**“STANDARD MAP GENERATION FROM SATELLITE IMAGE USING CONDITIONAL GAN”**

**A Major Project Report**

**Submitted in the Partial Fulfillment**

**of**

**the Requirements for the Degree of**

**Bachelor In Engineering of Information Technology**

**at**

**Everest Engineering College**

**Sanepa, lalitpur**

**Affiliated to Pokhara University**

**By**

**Amit Joshi(20120004)**

**Sabin Upreti(20120046)**

**Rupesh Gelal(20120046)**

**Suman Thapa(20120058)**



**2024**

**DECLARATION**

We hereby declare the project work entitled “**STANDARD MAP GENERATION FROM SATELLITE IMAGE USING CONDITIONAL GAN”** submitted to the Department of Computer and Information Technology Engineering, Everest Engineering college, Sanepa, in the partial fulfillment of the requirement for the award of the Degree of Bachelor of Engineering in Information Technology. The result embodied in this project have not been submitted to any other University or Institute and we are the only author of this complete work.

Amit Joshi (2012004) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Rupesh Gelal (20120046) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Sabin Upreti (20120047) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Suman Thapa (20120058) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**COPYRIGHT**

The author has agreed that the library, **Everest Engineering College(EEC),** Sanepa, lalitpur may make this report freely available for the inspection. Moreover, the author has agreed that permission for extensive copying of this project report for scholarly purpose may be granted by the lectures, who supervised the project works recorded herein or, in the absence, by the Head of Department wherein the project report was done. It is understood that the recognition will be given to the author of the report and to the Department of Information Technology, EEC in any use of the material of this project report. Copying or Publication or other use of this report for financial gain without the permission to copy or to make any other use of the material in this report in whole or in part should be addressed to the Head of the Department, department of Computer and Information Technology engineering.

**Acknowledgement**

We would like to express our heartfelt gratitude to our mentor Er. Santosh Bist ,of Computer and Information technology department of Everest Engineering college for his guidance which aided us in every step for the completion of our project titled “**Standard Map Generation From Satellite Image Using Conditional GAN”.**

We would like to express our special thanks to our team members for their time an effort they provided throughout the year. Without the guidance of our mentor, hard work and dedication of our team members, it wouldn’t have been possible to complete the work in time.

We would like to acknowledge that this project was completed entirely by me and teammates.

**Abstract**

Generation of maps from satellite images is conventionally done by a range of tools. Maps became an important part of life whose conversion from satellite images may be a bit expensive but Generative models can pander to this challenge. These models aim at finding the patterns between the input and output image. Image to image translation was employed to convert satellite image to corresponding map. Technique for image-to-image translations like Conditional adversarial networks was used to generate the corresponding human-readable maps for that region, which took a satellite image at a given zoom level as its input. The model was trained on Conditional Generative Adversarial Network which comprises of U-Net Generator model which generates fake images while the PatchGAN discriminator tried to classify the image as real or fake and both these models were trained synchronously in adversarial manner where both tried to fool each other and resulted in enhancing the model performance. The generator loss and discriminator loss and hyper parameter tuning were used to enhance and optimize model performance and provided further improved generated maps from the model.

**Keywords:**

*conditional generative adversarial network, datasets, patchGAN discriminator, U-Net gen-erator, image to image translation, satellite images, SSIM, PSNR, L1 loss, Generator and discriminator loss.*

Contents

[**CHAPTER 1:Introduction** 10](#_Toc177204707)

[1.1 Background 10](#_Toc177204708)

[1.2 Motivation: 10](#_Toc177204709)

[1.3 Project Objective 11](#_Toc177204710)

[1.4 Project Applications and Scope 11](#_Toc177204711)

[**2** **CHAPTER 2: Literature Review** 13](#_Toc177204712)

[**3** **CHAPTER 3: Methodology** 15](#_Toc177204713)

[3.1 Theoretical Details: 15](#_Toc177204714)

[3.1.1 Machine Learning Processes 15](#_Toc177204715)

[3.1.2 Generative Adversarial Network (GAN)and Convolution Neural Networks 15](#_Toc177204716)

[3.1.3 Conditional Gan 17](#_Toc177204717)

[3.1.4 Network Architecture 19](#_Toc177204718)

[3.2 Evaluation Plan: 21](#_Toc177204719)

[3.2.1 Peak-Signal-To-Noise-Ratio(PSNR) 21](#_Toc177204720)

[3.2.2 Structural similarity Index Measure(SSIM): 22](#_Toc177204721)

[3.3 Implementation Details: 23](#_Toc177204722)

[**4** **Chapter 4: Result and Analysis** 24](#_Toc177204723)

[4.1 Results 24](#_Toc177204724)

[4.2 Analysis 25](#_Toc177204725)

[4.2.1 Generator loss in GANs 25](#_Toc177204726)

[4.2.2 Generator Loss in Conditional GANs 26](#_Toc177204727)

[4.2.3 Discriminator Loss in GANs 26](#_Toc177204728)

[4.2.4 Discriminator loss in cGAN: 27](#_Toc177204729)

[4.2.5 Validation losses 27](#_Toc177204730)

[4.2.6 Training loss 28](#_Toc177204731)

[4.2.7 L1 loss 30](#_Toc177204732)

[**5** **Chapter: Conclusion** 32](#_Toc177204733)

[**6** **References** 33](#_Toc177204734)

**List of Figures**

Figure 1:GAN architecture 15

Figure 2: Conditional Generative Adversarial Network Model Architecture 17

Figure 3: Encoder-decoder vs U-net Generator 18

Figure 4: U-net architecture 18

Figure 5:Dimension of output image 19

Figure 6: 5x5 image matrix and 3x3 filter matrix 19

Figure 7:Feature Map after Convolution 20

Figure 8:Hyperbolic Tangent Function 21

Figure 9:ReLU and Leaky ReLU 21

Figure 10:SSIM and PSNR 23

Figure 11:Index page 24

Figure 12: Result page 25

Figure 13: Output 26

Figure 14:Generator and Discriminator loss 28

Figure 15: Validation Loss 29

Figure 16: Training Loss 31

Figure 17: L1 Loss 32

**List of ABBREVATIONS**

|  |  |
| --- | --- |
| API | Application Programming Interface |
| cGAN | Conditional Generative Adversarial Network |
| CNN | Convolutional Neural Network |
| GAN | Generative Adversarial Network |
| MNIST | Modified National Institute of Standards and Technology |
| PSNR | Peak Signal to Noise Ratio |
| SSIM | Structural Similarity Index |

# **CHAPTER 1:INTRODUCTION**

## 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 de- vices 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. Maps also 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.

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 re- sources.

We explore GANs in the conditional setting. Just as GANs learn a generative model of data, conditional GANs (cGANs) learn a conditional generative model. This makes cGANs suitable for image-to-image translation tasks, where we condition on an input image and generate a corresponding output image. Also, a striking effect of conditional GANs is that they pro- duce sharp images.

Hence, we have used cGANs to generate standard maps from satellite images. Standard maps are maps of high quality, high resolution, shows the vegetation, roads, mountains, water bodies, manmade structures and are of accurate scale and proportions.

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. 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.

## Motivation:

Some remote areas or regions don’t have access to map data. The production of electronic maps generally requires vectorization of paper maps first and then involves complex graphic editing manually which consumes a lot of manpower and resources. Also, there is a need for up-to-date standard maps for Civil engineer or in studies related to real state.

So, our motivation for this project is to find a solution to these problems.

## Project Objective

The main objective of our project is:

* To develop a system that takes satellite image as an input and generate standard maps with vegetation, roads, water bodies, and manmade structures of accurate scale and proportions.
* To train the model to produce better results by hyper-parameter tuning taking in account the generator loss, discriminator loss, PSNR and SSIM.

## Project Applications and Scope

* **Automatic conversion:** The need of time consuming updates and labor-intensive human work is decreased when standard maps are automatically generated from satellite pictures.
* **Image-to-image translation:** Using the image-to-image translation capabilities of the Conditional GAN technique transforms satellite photos to equivalent standard layer map images.

# **CHAPTER 2: LITERATURE REVIEW**

Image-to-Image translation has been a recent development and area of research in the field of generative modeling. 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 developments of the current era.

In 2014, Ian Good fellow 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 investigate conditional adversarial networks as a general-purpose solution to image-to-image translation problems. These networks not only learn the map- ping 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].

Mehdi Mirza and team introduced the conditional version of generative adversarial net- works in their paper, which was constructed by simply feeding the data, they wished to condition on to both the generator and discriminator to show that their model could generate MNIST digits conditioned on class label [8]. Wallace Lira and team introduced GAN Hopper in their paper which is an unsupervised image-to-image translation network that transforms images gradually between two domains, through multiple hops [9].

Dragos Costea and team proposed the Dual-Hop GAN (DH-GAN) to detect roads and in- tersections and used that to find its best covering road graph by applying a smoothing- based graph optimization procedure [10].

Timo Carras and team proposed an alternative generator architecture for generative adversarial networks, borrowing from style transfer literature. The new architecture leads to an automatically learned, unsupervised separation of high-level attributes such as head pose and freckles when trained on human faces. A new, highly varied and high-quality dataset of human faces were introduced with this paper [11].

Han Zhang and team introduced Stack GAN, which are used to generate 256×256 photo- realistic images conditioned on text descriptions. The hard problem is decomposed into more manageable sub-problems through a sketch-refinement process [12].

We use the conditional GAN architecture for our task of generating a map for a location given the satellite image of the location.

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:

• Convergence will be faster. Even the random distribution that fake images follow will have some patterns.

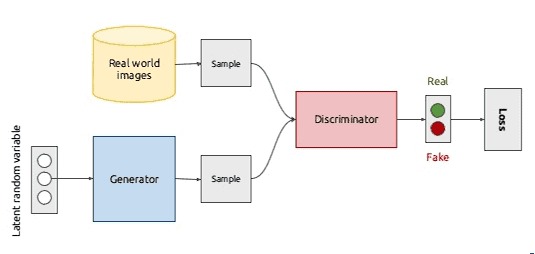
• We can control output of the generator at test time by giving the label for the image we want to generate.

# **CHAPTER 3: METHODOLOGY**

## Theoretical Details:

### Generative Adversarial Network (GAN) and Convolution Neural Networks

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 real one. The goal of the Discriminator is to be able to tell the difference between generated and real images.



**Figure 3.1.1‑2:GAN architecture**

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 Pz(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’)] (5.1)

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 Pdata(x) is y=1 and y’=D(x), so putting the values, we get;

L(D(x),1) = log(D(x)) (5.2)

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))) (5.3)

We also 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 get, for discriminator;

Discriminator = max[log(D(x)) + log(1-D(G(z)))] (5.4)

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

Generator=min[log(D(x)) + log(1-D(G(z)))] (5.5)

Writing it in same equations, we get;

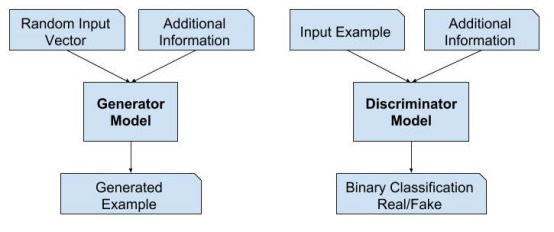
min(G)max(D)[log(D(x)+log(1-D(G(z)))] (5.6)

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) = Ex~P(x)[log(D(x))+ Ez~P(z)log(1-D(G(z)))] (5.7)

### 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.



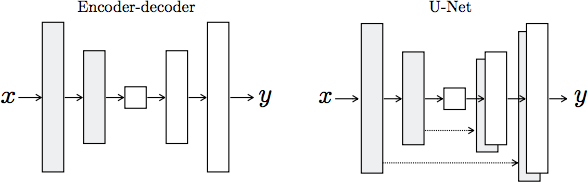
**Figure 3.1.2‑1: Conditional Generative Adversarial Network Model Architecture**

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;

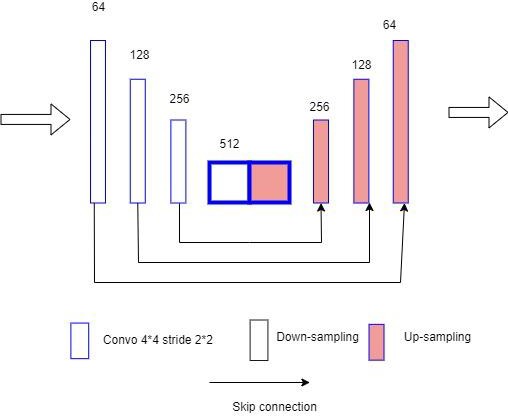
min(G)max(D) V(D,G) = Ex~P(x)[log(D(x|y))]+Ez~Pz(z)log(1-D(G(z|y)))] (5.8)

#### U-net Generator



**Figure 3.1.2‑2: 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 Res- Nets 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.



**Figure 3.1.2‑3: U-net architecture**

#### 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.

#### Algorithm for GAN

**for** number of training iterations **dofor** steps **do**

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

Sample minibatch of *m* examples *{x (1), …, x(m)}* from data generating distribution *pdata(x)*

Update the discriminator by ascending its stochastic gradient:

for

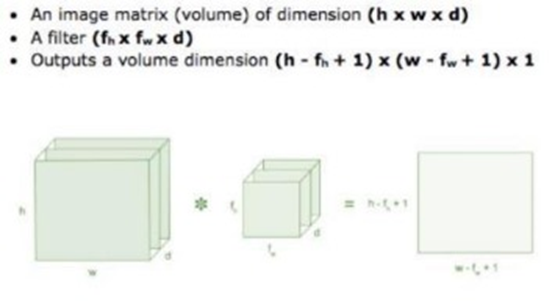
Sample minibatch of m noise samples *{z (1), …, z(m)} from noise prior pg(z).*

Update the generator by descending its stochastic gradient:

for

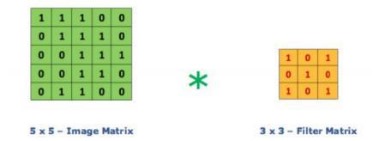
### Network Architecture

Both generator and discriminator use modules of the form Convolution –BatchNorm – ReLu. Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter.



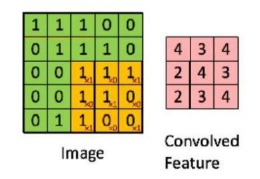
**Figure 3.1.3‑1:Dimension of output image**

Consider a 5x5 image matrix whose image pixel values are 0 and 1 , and 3x3 filter matrix as shown in the figure below:



**Figure 3.1.3‑2: 5x5 image matrix and 3x3 filter matrix**

Then the convolution of 5x5 Image matrix multiplied with 3x3 filter matrix which is called a feature map as output shown below:



**Figure 3.1.3‑3:Feature Map after Convolution**

We normalize the obtained layer by adjusting and scaling the activations. For example, when we have features from 0 to 1 and some from 1 to 1000, we normalize them to speed up learning. Batch normalization reduces the amount by what the hidden unit values shift around also known as covariance shift.

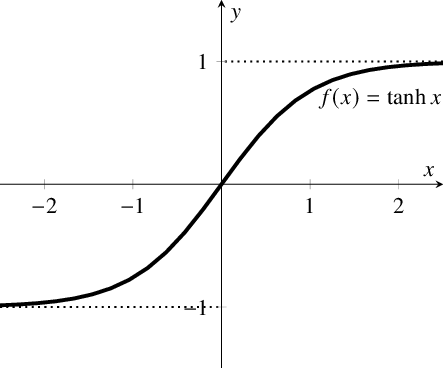
Batch normalization also reduces overfitting because it has a slight regularization effect. Similar to dropout, it adds some noise to each hidden layer’s activations. Therefore, by using batch normalization, we will need less dropout, which is a good thing because we are not going to lose a lot of information.

However, we should not depend only on batch normalization for regularization; it is always better to use it together with dropout.

Several types of nonlinear functions are used within the model to introduce non linearity in the outputs. Some such nonlinear activation functions used in the models are:

#### Hyperbolic Tangent Function-tanh:

Its mathematical formula is *f(x)=(1-exp(- 2x))/ (1- exp(-2x)).* Now its output is zero centered because its range is between -1 to +1 i.e. -1 < output < +1.

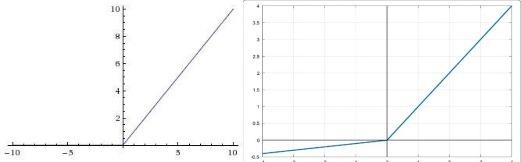


**Figure 3.1.3‑4:Hyperbolic Tangent Function**

#### ReLU, Rectified Linear Units:

Just *R(x) = max (0,x) i.e. if x < 0 , R(x) = 0* and if x>=0 then R(x)=x. Hence, as seeing the mathematical form of this function we can see that it is very simple and efficient almost all deep learning Models use ReLu nowadays. But its limitation is that it should only be used within hidden layers of a Neural Network Model.

Hence for output layers we should use a Softmax function for a Classification problem to com- pute the probabilities for the classes.

Another problem we see in ReLU is the Dying ReLU problem where some ReLU neurons essentially die for all inputs and remain inactive no matter what input is supplied, here no gradient flows and if large number of dead neurons are there in a neural network its performance is affected, this can be corrected by making use of what is called Leaky ReLU where slope is changed left of x=0 and thus causing a leak and extending the range of ReLU.

**Figure 3.1.3‑5:ReLU and Leaky ReLU**

## Evaluation Plan:

### Peak-Signal-To-Noise-Ratio(PSNR)

PSNR is derived from the mean squared error(MSE), which is a measure of a average squared differences between the original and the distorted images. PSNR represents the ratio between the maximum possible value of the pixel intensity and the power of the distortion (i.e, noise) that degrades the quality of the image.

The PSNR between two images *I1* and *I2* is given by:

Where,

* *MAXI* is the maximum possible pixel value of the image. For an 8-bit grayscale image, *MAXI* = 255, and for images normalized between 0 and 1, *MAXI* =1.
* *MSE*(Mean squared Error) is calculated as:

Where *m* and *n* are the dimensions of the image, and *I1*(*i,j)* and *I2*(*i,j)* are the pixel values of the original and distorted images at position(*i, j*).

### Structural similarity Index Measure(SSIM):

It is a method for predicting the perceived quality of digital television and cinematic pictures, as well as other kinds of digital images and videos. It is also used for measuring the similarity between two images. SSIM is designed to measure the similarity between two images based on the perception of structural information, which refers to the spatial relationship between pixels rather than their absolute intensities. Human vision is more sensitive to structural information that to individual pixels values, and SSIM takes this into account.

The SSIM between two images *I1* and *I2* is typically computed over small windows (usually 11x11) of the images. For a given window, the SSIM is calculated as :

Where,

* are the mean intensities (luminance) of the images *I1* and *I2*, respectively.
* and are the variances (contrast) of the images.
* is the covariance between *I1* and *I2* (measuring how the structure in the two images correspond).
* and are small constants too stabilize the division when the denominator is close to zero. These constants are based on the dynamic range of pixels values and typically defined as:

*C1* =*(K1 .L)2, C2=(K2 . L)2*

Where *L* is the dynamic range of pixel values (e.g, 255 for 8-bit images), and *K1* and *K2* are small constants.

Components of SSIM

* Luminance (*l)*: Measures the similarity of the average brightness between two images.
* Contrast (*c*): Compares the contrast (Variance) between the two images
* Structure(*s*): Measures the correlation between the two images structures by examining their co variance.

The overall SSIM is then given as the product of these three components:

*SSIM(I1, I2) = l(I1, I2) . c(I1, I2) . s(I1, I2)*

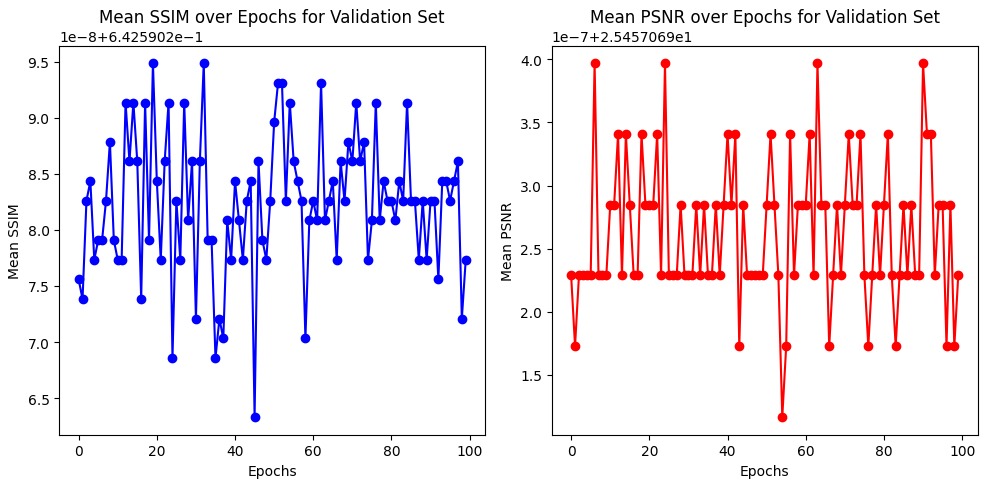


Figure 3.2.2‑1:SSIM and PSNR

## Implementation Details:

We have used Google Collab to train our algorithm. Google Colab is an as-a-service version of Jupyter Notebook that enables you to write and execute Python code through your browser.

Colab notebooks are stored in a Google Drive account and can be shared with others users, similar to other Google Drive files. The Notebooks also include an autosave feature, but they do not support simultaneous editing, so collaboration must be serial rather than parallel.

Google Colab eliminates the need for complex configuration setup and installation, as it runs right in the browser. It also includes pre-installation Python Libraries that requires no setup to use.

To train the cGAN model, we'll need a dataset of paired satellite images and corresponding map features. This dataset should be large enough to capture the variability in satellite images and map features..

Then, We've used A100 , L4, T4, TPU v2-8 GPU to train model. It falls under google colab pro( which is paid version).

Dataset details:

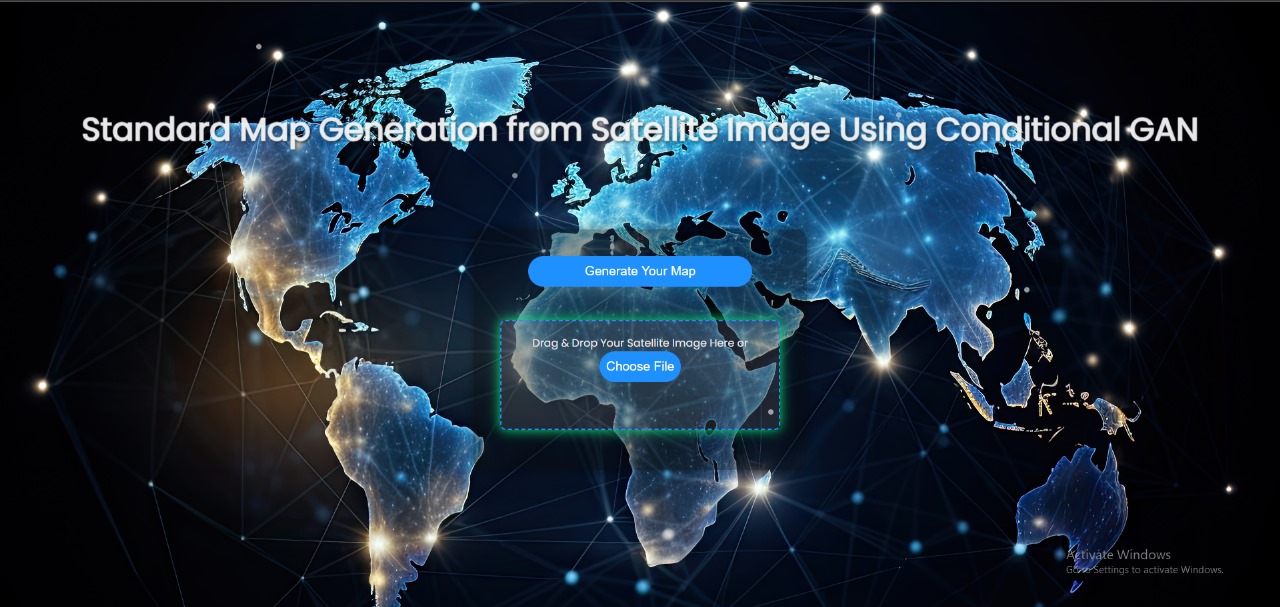
* Kaggle(limited 1098 images—Satellite and Mask images side by side in combined fashion)
* Rotation augmented used (in angle 90, 180, 270)
* Image size:(600,1200,3)

Training Details:

* Mini-Batch = 32
* epoch = 100
* Kernel size = 4
* Strides = 2 \* 2
* LeakyReLu, alpha(slope) = 0.2
* Weight initialized from normal distribution with mean = 0 and stddev = 0.02
* Adam optimizer with hyperparameter Tuning:
* Found learning rate 0.0002 to work good
* Momentum term (beta) = 0.5

# **Chapter 4: Result and Analysis**

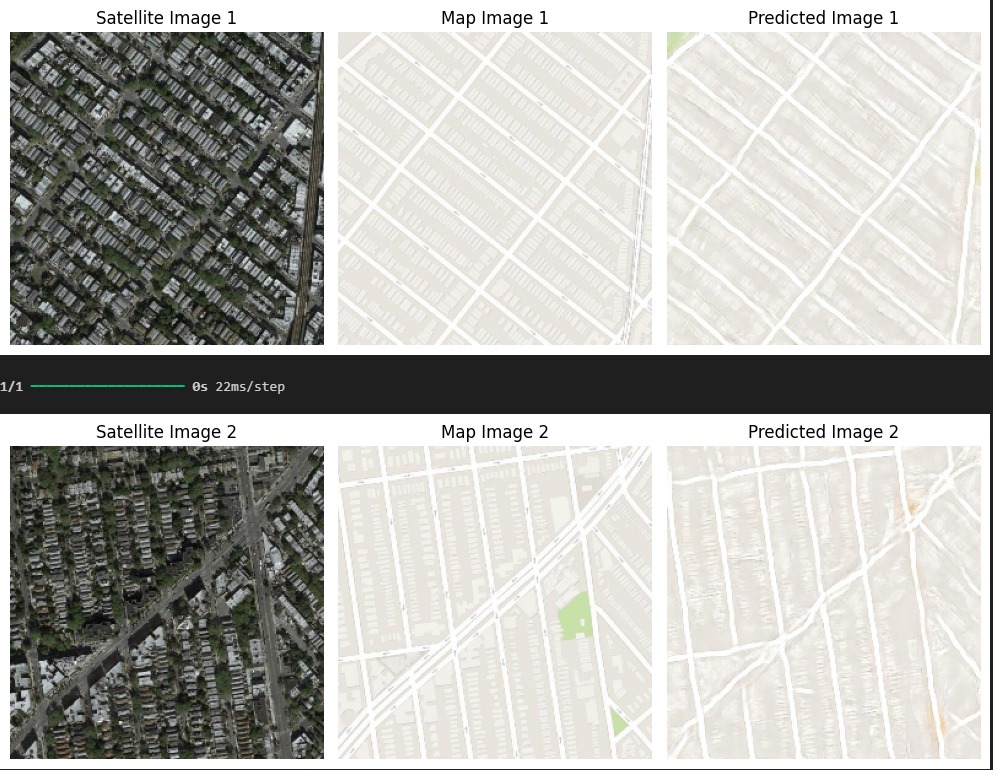
## Results



**Figure 3.2.2‑1:Index page**



**Figure 3.2.2‑2: Result page**



**Figure 3.2.2‑3: Output**

## Analysis

While training the cGAN model, we define a loss function and an optimizer. The Loss function should be an combination of the generator loss and the discriminator loss. The optimizer should be a variant of stochastic gradient descent.

### Generator Loss in Conditional GANs

For Conditional GANs, the generator loss is adjusted to account for the conditioning information *y* (such as class labels or auxiliary data). The loss function is:

Where:

* *y* represents the conditioning information, sampled from the distribution *py*(*y*).
* is the generator function that produces data conditioned on both the latent vector *z* and the conditioning information *y*.
* ) is the discriminator's probability that the generated data is real given the conditioning information *y*.

### Discriminator loss in cGAN:

In a conditional GAN, both the generator and discriminator receive additional information (a condition) along with the input. This condition could be any auxiliary information, like class labels in a labeled dataset.

In a cGAN, the discriminator *D* also takes the condition *y* (e.g., a class label) as input. The loss function is:

Where,

* represents the expected log probability that the discriminator correctly classifies a real sample *x* (conditioned on *y*) as real.
* represents the expected log probability that the discriminator correctly classifies a generated sample (conditioned on *y*) as fake.

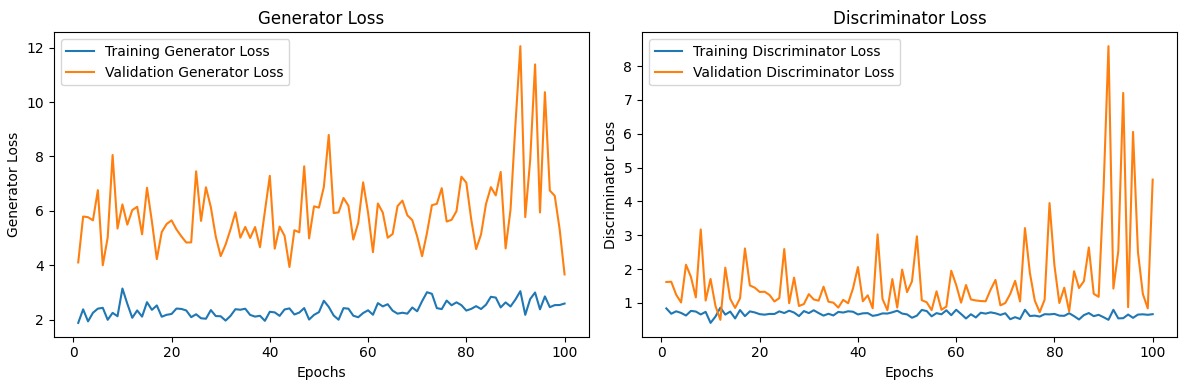
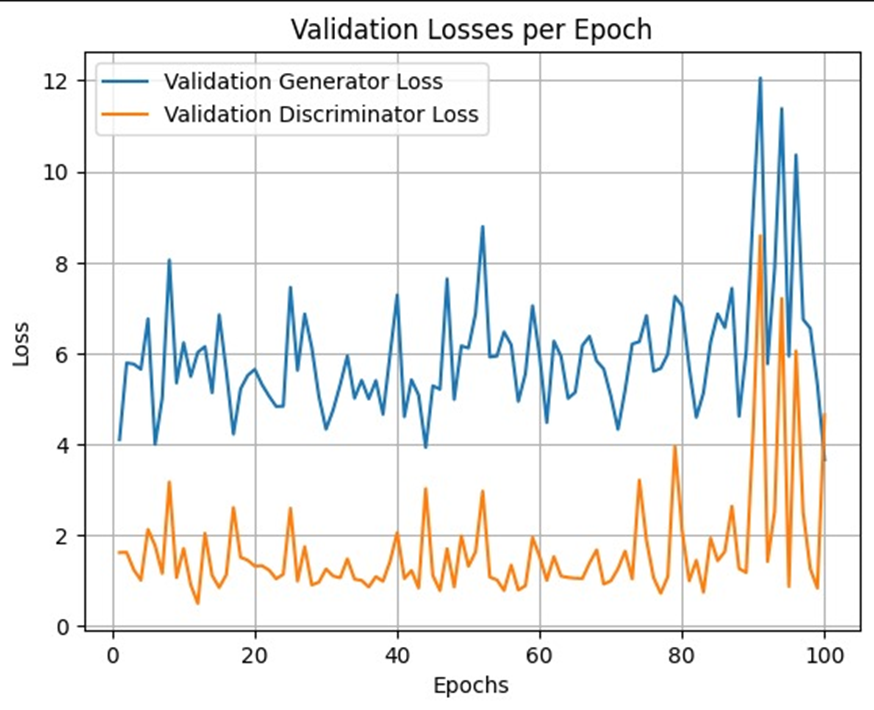


Figure 4.2.4‑1:Generator and Discriminator loss

### Validation losses



**Figure 4.2.5‑1: Validation Loss**

### Training loss

Training loss refers to the measure of how well a model is performing on the training dataset during the training process. It provides an indication of how well the model's predictions match the actual labels or target values in the training data. Training loss is crucial for understanding the learning progress of the model and for diagnosing issues such as overfitting or underfitting.

#### In GANs (Generative Adversarial Networks)

In GANs, there are two separate models: the generator and the discriminator. Each has its own loss function:

* Generator Loss: The generator's goal is to produce data that is indistinguishable from real data. Its loss function is often the negative log probability of the discriminator being fooled.
* Discriminator Loss: The discriminator's goal is to correctly classify real and fake data. Its loss function is:

#### In cGANs (Conditional GANs)

Conditional Gans extends the GAN framework by conditioning on additional information, such as class labels.

* Generator Loss:
* Discriminator loss:

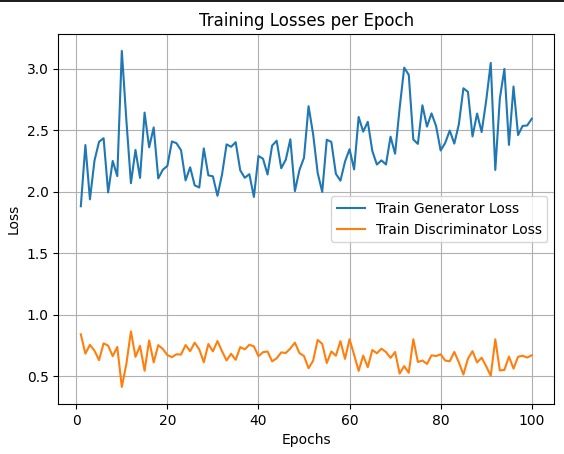


Figure 4.2.6‑1: Training Loss

### L1 loss

Penalizes the generator based on the pixel-wise difference between the generated image and the ground truth image. This encourages the generator to produce images that are not only realistic but also close to the actual target image.

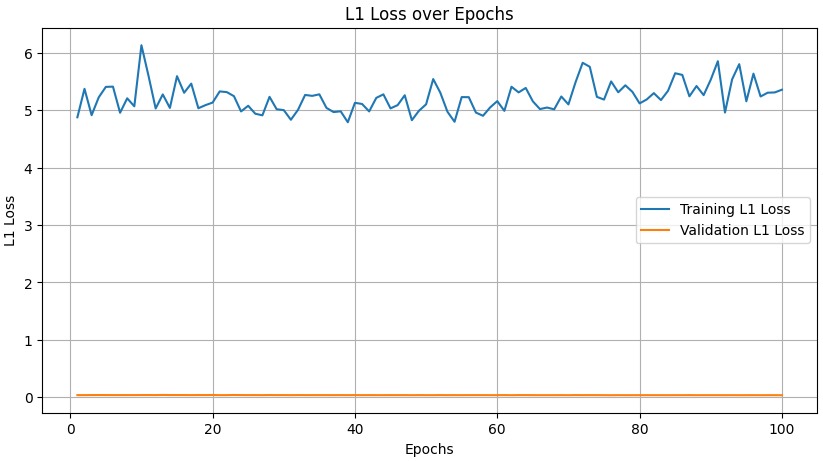


Figure 4.2.7‑1: L1 Loss

# **Chapter: Conclusion**

The translation of satellite photos into equivalent standard layer maps has shown encouraging results in the study on standard map production from satellite images using Conditional GAN. With this method, the mapping between the two domains is learned using the power of Conditional GANs, producing visually appealing maps that closely resemble the original satellite photos.

In this project, there are a number of benefits to using Conditional GANs. First off, Conditional GANs may produce excellent maps with visuals that closely resemble the original satellite photos. In order to do this, the model is trained on the input satellite photos, which enables it to discover the fundamental structure of the maps. Furthermore, Conditional GANs are capable of learning how the two domains are mapped, producing maps that match the input satellite photos.

To sum up, the research study utilizing Conditional GAN to generate standard maps from satellite pictures has shown encouraging outcomes. The method has demonstrated its applicability in a number of scenarios and may be extended to incorporate deep learning architectures and more kinds of satellite photos. Creating excellent maps that aesthetically resemble the input satellite photos is one of the many benefits of using conditional neural networks (GANs). Overall, the experiment has produced encouraging results and has illustrated the potential of Conditional GANs for standard map production from satellite imagery.

# **References**

1. I.Goodfellow, “arxiv,”10June2014. [Online].

Available: https://arxiv.org/abs/1406.2661. [Accessed 2023].

1. J.-Y. Z. Phillip Isola, “arxiv,” 21 November 2016. [Online].

Available: https://arxiv.org/abs/1611.07004. [Accessed 2023] [3] “arxiv,” 21 January 2021. [Online].

Available: https://arxiv.org/abs/2101.08629. [Accessed 2023]

1. M. D. M. M. A. R. B. C. Hajar Emami, “SPA-GAN: Spatial Attention GAN for,” 2020.
2. 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 GroundLevel Views and Image Features From Overhead Imagery Using Conditional Generative Adversarial Networks,” California, 2018.
3. M. Mirza, “Conditional Generative Adversarial Nets,” Montreal, 2014
4. W. Lira, “GANHopper: Multi-Hop GAN for Unsupervised Image-to- Image Translation,” 2020.
5. D. Costea and A. Marcu, “Creating Roadmaps in Aerial Images with GAN,” 2017.
6. T. Carras and S. Laine, “A Style-Based Generator Architecture for Generative Adversarial Networks,” 2019
7. H. Zhang and T. Xu, “StackGAN: Text to Photo-realistic Image Synthesis,” Hong Kong, 2017.
8. https://www.arxiv-vanity.com/papers/1611.07004/
9. https://www.kaggle.com/code/mastersniffer/pix2pix-maps-on-multiple-gpu- s/notebook
10. https://docslib.org/doc/11787646/arxiv-1611-07004v3-cs-cv-26-nov-2018- would-require-very-different-loss-formulations