aerial grids of drones and balloons, or low orbit satellites. This goal cannot be achieved without fast and detailed analysis for different terrains and building characteristics. Current methods, such as ray tracing simulations, empirical mathematical models or practical measurement surveys, are neither generally applicable nor fast enough. In this thesis we propose two solutions which are suitable for online analysis, rapid deployment and accurate enough for predicting excessive path loss in wireless network planning. First, we propose using convolutional neural networks, such as VGG-16, and we compare the results for estimating the probability distribution of excessive path loss with satellite and height map inputs. Second, we develop a conditional Generative Adversarial Network for predicting point-wise path loss. We propose using adversarial loss function for training generator and discriminator in predicting path loss maps of a region. We show the results for satellites images and compare it with the height maps as an input for the proposed approach.

Keywords: Deep learning, height maps, satellite images, GANS, channel parameter estimation,wireless network, regression, excessive path loss, air-to-ground communication system.

Chapter 1

Introduction

Estimation of channel path loss values is crucial for wireless network operators in the process of planning the wireless network infrastructure in a certain region. Adequate and precise information is required to produce a suitable plan for the coverage of the desired region. Path loss is defined as the difference between the transmitted power and the received power. It is affected by physical phenomena, such as reflection, refraction, diffraction and absorption in different media; and it depends on regional characteristics, such as urban, sub-urban and metropolitan. The type of the terrain, the distance from the transmitter to the receiver, and the frequency of the signal are examples of other important factors. In the process of planning the infrastructure of the wireless system in a region, the path loss values should be estimated with sufficient accuracy.

When planning the locations of the base stations, it is required that most of the region has a path loss that does not exceed a certain threshold; this guarantees that the received signal is robust and reliable enough for operation. In most cases, it is not possible to achieve a 100% coverage, where all path loss values everywhere are less than the maximum path loss threshold. The goal of maximizing the coverage can be achieved not just by increasing the transmitted power but by adjusting the 3-D positions and the orientations of the transmitters more effectively. To decide which positions are more appropriate for deployment and achieving the best coverage, field measurements are necessary. Since field measurement require time and resources, alternative mechanisms have been developed. One such alternative is ray tracing simulations, which can provide accurate path loss values when the 3-D model of the region is available. The downside of ray tracing simulations is the high computational cost [1]. Another alternative is the use of mathematical models, which incorporate signal frequency and region characteristics, such as rural, urban, sub-urban regions, etc., to estimate the channel behavior [9].

The above-mentioned approaches are very limited when rapid solutions are required as in, for instance, natural disasters that damage infrastructure, sudden network overload due to traffic, social events, or in cases where we need to modify incapable systems as quickly as possible while maintaining a high quality of service.

In this thesis, we offer two solutions for path loss prediction based on deep learning methods. Our solutions are accurate and general enough to be practically used for this very crucial task, instead of hours of simulations to try and test the results of a certain positioning plan. With our method, we can get this information in a matter of seconds using the satellite image or the height map image of the desired region. This allows many trials for different plans to find the best suitable deployment in a matter of minutes or hours instead of days for just one region.

Our first solution uses a Convolutional Neural Network (CNN). We compare the behavior of the neural network with the case of using satellite images and using height maps as an input. It is well known that the heights and distribution of the buildings in an area is the most important regional characteristics affecting the path loss values. Existing mathematical models try to incorporate this information in some way. In [10], the building heights, the separation between them and the streets formations are utilized. Features extracted from the 3-D region models are used to estimate the path loss distribution directly with the help of machine learning techniques [11]. Support vector machine, dimensionality reduction and artificial neural networks have also been used to predict the path loss values. This motivates us to use height map images, which include all the 3-D building information, as an input to CNN to predict the path loss values. While previous machine learning based methods try to extract features from an height map image, we directly use it as an input and let the deep network extract the features during training. This allows the network to decide on the suitable features from the image on its own, instead of handpicking the features. At the end, we achieved high accuracy on a dataset that is very diverse and general enough to offer an insightful evaluation.

Our second solution uses conditional adversarial training for a deep neural network in the same way the conditional style Generator Adversarial Networks (GAN) is used. The goal to predict the point-wise path loss values, instead of regional distributions, using height maps. We compare the results obtained when height maps are used as input and when satellite images are as input. We also check how GANs perform in comparison to regular convolutional networks at estimating the path loss distributions.

We have generated a dataset of height maps, satellite images, and point-wise path loss images corresponding to different regions from New York City. A point-wise path loss image shows path loss values in the entire region. The ground-truth path loss is obtained using ray tracing simulations. We use height maps and satellite images as input, and point-wise path loss images as the desired output. Further details about the dataset are discussed in Chapter 4.

The contributions of this thesis are as follows.

- We show that CNNs, such as VGG-16, can be successfully used to predict path loss distributions, when height maps or satellite images are used as input.
- We show that GANs can be successfully used to estimate point-wise path loss values from height maps or satellite images.
- The proposed methods are much faster than ray tracing simulations and accurate enough to be used as an alternative in wireless network planning.

This thesis is organised as follows. In Chapter 2, outdoor propagation models are presented to serve as a background on wireless communication parameter estimation problem. The proposed approach based on CNNs, including dataset generation, network architecture and the results, is presented in Chapter 3. The proposed approach based on conditional GANs, including the network parameters and architecture, is explained in Chapter 4. The thesis is concluded in Chapter 5.

Chapter 2

Literature Review

In this chapter, we discuss the empirical models that are used to try to estimate the path losses across different regions and their limitations for outdoor wireless communication planning. Then, we will discuss the machine learning methods that have been lately proposed in the literature.

2.1 Empirical Models

First, the widely used free-space path loss ($FSPL$), which is the simplest model, is formulated as shown below:

$$FSPL = 20\log(f_c) + 20\log(d) + 20\log(4\pi/c), \tag{2.1}$$

where c stands for speed of light in vacuum (m/s), d is the distance between the transmitter and the receiver in km and f_c is the frequency in GHz, so in order to convert it to our dataset units, which are d in meters and f_c is in MHz, the constant becomes -27.55 as it can be seen in Equation (2.2).

$$FSPL = 20\log(f_c) + 20\log(d) - 27.55, \tag{2.2}$$

We can see from Equation (2.2) that the path loss depends only on the frequency of the signal and the distance between the receiver and the transmitter showing how simple this model is, this is of course ignoring many details and characteristics that affect the real measurements.

Second, we will talk about Okumura model [12], which is one of the most famous models, and it was base for other models specialized in open areas as it can not deal with areas with very high blocking buildings. The model is suitable for free areas with no blockage by trees or buildings, also suburban areas like villages or scattered house with minimal blockage near the transmitter or urban areas that have medium blockage like towns with two or more storey buildings.

It can be applied with 150 - 1920 MHz frequencies in urban areas. The height of the transmitter is allowed to be between 30 - 1000 meters. The receiver is allowed to be less than 10 meters. To use the Okumura model for path loss calculation, the median path loss is calculated by the following equation:

$$PL = L_F + A_{mu}(f, d) - G(h_{te}) - G(h_{re}) - G_{AREA}, \tag{2.3}$$

where L_F stands for free-space path loss, median attenuation relative to free space is $A_{mu}(f, d)$, transmitter altitude gain is $G(h_{te})$, the receiver altitude gain is $G(h_{re})$ and environmental gain is G_{AREA} such as urban, rural, etc. Furthermore, the equations below are important to calculate the Okumura model path loss for that environment:

$$G(h_{te}) = 20 \log(h_{te}/200), \tag{2.4}$$

and this can be applied for transmitter height between 30m and 1000m, while for receiver altitude less than 3m we can use Equation (2.5), but for higher altitudes for the receiver we use Equation (2.6),

$$G(h_{re}) = 10 \log(h_{re}/3), \tag{2.5}$$

$$G(h_{re}) = 20 \log(h_{re}/3). \tag{2.6}$$

Correction factors like terrain related parameters can be added using a graphical form to allow for street orientation as well as transmission in suburban, open areas, and over irregular terrain. Irregular terrain is divided into rolling hilly terrain, isolated mountain, general sloping terrain and mixed land-sea path. The terrain related parameters must be evaluated to determine the various corrections factors.

We now look at Hata model, which is a limited version of Okumura model as it is limited for quasi-smooth terrain [12], with limited transmitter altitude between 30 - 200 meters and transmission distance of less than 20 km. The Hata model equation to estimate path loss for urban areas is shown as follows [13]:

$$PL = 69.55 + 26.16\log(f_c) - 13.82h_{te} - a(h_{re}) + (44.9 - 6.55\log(h_{te}))\log(d), \quad (2.7)$$

where f_c indicates the frequency in MHz, h_{te} and h_{re} are the transmitter and receiver altitudes in meters, respectively, the distance between transmitter and receiver is d, and $a(h_{re})$ is the correction factor, which is 0 for 1.5m height.

For small and medium cities, $a(h_{re})$ is as follows:

$$a(h_{re}) = (1.1\log(f_c) - 0.7)h_{re} - 1.56\log(f_c) + 0.8, \quad (2.8)$$

while for large cities:

$$a(h_{re}) = \begin{cases} 3.2(\log(11.75h_{re}))^2 - 4.97 & f_c > 300 \\ 8.29(\log(1.54h_{re}))^2 - 1.1 & f_c \leq 300 \end{cases} \quad (2.9)$$

Some studies have suggested that the path loss experienced at 1845 MHz is approximately 10 dB larger than those experienced at 955 MHz [12]. This is why Cost231-Hata model [10] was introduced to help with this deviation from the Okumura model and we can see the formulation below:

$$PL = 46.3 + 33.9\log(f_c) - 13.82h_{te} - a(h_{re}) + (44.9 - 6.55\log(h_{te}))\log(d) + C, \quad (2.10)$$

where C is equal to zero for suburban areas and three for metropolitan areas. Hence, from the empirical methods we can see the limit of approximation it can reach and also the constrains for their application, that includes the types of terrains and the signal frequency. For each kind of terrain, the suitable model has to be picked considering how far it matches the standard terrain. On the other hand, in our method we simply train the model for the specific frequency for any terrain omitting the need to research to reach an acceptable approximation to the real measurements.

2.2 Ray Tracing Simulations

From the models discussed so far, we can deduce that the complexity and computational cost for the empirical models are very low but this comes with the price of low accuracy for the estimations they provide. On the other hand, we have the high accuracy methods which are the ray tracing simulations that use the 3-D models of the regions [1, 14, 15]. Using these simulations, we can get the characteristics of the wireless channel, such as the path losses and delay spreads. Obviously, this comes at the cost of high computational needs and the long simulations. In the case of urgent planning for large areas, the simulations prove to be very hindering.

Many ray tracing techniques and algorithms exist with their basic and advanced version [16]. 3-D models are normally used for maximum accuracy when compared to real data. For our simulations and data generation, we use "Wireless InSite[1]" software for 3-D models to get the highest accuracy to real measurements and we can see how the software performs compared to real data in [1] as shown in Figure 2.2.1. In this experiment, measurements were done in a town called Banja Luka where "Rohde&Schwarz FSH" spectrum analyzer was used with "TS-EMF" antenna system and "RFEX" software package. Electric field strength values were obtained using grids of 5m resolution for 115000 receivers for frequencies 900 and 1800 MHz.

[1] https://www.remcom.com/wireless-insite-em-propagation-software

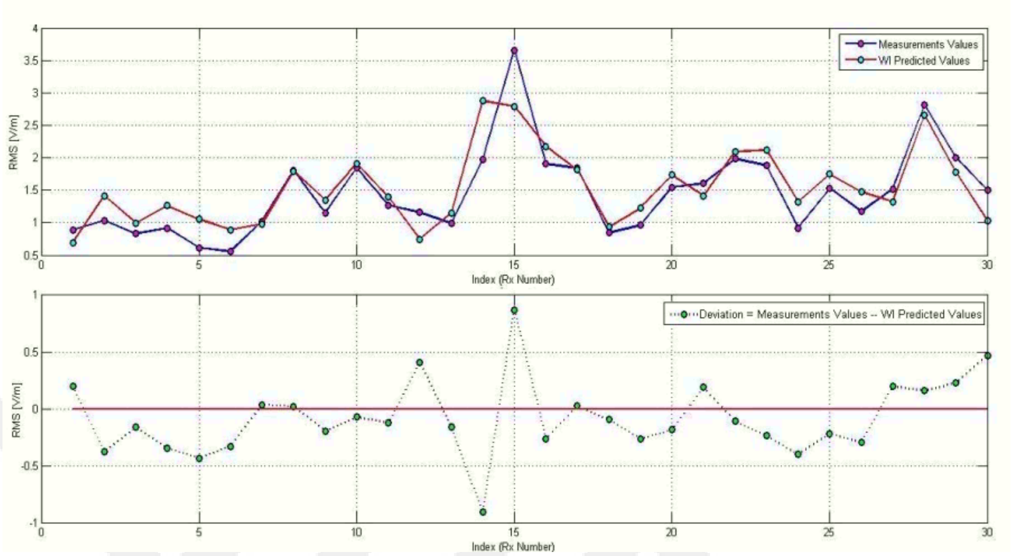


Figure 2.2.1. Comparison of measurement results and software predicted values [1].

2.3 Machine Learning Methods

In the recent years, many researchers have investigated using machine learning techniques to try to solve the problem of estimating wireless channel parameters. One of these experiments utilized random forest and K-nearest-neighbour algorithms [2]. For the lack of data, transfer learning was the appropriate solution, where the model was trained on the same dataset but for different frequencies. The goal was estimating the delay spread and path loss using supervised regression. Wireless InSite software was used to generate the datasets. 20k samples divided to different datasets with different frequencies were produced to test the effect of transfer learning which appeared to be very beneficial to the accuracy of the results of the path loss predictions as demonstrated in Figure 2.3.1.

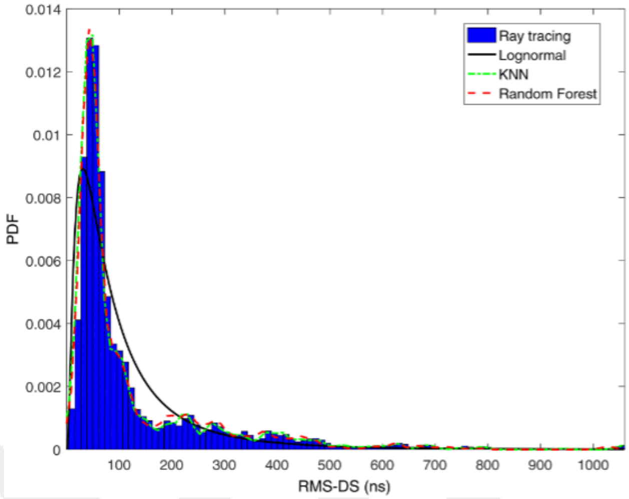


Figure 2.3.1. Comparison between empirical models and machine learning techniques with the simulation data for delay spread prediction [2].

Also in Figure 2.3.2 we can see the accuracy in predicting path loss for the empirical models versus the machine learning techniques.

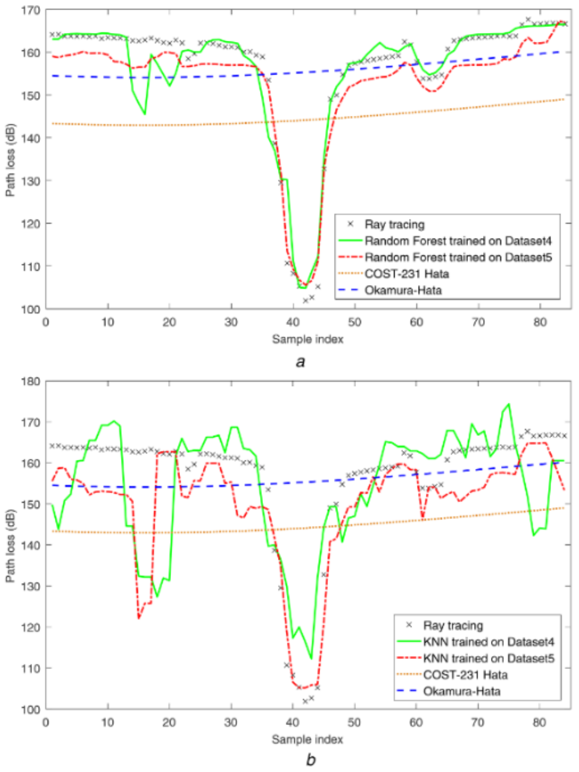


Figure 2.3.2. Comparison between empirical models and machine learning techniques with the simulation data for path loss prediction [2].

2.4 Deep Learning Methods

In the recent years, due to the advancement of graphical processing units technology, the use of deep learning has flourished greatly. Deep learning is the process of training a deep neural network on a dataset for many iterations so that the network could be able to imitate the natural neural network behavior of learning the implicit information that is available in the dataset being studied. The process aims for the network to gain the ability to predict the correct answer when used on unknown data that has not been introduced to it before. This method was used in calculating path loss distribution by trying to classify the path loss exponent and standard deviation of shadowing using satellite images as a source of information for the network [17]. Wireless InSite was used to generate the dataset for 900MHz where they were able to achieve 88% and 76% accuracy. In this work, the network was trained on specific transmitter heights and frequencies. VGG-16 [7] network and Resnet-50 [18] were used and in order to match the sizes of these networks inputs, resizing the images to 224x224x3 had to be done. This work demonstrated the ability of deep learning to be deployed in real time and achieve robust and usable results for wireless network planning.

Another example for deep learning usage was the work in [3] where path loss distribution was estimated for a certain region using satellite images only as an input for different frequencies and transmitter heights. The histogram of the path losses was divided into 8 bins, each having 10 dB width. VGG-16 network was trained in a regression scheme, where they used Mean Squared Error (MSE) for performance evaluation and in Figure 2.4.1 the scatter plot demonstrates how well the network was able to predict the histogram bins for the 300m high transmitter and 900MHz signal frequency.

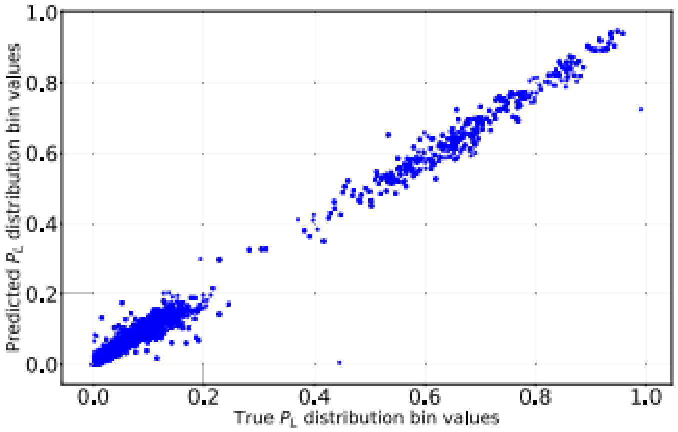


Figure 2.4.1. Scatter plot for the transmitter at height of 300m and signal frequency of 900MHz for VGG-16 network [3].

2.4.1 GAN Methods

The main problem with using deep learning was how to define the appropriate loss for the problem being studied. A very successful solution for the problem was using another network to learn the loss on its own, this technique is the GAN method [19], where a network called generator is trained to try and deceive another network called discriminator which is itself being trained at the same time to not to be deceived by the former network [20]. The adversarial training achieves very good results mainly because the discriminator defines and learns the appropriate loss for the training. In this method, the generator tries to minimize two losses at the same time, the mean absolute loss and the loss coming from the discriminator. Conventional GANs use noise as an input without any conditions and try to generate an appropriate output from it. Another type of GANs is conditional GAN where the input is a condition that holds clues to the appropriate output wanted [21] like style transferring [5], semantic imaging [22], producing a super-resolution images [23], image to image transferring [4], etc. Even for speech, GANs were used successfully [24]. Style GANs are the type of GAN where the generator is able to infer a style on a certain image to manipulate the style in it as shown in Figure 2.4.2. We aim to apply this process on height maps to convert them into path loss images, which is explained in our second approach in Chapter 4.

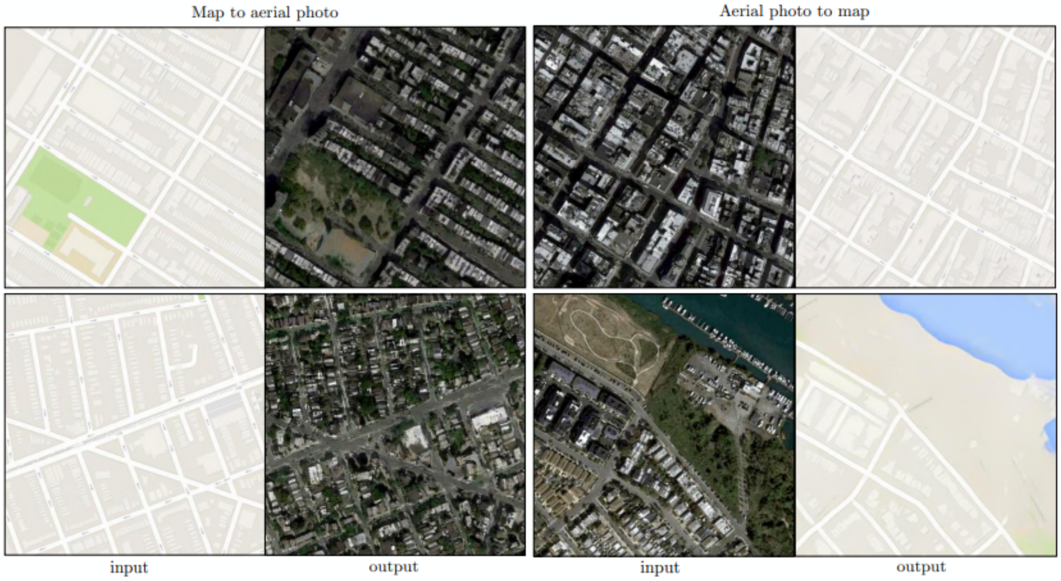


Figure 2.4.2: Sample result of style GAN [4, 5].

As an example for using GAN in the wireless communications field, a GAN was used for wireless channel modelling aiming to eliminate the need for field experts [6]. A typical additive white Gaussian noise channel was assumed in the simulations. Accuracy of the trained model was not promising, which can be understood by comparing the Probability Distribution Function (PDF) of the real and the generated channel response samples as shown in Figure 2.4.3.

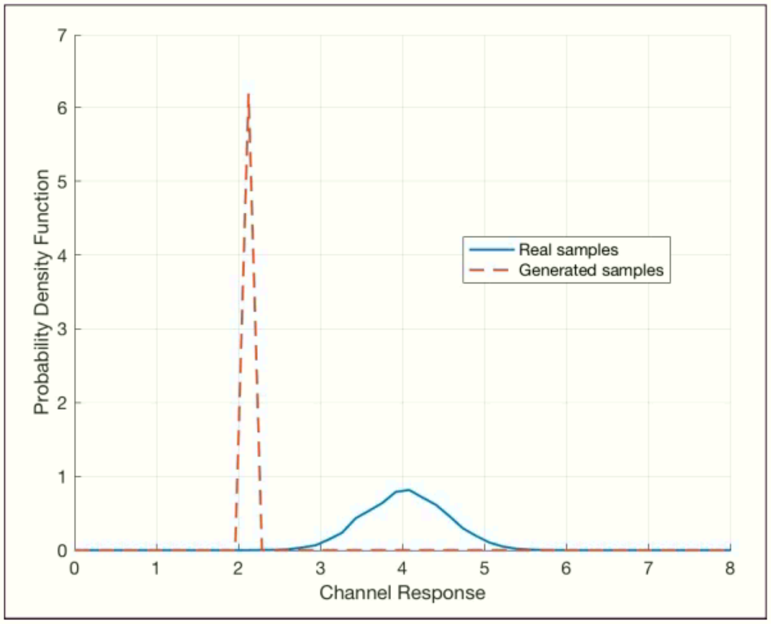


Figure 2.4.3: The PDF of real channel response vs. the generated channel response [6].

Chapter 3

Path Loss Prediction Using CNN approach

As we mentioned, our first solution is using CNN to predict the excessive path loss PDF as a regression problem benefiting from the ability of the CNN to extract meaningful features for accurate estimation.

3.1 Dataset Generation

Dataset preparation is the most crucial element for machine learning based solutions, as it is the key for training and learning for the model being created. We prepared a dataset which is the first of its kind. It consists of satellite images from "Google Maps" orthorectified to match with their corresponding height maps plane view, so that any pixel in the satellite image and the same pixel in the height map point to the same location in the region.

We used "Sketchup[1]" to obtain the 3-D models to generate the height maps from it using "PlaceMaker[2]". We obtained 1000 satellite images and height maps

[1] https://www.sketchup.com/

[2] https://www.suplacemaker.com/

from different regions in New York city, and each region is approximately 1.8 km by 1.8 km in size. Then, we used the 3-D models to do ray tracing simulations with Wireless InSite software. As for the simulation setup, the grid consists of 12100 (110x110) receivers, all at 1.5m above the ground, the power of the transmitter is 60 dBm and we used 300m and 80m as different heights for the isotropic transmitter. We can see all the details of the simulation in the table below:

Table 3.1: Simulation Parameters

Parameters	Values
Transmission Frequency	900 MHz
Transmit Power	+60 dBm
Transmitted Signal	Sinusoid
Transmitter Antenna Height	300m,80m
Transmitter location	region center
Receiver Antenna Height	1.5 m
Antenna Polarization	Vertical
Antenna Radiation Pattern	Omni-directional
Bandwidth	8 MHz
d_0	57.28 m
$PL(d_0)$	63.44 dB
Number of receivers	12100
Receivers grid	110x110
region size	1.8x1.8km

The received power is obtained for each receiver in decibel (dB) with the following equation:

$$PL_n = 10\log(P_T) - 10\log(P_{R_n}), \tag{3.1}$$

where PL_n is the path loss at the n-th receiver, P_T is the transmitted power and P_{R_n} is the received power at the n-th receiver. Then, we calculate the excessive path loss at the n-th receiver η_n using this equation:

$$\eta_n = PL_n - FSPL_n, \tag{3.2}$$

where $FSPL_n$ is the free space path loss for the n-th receiver, which also can be calculated by

$$FSPL_n = 20\log(d_n) + 20\log(f_{MH_z}) - 27.55, \tag{3.3}$$

where d_n is the distance between the n-th receiver and the transmitter, and f_{MH_z} is the frequency of the transmitted signal. The simulation is performed at frequency 900 MHz. Since we have the path loss for each receiver, we can calculate the PDF of these path losses, so we divided the PDF into 8 bins to discretize the function. Each bin represents a certain interval of path losses as shown in Table 3.2.

Table 3.2: Probability Distribution Function bin intervals for excessive path loss.

Bin	Interval
0	$\eta < 20$
1	$20 \leq \eta < 30$
2	$30 \leq \eta < 40$
3	$40 \leq \eta < 50$
4	$50 \leq \eta < 60$
5	$60 \leq \eta < 70$
6	$70 \leq \eta < 80$
7	$80 \leq \eta$

Firstly, we obtain the 3-D model of the regions in Collada (DAE) format that Wireless InSite needs to perform the simulation. To generate the height map from this format, we need to decode the 3-D file into x,y,z data with surfaces information to be able to visualize the models. For that purpose, we use the Stanford Triangle Format (PLY) or in other name the polygon file format. We use "MeshLab[3]" to do this transformation.

We use "CloudCompare[4]" software that is able to visualize the surfaces and the buildings accurately capture the height map. It can show the height maps using color-maps for good visualisation. For training, we decided on the white color map where the intensity of the color depends on the height and each sample is normalized so that the maximum height in the region has pixel intensity value of 255.

We were not able to do this process with a script, so it was very tedious to

[3] "https://www.meshlab.net/"

[4] "https://www.danielgm.net/cc/"

create a large dataset one by one. We used "Actiona[5]" automation tool for the repetition process.

We had to preprocess the satellite images to match the same region with the height maps. Hence, we warped the image by applying an appropriate transformation matrix. We used least mean square algorithm to get the homography matrix, used for the transformation, by hand picking the matching pixels in the satellite images and their corresponding height maps. Since the same transformation can be applied for the rest of the dataset, the process could be generalised for the dataset easily. Due to software limitations, the height maps often exceeded the region that was present in the satellite images, so we had to crop the height maps so that they would represent the exact same region to achieve fair and correct comparison.

The final step was global normalization of the height maps. The way "CloudCompare" works is it produces the height map normalized to its local highest building so that each height map has a 255 value pixel assigned to the highest building in the region which means that a pixel intensity in one sample does not correspond to the same height for the same pixel intensity in another sample. As a result, we had to renormalize the dataset so that the highest pixel value 255 is assigned to the highest building in the whole dataset which is the Empire State Building of height 381 meters and resulting in a global meaningful values for the intensities across the whole dataset for adequate training process. We can summarize the steps in the following points:

- Converting 3-D models to (PLY) by MeshLab.
- Using Stanford Triangle Format in CloudCompare to produce the height maps.
- Using Actiona to automate height maps generation.
- Warping operation to orthorectify the satellite images to coincide with the height maps.

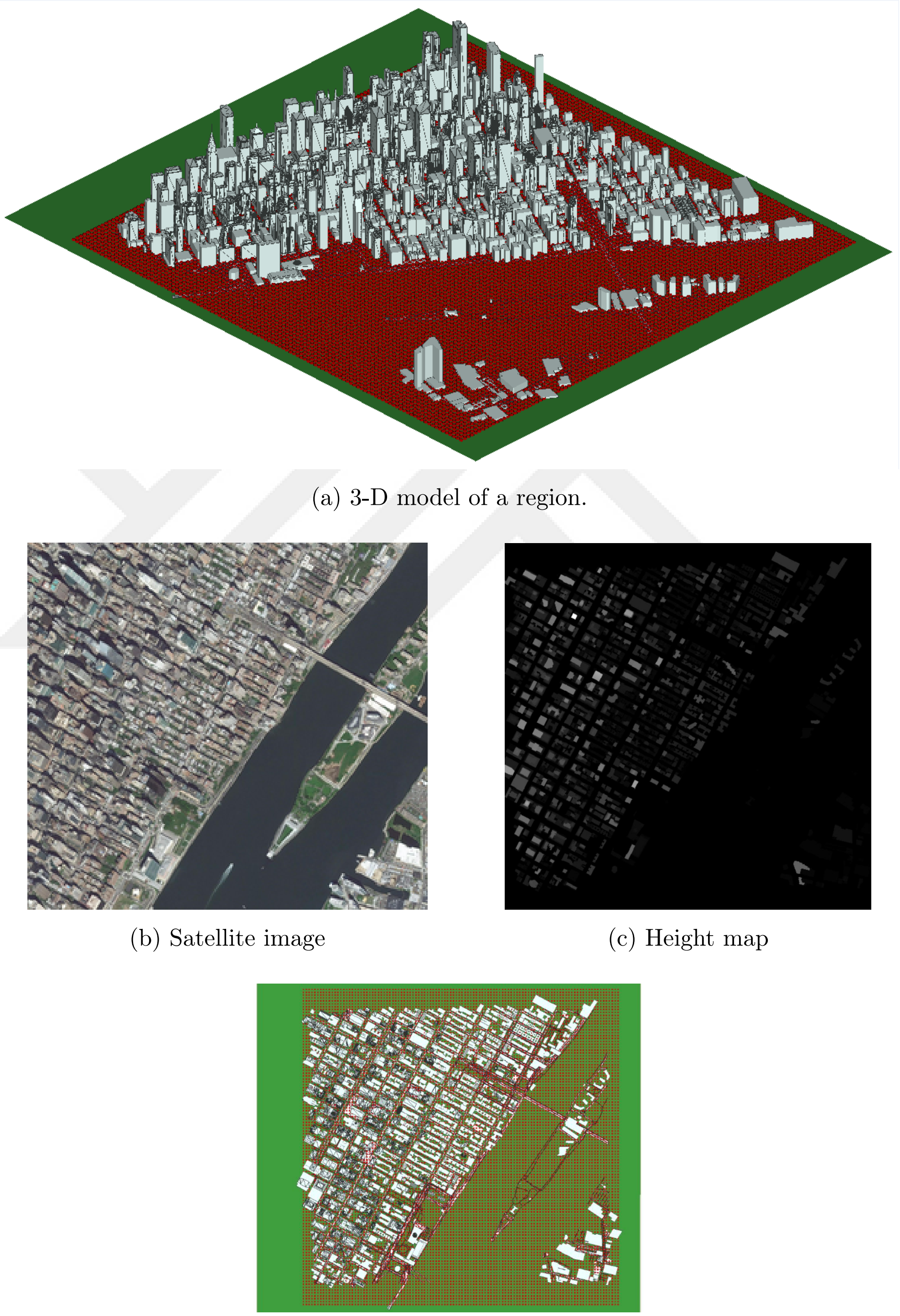
[5] " https://actiona.tools"

- Height maps intensity normalization so that each pixel intensity matches the same power value across the whole dataset.

- Checking which receivers are outdoor receivers and do not lie inside a building.

- Calculating the PDF of each sample with bin width 10 dB.

After all these processes, we obtained height maps normalized so that any pixel value across the dataset corresponds to a specific height with margin of error 2m resulting from the quantisation of values. In addition, the height maps are cropped and matched with the satellite images so that each pixel location in the height map points at the same place that the same pixel location in the satellite image points at. We can see samples for the height maps and satellite images obtained from the corresponding 3-D models and their receivers locations in Figures 3.1.1 and 3.1.2.



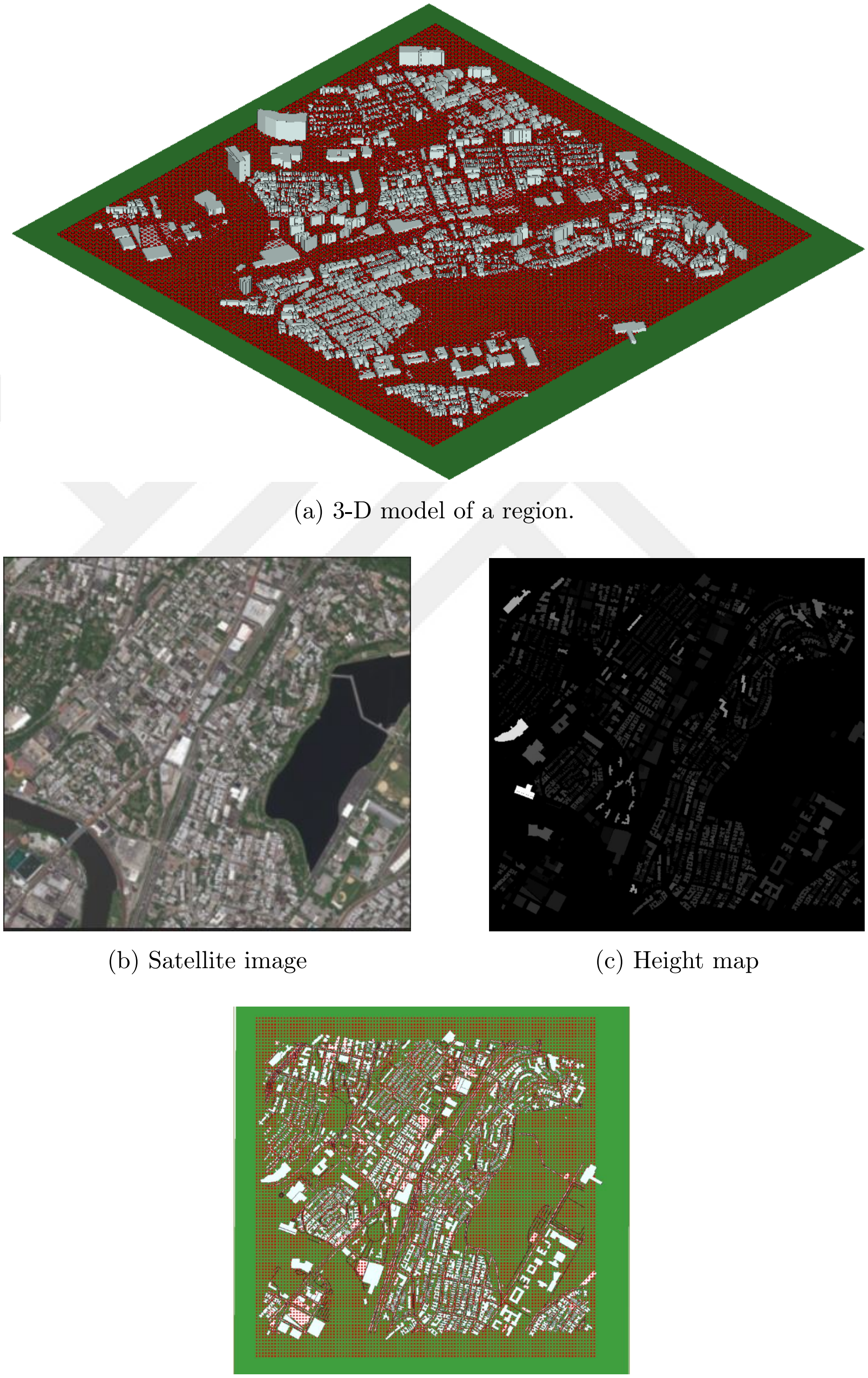
(a) 3-D model of a region.

(b) Satellite image

(c) Height map

(d) Receiver locations image

Figure 3.1.1: A sample 3-D model, satellite image, height map, and receiver locations image for a certain region.



(a) 3-D model of a region.

(b) Satellite image

(c) Height map

(d) Receiver locations image

Figure 3.1.2: A sample 3-D model, satellite image, height map, and receiver locations image for a certain region.

3.2 Network Architecture

One of the most successful convolutional classification networks is VGG-16 which achieved very high classification score in the ImageNet Large Scale Visual Recognition Challenge 2014 (ILSVRC2014) where it achieved a top 5 score of more than 92.7% test accuracy. The ImageNet is a dataset of more than 14 million images containing more than 22000 categories and it was trained on it for weeks, but in the annual competition they used only 1000 categories with 1000 images at a resolution of 256x256.

Our plan is to to be able to use this network with its more than 100 million trainable parameters and deep architecture shown in Figure 3.2.1 at the optimum performance. For that purpose, we apply transfer learning, in which we use the weights of the already trained network for initialization at the beginning of training. Then, we train and fine-tune the model on our own dataset. This technique saves a lot of effort and time and gives us a chance to utilize the well-trained filters especially the initial filters at the first layers of the network that are very hard to train in deep networks. This approach is much more effective than initializing with random weights and retraining, since these filters got their accurate weights from millions of samples and days of training.

For training loss, we used categorical cross entropy, and for the optimizer we used Adam optimizer at learning rate 0.0001.

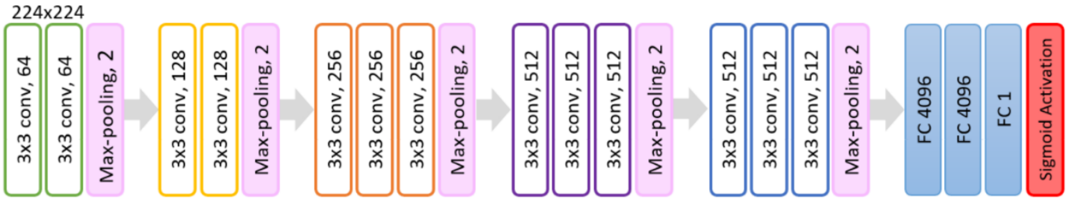


Figure 3.2.1. VGG-16 architecture [7].

As for the training details, we divided the data into %70 for training and %30 for testing, the images were normalized to [-1, 1] interval before training. We trained on RTX 2060 Nvidia GPU, and the training went smoothly and converged easily.

3.3 Experimental Results

We mentioned above that we trained the network on two datasets, each for a specific height for the transmitter to continue the work of [3] but for excessive path loss instead of normal path loss using both satellite images and height maps as an input. The aim is to study the ability of the CNN to benefit from the height information available in the height maps and if it will be able to achieve better accuracy for the estimation using this information or not.

We provide below random samples for the PDF predictions by the network for the height maps and satellite images as inputs, for both transmitter heights 300m and 80m. First, we show the 300m dataset results. We show a random result for satellite image input in Figure 3.3.1. Then, we show for the same dataset but for height map input a sample result as well in Figure 3.3.2. We also show the worst predictions for the PDF for the satellite image input and for the height map input in Figures 3.3.3 and 3.3.4, respectively. As for the general accuracy and behavior of the network, we provide the scatter plots that show the predictions versus the true values for all the testing dataset. We give the scatter plot of the satellite image input and height map input in Figures 3.3.5 and 3.3.6, respectively. Next, we analyze the 80m dataset results. In the same order as in the 300m dataset results, we show a random sample prediction result for the satellite image input in Figure 3.3.7, then for height map input in Figure 3.3.8. The worst cases of PDF predictions are provided for satellite image input and height map input in Figures 3.3.9 and 3.3.10, respectively. Finally, We give the scattering plots for both inputs in Figure 3.3.11 and 3.3.12.

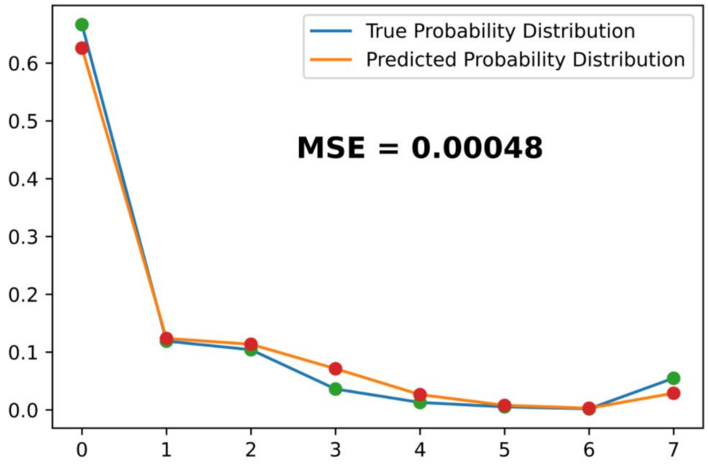


Figure 3.3.1: A random sample of predicted PDF vs. true PDF for satellite image input at height 300m.

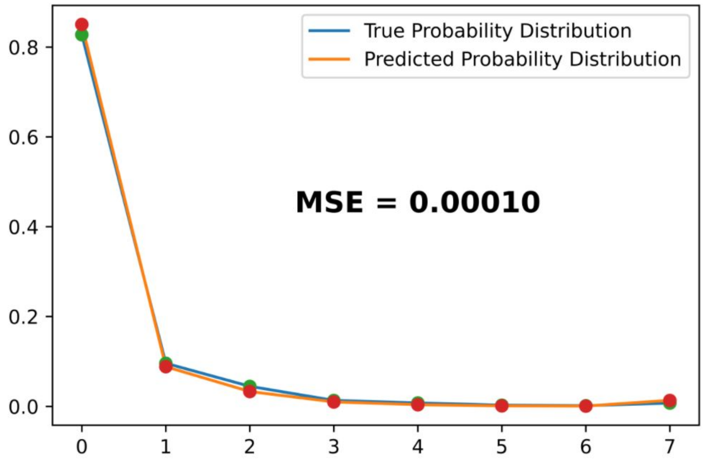


Figure 3.3.2: A random sample of predicted PDF vs. true PDF for height map input at height 300m.

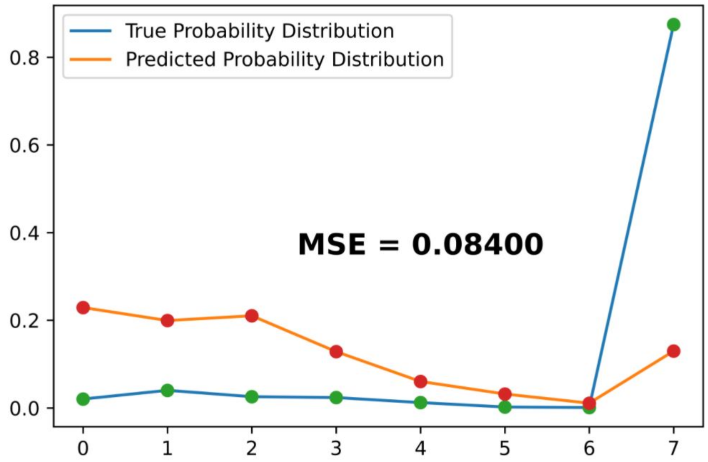


Figure 3.3.3: The sample of lowest MSE of predicted PDF vs. true PDF for satellite image input at height 300m.

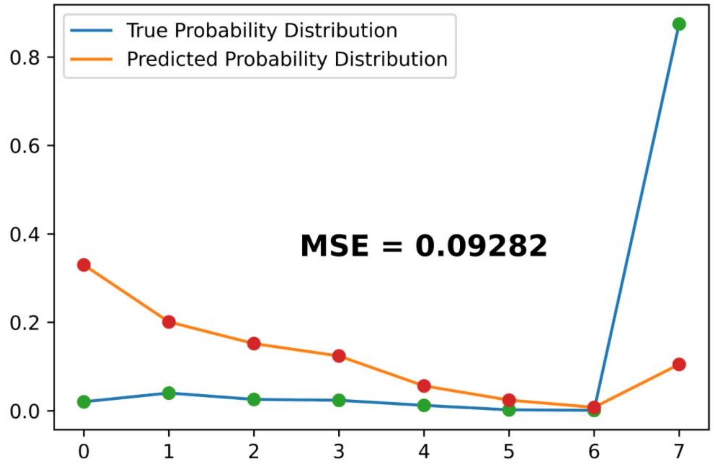


Figure 3.3.4: The sample of lowest MSE of predicted PDF vs. true PDF for height map input at height 300m.

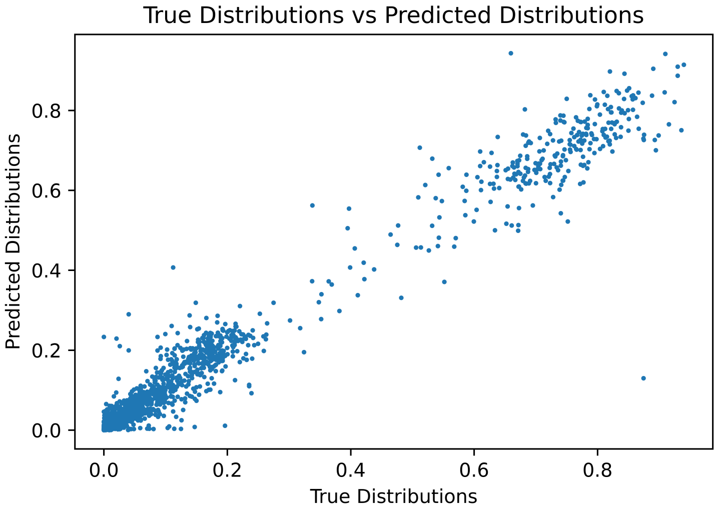


Figure 3.3.5: A scatter plot of predicted PDF vs. true PDF for satellite image input at height 300m.

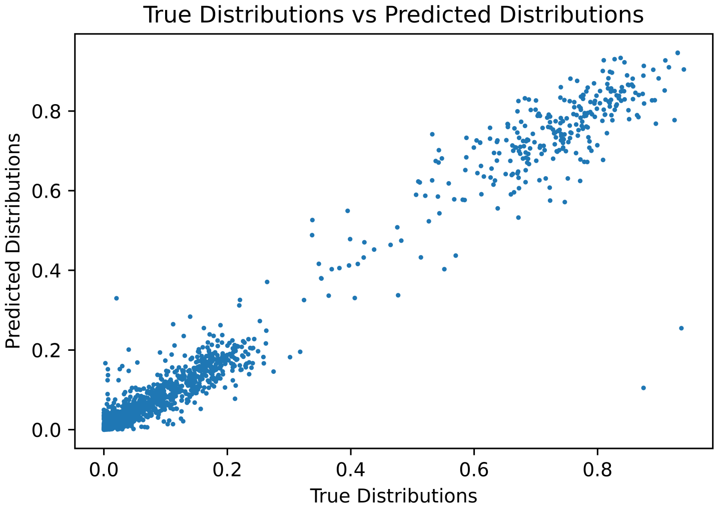


Figure 3.3.6: A scatter plot of predicted PDF vs. true PDF for height map input at height 300m.

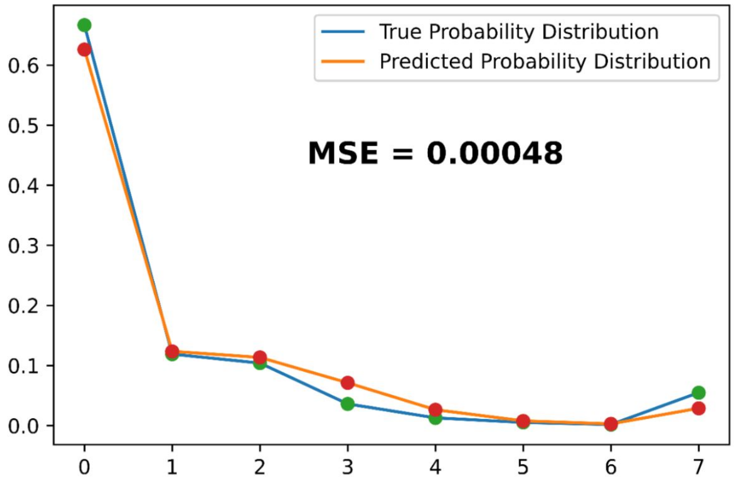


Figure 3.3.7: A random sample of predicted PDF vs. true PDF for satellite image input at height 80m.

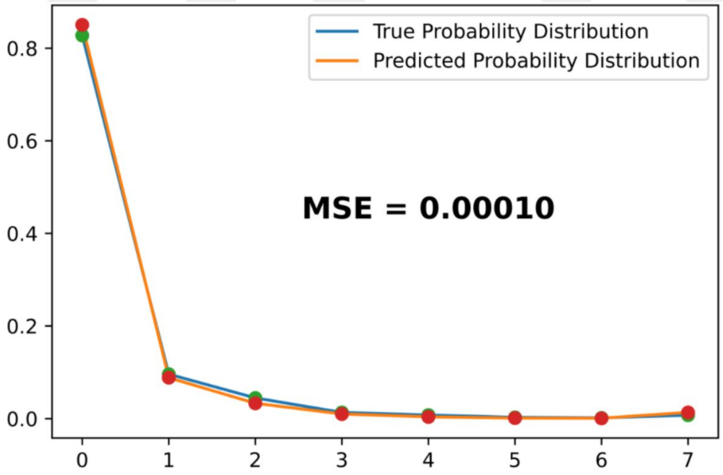


Figure 3.3.8: A random sample of predicted PDF vs. true PDF for height map input at height 80m.

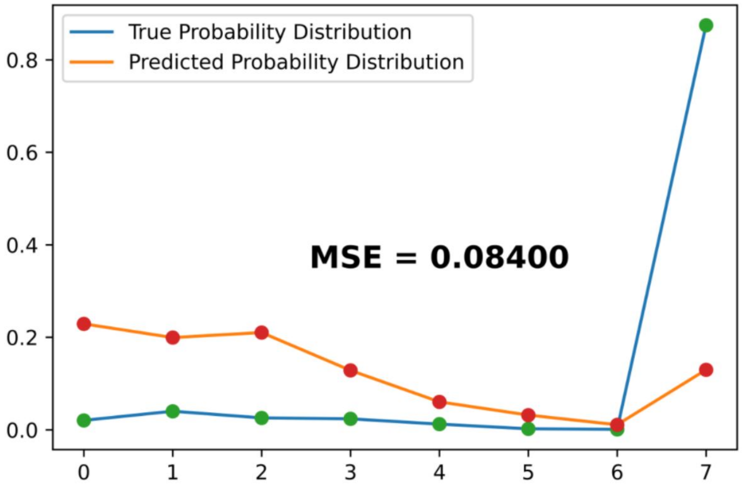


Figure 3.3.9: The sample of lowest MSE of predicted PDF vs. true PDF for satellite image input at height 80m.

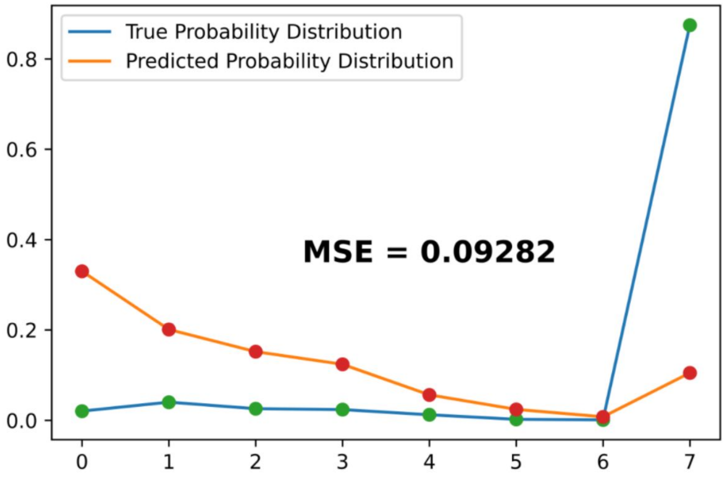


Figure 3.3.10: The sample of lowest MSE of predicted PDF vs. true PDF for height map input at height 80m.

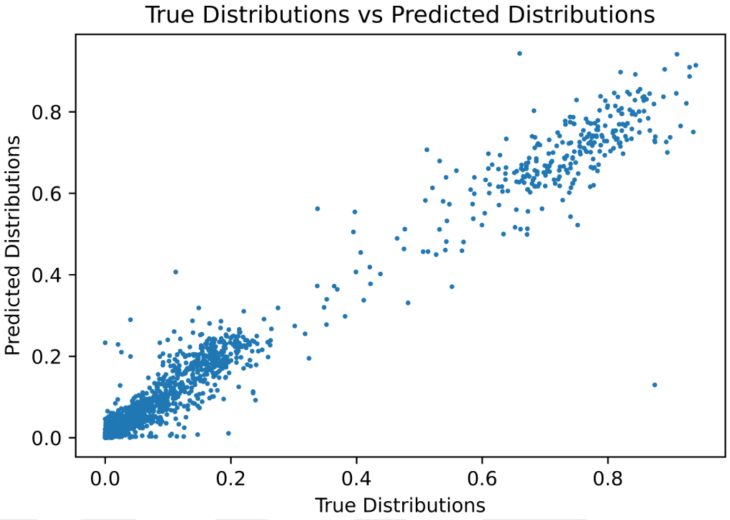


Figure 3.3.11: A scatter plot of predicted PDF vs. true PDF for satellite image input at height 80m.

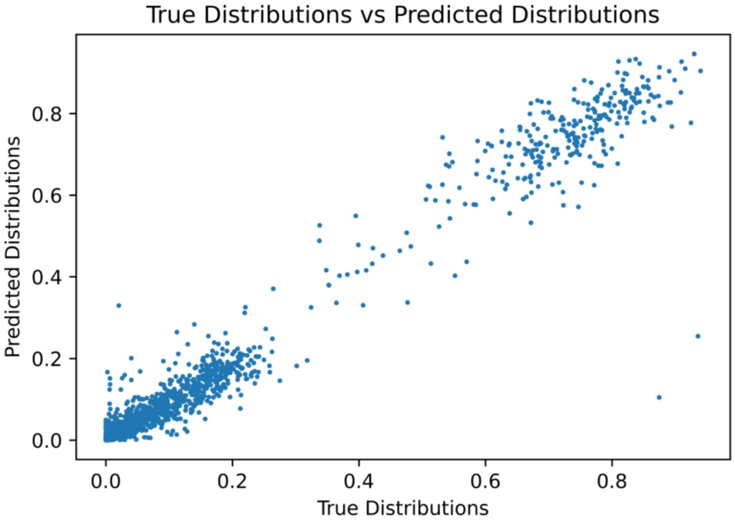


Figure 3.3.12: A scatter plot of predicted PDF vs. true PDF for height map input at height 80m.

From the figures, we can see that there is no special preference or success for one input against the other whether this is for 300m height or 80m height. This is especially obvious when we look at the scatter plots of the height map input vs. the scatter plot for the satellite image input, where we see that the results and the general trend of the predictions are almost the same for both the height map and the satellite images. Although the model performs noticeably better for 300m dataset but this is because of the inherent nature of the distributions that it is trying to predict. Since for lower heights the distribution gets more complicated and varies very widely from one region to the other unlike the 300m dataset where most of the distributions are typically concentrated at the first bins and they rarely look otherwise, therefore the 300m dataset would understandably get better results than the 80m dataset, we can see that in the tables showing the MSE results.

Table 3.3: Excessive Path Loss Prediction for 300m transmitter.

	Mean Squared Error (MSE)		
Architecture	Satellite image	Height map	Variance
VGG-16	7.2×10^{-4}	7.5×10^{-4}	0.051

Table 3.4: Excessive Path Loss Prediction for 80m transmitter.

	Mean Squared Error (MSE)		
Architecture	Satellite image	Height map	Variance
VGG-16	1.8×10^{-3}	1.4×10^{-3}	0.053

3.4 Discussion

In the previous section, we showed random samples and results from different datasets. Figure 3.3.1 showed a random example from the 300m dataset for satellite image input, then Figure 3.3.2 also showed a random example for the same dataset but for height map input. By comparing the two figures, we can see that they generally look similar. We repeated the same comparison for samples with the worst prediction performance, and they also showed similar behavior. We can confirm this similarity in Figure 3.3.5 and Figure 3.3.6. These plots demonstrated the general behavior of both the satellite image input and the height map input, respectively. We can confirm from the scatter plots that the learning from both inputs produced similar results, proving that neither of the inputs was more beneficial than the other. We repeated the same comparison for the 80m dataset and we can reach the same conclusion as well.

To confirm the inability of CNNs to obtain the additional information available in the height maps, we tried networks other than VGG-16 and other dataset manipulations like using the height map and satellite images concatenated with each other, the outcome of which did not differ much from the results discussed.

We can see that the CNNs were not able to extract the height information from the height maps and they only used the building distribution information that was already available in satellite and aerial images. As a result, we believe that the use of height maps as a source of information to train CNNs is not the most optimum way for estimating the PDF of the excessive path loss. The reason for that is that height maps require more effort and resources to be obtained than the effort needed for acquiring satellite images as they are now widely available from many different sources.

Chapter 4

Pointwise Path Loss Prediction Using GAN Approach

Our Second solution is using conditional style GAN like in [25] where we try to use the underlying structure of the buildings in either satellite images or height maps and especially height maps to try to predict the excessive path loss image. Instead of using the image to deduce the probability distribution of the path losses, we wanted to gain the ability to estimate the exact path loss at each point across the area and how they are distributed across the region using only an image as an input whether it is a height map or a satellite image. We considered it as a problem of synthesizing an image where the pixels represent the excessive path loss values utilizing input image as a guide or a clue for the desired output. This is a very suitable task for the GAN as they are the best candidate for image synthesis at present with many successful applications in synthesizing an artificial image.

4.1 Dataset Generation for GAN Training

The dataset prepared for this experiment was based on the dataset already created for the previous experiment as mentioned in Chapter 3. We used the calculated excessive path losses and distributed them on a grid in a square matrix shape whose size is 110x110, which is the number of receivers for each region. Then, we resized the produced matrix to a size of 256x256 to achieve the same size as the network input. We used the nearest neighbour interpolation algorithm since we do not want any values to be modified during the resizing process. We prepared these path loss images for 300m, 80m and 40m transmitter heights. In Figure 4.1.1, we show the satellite image, the corresponding height map and the produced excessive path loss image.

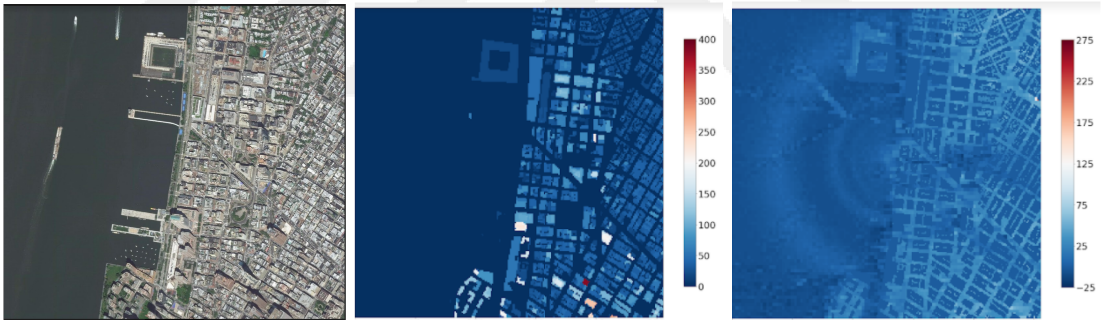


Figure 4.1.1: Satellite images, their corresponding height map, where the colorbar is in meters, and path loss image, where the color bar is in dB. Images are shown for a certain region.

4.2 Network Architecture

Due to the obvious success of GAN networks in image synthesis, either in style transfer, super resolution or even producing fake images that look exactly like real images, we picked a very successful style GAN which was very general as an essential characteristic of this network and we edited the network to match our application. We used the idea of U-net in the generator part as a building block instead of a simple encoder-decoder scheme. Due to the high level of information that is shared between the output and the input, transferring the information across the bottleneck of the network helps the network to converge easily. This is why the skip connections play a crucial part in the generator. Hence, we add skip connections across all the layers from the encoder to their corresponding layers in the decoder part as shown in Figure 4.2.1. We used Adam optimizer for training both the generator and discriminator with learning rate 0.0001 and batch size of 32.

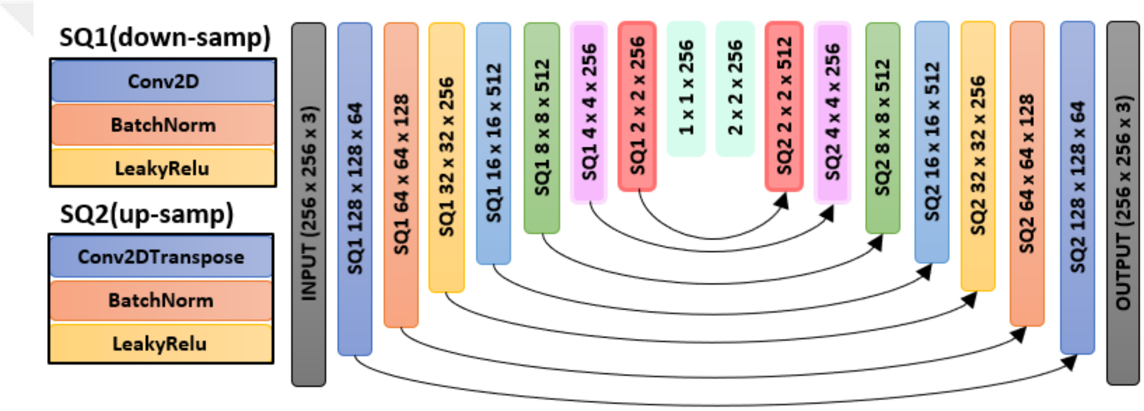


Figure 4.2.1: The generator architecture sequence 1 (SQ1) has 2-D convolutional layers with strides of 2 and padding (same). Sequence 2 (SQ2) has transposed convolutional layer with strides of 1 and padding (same).

We use 18 convolutional blocks where we begin with decreasing the width and length of the layers as in conventional encoder-decoder networks while increasing the depth of the layers at the same time until we reach the bottleneck. We use stride of 2 at first to increase the receptive field of the first layers while keeping the number of parameters relatively small, which helps with overcoming the overfitting problem. The convolutional blocks include batch normalisation and leaky relu activation. In the decoder part, we use transposed convolution that works as a counter for the convolution to go back to the original size of the image while also decreasing the depth of the layers to go back to the original depth of the image and we add dropout as a source of regularization.

As for the discriminator, we used a patch discriminator [26] that decides on each patch of the generator output as true or fake and we again use strides of

2 at the beginning to decrease the number of useless parameters and to avoid over-complicating the layers to try to manage the over-fitting problem in the discriminator training. This type of discriminator looks at each patch concentrating on the high frequency structures. Therefore, while training we add L1 loss as a constrain for the low frequency structure, a method used many times in the literature to balance between the two types of information needed for the image synthesis. The input of the discriminator is 6 channels instead of 3 channels because it takes an extra image as a condition working as a clue for the output. As a result, we concatenate the input image, whether it was a satellite image or a height map, with the generated image or the true image going through the discriminator network. The discriminator consists of 8 blocks of convolution with batch normalisation and leaky relu until it goes down to the size 5x5x1. We divided the data into 80% and 20% partitions for training and testing datasets, respectively, for a total of 997 images. We applied data augmentation on the training images which were rotated and flipped simultaneously adding 8 samples for each image. We trained on Tensorflow 2.0 on Nvidia RTX 2060 GPU where the training goes for about 12 hours for 500 epochs.

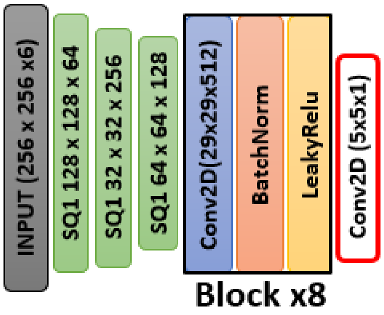


Figure 4.2.2: Discriminator architecture. The block shown is repeated 8 times and the 2-D convolution has strides of 1.

4.3 GAN Training Method

Training a GAN network is very tricky and here we explain our method for training our network. Our final aim from this training is for the generator to be able to deceive a well trained discriminator driving it to decide that the generator output is true. As for the generator training, we apply the generator on the input, which is either a height map or satellite image. The output is passed to the discriminator and compared with a matrix of ones that measures how much the generator was successful in deceiving the discriminator into deciding that the generator output is true and not fake. Therefore, in a perfect case the discriminator would decide that all the patches are true and give a matrix of ones as an output so that the loss would be zero in that case. We calculate the loss using a sigmoid cross entropy loss. Then, we have the low frequency structure constrained by mean absolute error between the generator output and the ground truth. Finally, the two losses are added by a weighted summation where lambda equal to 100 is multiplied with L1 loss to balance the losses as in [4], then, this sum is applied as a gradient to the weights of the generator. Figure 4.3.1 demonstrates the previous explanation.

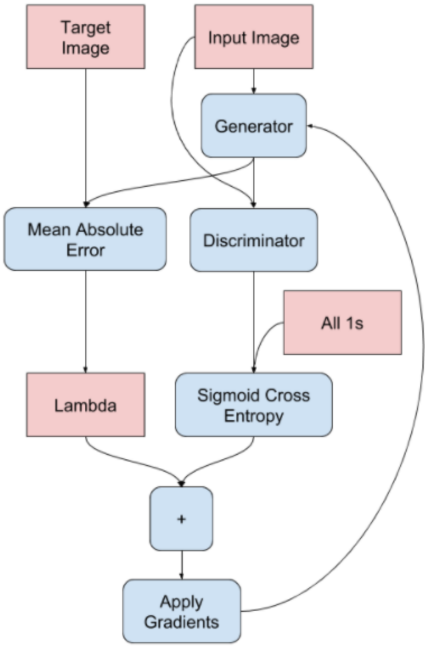


Figure 4.3.1: Training method of the generator [8].

A crucial step was added to the generator training for the height map input. Normally we measure the L1 loss between the ground truth and the generated image. For our training, we included the pixels pointing to outdoor spaces only, excluding the indoor of the buildings from the loss calculation during the training. We added this detail believing that this should help the network focus on the desired information, which is the outdoor regions, and not the whole image. This method was not applied for the satellite training because we assume that we do not know the buildings exact boundaries. For the discriminator training, the goal is for the network to get better at separating the true path loss images from the fake ones. The discriminator receives the generator output as input and compares its output with a matrix of zeros. This is because the output should be zero for these images, meaning that these are fake images. Then, the real path loss images are passed to the discriminator as well and this time the output is compared to a matrix of ones. Sigmoid cross entropy loss is calculated for both fake and real images and then both losses are added and their gradient is back propagated to update the weights of the discriminator.

Using this method of training lets the discriminator learn the appropriate loss for this training on its own as opposed to using only L1 or L2 loss. This aims only at reducing the difference between the prediction and the ground truth with blurry details focusing only on the low frequency component and ignoring the high frequency details of each image.

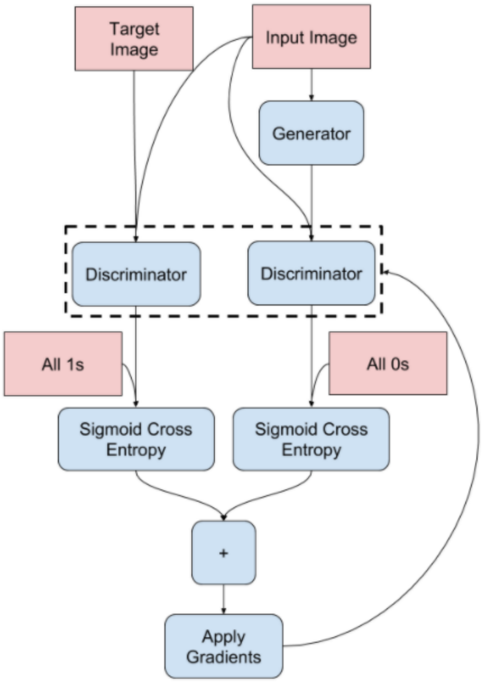
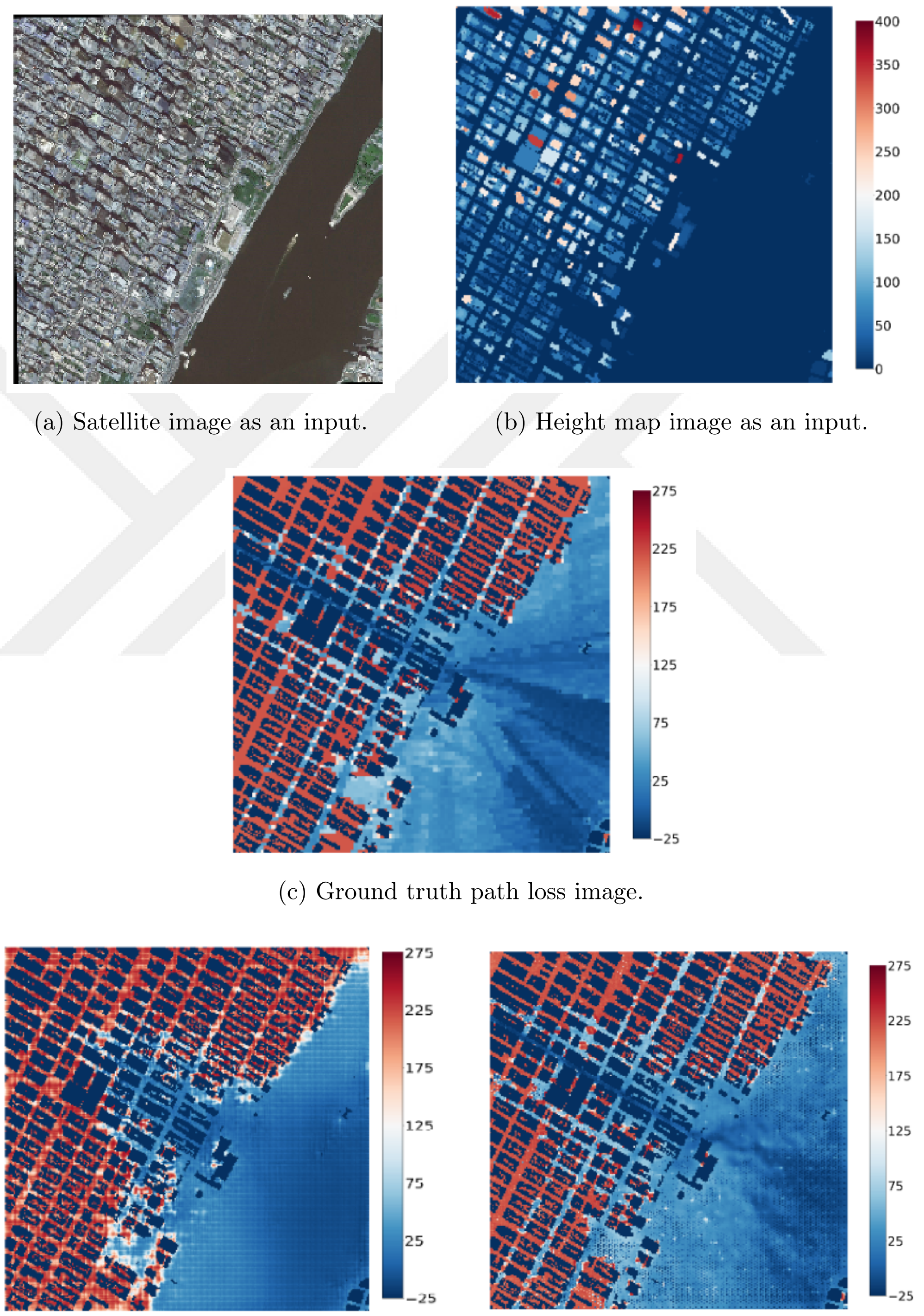


Figure 4.3.2: Training method of discriminator [8].

In Figures below, in order to highlight the performance of the proposed method for the 40m dataset, we show the satellite image of a region for a random sample, we then show its height map image for the same region. The color bar located beside the height map shows the height in meters. In the second row, we show the ground truth of the path loss image that we aim to predict perfectly from the mentioned inputs. In the third row, we demonstrate the prediction of the path loss image for the satellite input image on the left and the height map input image on the right. All the path loss images are accompanied by a color bar that shows the values in dB. In the path loss images we assigned the pixels for the indoor areas to -25dB so that the contrast could help with the visualisation of the effect of buildings since we are not predicting the indoor areas of the regions. Also, there is no path loss results that are near to that value (-25) so this should be a source of confusion. For each region input and output images figure, we show another figure that demonstrates a PDF comparison between the results of the satellite image prediction and the height map image prediction. This illustrates the success of the network to predict the path loss in a more detailed approach. Four random regions for the 40m testing dataset are shown. Inputs, outputs and the ground truth images are shown in Figures 4.3.3, 4.3.5, 4.3.7, and 4.3.9 and the corresponding PDF comparisons are shown in Figures 4.3.4, 4.3.6, 4.3.8, and 4.3.10, respectively.

For the 80m dataset, we show the satellite image of the region in the first row. In the second row, we show the ground truth path loss image, and beside it the predicted path loss image of the network for the same region. In the third row, we present a plot for path loss PDF of the the ground truth and for the predicted image for the outdoor regions. We showed examples for the 80m dataset in Figures 4.3.11 - 4.3.14. In the same manner, the results of the 300m dataset are displayed in Figures 4.3.15 - 4.3.18.



(a) Satellite image as an input.

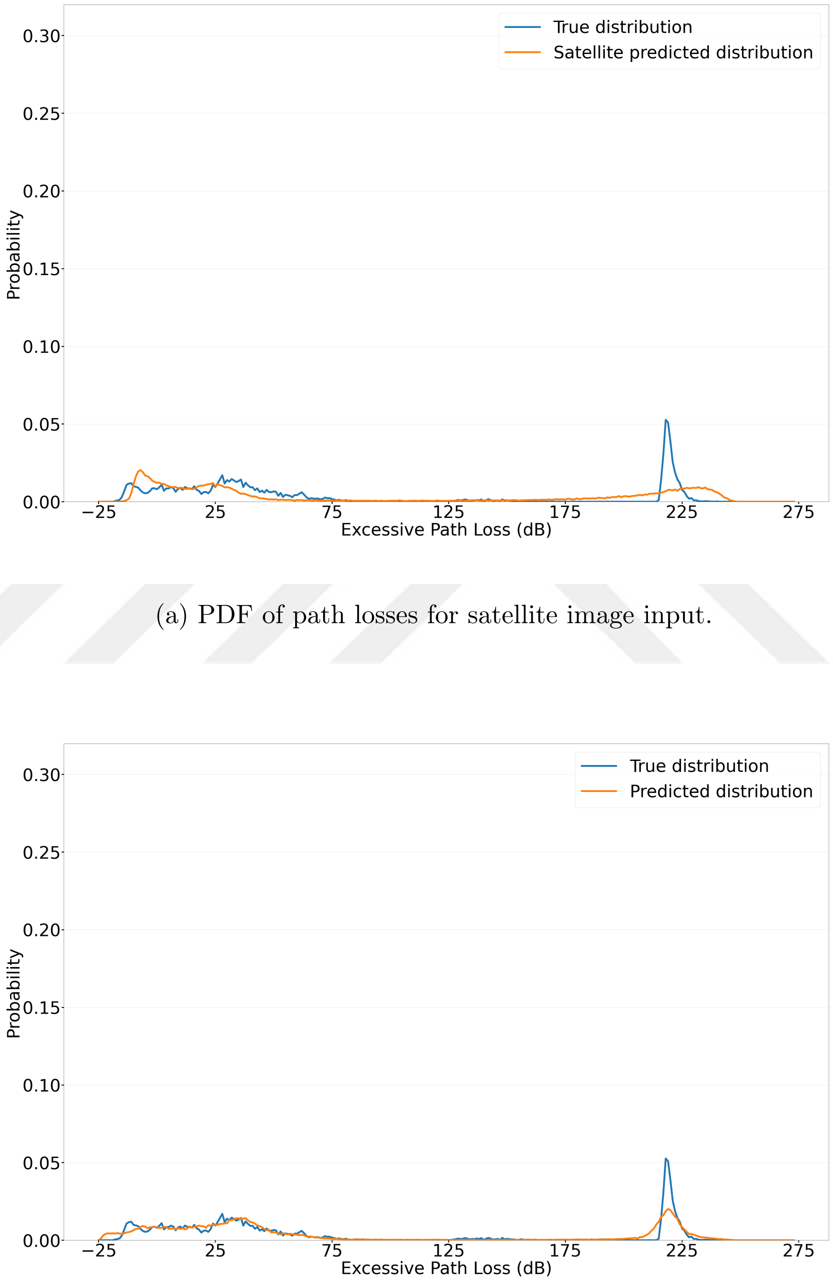
(b) Height map image as an input.

(c) Ground truth path loss image.

(d) Predicted path loss image for satellite input.

(e) Predicted path loss image for height map input.

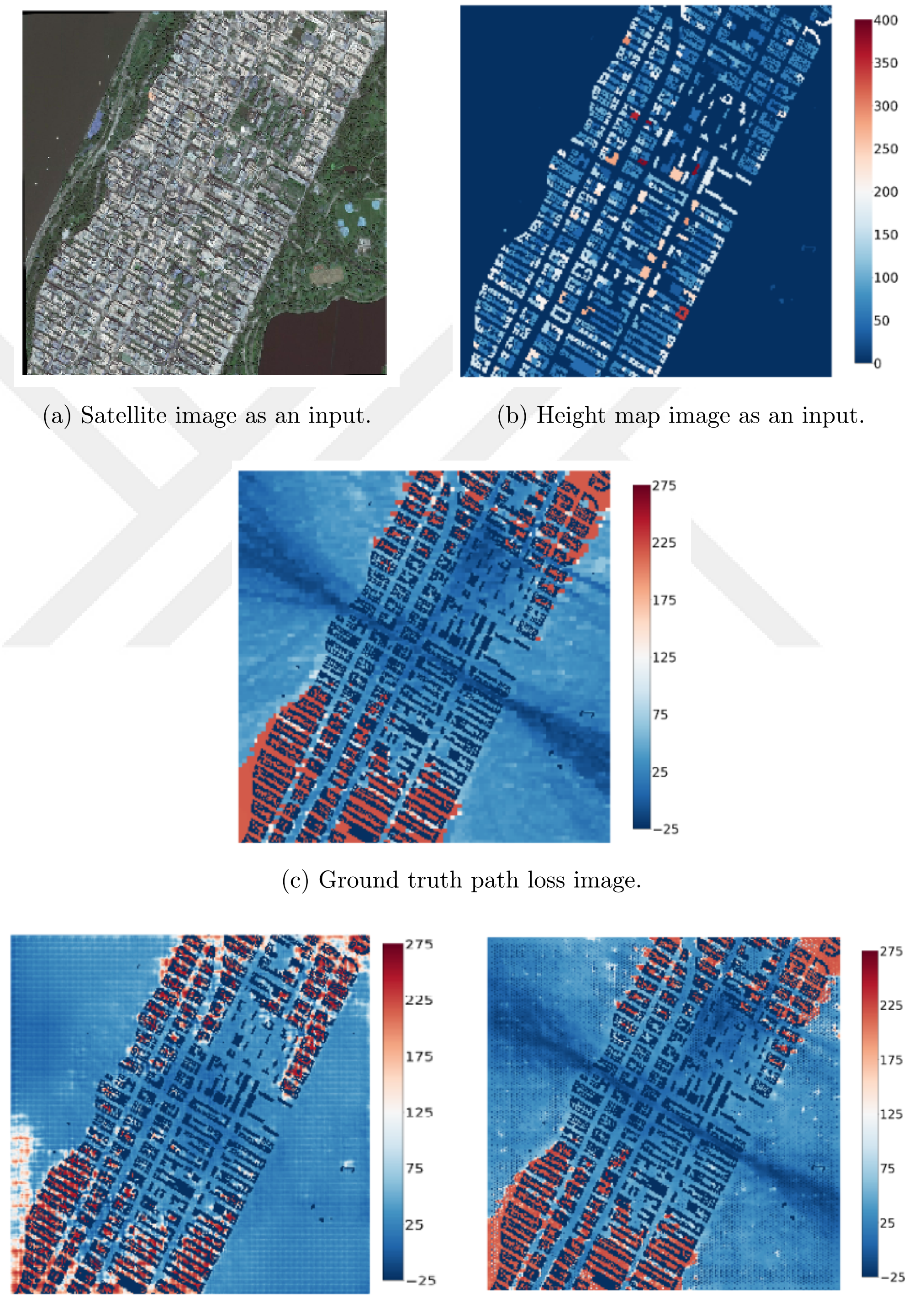
Figure 4.3.3: One of the 40m results is shown. Height map color-bar is in (m), path loss image color bar is in (dB).



(a) PDF of path losses for satellite image input.

(b) PDF of path losses for height map image input,

Figure 4.3.4: PDF comparison between satellite and height map as inputs for the region shown in Figure 4.3.3.



(a) Satellite image as an input.

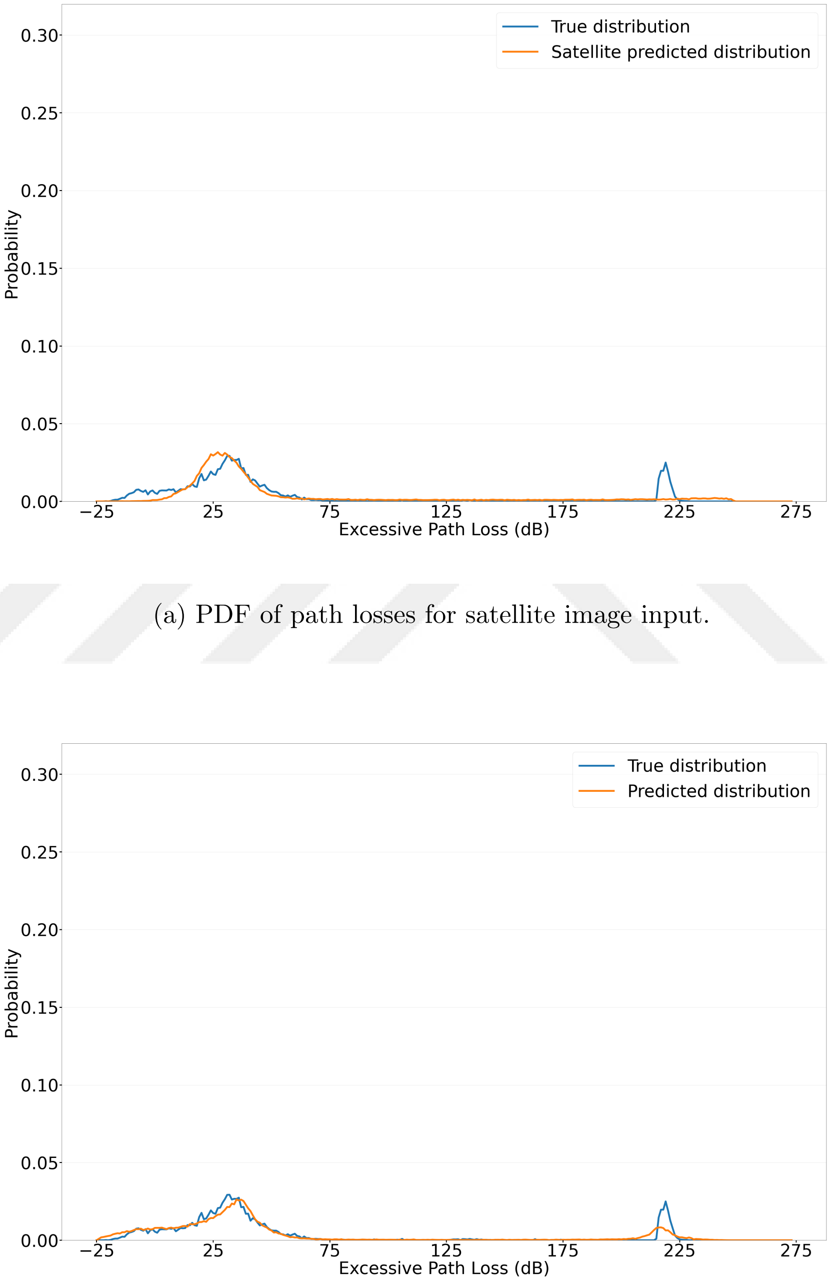
(b) Height map image as an input.

(c) Ground truth path loss image.

(d) Predicted path loss image for satellite input.

(e) Predicted path loss image for height map input.

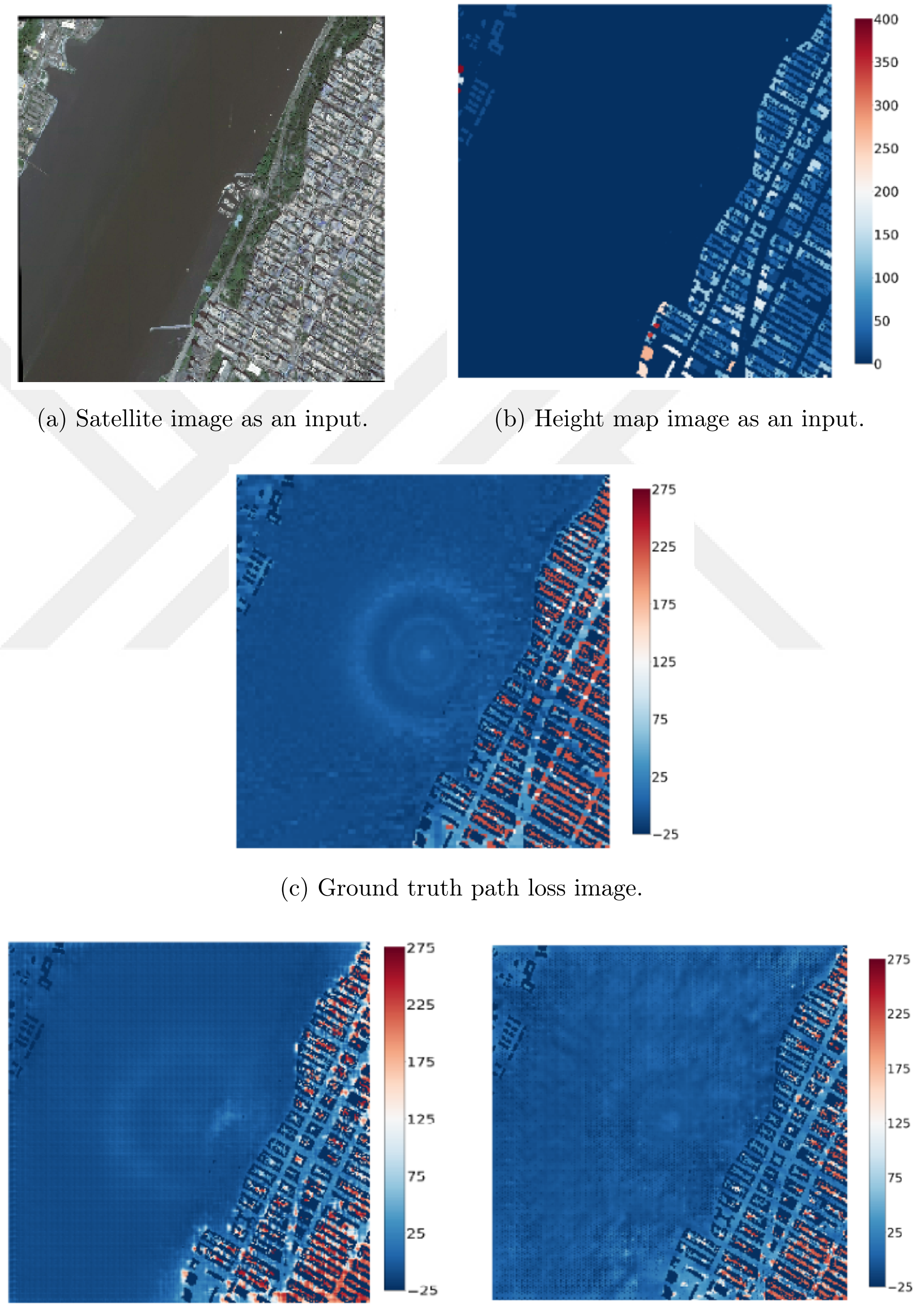
Figure 4.3.5: One of the 40m results is shown. Height map color-bar is in (m), path loss image color bar is in (dB).



(a) PDF of path losses for satellite image input.

(b) PDF of path losses for height map image input.

Figure 4.3.6: PDF comparison between satellite and height map as inputs for the region shown in Figure 4.3.5.



(a) Satellite image as an input.

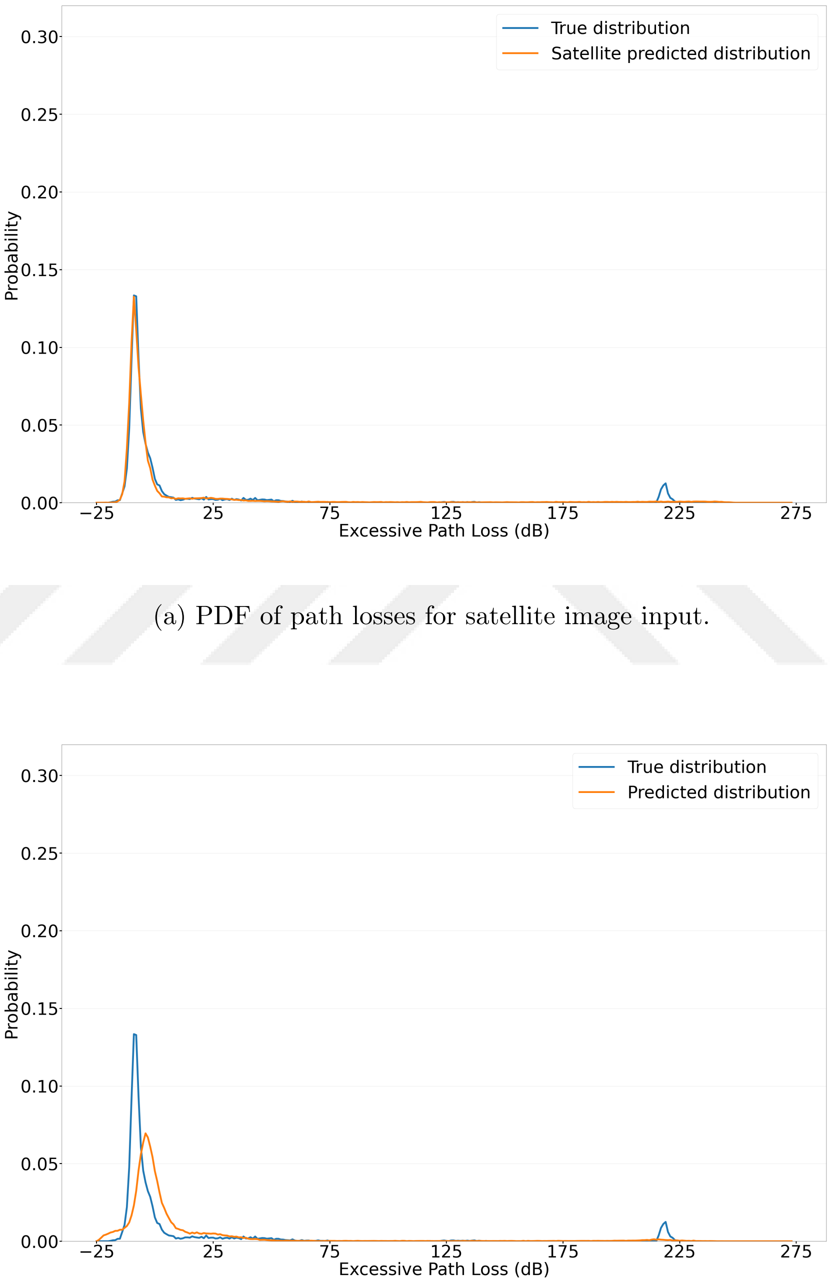
(b) Height map image as an input.

(c) Ground truth path loss image.

(d) Predicted path loss image for satellite input.

(e) Predicted path loss image for height map input.

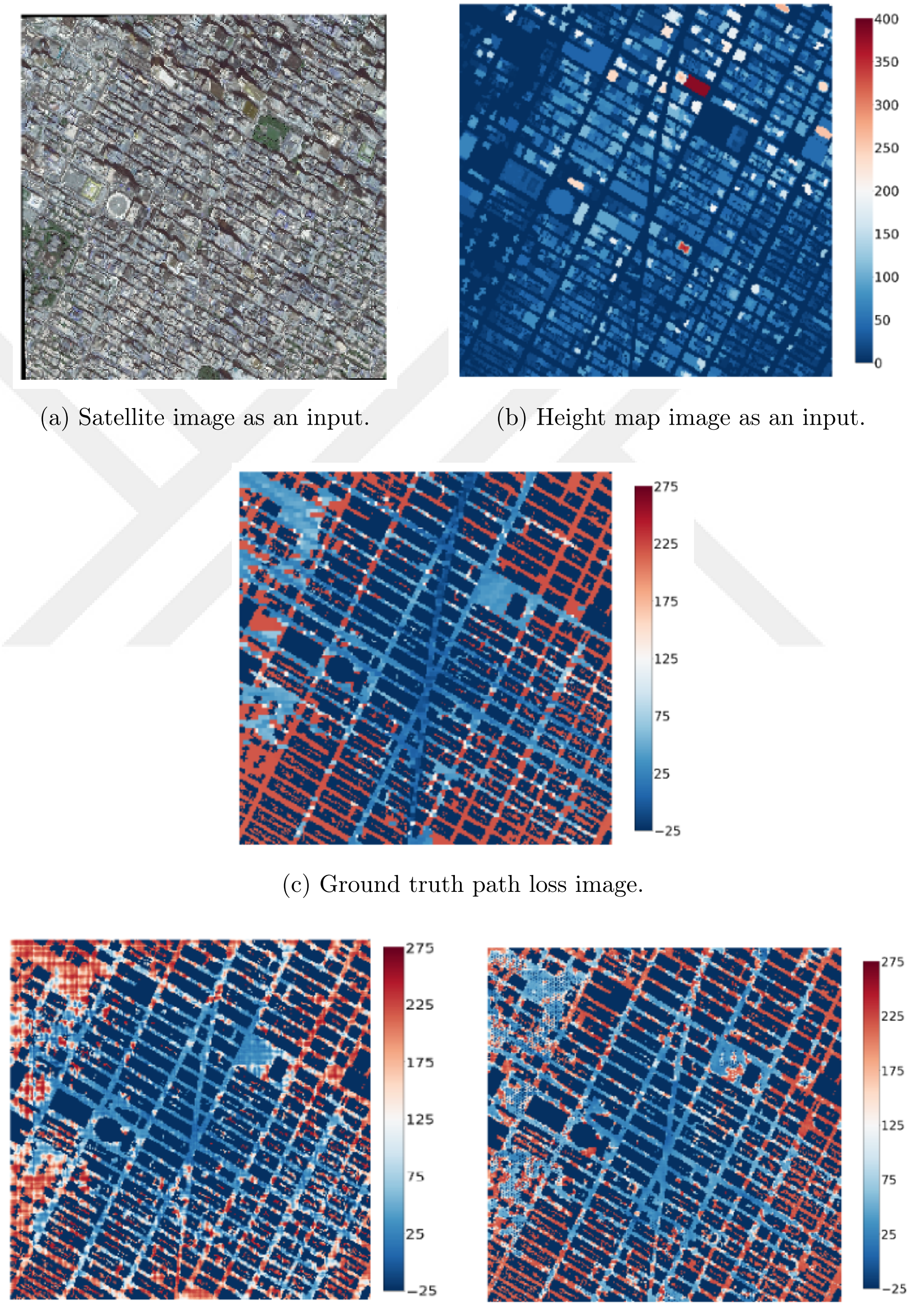
Figure 4.3.7: One of the 40m results is shown. Path loss image color bar is in (dB).



(a) PDF of path losses for satellite image input.

(b) PDF of path losses for height map image input.

Figure 4.3.8: PDF comparison between satellite and height map as inputs for the region shown in Figure 4.3.7.



(a) Satellite image as an input.

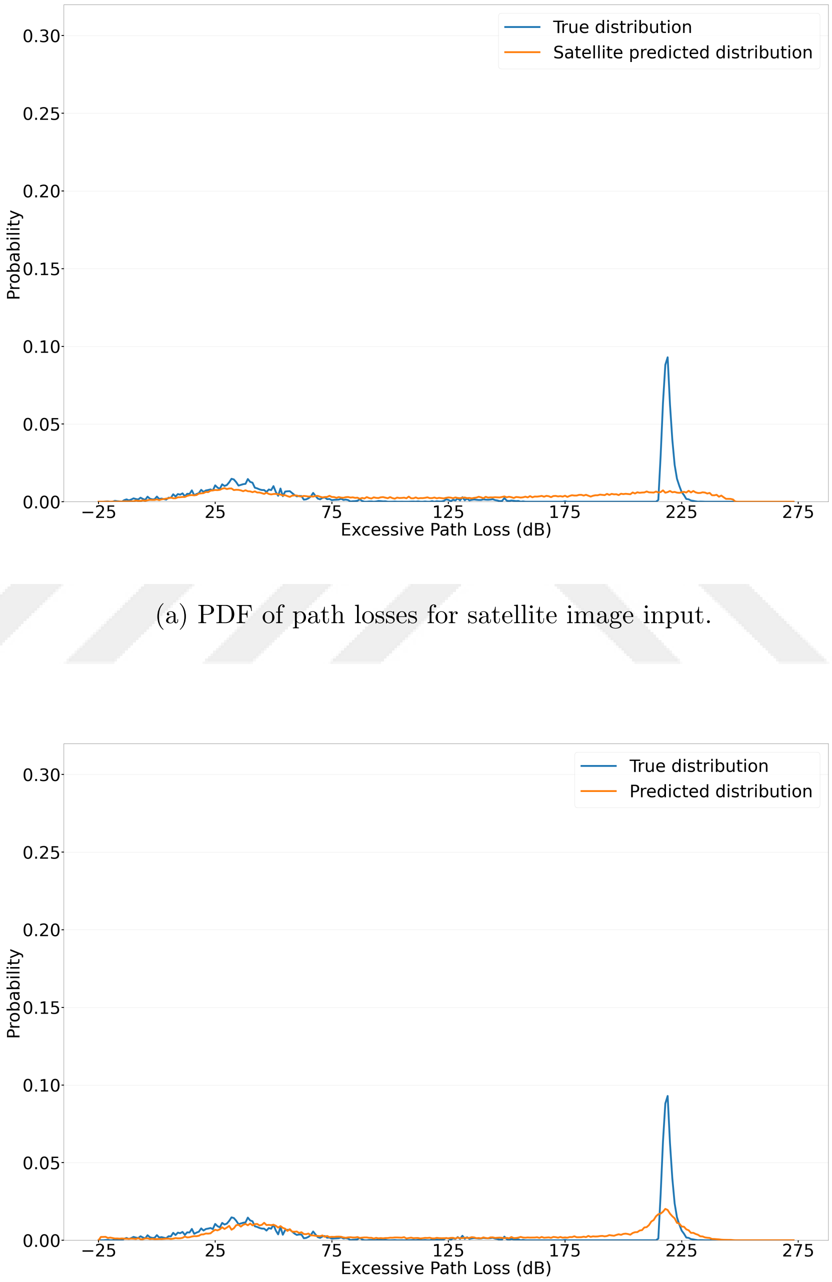
(b) Height map image as an input.

(c) Ground truth path loss image.

(d) Predicted path loss image for satellite input.

(e) Predicted path loss image for height map input.

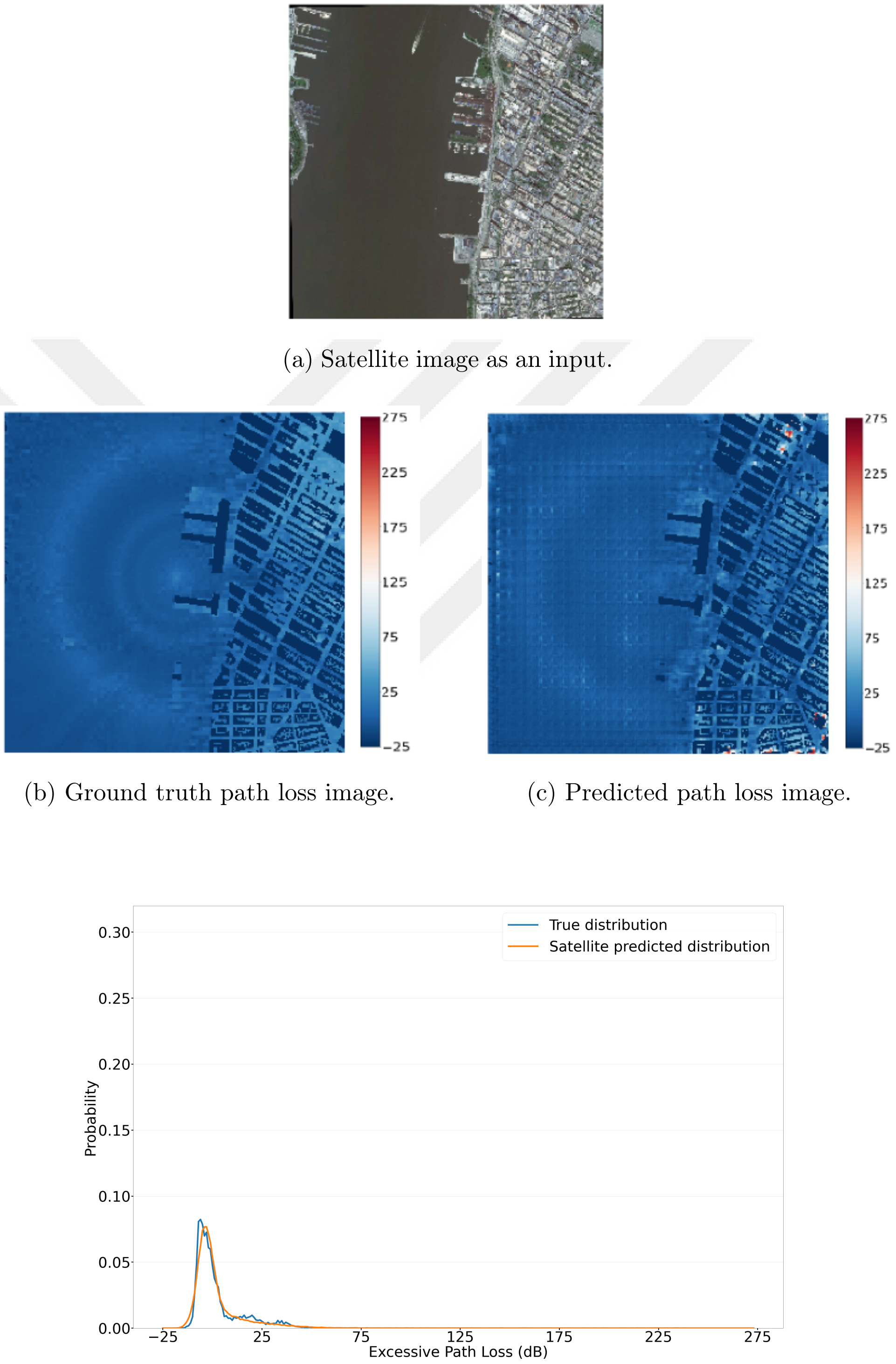
Figure 4.3.9: One of the 40m results is shown. Height map color-bar is in (m), path loss image color bar is in (dB).



(a) PDF of path losses for satellite image input.

(b) PDF of path losses for height map image input.

Figure 4.3.10: PDF comparison between satellite and height map as inputs for the region shown in Figure 4.3.9.



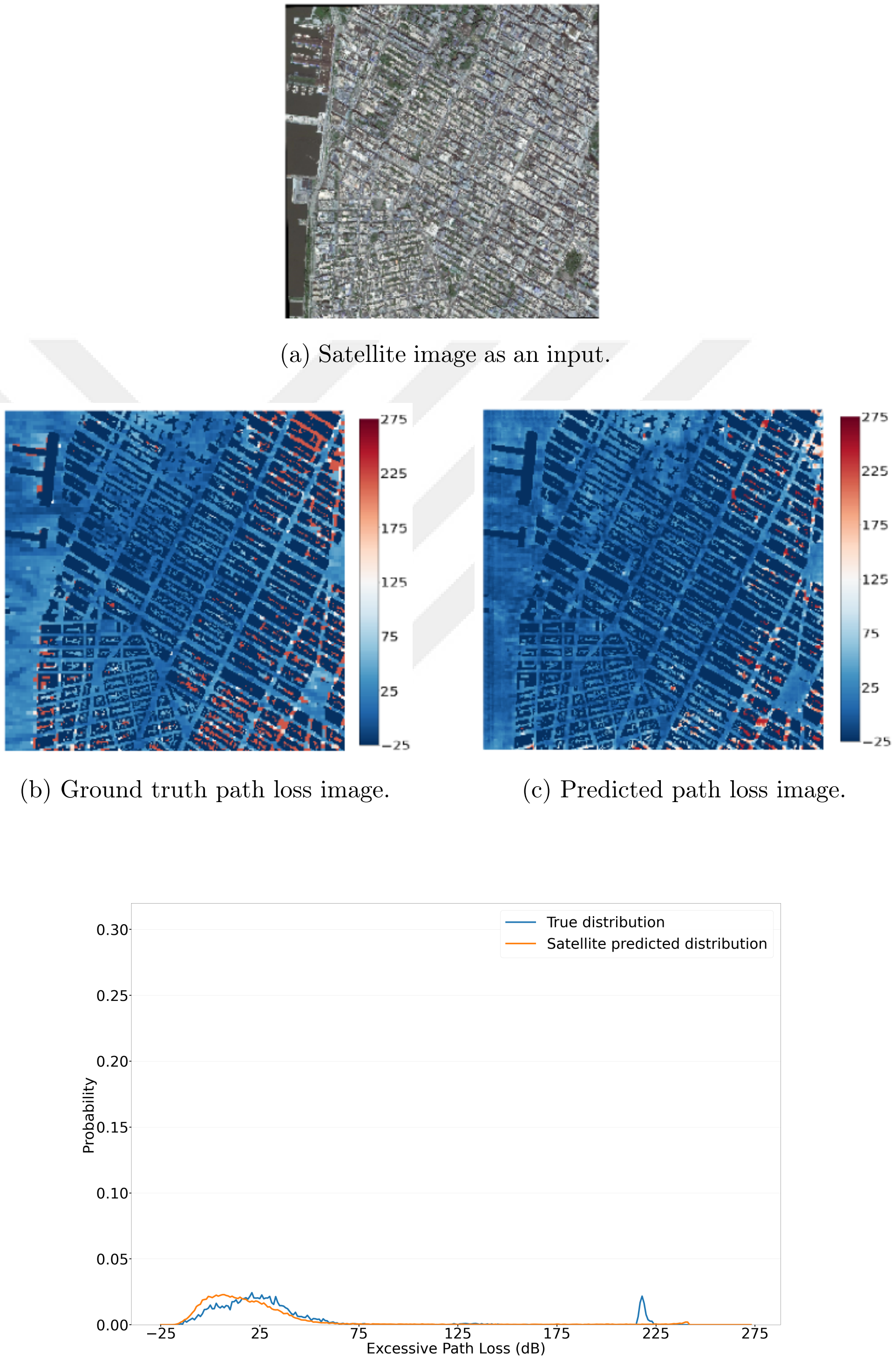
(a) Satellite image as an input.

(b) Ground truth path loss image.

(c) Predicted path loss image.

(d) PDF of 300-bin

Figure 4.3.11: One of the 80m results is shown. Path loss image color bar is in (dB).



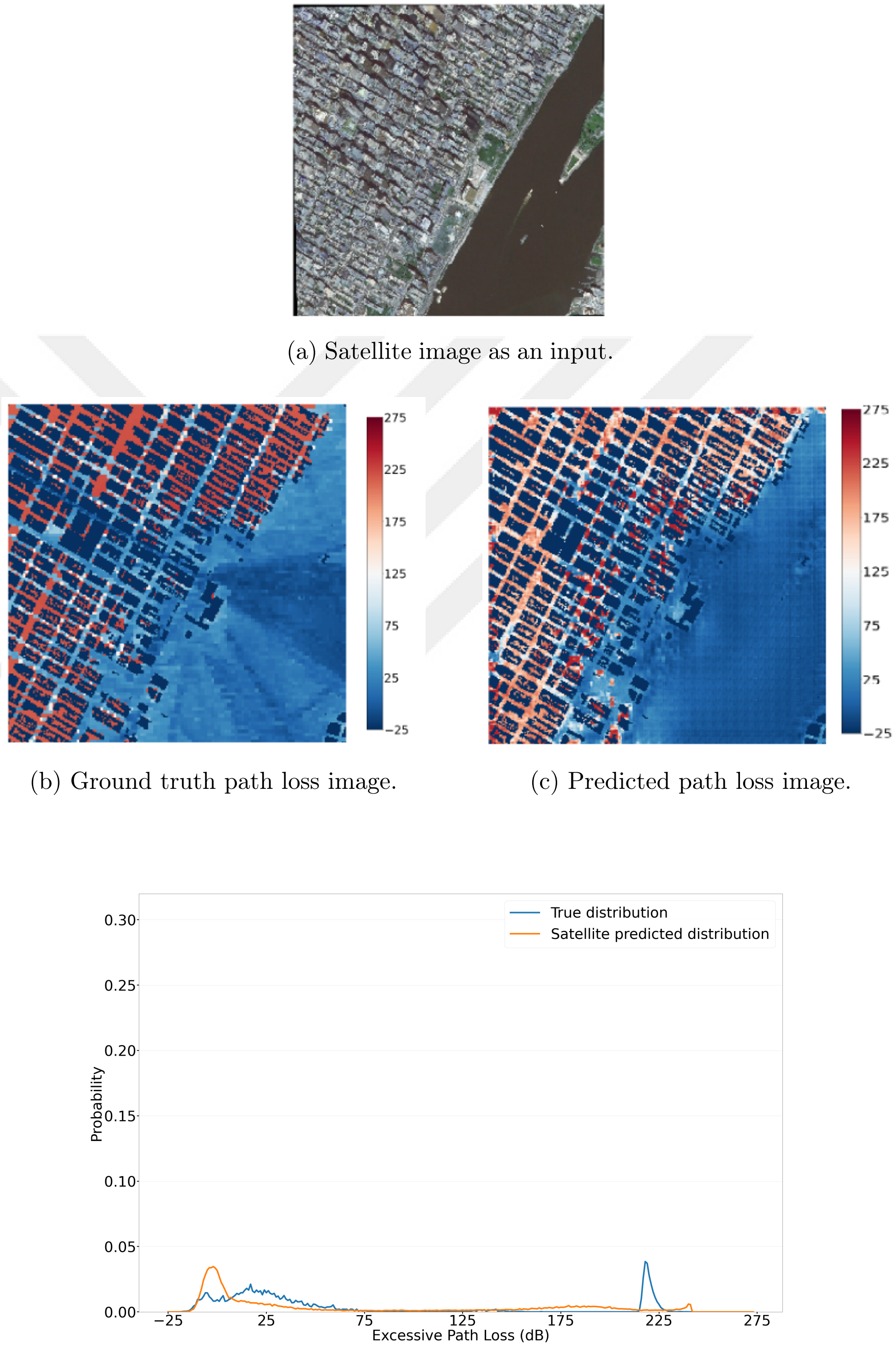
(a) Satellite image as an input.

(b) Ground truth path loss image.

(c) Predicted path loss image.

(d) PDF of 300-bin

Figure 4.3.12: One of the 80m results is shown. Path loss image color bar is in (dB).



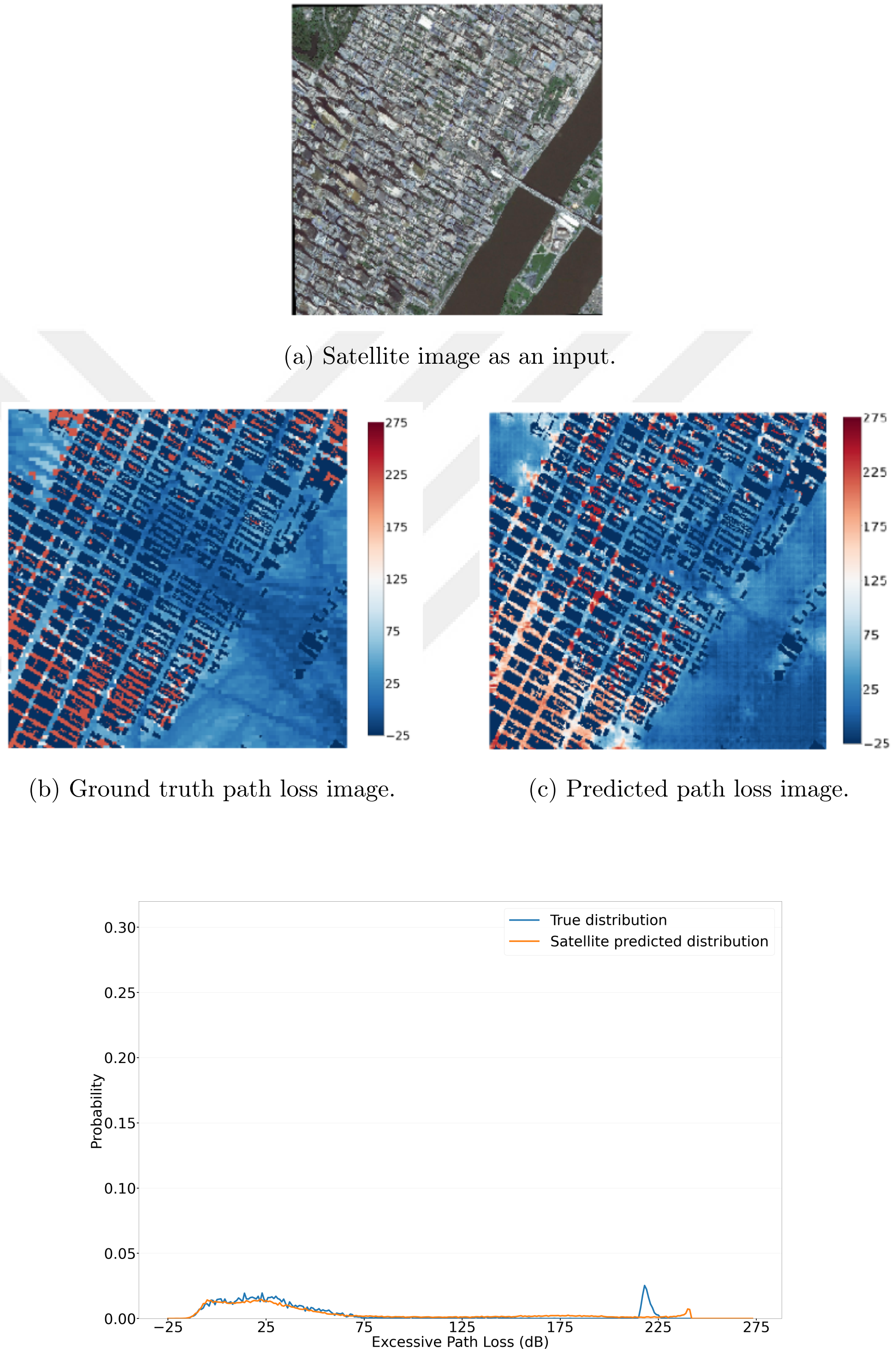
(a) Satellite image as an input.

(b) Ground truth path loss image.

(c) Predicted path loss image.

(d) PDF of 300-bin

Figure 4.3.13: One of the 80m results is shown. Path loss image color bar is in (dB).



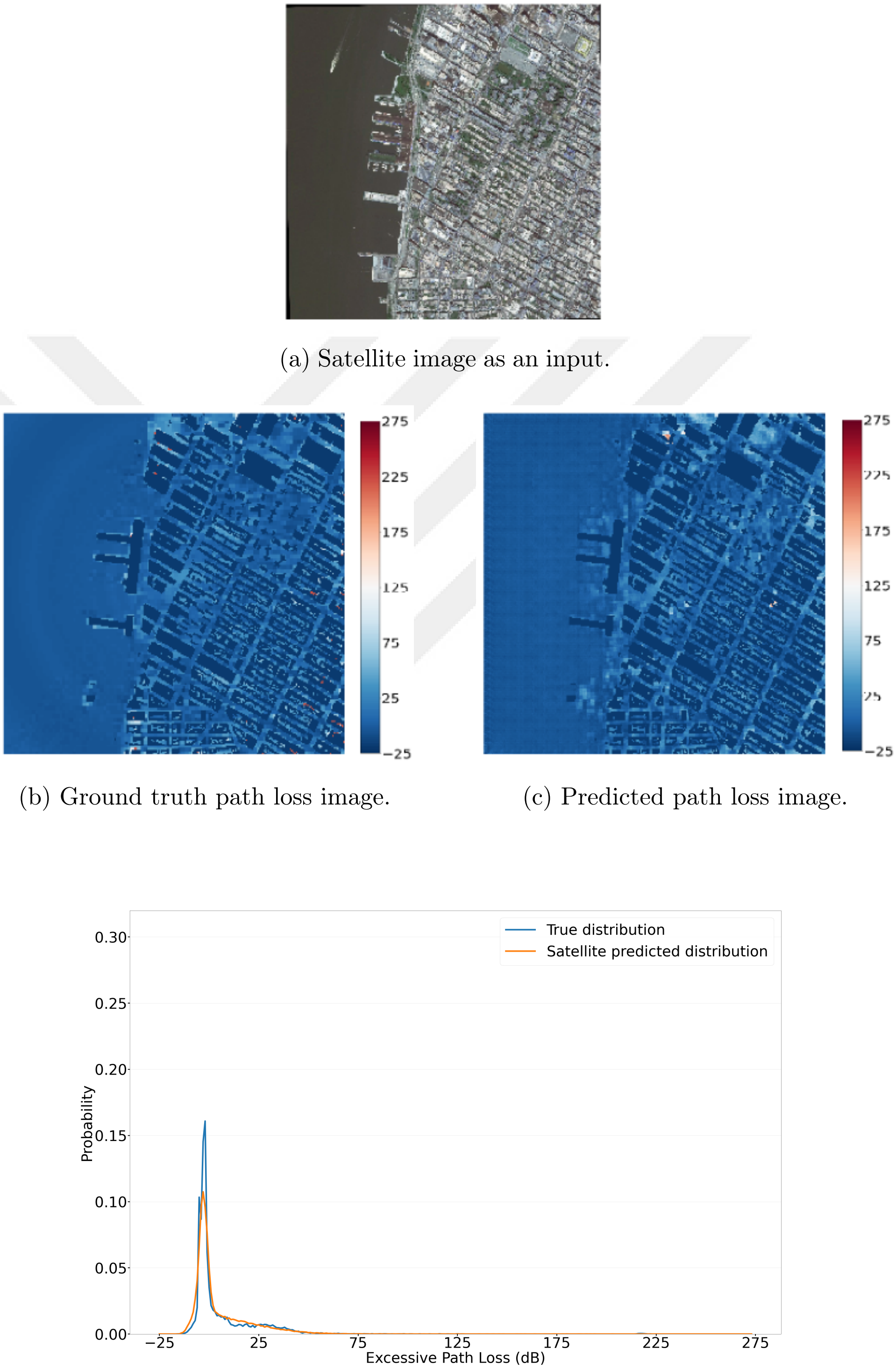
(a) Satellite image as an input.

(b) Ground truth path loss image.

(c) Predicted path loss image.

(d) PDF of 300-bin

Figure 4.3.14: One of the 80m results is shown. Path loss image color bar is in (dB).



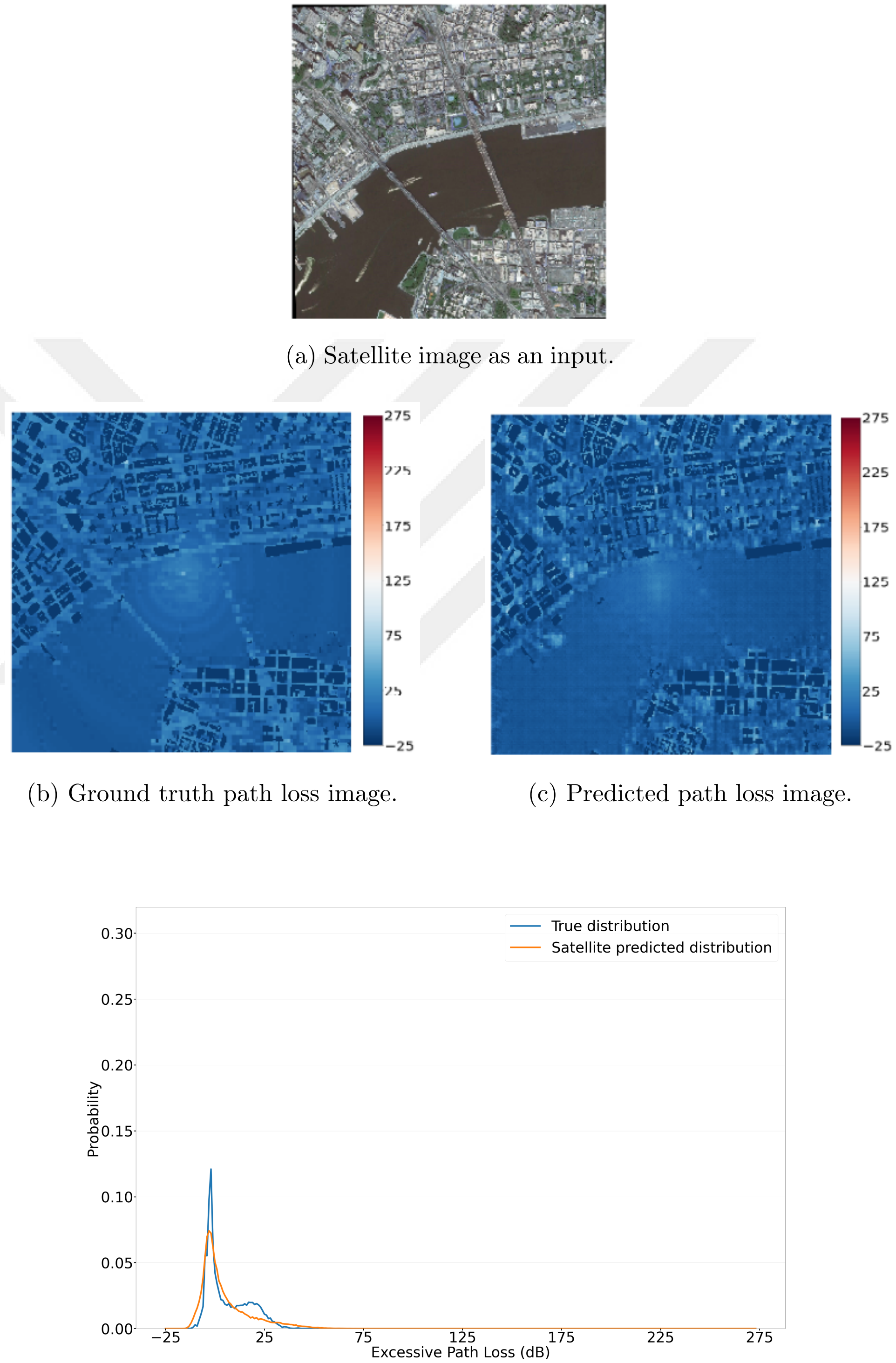
(a) Satellite image as an input.

(b) Ground truth path loss image.

(c) Predicted path loss image.

(d) PDF of 300-bin

Figure 4.3.15: One of the 300m results is shown. Path loss image color bar is in (dB).



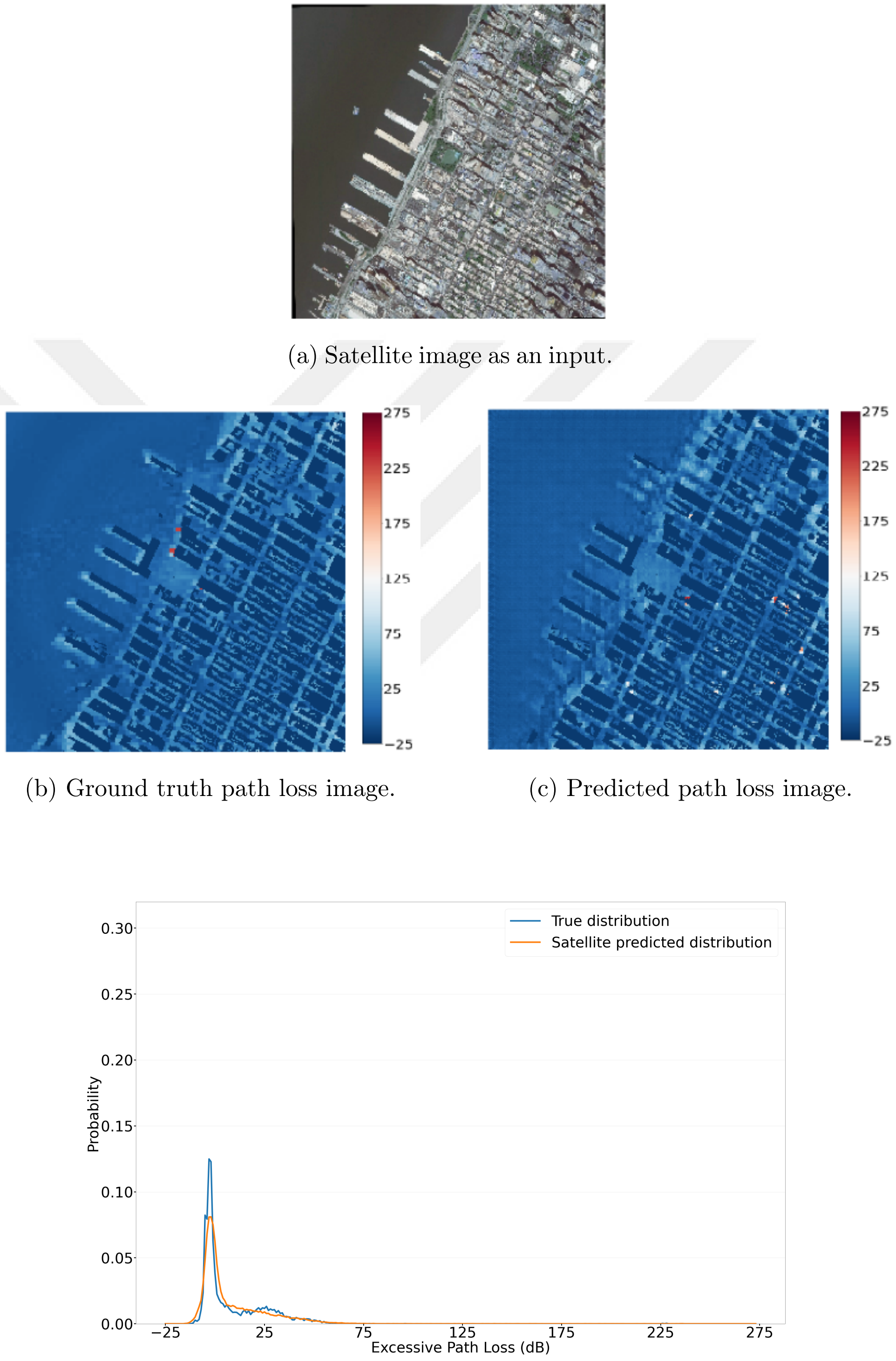
(a) Satellite image as an input.

(b) Ground truth path loss image.

(c) Predicted path loss image.

(d) PDF of 300-bin

Figure 4.3.16: One of the 300m results is shown. Path loss image color bar is in (dB).



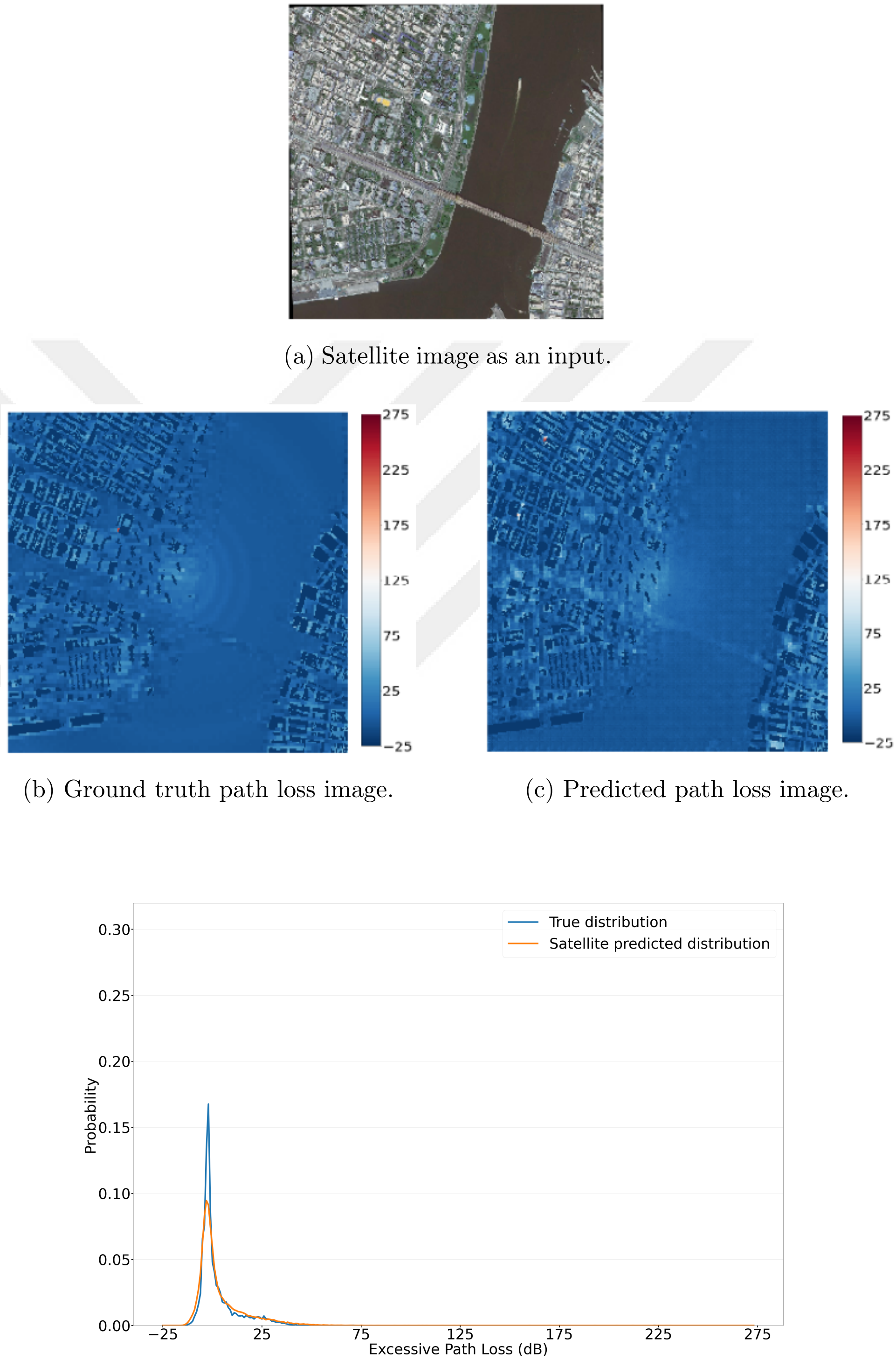
(a) Satellite image as an input.

(b) Ground truth path loss image.

(c) Predicted path loss image.

(d) PDF of 300-bin

Figure 4.3.17: One of the 300m results is shown. Path loss image color bar is in (dB).



(a) Satellite image as an input.

(b) Ground truth path loss image.

(c) Predicted path loss image.

(d) PDF of 300-bin

Figure 4.3.18: One of the 300m results is shown. Path loss image color bar is in (dB).

4.4 Discussion

For the evaluation part of the network, we use MSE. We test two parameters, the average MSE where we compare the generated image and the true image, also the probability distribution in the same way we evaluated the previous experiment, where we calculate the PDF of the predicted and true excessive path losses then get the MSE between the two functions. We set the buildings and the indoor receivers in the path loss image to the minimum value which is -25 to show higher contrast between the indoor and the outdoor for better visualisation of the distribution of the path losses around the buildings. We also only consider the outdoor receivers for evaluation and we do not consider the indoor ones for all the evaluated parameters.

We trained for the satellite images at transmitter heights 300 and 80 meters and we showed the results in Table 4.1, then for both height maps and satellite images for the 40 meters and we can see the results in Table 4.2. Since the average point-wise MSE is higher than the variance, we can understand that the satellite images were not very successful at predicting the point wise path loss. On the other hand, the average MSE results regarding the probability distribution estimation are very small with very small resolution where the bin represents 1 dB. In [3], the MSE was one order of magnitude higher and the bin represents 30 dB. This proves that our work achieves better accuracy at a more difficult challenge.

Table 4.1: Point-wise Excessive Path Loss Results.

	Satellite Predictions	
Dataset	**80 m**	**300 m**
Average point-wise MSE	745	410
Average PDF 300-bin MSE	1.28×10^{-5}	4.23×10^{-5}
Path loss receiver variance	520	277

It is clear from the previous figures showing the 80m results that the network is successful in giving a general view for the distribution of the path loss across the region and they can demonstrate the trend of the path losses and the regions that suffer the most from high path loss. However, the accuracy of the pixel values is not very good, especially in the very challenging samples. On the other hand, the results are very informative when we look at the path loss image as they give a general view of the distribution of the path losses and can demonstrate the trend of the path losses and how they fare across the area, they illustrate where the high path losses are concentrated, around what values they lie, and how that would change when changing the location of the transmitter. The limitation of results regarding the pixel accuracy were expected because here we used the satellite images as an input that lack the height information needed for an accurate path loss image.

As for the 300m results, we found that the predicted path loss images were better, even though they do not fare well in the MSE tables. This is probably due to the very low standard deviation of the data that the network struggles with. This is also obvious when we look at the PDFs of the samples for the 300m height, where the true distribution is very narrow at the beginning of the function, so that any mismatch with the predicted function results in a very high MSE. When we observe the prediction as a whole and look at the PDFs we can see that the network is successful in predicting it. When the results are compared with [3], MSE is smaller by an order of magnitude, showing that GAN approach is much better at estimating the PDF of the path losses than using the conventional CNN models.

For the comparison between satellite and height map input in the 40m transmitter height dataset, we can see that the height map input was able to influence the network to capture the shadowing effect especially in Figures 4.3.5 and 4.3.3. Also by looking at the predicted probability distributions, we can notice that the height map input is better at predicting the high path loss probabilities better than the satellite input. Even though we can notice sometimes the predicted high path losses are not as concentrated as in the ground truth probability distribution, it still captures the ground truth behaviour at the right interval with slightly wider probability band and this is clear in Figures 4.3.4 and 4.3.10.

Another interesting point is that, compared to the 300m and 80m datasets, the 40m dataset behaves better regarding the variance of the path losses, where in the former case MSE was larger while in the latter it is almost equal to the variance, as demonstrated in Table 4.2, showing that the network performed better in the 40m dataset. This is because this dataset has high variations in the values, which makes it possible for the network to better adapt to these variations. On the contrary, in the higher altitude datasets the path losses show peaked distributions, which makes a slight mismatch in the prediction causing a high overall MSE.

Table 4.2: Point-wise Excessive Path Loss Comparison For 40m Transmitter Height.

	40m Predictions	
Dataset	**Satellite Input**	**Height Map Input**
Average point-wise mse	2770	2800
Average PDF 300-bin mse	3.04×10^{-5}	4.9×10^{-5}
Path loss receiver variance	2750	2750

Chapter 5

Conclusion

In this thesis, we proposed two methods in order to estimate the PDF of excessive path loss for a specific region. We used height map images and satellite images as inputs. We proposed using GAN and CNN models for the supervised estimation problem. The methods prove their real time applicability and provide a solution to the high computational problem of ray tracing simulations. Our two methods offered the way to counter these problems and offered the ability for adequate urgent planning of communication systems without affecting the accuracy of the information obtained. First for the CNN model, it was successful in obtaining satisfying accuracy in predicting the path loss distribution from both the satellite images and height maps. However it failed to benefit from the height information, whether it is the height of the buildings or their distribution across the region. We tried different networks and different feature manipulations but all proved that the inherent characteristics of CNN are not suitable for this kind of information.

As for the GAN models, we found that the GAN was able to synthesize the path loss image and successfully predict path loss value of each receiver directly from an image in a point-wise manner. This GAN-based solution was applied for the first time in literature, as far as we know. It was also able to predict the general distribution of the path losses across the region and to generate visually informative path loss images showing the general shadowing effects of the buildings. We also found that, unlike CNNs, the GAN model was able to benefit from the height map images and its extra building height information that is missing from the satellite images.

This work may be extended in several different ways. Future work includes further experimentation on GAN-based approach, such as trying other networks for the generator and the discriminator as well. The work in this thesis was limited to estimating simulated data. We believe it would be very insightful to continue those approaches using real measurements. Further analysis is also required to explain mathematically the ability of GANs to benefit from the height information in the height maps and, on the other hand, to explain the inability of the classical CNNs to benefit from it.

Bibliography

[1] P. Medeđović, M. Veletić, and Ž. Blagojević, "Wireless insite software verification via analysis and comparison of simulation and measurement results," in *2012 Proceedings of the 35th International Convention MIPRO*, pp. 776–781, 2012.

[2] G. Yang, Y. Zhang, Z. He, J. Wen, Z. Ji, and Y. Li, "Machine-learning-based prediction methods for path loss and delay spread in air-to-ground millimetre-wave channels," *IET Microwaves, Antennas & Propagation*, vol. 13, no. 8, pp. 1113–1121, 2019.

[3] O. Ahmadien, H. F. Ates, T. Baykas, and B. K. Gunturk, "Predicting path loss distribution of an area from satellite images using deep learning," *IEEE Access*, vol. 8, pp. 64982–64991, 2020.

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

[5] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2223–2232, 2017.

[6] Y. Yang, Y. Li., W. Zhang, F. Qinand P. Zhu, and C. Wang, "Generative-adversarial-network-based wireless channel modeling: Challenges and opportunities," *IEEE Communications Magazine*, vol. 57, no. 3, pp. 22–27, 2019.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[8] “Image to image GAN using Tensorflow.” `https://www.tensorflow.org/tutorials/generative/pix2pix`. Accessed: 2021-01-20.

[9] A. Al-Hourani, S. Kandeepan, and A. Jamalipour, “Modeling air-to-ground path loss for low altitude platforms in urban environments,” in *IEEE Global Communications Conference*, pp. 2898–2904, 2014.

[10] D. J. Cichon and T. Kürner, “Propagation prediction models,” *COST-231 TD*, vol. 95, no. 66, pp. 115–207, 1995.

[11] M. Piacentini and F. Rinaldi, “Path loss prediction in urban environment using learning machines and dimensionality reduction techniques,” *Computational Management Science*, vol. 8, no. 4, pp. 371–385, 2011.

[12] Y. Singh, “Comparison of okumura, hata and cost-231 models on the basis of path loss and signal strength,” *International Journal of Computer Applications*, vol. 59, no. 11, 2012.

[13] M. Hata, “Empirical formula for propagation loss in land mobile radio services,” *IEEE Transactions on Vehicular Technology*, vol. 29, no. 3, pp. 317–325, 1980.

[14] B. Ai, K. Guan, R. He, J. Li, G. Li, D. He, Z. Zhong, and K. M. S. Huq, “On indoor millimeter wave massive mimo channels: Measurement and simulation,” *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 7, pp. 1678–1690, 2017.

[15] M. Zhu, A. Singh, and F. Tufvesson, “Measurement based ray launching for analysis of outdoor propagation,” in *2012 6th European Conference on Antennas and Propagation (EUCAP)*, pp. 3332–3336, 2012.

[16] T. K. Geok, F. Hossain, M. N. Kamaruddin, N. Z. Abd Rahman, S. Thiagarajah, A. T. W. Chiat, J. Hossen, and C. P. Liew, “A comprehensive review of efficient ray-tracing techniques for wireless communication,” *International Journal on Communications Antenna and Propagation (IRECAP)*, vol. 8, no. 2, pp. 123–136, 2018.

[17] H. F. Ates, S. M. Hashir, T. Baykas, and B. K. Gunturk, “Path loss exponent and shadowing factor prediction from satellite images using deep learning,” *IEEE Access*, vol. 7, pp. 101366–101375, 2019.

[18] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern recognition*, pp. 770–778, 2016.

[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*, pp. 2672–2680, 2014.

[20] G. Biau, B. Cadre, M. Sangnier, U. Tanielian, *et al.*, “Some theoretical properties of gans,” *Annals of Statistics*, vol. 48, no. 3, pp. 1539–1566, 2020.

[21] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” *arXiv preprint arXiv:1411.1784*, 2014.

[22] R. A. Yeh, C. Chen, T. Yian Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, “Semantic image inpainting with deep generative models,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5485–5493, 2017.

[23] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, “Photo-realistic single image super-resolution using a generative adversarial network,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4681–4690, 2017.

[24] S. Yang, L. Xie, X. Chen, X. Lou, X. Zhu, D. Huang, and H. Li, “Statistical parametric speech synthesis using generative adversarial networks under a multi-task learning framework,” in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 685–691, 2017.

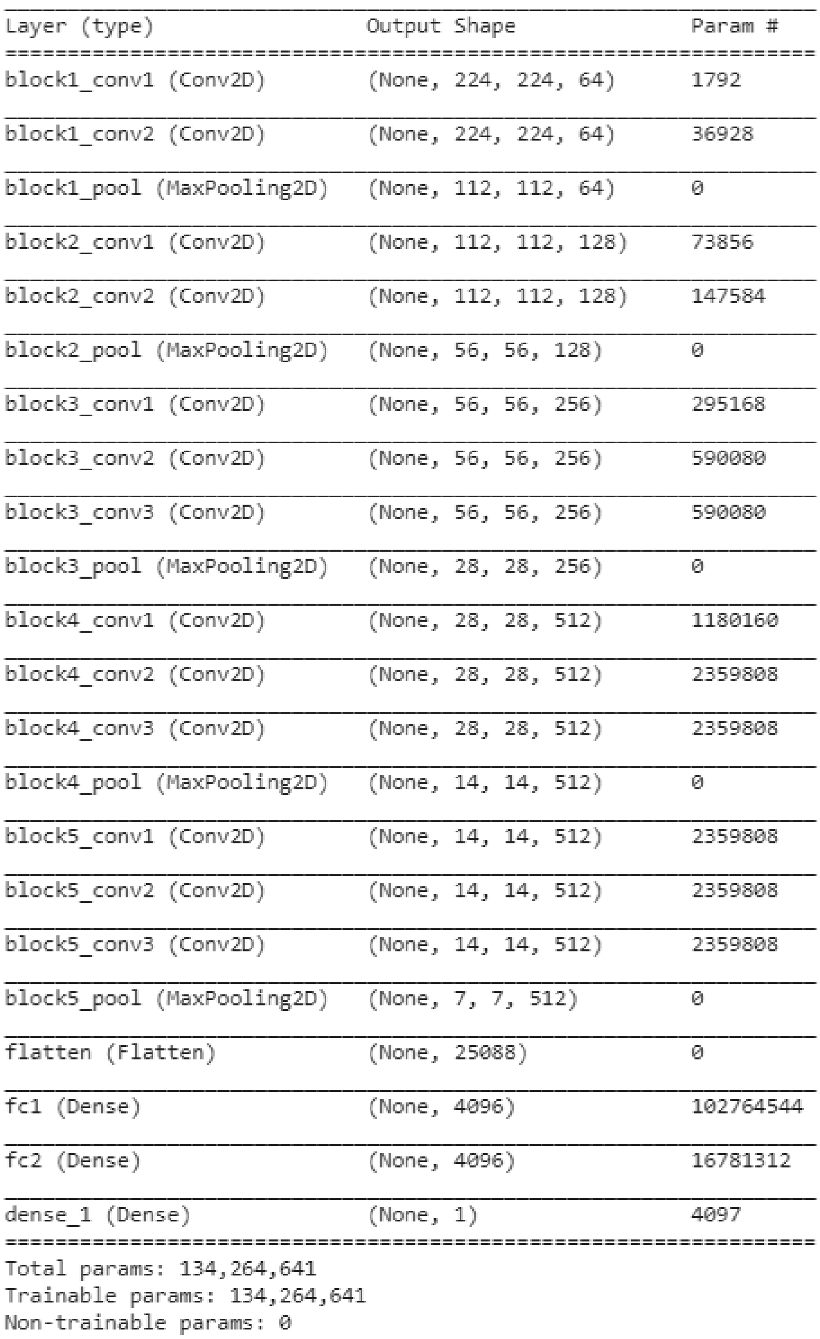
[25] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2223–2232, 2017.

[26] C. Li and M. Wand, “Precomputed real-time texture synthesis with markovian generative adversarial networks,” in *European Conference on Computer Vision*, pp. 702–716, Springer, 2016.

Appendix A

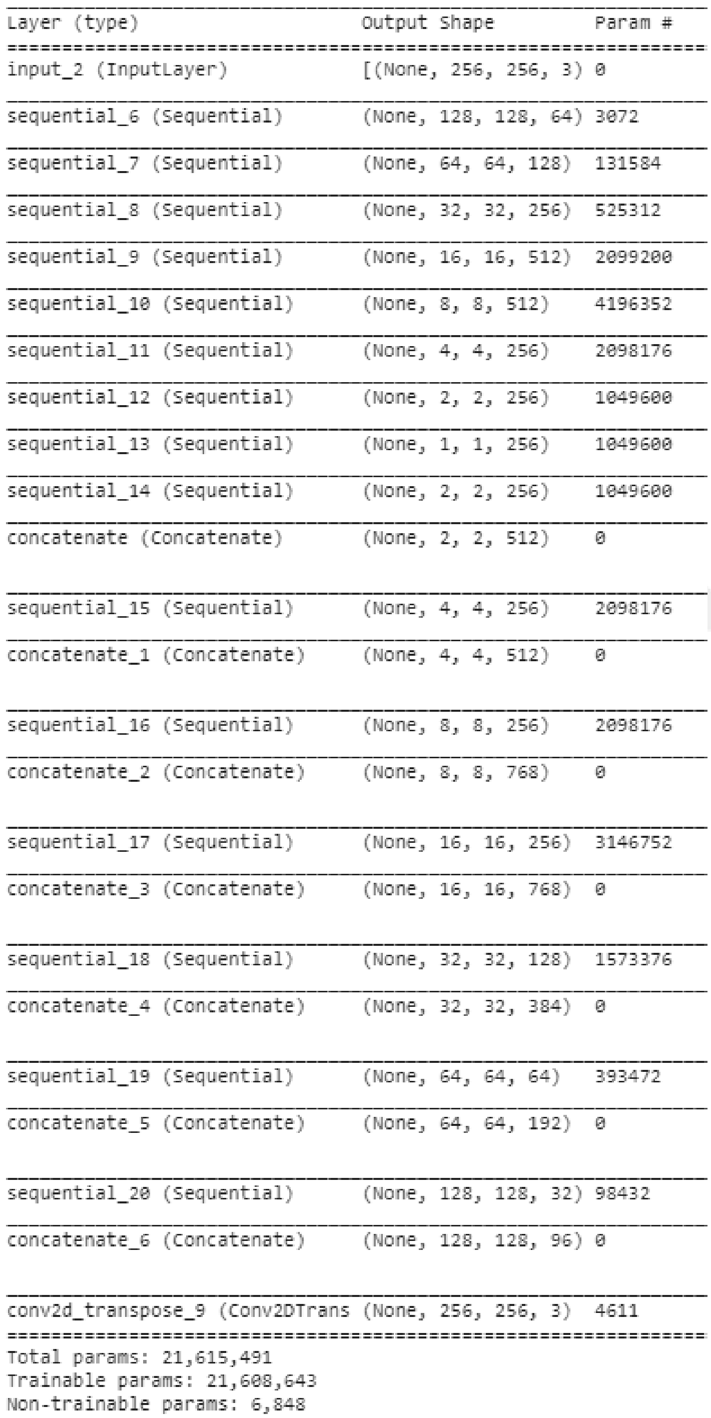
Networks Summaries

A.1 VGG-16 Network



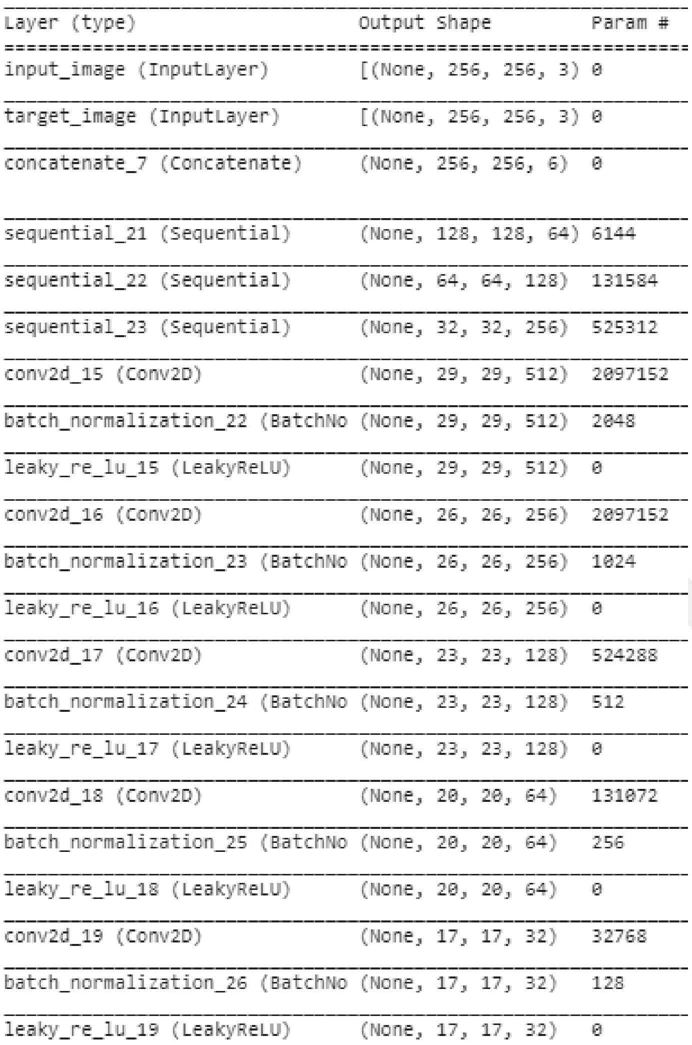
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792      
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928     
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0         
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856     
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584    
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0         
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168    
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080    
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080    
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0         
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160   
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808   
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808   
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0         
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808   
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808   
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808   
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0         
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0         
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544 
_________________________________________________________________
fc2 (Dense)                  (None, 4096)              16781312  
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 4097      
=================================================================
Total params: 134,264,641
Trainable params: 134,264,641
Non-trainable params: 0
```

A.2 GAN Generator

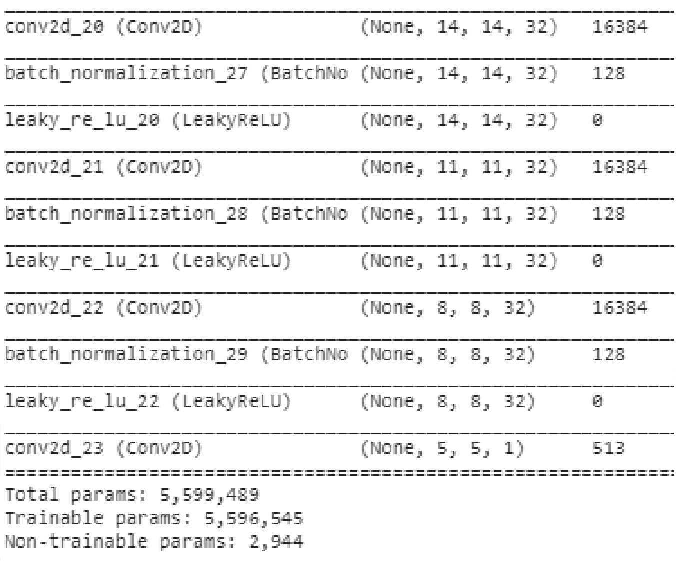


```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 256, 256, 3) 0
_________________________________________________________________
sequential_6 (Sequential)    (None, 128, 128, 64) 3072
_________________________________________________________________
sequential_7 (Sequential)    (None, 64, 64, 128)  131584
_________________________________________________________________
sequential_8 (Sequential)    (None, 32, 32, 256)  525312
_________________________________________________________________
sequential_9 (Sequential)    (None, 16, 16, 512)  2099200
_________________________________________________________________
sequential_10 (Sequential)   (None, 8, 8, 512)    4196352
_________________________________________________________________
sequential_11 (Sequential)   (None, 4, 4, 256)    2098176
_________________________________________________________________
sequential_12 (Sequential)   (None, 2, 2, 256)    1049600
_________________________________________________________________
sequential_13 (Sequential)   (None, 1, 1, 256)    1049600
_________________________________________________________________
sequential_14 (Sequential)   (None, 2, 2, 256)    1049600
_________________________________________________________________
concatenate (Concatenate)    (None, 2, 2, 512)    0

_________________________________________________________________
sequential_15 (Sequential)   (None, 4, 4, 256)    2098176
_________________________________________________________________
concatenate_1 (Concatenate)  (None, 4, 4, 512)    0

_________________________________________________________________
sequential_16 (Sequential)   (None, 8, 8, 256)    2098176
_________________________________________________________________
concatenate_2 (Concatenate)  (None, 8, 8, 768)    0

_________________________________________________________________
sequential_17 (Sequential)   (None, 16, 16, 256)  3146752
_________________________________________________________________
concatenate_3 (Concatenate)  (None, 16, 16, 768)  0

_________________________________________________________________
sequential_18 (Sequential)   (None, 32, 32, 128)  1573376
_________________________________________________________________
concatenate_4 (Concatenate)  (None, 32, 32, 384)  0

_________________________________________________________________
sequential_19 (Sequential)   (None, 64, 64, 64)   393472
_________________________________________________________________
concatenate_5 (Concatenate)  (None, 64, 64, 192)  0

_________________________________________________________________
sequential_20 (Sequential)   (None, 128, 128, 32) 98432
_________________________________________________________________
concatenate_6 (Concatenate)  (None, 128, 128, 96) 0

_________________________________________________________________
conv2d_transpose_9 (Conv2DTrans (None, 256, 256, 3)  4611
=================================================================
Total params: 21,615,491
Trainable params: 21,608,643
Non-trainable params: 6,848
```

A.3 GAN Discriminator



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_image (InputLayer)     [(None, 256, 256, 3) 0
_________________________________________________________________
target_image (InputLayer)    [(None, 256, 256, 3) 0
_________________________________________________________________
concatenate_7 (Concatenate)  (None, 256, 256, 6)  0

_________________________________________________________________
sequential_21 (Sequential)   (None, 128, 128, 64) 6144
_________________________________________________________________
sequential_22 (Sequential)   (None, 64, 64, 128)  131584
_________________________________________________________________
sequential_23 (Sequential)   (None, 32, 32, 256)  525312
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 29, 29, 512)  2097152
_________________________________________________________________
batch_normalization_22 (BatchNo (None, 29, 29, 512)  2048
_________________________________________________________________
leaky_re_lu_15 (LeakyReLU)   (None, 29, 29, 512)  0
_________________________________________________________________
conv2d_16 (Conv2D)           (None, 26, 26, 256)  2097152
_________________________________________________________________
batch_normalization_23 (BatchNo (None, 26, 26, 256)  1024
_________________________________________________________________
leaky_re_lu_16 (LeakyReLU)   (None, 26, 26, 256)  0
_________________________________________________________________
conv2d_17 (Conv2D)           (None, 23, 23, 128)  524288
_________________________________________________________________
batch_normalization_24 (BatchNo (None, 23, 23, 128)  512
_________________________________________________________________
leaky_re_lu_17 (LeakyReLU)   (None, 23, 23, 128)  0
_________________________________________________________________
conv2d_18 (Conv2D)           (None, 20, 20, 64)   131072
_________________________________________________________________
batch_normalization_25 (BatchNo (None, 20, 20, 64)   256
_________________________________________________________________
leaky_re_lu_18 (LeakyReLU)   (None, 20, 20, 64)   0
_________________________________________________________________
conv2d_19 (Conv2D)           (None, 17, 17, 32)   32768
_________________________________________________________________
batch_normalization_26 (BatchNo (None, 17, 17, 32)   128
_________________________________________________________________
leaky_re_lu_19 (LeakyReLU)   (None, 17, 17, 32)   0
```



```
_________________________________________________________________
conv2d_20 (Conv2D)           (None, 14, 14, 32)   16384     
_________________________________________________________________
batch_normalization_27 (BatchNo (None, 14, 14, 32)   128       
_________________________________________________________________
leaky_re_lu_20 (LeakyReLU)   (None, 14, 14, 32)   0         
_________________________________________________________________
conv2d_21 (Conv2D)           (None, 11, 11, 32)   16384     
_________________________________________________________________
batch_normalization_28 (BatchNo (None, 11, 11, 32)   128       
_________________________________________________________________
leaky_re_lu_21 (LeakyReLU)   (None, 11, 11, 32)   0         
_________________________________________________________________
conv2d_22 (Conv2D)           (None, 8, 8, 32)     16384     
_________________________________________________________________
batch_normalization_29 (BatchNo (None, 8, 8, 32)     128       
_________________________________________________________________
leaky_re_lu_22 (LeakyReLU)   (None, 8, 8, 32)     0         
_________________________________________________________________
conv2d_23 (Conv2D)           (None, 5, 5, 1)      513       
=================================================================
Total params: 5,599,489
Trainable params: 5,596,545
Non-trainable params: 2,944
```

DEEP LEARNING BASED PREDICTION OF PATH LOSS VALUES AND DISTRIBUTIONS FROM SATELLITE IMAGES

ORIGINALITY REPORT

9%	6%	5%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	eprints.utm.my Internet Source	1%
2	Submitted to Nashville State Community College Student Paper	1%
3	www.arehna.di.uoa.gr Internet Source	1%
4	Submitted to Eastern Mediterranean University Student Paper	<1%
5	Regev Cohen, Yi Zhang, Oren Solomon, Daniel Toberman, Liran Taieb, Ruud JG van Sloun, Yonina C. Eldar. "Deep Convolutional Robust PCA with Application to Ultrasound Imaging", ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019 Publication	<1%
6	arxiv.org Internet Source	<1%