TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

HIMALAYA COLLEGE OF ENGINEERING

A MAJOR PROJECT MID DEFENSE REPORT

ON

"STANDARD MAPS GENERATION FROM SATELLITE IMAGES USING CONDITIONAL GAN"

[CT 707]

<u>SUBMITTED TO:</u>

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Chyasal, Lalitpur

<u>SUBMITTED BY:</u>

Anjan Prasad Sharma (HCE074BCT005)

Anudit Adhikari (HCE074BCT006)

Arish Prasain (HCE074BCT008)

Shubhashish Basnet (HCE074BCT036)

November, 2021

ABSTRACT

Automatically generating maps from satellite images is a challenging and important task. It is a very time consuming task. Maps have business incentive to organizations in numerous divisions of the economy: ride-sharing companies like Tootle, food delivery companies like Foodmandu and numerous different areas of the economy. Furthermore, this technique can also be applicable to drone images for rural and semi-urban parts of the country.

As we know, for the map-related services and functionalities, Google Maps is already there but it is generally updated in 1-3 years. The satellite data on Google Maps is typically between 1 to 3 years old. Moreover, for countries like Nepal, map data are not available for many regions. Thus, we propose a Conditional Generative Adversarial Network (cGAN) model which compresses the images down to a learned embedding. Our model translates the satellite image to the corresponding standard layer map image using cGAN where the generator translates by means of a series of convolutions to the standard layer of a map and the discriminator input is the concatenation of the real/generated map and the satellite image. Since we have access to the real map for a given satellite image, we are able to assign a quantitative metric to the quality of the generated images. Moreover, using U-net Generator and PatchGAN Discriminator, and calculating the reconstruction loss, generator loss and discriminator loss will also provide further improved results.

ACKNOWLEDGEMENT

First of all, we would like to thank Himalaya College of Engineering and Tribhuvan University for providing us this opportunity to do this project from which we can gain a lot of skill in researching and on various other fields after completing this project. And we would sincerely like to express our gratitude to **Er. Ashok GM** sir, Head of Department, **Er. Narayan Adhikari Chettri** sir, project coordinator and **Er. Madhu Nyoupane** sir, project supervisor who are providing their invaluable guidance, comments, and suggestions throughout the project task.

Group Members

Anjan Prasad Sharma (HCE074BCT005)

Anudit Adhikari (HCE074BCT006)

Arish Prasain (HCE074BCT008)

Shubhashish Basnet (HCE074BCT036)

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1 Background

In today's modern society, the electronic map plays an important role in daily life such as travel, navigation, geographic information query, commercial purposes and other services. Creating accurate maps has been a major challenge for companies that want to sell smart devices such as mobile phones, watches, laptops, etc. since a factor for what makes a device smart is its ability to locate itself and inform its user via a human-readable interface. More importantly, maps have a huge humanitarian value. For e.g., in context of our country, generating an accurate map of hospitals in rural areas of the country can help to the timely delivery of resources such as medicines, funding, hospitals equipment, etc. to the people in these places. Maps can also be used in autonomous vehicles and self-driving cars like Tesla. An accurate map must reflect all changes on the ground in a timely manner.

However, there are still some blind spots in the coverage of electronic maps (such as some remote areas), which limits the service level of geographic information data for users and the guidance level for socioeconomic and political purposes. At the same time, the production of electronic maps generally requires vectorization of paper maps first and then involves complex graphic editing manually by industry standards, which consumes a lot of manpower and resources.

Up-to-date geospatial data is continuously collected from flyover imaging air-crafts or satellites. Currently, there is considerable latency between changes to geographic/road conditions on the ground and the publicly available human-readable maps. One way to reduce this latency is to automate the process of human-readable map generation from a satellite image of a given location at a specific zoom level and resolution. In this work, we emphasize the importance of human-readability of a map and aim to construct accurate human-readable maps directly from a satellite/aerial image of the location. The satellite/aerial image specifies the zoom level and resolution of the required map. If a generative model can be trained to directly convert remote sensing images into corresponding electronic maps, the production of electronic maps will be accurate and rapid, thereby further improving the service level for society.

1.2 Objectives

The general objective of this project is to develop an application:

a. To generate a standard layer of maps from satellite images using conditional-GAN.

2. LITERATURE REVIEW

Image-to-Image translation has been a recent development and area of research in the field of generative modelling. In the current modern society of internet and technology, there have been vast developments in various fields where almost everything is now computerized. Not only that, but they have been trained to think and act like humans and intelligence. In such sector, image to image translation is one of the important development of the current era.

In 2014, Ian Goodfellow and his colleagues from University of Montreal introduced Generative Adversarial Networks (GANs). It was a method of learning an underlying distribution of the data that allowed generating artificial objects that looked strikingly similar to those from real life [1]. Philip Isola and his team investigates conditional adversarial networks as a general-purpose solution to image-to-image translation problems. These networks not only learn the mapping from input image to output image, but also learn a loss function to train this mapping. This makes it possible to apply the same generic approach to problems that traditionally would require very different loss formulations [2]. Yingxue Pang and team state that image-to-image translation aims to transfer images from a source domain to a target domain while preserving the content representations [3].

Hajar Emami and team introduced the attention mechanism directly to the generative adversarial network (GAN) architecture and proposed a novel spatial attention GAN model (SPA-GAN) for image-to-image translation [4]. GANs have been also used to create spoof satellite images and spoof images of the ground truth conditioned on the satellite image of the location [4]. Conditional GANs have also been used to generate ground-level views of locations from overhead satellite images [6].

In this project, we use the conditional GAN architecture for our task of generating a map for a location given the satellite image of the location. We also develop code that provides a feasible method to generate new datasets from the publicly available Google Earth Engine and the Google Maps APIs.

GANs' successful ability to model high-dimensional data, handle missing data, and the capacity of GANs to provide multi-modal outputs or multiple plausible answers is the reason behind its popularity.

Perhaps the most compelling reason that GANs are widely studied, developed, and used is because of their success. GANs have been able to generate photos so realistic that humans are unable to tell that they are objects, scenes, and people that do not exist in real life.

Some of the main benefits of GAN over other algorithms are:

a. Convergence will be faster. Even the random distribution that fake images follow will have some patterns.
b. We can control output of the generator at test time by giving the label for the image we want to generate.

3. SYSTEM ANALYSIS AND FEASIBILITY STUDY

3.1 Feasibility analysis

3.1.1 Economic Feasibility

This project is economically feasible because it only required simple algorithms to build without any budget problems and is also economical to use as it can be used for free. As our project only requires a personal computer connected to the internet with no other sophisticated devices required, that too with the minimum human resources required, our project comes under the budget.

3.1.2 Technical Feasibility

This project is technically feasible. This project is planned to be developed by using Python programming by us along with some libraries like NumPy, Pandas, Keras, matplotlib. Furthermore for the datasets, it can be obtained from the Google Earth Engine and Google Maps API. We can also scrap map-data using open satellite maps if we cannot obtain API keys from Google.

3.1.3 Operational Feasibility

This system can be used by any person who has an access to the internet. It will be having a very user-friendly interface so that it becomes easier for users to use this application.

3.1.4 Social Feasibility

An application must be socially feasible and it should not harm any of the users using it. The acceptance of the society or the users is a must for a product to grow successfully. Our project does not tend to hurt or harm anyone, hence it is socially feasible.

3.2 REQUIREMENT ANALYSIS

3.2.1 Functional Requirement

These are the requirements that the end user specifically demands as basic facilities that the system should offer.

i. Users should be able to acquire maps of any place having satellite images.

ii. The system should be able to generate desired maps for every satellite image.

3.2.2 Non Functional Requirement

A non-functional requirement defines the quality attribute of a software system. These are basically the quality constraints that the system must satisfy for better performance and quality. They are also called non-behavioural requirements

i. The system should be easy to use and user friendly.

ii. The system should be efficient, and should deliver the result as soon as possible.

iii. The system should be very reliable in context of the results it shows, i.e. these must not be any fake information.

iv. The system is expected to be scalable as it can be used over large dataset.

v. Our system is supposed to be flexible in a sense that it can be further used in other scopes as well. It is not confined within the satellite images as the input.

4. SYSTEM DESIGN AND ARCHITECTURE

4.1 System block diagram

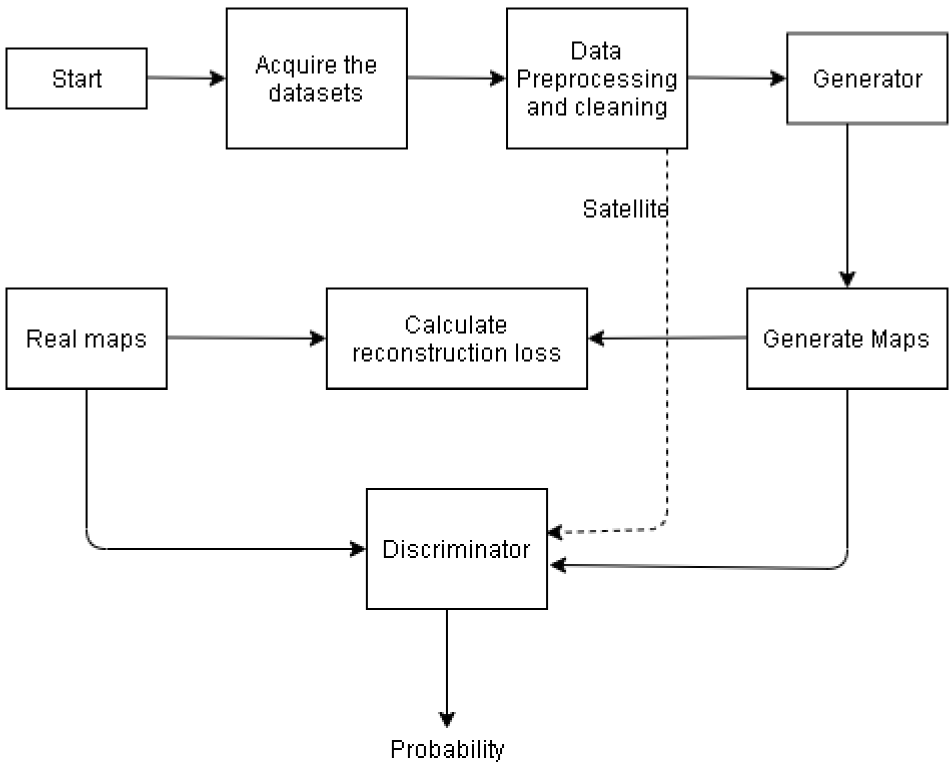

Figure 1 System Block diagram

4.2 Use case diagram

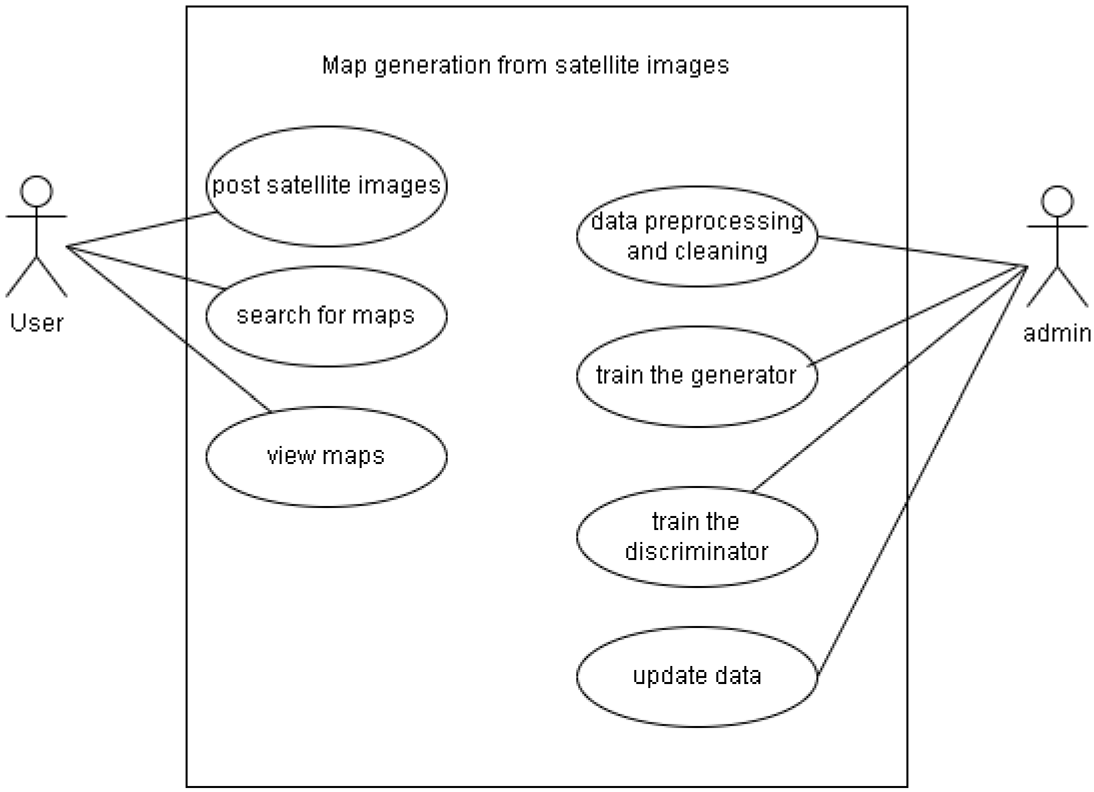

Figure 2 Use Case Diagram

4.3 DFD level 0

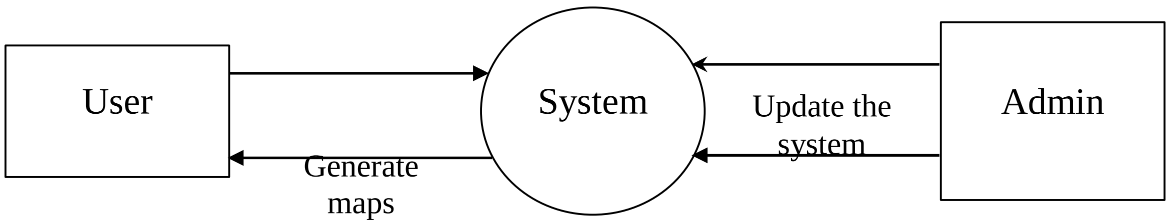

Figure 3 DFD Level 0

5. METHODOLOGY

5.1 Generative Adversarial Network (GAN)

The methodology of the GAN is that there are two networks — a Generator and a Discriminator. They play a game against each other. The objective of the Generator is to produce an object, say, a picture of a person that would look like a real one. The goal of the Discriminator is to be able to tell the difference between generated and real images.

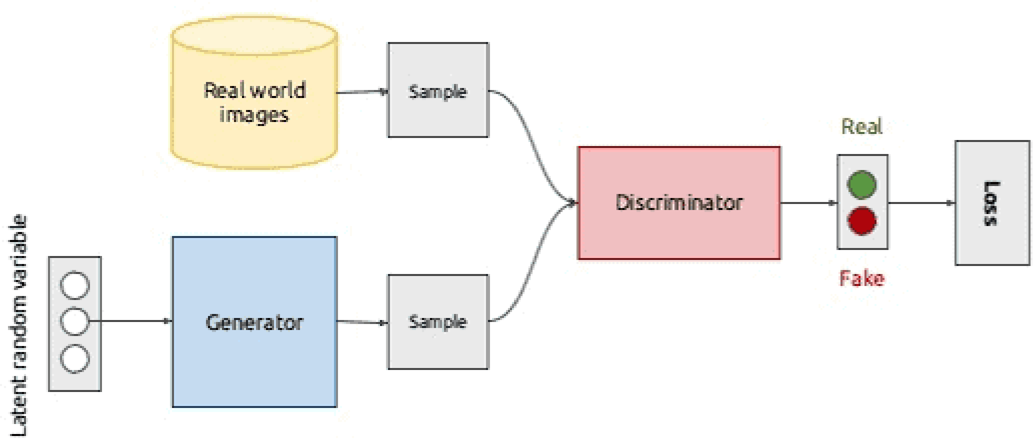

Figure 4 GAN Architecture

5.2 GANs and Convolution Neural Networks

GANs typically work with image data and use Convolutional Neural Networks, or CNNs, as the generator and discriminator models.

We have two models for GAN, generative model and discriminative model. The generative model has been trained on some data, let's say 'x', sampled from some true distribution, say D, is the one which given more random values, Z, produces a distribution D', which is close to D. Here we have trained the datasets on a certain probability distribution function. The generator generates random images, on the same probability distribution function, which is be close to the original dataset.

The discriminative model is the one which discriminates between two different classes of data. The main objective of this is to classify whether the generated image is fake or not fake.

Let us consider a set of dataset on the distribution D, which is D(x), where x is a single element of the dataset. Let $P_z(z)$ be the distribution of random variable where z is a single element of P.

The generative adversarial network works on the value of loss function, which is given by;

$$L(y',y) = [y \log y' + (1 - y) \log (1 - y')]$$

Here, y' is the reconstructed image from the generator and y is the original image from the trained dataset.

The label that is coming from $P_{data}(x)$ is y=1 and y'=D(x), so putting the values, we get;

$$L(D(x),1)= \log (D(x))$$

Also, for the data coming from the generator, the label y=0 and y'=D(G(z)), so in this case, we get;

$$L(D(G(z),0)= \log(1-D(G(z)))$$

We know that the objective of the discriminator is to classify the fake and the real datasets. For this, the above two equations should be maximized. For discriminator, the condition of maximization is D(G(z))=0, so we get, for discriminator;

$$\text{Discriminator}= \max [\log(D(x))+\log(1-D(G(z)))]$$

For the generator, the above equations should be minimized, and its condition is D(G(z))=1, so for generator, we get;

$$\text{Generator} = \min [\log(D(x))+\log(1-D(G(z)))]$$

Writing it in same equations, we get;

$$\min_{(G)} \max_{(D)} [\log(D(x))+\log(1-D(G(z)))]$$

The function we obtained from above is for a single instance of dataset x. if we have to consider all the instances, then

$$\min_{(G)} \max_{(D)} V(D,G)= E_{x\sim P(x)} [\log(D(x))]+ E_{z\sim Pz(z)} \log(1-D(G(z)))]$$

5.3 Conditional GAN

The generative model is trained to generate new examples from the input domain, where the input, the random vector from the latent space, is provided with (conditioned by) some additional input. The additional input could be a class value, such as male or female in the generation of photographs of people, or a digit, in the case of generating images of handwritten digits.

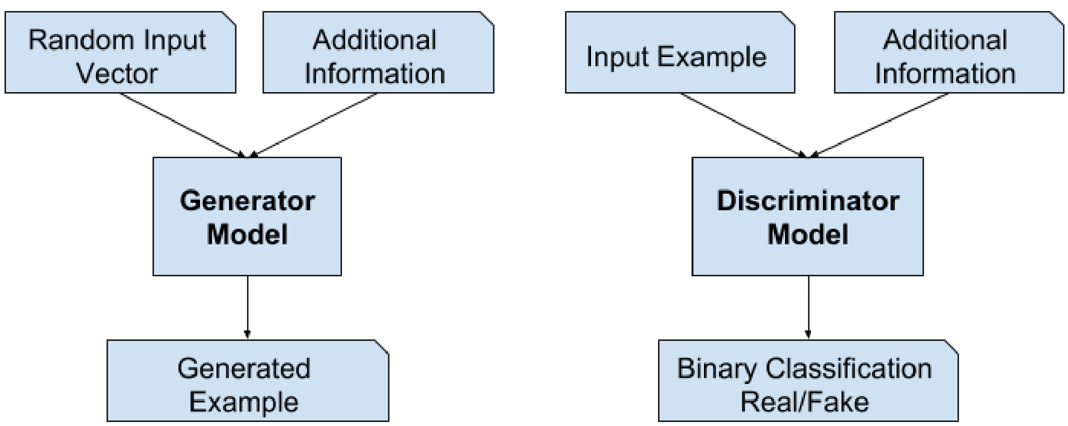

Figure 5 Conditional Generative Adversarial Network Model Architecture

The problem of simple GAN is that we do not have control over its output. So in order to solve that problem, conditional GAN is used. We simply add labels in the GANs as a condition.

From above expression, using label as y, we get, the expression of conditional GAN as;

$$\min_{(G)} \max_{(D)} V(D,G) = E_{x\sim P(x)} [\log(D(x|y))] + E_{z\sim Pz(z)} \log(1-D(G(z|y)))]$$

5.4 U-net Generator

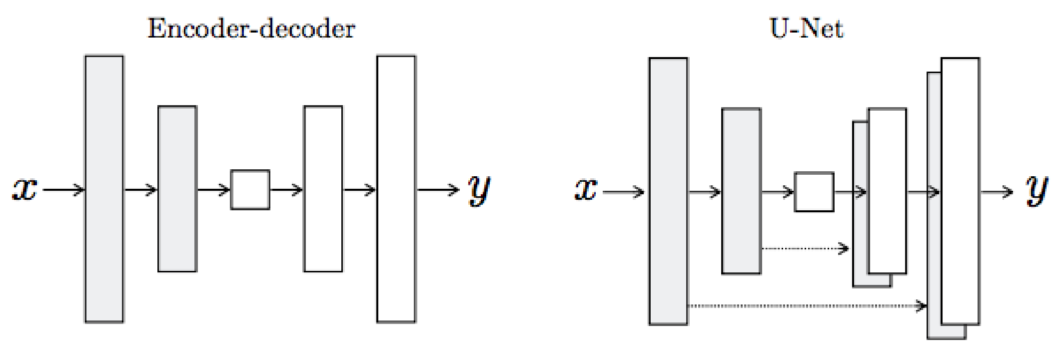

Figure 6 Encoder-decoder vs. U-Net Generator

The Generator takes in the Image to be translated and compresses it into a low-dimensional, “Bottleneck”, and vector representation. The Generator then learns how to up sample this into the output image. As illustrated in the image above, it is interesting to consider the differences between the standard Encoder-Decoder structure and the U-Net. The U-Net is similar to ResNets in the way that information from earlier layers are integrated into later layers. The U-Net skip connections are also interesting because they do not require any resizing, projections etc. since the spatial resolution of the layers being connected already match each other.

5.5 PatchGAN Discriminator

The PatchGAN discriminator works by classifying individual (N x N) patches in the image as "real vs. fake", opposed to classifying the entire image as "real vs. fake". This enforces more constraints that encourage sharp high-frequency detail. Additionally, the PatchGAN has fewer parameters and runs faster than classifying the entire image.

5.6 Algorithm for GAN

for number of training iterations **do**

for *k* steps **do**

- Sample minibatch of *m* noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from noise prior $p_g(z)$.

- Sample minibatch of *m* examples $\{x^{(1)}, \ldots, x^{(m)}\}$ from data generating distribution $p_{data}(x)$

- Update the discriminator by ascending its stochastic gradient :

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right]$$

end for

- Sample minibatch of m noise samples {z(1), … , z(m)} from noise prior pg(z).

- Update the generator by descending its stochastic gradient :

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$$

end for

5.7 Hyper-parameters

As suggested by the paper [2], the values of the hyper-parameters to train the model:

Batch size = 1

Input and output image size = 256 X 256

Learning rate = 0.0002

Momentum [$\beta 1$, $\beta 2$] = [0.5, 0.999]

λ_L1 = 100

6. WORK ANALYSIS

A. WORK COMPLETED

We have been working on our project efficiently. Until now, we have constructed our convolutional neural network for the generator and deconvolutional neural network for the discriminator, which corrects each other. We have trained our datasets on this neural network.

B. WORK REMAINING

We still have to train more datasets as much as we can for our output to appear more optimally. We still are optimizing the model for us to obtain the output for the threshold we want.

7. REFERENCES

[1] I. Goodfellow, “arxiv,” 10 June 2014. [Online]. Available: https://arxiv.org/abs/1406.2661. [Accessed 2021].

[2] J.-Y. Z. Phillip Isola, “arxiv,” 21 November 2016. [Online]. Available: https://arxiv.org/abs/1611.07004. [Accessed 2021].

[3] “arxiv,” 21 January 2021. [Online]. Available: https://arxiv.org/abs/2101.08629. [Accessed 2021].

[4] M. D. M. M. A. R. B. C. Hajar Emami, “SPA-GAN: Spatial Attention GAN for,” 2020.

[5] B. Z. Chunxue Xu, “Satellite Image Spoofing: Creating Remote,” Oregon, 2018.

[6] Y. Z. S. N. Xueqing Deng, “What Is It Like Down There? Generating Dense Ground-Level Views and Image Features From Overhead Imagery Using Conditional Generative Adversarial Networks,” California, 2018.