

**CS6611 CREATIVE AND INNOVATIVE PROJECT**

**SATELLITE IMAGE DOMAIN  
TRANSLATION USING GENERATIVE  
ADVERSARIAL NETWORKS**

**Final Report**

**Gokul.S 2018103026  
Srihari.S 2018103601**

## INDEX

CHAPTER NO	TOPIC	PAGE NUMBER
1	<i>Introduction</i> 1.1 Motivation 1.2 Problem Scope 1.3 Problem Statement 1.4 Contribution 1.5 Overview	3
2	<i>Related works</i>	5
3	<i>System design</i> 3.1 System Use-case Diagram 3.2 Overall Block Diagram 3.3 Detailed Module-wise Description	8
4	<i>Experimental Results</i> 4.1 System Development 4.2 Dataset Description 4.3 Working of the System	15
5	<i>Result analysis</i> 5.1 Final System Testcase Table 5.2 Evaluation Metrics 5.3 Epoch-wise Evaluation Metric Values 5.4 Graphical Analysis of Observed Evaluation metric values	20
6	<i>Conclusion</i> 6.1 Future Scope 6.2 Conclusion	28

# SATELLITE IMAGE DOMAIN TRANSLATION USING GENERATIVE ADVERSARIAL NETWORKS

## *Abstract*

The Pix2Pix model which we are going to devise is a type of conditional GAN, where the generation of the output image is conditional on an input, in this case, a source image. The discriminator is provided both with a source image and the target image and must determine whether the target is a plausible transformation of the source image. We attempt to introduce a new strategy of training the discriminator network by using a different approach. The input source image and the input target image are concatenated and then given as input to the first convolutional layer of the discriminator network whose output is then split to retrieve the generated sample. This  $(256*256*3)$  is then compared with the ground truth image and the final prediction  $(16*16*1 \rightarrow \{0,1\})$  is made. Because the generator is being via adversarial loss, which encourages the generator to generate plausible images in the target domain, the training time substantially increases. We handle this enormous train time by using batch training and batch normalization on a Graphics Processing Unit (GPU) backend. This encourages the generator model to create plausible translations of the source image. In particular, we attempt to translate images from their satellite form to their aerial map form. In order to achieve this, we will make use of a compressed NPZ Maps dataset. Nowadays, Pix2Pix GANs are further being applied on a range of image-to-image translation tasks such as converting black and white photographs to colour, sketches of products to product photographs, etc. We evaluate the performance of our network using Fréchet-Inception Distance, Manhattan Norm, Mean - Square Error and Structural Similarity in order to check the quality and credibility of the generated image.

## Chapter 1: INTRODUCTION

### **1.1 Motivation**

Many image processing and computer vision tasks, such as image segmentation, stylization, and abstraction, can be posed as image-to-image translation problems, which convert one visual representation of an object or scene into another as well as other interesting applications like face morphing [1]. Conventionally, these tasks have been tackled separately due to their intrinsic disparities. The primary focus has been to tackle the domain of geophysical surface exploration, especially in the field of celestial structure detection along with edifice observation for the defence forces and intelligence bureaus. The objective is infeasible and impractical on developing an object detection model that works on satellite images. In order to come around this, we use a variant of

the conditional adversarial network, called the Pix2Pix GAN to resolve this issue by generating the aerial map version of the input image. Further, on applying the Ramer-Douglas-Peucker's Contour Approximation Algorithm [2] on the artificially generated image, the structure of the surface that is being worked upon would be predicted more accurately unlike the existing system which uses a basic Conditional Generative Adversarial network. The user gives the image returned by a radar satellite and it gets converted into its corresponding aerial map version. The model uses image embeddings and synthesizes an image in correspondence with the input. Some models learn to achieve the same task using 3 novel losses [3]. We will further demonstrate the ability of the model to synthesize a plausible detection of the edifices present in the image irrespective of the part of the day the image was shot. This idea has enormous implications in military applications, fighting wildfires, space exploration, etc. In future work, we aim to further scale up the model to higher resolution images and implement its reverse translation as well.

## ***1.2 Problem Scope***

Image-to-image translation is to learn a mapping between images from a source domain and images from a target domain. After much studies, it has been found that more than about 98% of the land, environment and mineral composition/structure of the universe still remains completely unexplored to date. This is still an area where the rate of growth in our existing technology remains dull. This problem is being attempted to be solved from the perspective of image domain translation since intricate details and information would start to become evident along the way. For applications in which data collection seem expensive data augmentation could be the way to go about [4]. In order to achieve this objective efficiently, we make use of the state-of-the-art Generative Adversarial Networks framework, in particular, the Pix2Pix GAN. This network also comes in handy for intelligence services and military forces to secret detect trenches and attack spots. Recently icon-colorization has also become popular which can be efficiently solved using triple conditional generative adversarial networks [5].

## ***1.3 Problem Statement***

Given an input satellite image from a military satellite, the model generates a plausible aerial map version of the image using the latent sample space that it has learned during the training. This is further given as input to the edifice detection module which detects the hidden trenches and edifices present in the image. An interesting methodology could be to utilize the adversarial loss to match the distributions of the translated and the target image sets [6]. This can be passed on to the

terrestrial military forces by the secret services to facilitate efficient attacks. We could extend these networks to work on a higher dimension as well [7].

## ***1.4 Contribution***

In this paper, the primary contributions are the use of a channel-wise concatenation strategy for feeding images to the discriminator to distinguish real images from its perfectly plausible transformations in the aerial map domain. This has ensured the discriminator network to learn to map pixel intensity values from the corresponding channel and compare them rigorously to come to a conclusion. The other extremely vital module that we've supplemented to the architecture is the addition of a smart and accurate edifice detection module that detects the edifices present in the generated image by using appropriate contour approximation algorithms. This can be of monumental use to industry experts from multiple domains like Intelligence Bureau, Secret Services, International Space Agencies, Space Exploration Teams, etc.

## ***1.5 Overview***

Chapter 2 conducts extensive analysis on the challenging generative and discriminative tasks with six recent independent works, demonstrating the superiority and wide application scope of our GAN framework. Chapter 3 covers the refined overall system design along with module-wise design and their corresponding implementation methodology, pseudocode and block diagrams. Chapter 4 discusses the experimental results by summarizing the overall system performance with respect to four evaluation metrics attacking each and every qualitative aspect of the generated output image. It also includes a detailed workflow of the system with some execution snapshots. Chapter 5 analyses the experimental results by presenting the final system test case table and performs a graphical analysis of the specified evaluation metrics which amounts to an exhaustive ablation study to evaluate each component of the proposed methodology. The last chapter formally concludes the overall argument, gives few insights regarding the future scope and mentions the references to the aforementioned significant works of contemporaries.

## **Chapter 2: RELATED WORKS**

[8] Proposed an image-to-image translation model by combining GAN and latent space learning to arrive at Consistent Embedded Generative Adversarial Networks (CEGAN) generating both realistic and diverse images. It captures the full distribution of potential multiple modes of results by enforcing tight connections in both the real image space and latent space. In order to improve the quality of generated images, variant GANs have been proposed, with better training objectives,

combination with auto-encoders and multi-stage generation. To achieve realism, unlike existing GANs models that have their discriminators attempt to differentiate between real images from the dataset and fake samples produced by the generator, the discriminator in this model distinguishes the real images and fake images in the latent space to alleviate the impact of the redundancy and noise in generated images. The ability of the model to produce more diverse and realistic results is strengthened by learning a low-dimensional latent code due to the multiple distribution in the latent space. However, all above methods have focused on generating a single result conditioned on the input and these techniques usually assume a deterministic or unimodal mapping. As a result, they fail to capture the full distribution of possible outputs. Even if the model is made stochastic by injecting noise, the network usually learns to ignore it. Their evaluation based on AMT – Amazon Mechanical Turk Perpetual Study is subjective to human errors.

[9] Adopted the self-attention network thus producing better results than the convolution-based GAN. Verified the effectiveness of the self-attention network in unsupervised image-to-image translation tasks. Accomplishes the transformation from a source domain to a target domain given unpaired training data. The network architecture is devised by combining several self-attention blocks into the generator and discriminator of the Multimodal Unsupervised Image-to-Image Translation (MUNIT) model. Long range dependency helps to not only capture strong geometric change but also generate details using cues from all feature locations. Combined with self-attention, the generator can translate images in which fine details at every position are carefully coordinated with fine details in distant portions of the image. Furthermore, the discriminator can also more accurately enforce complicated geometric constraints on the global image structure. Has overcome the limitations of CycleGAN, the most representative unsupervised image translation method, by changing the high-level semantic meaning. It fails to capture strong geometric changes between domains, or it produces unsatisfactory results for complex scenes, compared to local texture mapping tasks such as style transfer.

[10] Incorporated the attention mechanism into image-to-image translations which helps the generative network to attend to the regions of interest and produce more realistic images. The SPA-GAN model achieved this by explicitly transferring the knowledge from the discriminator to the generator to force it focus on the discriminative areas of the source and the target domains. SPA-GAN has feedback connections as compared to the CycleGAN model with no feedback attention. Both frameworks learn two inverse mappings through one generator and one discriminator in each domain. In SPA-GAN, the discriminator network is deployed to highlight the most discriminative regions between real and fake images in addition to the classification. These discriminative regions

illustrate the areas where the discriminator focusses on in order to correctly classify the input image. However, in SPA-GAN the discriminator generated the attention maps in addition to classifying its input as real or fake. These attention maps were looped back to the input of the generator. While CycleGAN is trained using the adversarial and cycle consistency losses, SPA-GAN integrates the adversarial, modified cycle consistency and feature map losses to generate more realistic outputs. However, most data is inherently non – IID and most assumptions of this model are too simplistic (E.g.: Euclidean Distance). It also struggles with modelling spatial complexities (E.g.: Interplay of short versus long-distance dependencies).

[11] Uses the loss format advocated by Wasserstein GAN rather than the sigmoid cross-entropy loss used in the original GAN. It is proven that the former performs better in terms of generator convergence and sample quality, as well as in improving the stability of the optimization. L1 distance is adopted to measure the recovery error, which is added to the GAN objective to force the translated samples to obey the domain distribution. DualGAN is constructed with identical network architecture for  $G_A$  and  $G_B$ . The generator is configured with equal number of downsampling (pooling) and upsampling layers. In addition, the generator is configured with skip connections between mirrored downsampling and upsampling layers, making it a U-shaped net. Such a design enables low-level information to be shared between input and output, which is beneficial since many image translation problems implicitly assume alignment between image structures in the input and output (e.g., object shapes, textures, clutter, etc.). Without the skip layers, information from all levels has to pass through the bottleneck, typically causing significant loss of high-frequency information. However, DualGAN uses an unsupervised learning approach for image-to-image translation based on unpaired data, but with different loss functions. Figuring the right combination of these loss functions is a tedious task and wastes a huge amount of time which otherwise could've been used to train the actual GAN model itself.

[12] For some specific image-to-image translation tasks such as hand gesture-to-gesture translation and person image generation tasks, the image domains could be arbitrarily large. Eg: hand gestures and human bodies in the wild can have arbitrary poses, sizes, appearances, structures, locations, and self-occlusions. Earlier approaches are not effective in solving these specific situations. Although several other works have been proposed to generate persons, birds, faces and scene images based on controllable structures, i.e., object key points, human skeletons and semantic maps, controllable structures provide four types of information to guide the image generation process, i.e., category, scale, orientation, and location. Although significant efforts have been made to achieve controllable image-to-image translation in the area of computer vision, there has been

very limited research on universal controllable image translation. That is, the typical problem with the aforementioned generative models is that each of them is tailored for a specific application, which greatly limits the generalization ability of the proposed models. To handle this problem, the unified GAN model is used, which can be tailored for handling all kinds of problem settings of controllable structure guided image-to-image translation, including object key point guided generative tasks, human skeleton guided generative tasks and semantic map guided generative tasks, etc. Since this model architecture covers a wide range of applications and considers multiple independent aspects of the image generation process, it is extremely general and its performance when it comes to specific image transformation tasks like the ones that we intend to solve isn't the best available.

[13] Approaches the image-to-image translation problems by adding a controller to Pix2Pix model to improve the classification performance. The improved Pix2Pix model is composed of three parts: generator, discriminator and controller. The input to the controller is the remote sensing image classification map, and the output is the reconstructed version based on the classification map obtained by generator (controller is also considered as the reverse mapping of the generator). The aim of the controller is to adjust the classification performance by the reconstruction error, which is established on the relation between the classification performance and the smaller the reconstruction error, the better the classification performance will be. U-net structure is used for the controller due to the advantages of the cross-layer connection in preserving the high frequency characteristics of the image. The optimization objective of the controller is to minimize the reconstruction error. It can only generate real images based on simple labels, but complex features are difficult to learn or the image is difficult to match well with the label. In other words, one important drawback here is the ignorance of explicit constraints between classification performance and reconstruction error.



## Chapter 3: SYSTEM DESIGN

### 3.1 System Use Case Diagram:

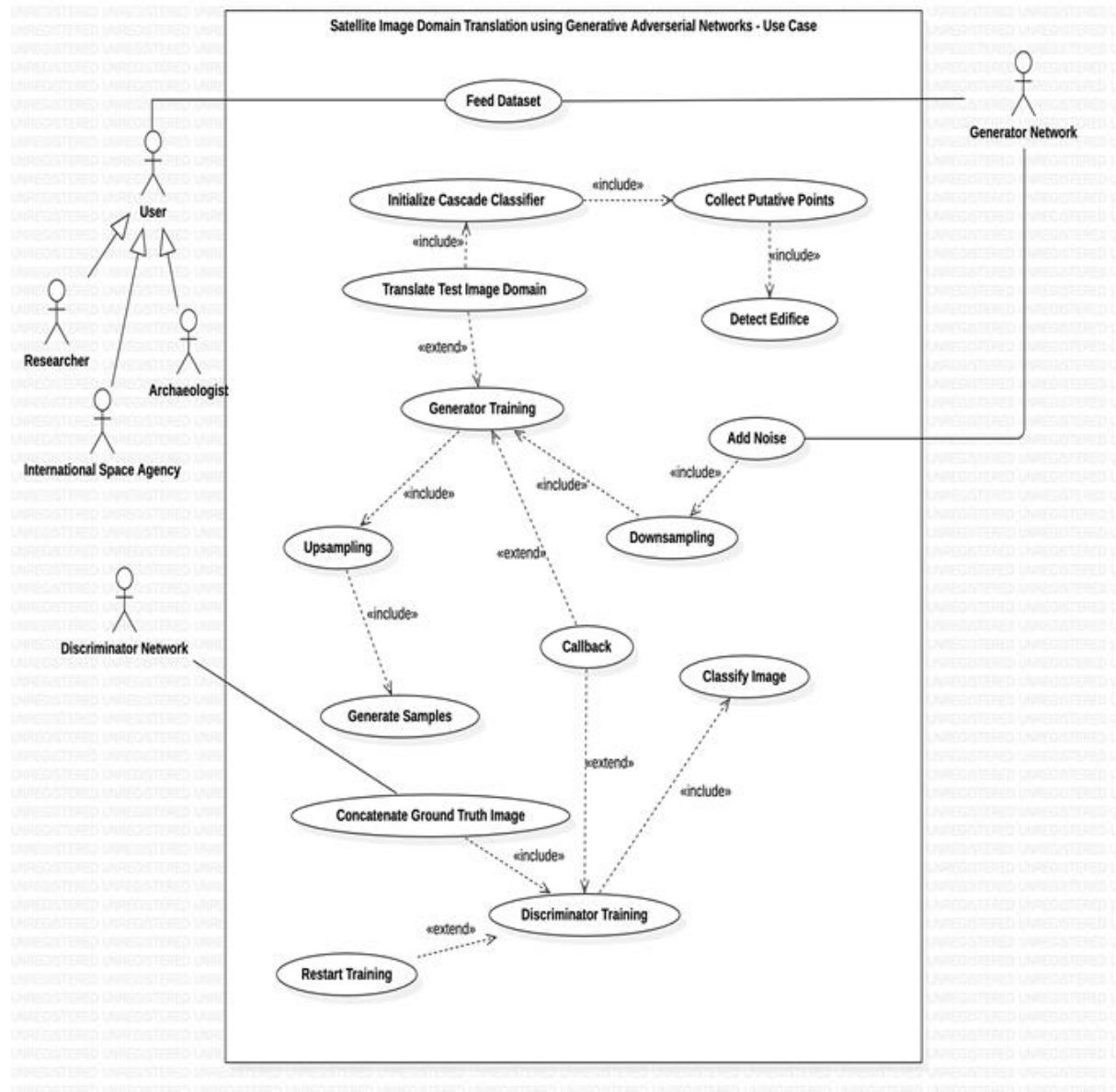
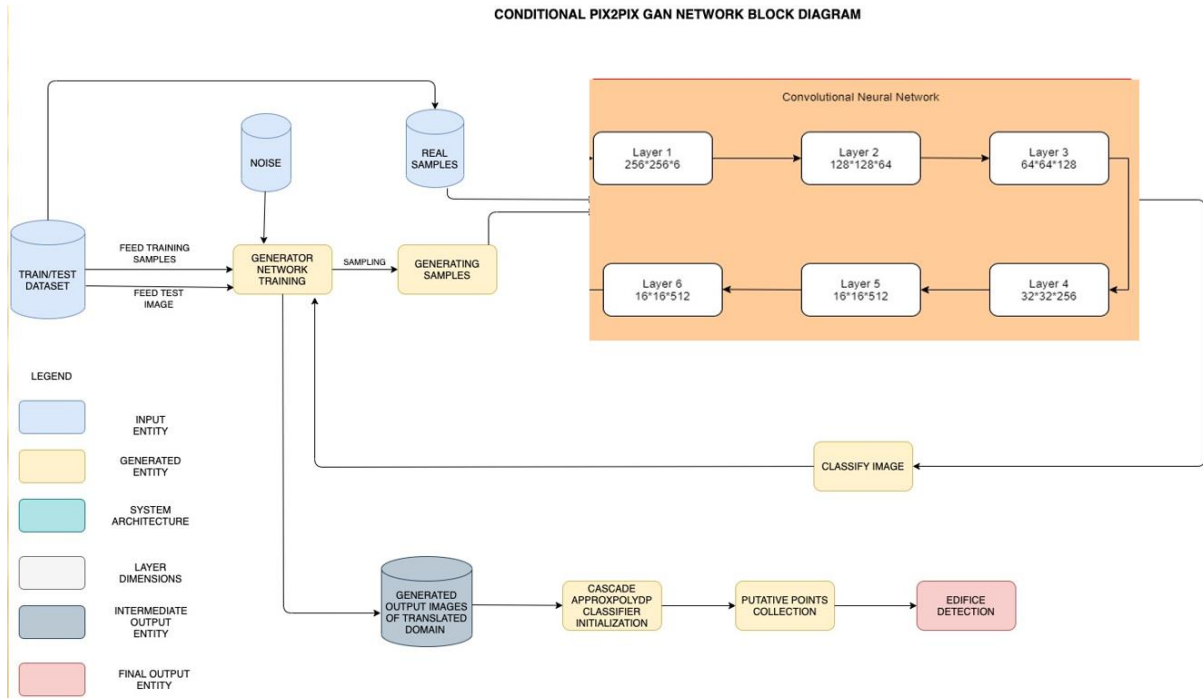


Figure.1 Use-Case Diagram

### 3.2 Overall Block Diagram:

#### List of Modules:

- Generator Module
- Discriminator Module
- Edifice Detection Module.



**Figure.2 Overall Block Diagram**

### Equation Involved:

The generator tries to minimize the following function while the discriminator tries to maximize it:

$$V(D, G) = E_x [\log (D(x))] + E_z [\log (1 - D(G(z)))]$$

**Equation.1 Generator Loss / Discriminator Objective Function**

In this function:

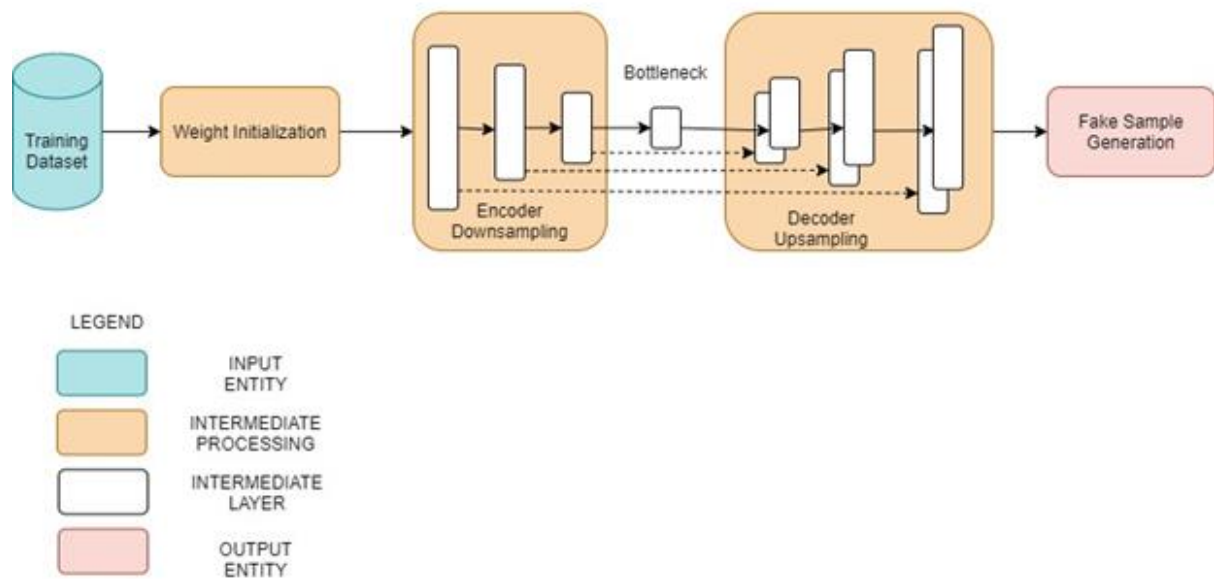
- $D(x)$  is the discriminator's estimate of the probability that real data instance  $x$  is real.
- $E_x$  is the expected value over all real data instances.
- $G(z)$  is the generator's output when given noise  $z$ .
- $D(G(z))$  is the discriminator's estimate of the probability that a fake instance is real.
- $E_z$  is the expected value over all random inputs to the generator (in effect, the expected value over all generated fake instances  $G(z)$ ).
- The formula derives from the cross-entropy between the real and generated distributions.

The generator can't directly affect the  $\log(D(x))$  term in the function, so, for the generator, minimizing the loss is equivalent to minimizing  $\log(1 - D(G(z)))$ .

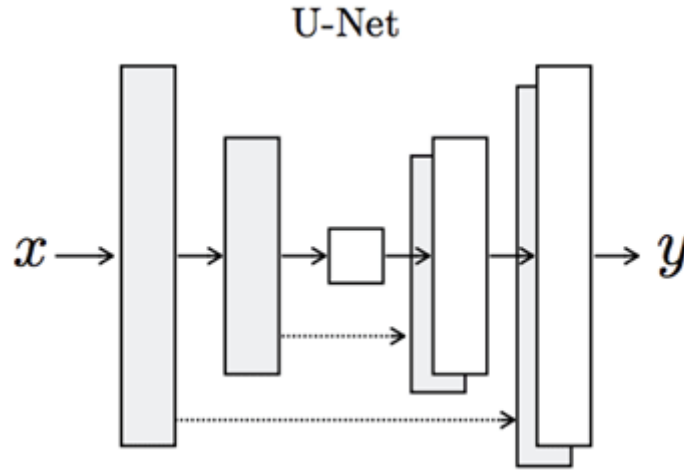
### 3.3 Detailed Module Wise Description:

#### 3.3.1 Generator Module:

<b>INPUT</b>	Training Dataset, Noise for the dataset, Discriminator Predictions
<b>PROCESS</b>	<ul style="list-style-type: none"> <li>• Generator network takes a real time dataset and generates a sample of data. It is an encoder-decoder model using a U-Net architecture.</li> <li>• Training is done via adversarial loss and the generator is updated via L1 loss measured between the generated image and the expected output.</li> <li>• This additional loss encourages it to create plausible translations of the source image.</li> <li>• The generator module analyses the distribution of this data in such a way that after the training phase the probability of the discriminator making an error maximizes.</li> <li>• It does this by first downsampling or encoding the input image down to a bottleneck layer, then upsampling or decoding the bottleneck representation to the size of the output image.</li> </ul>
<b>OUTPUT</b>	Fake Samples



**Figure.3 Generator Module Block Diagram**



*Figure.4 U-Net architecture of Generator Module*

***Sequence of Steps:***

- The training dataset is downloaded and transformed from its normal form to its Numpy Compressed form.
- Using an appropriate train - test ratio, we split the dataset into two parts.
- Images from the train split are directly chosen and they form a part of the real samples part.
- Each image has dimensions  $256 * 256 * 6$  since it is a concatenated version of the satellite image with its aerial form. Hence, we split it into two  $256 * 256 * 3$  images and give the satellite image as input to the generator network.
- Generator network generates fake images for the given input images.
- These fake images along with their real counterparts are given as input to the discriminator module.

***Pseudocode:***

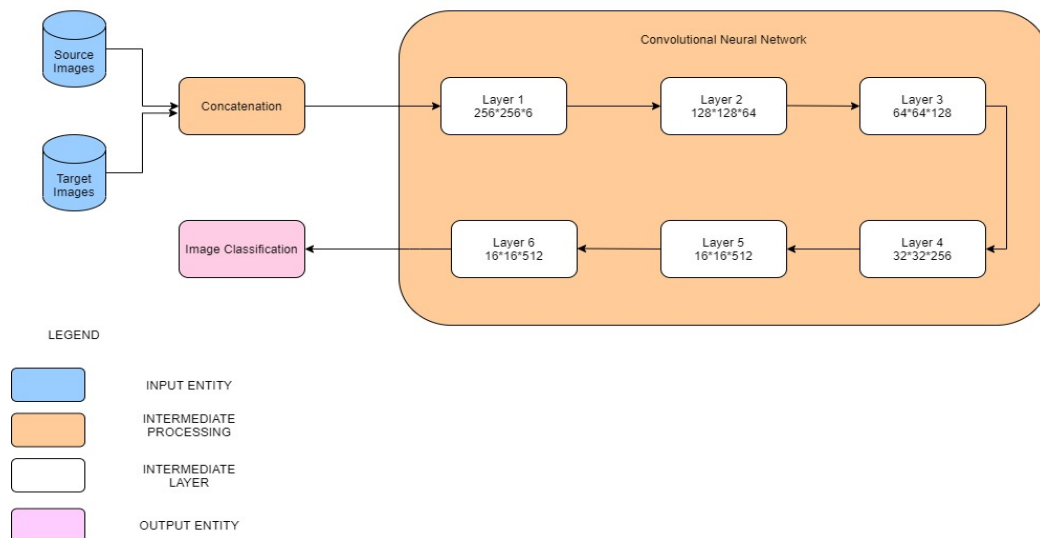
```

generator():
    Initialize random normal weights
    Input_img = input()
    Define encoder block
    Create bottleneck using conv_layer
    conv_layer = Activation(conv_layer)
    Define decoder block
    Bottleneck output using conv2d_transpose_layer
    output_layer = Activation(conv2d_transpose_layer)
    return model

```

### 3.3.2 Discriminator Module

<b>INPUT</b>	Training Dataset (Real Samples), Generated Samples (Fake Samples)
<b>PROCESS</b>	<ul style="list-style-type: none"> <li>Discriminator network decides whether the data is generated or directly taken from the real sample using a binary classification problem with the help of the sigmoid activation function that gives the output in the range 0 to 1.</li> <li>In adversarial, the model is trained in an adversarial setting and network simply means for the training of the model we use the neural networks as AI algorithms.</li> <li>It is optimized using binary cross entropy, and a weighting is used so that updates to the model have half (0.5) the usual effect.</li> </ul>
<b>OUTPUT</b>	Binary Classification of Images as Real or Fake



**Figure.5 Discriminator Module Block Diagram**

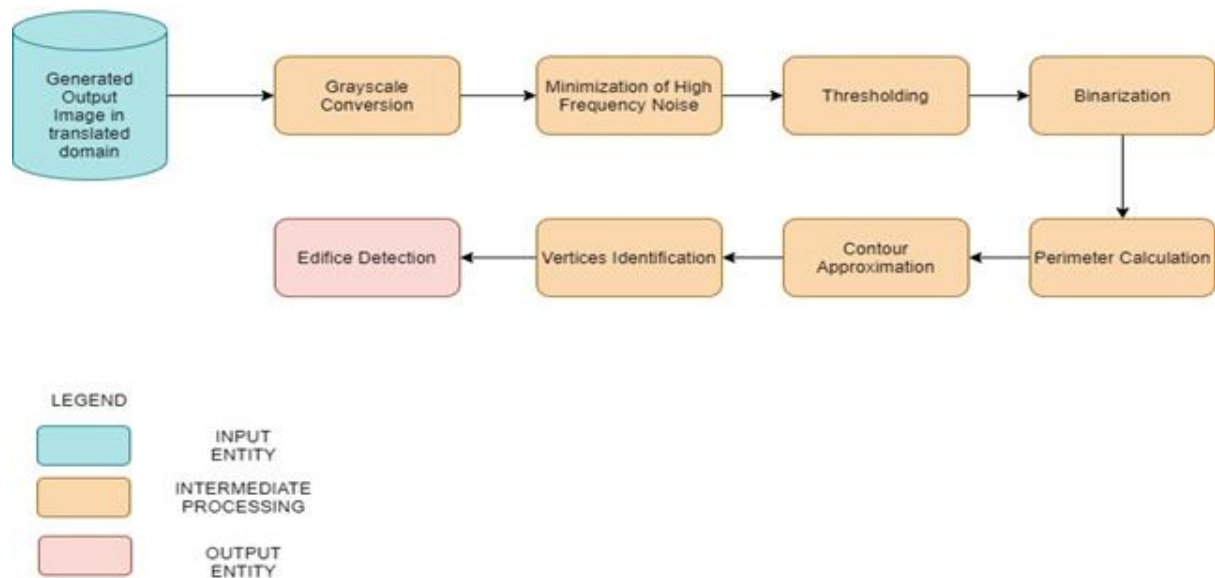
#### Sequence of Steps:

- Source input and generator output is connected to the discriminator input.
- We create an object of the Keras.Models class by giving the source input, the generated image and the classification output.
- This is then trained and optimized in-order to predict which image is fake and which one is real.
- As the discriminator gets more and more intelligent, the generator starts generating high quality images close to the source input image which in turn makes the discriminator's task challenging.

### ***Pseudocode:***

```
discriminator():  
    Initialize random normal weights  
    Source_img = input()  
    Target_img = input()  
    Merged_img = Concatenate(Source_img, Target_img )  
    For i in len(layers):  
        Define CNN layer including activation functions  
        Activated_layer[i] = Activation(layers[i])  
        layer[i] = Batch_normalization(Activated_layer[i])  
    patchout = Activation(layers[len(layers)-1])  
    define optimizer  
    compile model  
    return model
```

### ***3.3.3 Edifice Detection Module:***



***Figure.6 Edifice Detection Module Block Diagram***

<b>INPUT</b>	Generated Samples in the output domain
<b>PROCESS</b>	<ul style="list-style-type: none"> <li>• On obtaining the generated samples, it is visualized with the aid of ApproxPolydp feature enabling us to recognize the polygons in this image.</li> <li>• The putative points are figured out in the resulting image thereby enabling us to figure out the erections</li> </ul>
<b>OUTPUT</b>	Edifice Detection

***Pseudocode:***

```

edificeDetector():
    img = input()
    convert_to_grayscale(img)
    img = threshold(img)
    contours = findContours(img)
    for i in contours:
        approx = approxPolyDP(i)
        drawContours(img,approx)
        vertices = len(approx)
        if not aspectRatio() within specified limits:
            highlightEdifice()

```

***Algorithm involved:*** Ramer-Douglas-Peucker's Contour Approximation Algorithm

This algorithm is commonly known as the Ramer-Douglas-Peucker algorithm, or simply the split-and-merge algorithm. Contour approximation is predicated on the assumption that a curve can be approximated by a series of short line segments. This leads to a resulting approximated curve that consists of a subset of points that were defined by the original curve.

## **Chapter 4: EXPERIMENTAL RESULTS**

### ***4.1 System Development***

The proposed system implements a carefully designed architecture that outperforms the contemporary works by a significant amount and it provides an easy-to-use interface through which users can get their queries resolved. Once the user moves to the main dashboard from the opening GUI window, he is provided with assistance to load the fully trained generator and discriminator

models post which he can summarize the performance of the overall GAN network. On moving to the testing dashboard, he is allowed to utilize functionalities pertaining to loading a completely new test image returned by a satellite, generate its aerial map view, detect edifices within the generated image and finally calculate the associated Frechet – Inception Distance.

*Table.1 Core libraries used and their related dependencies*

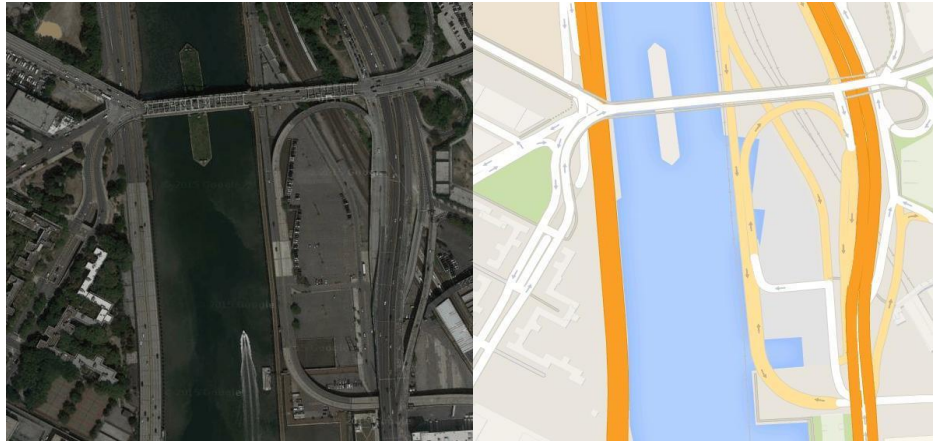
Pillow==8.1.2	imageio==2.9.0	Keras==2.4.3	scikit-image==0.18.1	kiwisolver==1.3.1
Markdown==3.3.4	matplotlib==3.4.1	mpmath==1.2.1	ndindex==1.5.1	networkx==2.5.1
numpy==1.19.5	PyYAML==5.4.1	tensorflow==2.4.1	opt-einsum==3.3.0	pandas==1.2.3
PyQt5==5.15.4	PyQt5-Qt5==5.15.2	PyQt5-sip==12.8.1	PyQt5Designer==5.14.1	python-dateutil==2.8.1
rsa==4.7.2	tensorboard==2.4.1	scipy==1.6.2	six==1.15.0	sympy==1.7.1
cv2-plt-imshow==0.0.1	tensorboard-plugin-wit==1.8.0	opencv-python==4.5.1.48	tensorflow-estimator==2.4.0	KerasPreprocessing==1.1.2

## 4.2 Dataset Description

**Link:** <https://www.kaggle.com/vikramtiwari/pix2pix-dataset>

**Description:** The overall dataset folder has two sub folders named train and validation. The train folder has 1096 images of dimensions 256 \* 512 \* 3 where [:,256] corresponds to the satellite image part and[:,256:] corresponds to the target aerial view form. The validation folder follows the same structure except the fact that it has 1098 images of similar dimensions. The train folder is used to train the generator to develop images that are indistinguishable from the real target image, i.e, to confuse the discriminator progressively and make its job of distinguishing real from fake image tougher as training progresses.





*Figure.7 Concatenated Input image used for training*

### 4.3 Working of the System

- Navigate to the directory containing the project files and activate the virtual environment containing all the necessary libraries and dependencies pre-installed.

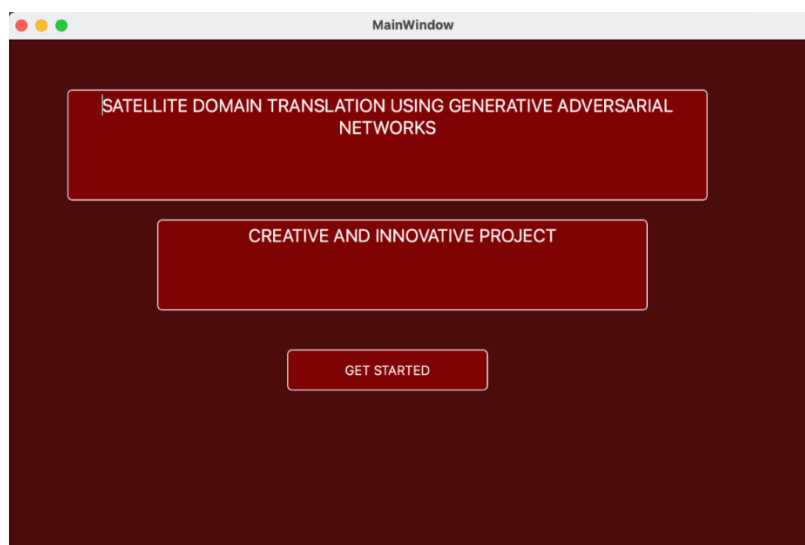
```

Codes — python UI.py — 80x24
[(base) gokul@Gokuls-MacBook-Pro User-Interface % ls
Codes          PyQtDesignerFiles      requirements.txt
Model          interface                virt
[(base) gokul@Gokuls-MacBook-Pro User-Interface % source virt/bin/activate
[(virt) (base) gokul@Gokuls-MacBook-Pro User-Interface % cd Codes
[(virt) (base) gokul@Gokuls-MacBook-Pro Codes % ls
GAN.py         Pix2Pix.ipynb    UI.py           __pycache__
[(virt) (base) gokul@Gokuls-MacBook-Pro Codes % python UI.py

```

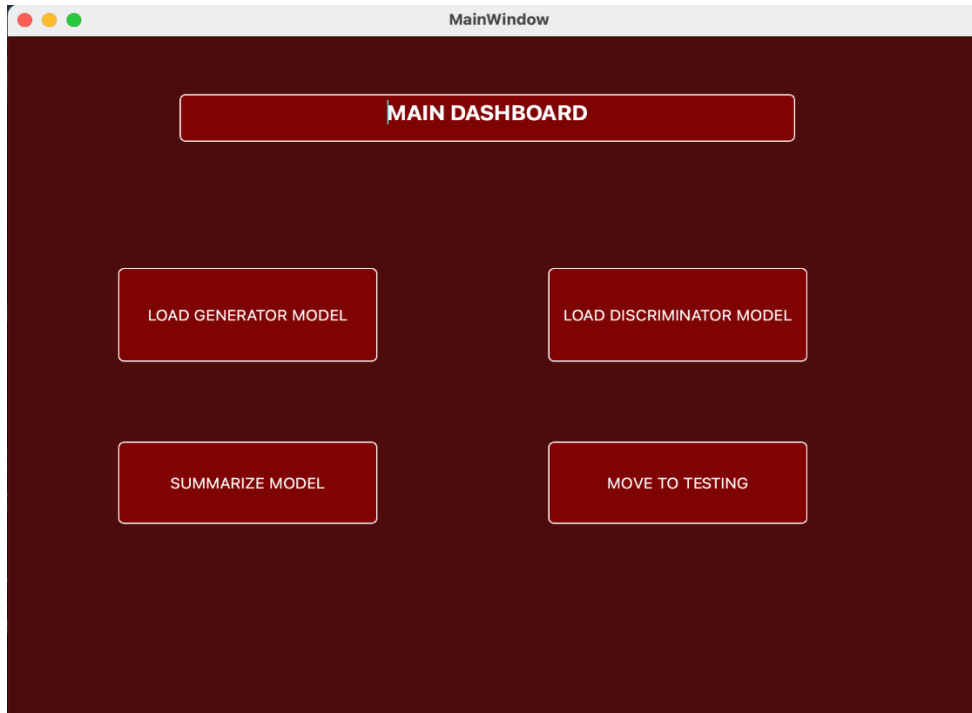
*Figure.8 Terminal*

- Navigate to the main-dashboard from the opening window using the appropriate buttons (Refer Figure 9).



*Figure.9 GUI Open Window*

- Load the generator, discriminator and GAN model using the appropriate buttons in that order (*Refer Figure 11*).
- Summarize the performance of the overall model using the summarize model button which opens a window containing the input, target and generated output images (*Refer Figure 10*).



***Figure.10 Main dashboard***

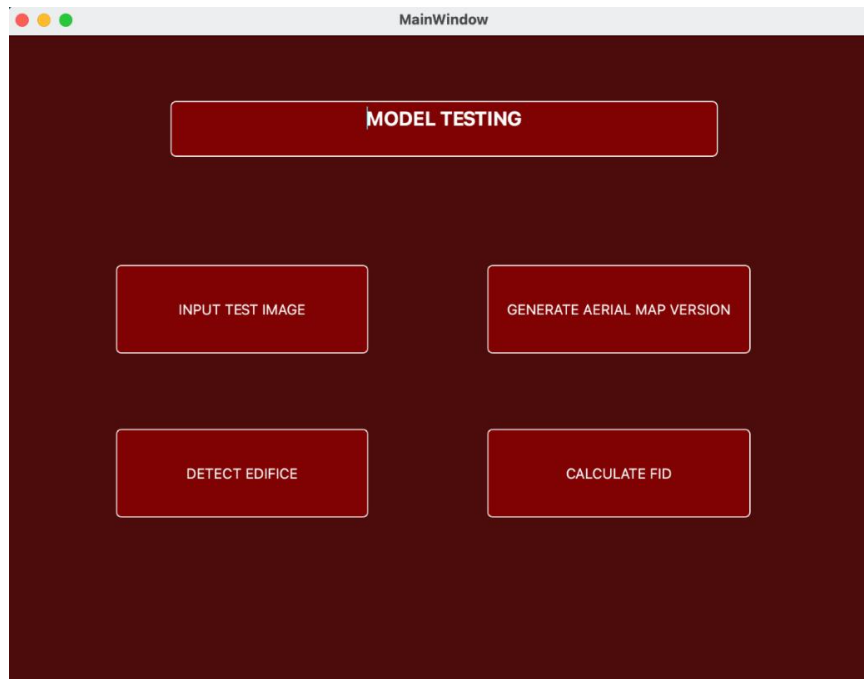
```

GENERATOR MODEL LOADED SUCCESSFULLY...!!
Input = (None, 256, 256, 3)
Output = (None, 256, 256, 3)
DISCRIMINATOR MODEL LOADED SUCCESSFULLY...!!
Input = [(None, 256, 256, 3), (None, 256, 256, 3)]
Output = (None, 16, 16, 1)
GAN MODEL CREATED SUCCESSFULLY...!!
Input = (None, 256, 256, 3)
Output = [(None, 16, 16, 1), (None, 256, 256, 3)]

```

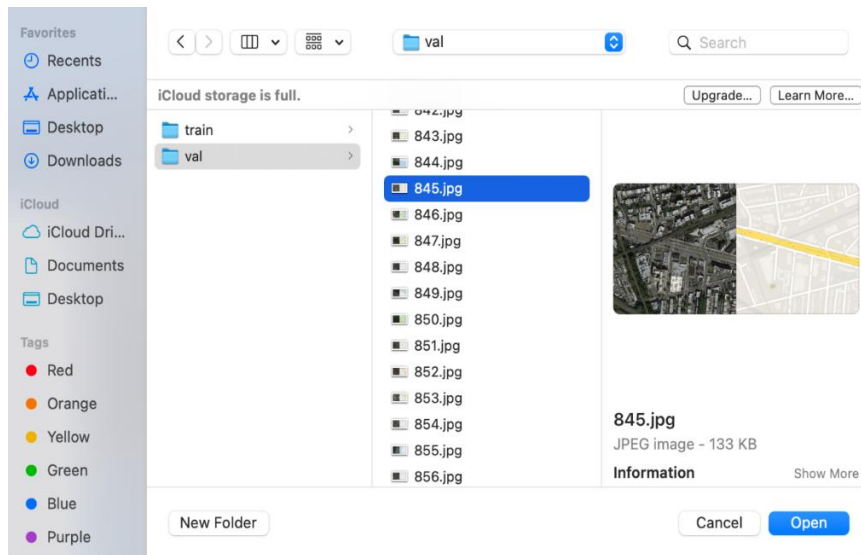
***Figure.11 Loaded Models***

- Upon reaching the testing dashboard, the user can input his test image using the corresponding button which opens the dedicated file browser where-in the user give his input choice (*Refer Figure 13*).
- On clicking the Generate Aerial Map Version button, the trained generator gives out the output which is the input image in a translated domain. (Aerial Map)



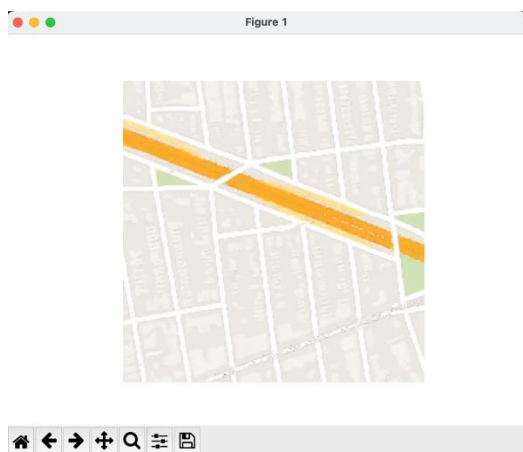
**Figure.12 Testing Dashboard**

- Detect Edifice button can be used to detect edifices present in the generated output image (*Refer Figure 12*).
- On clicking the Calculate FID button the Fréchet Inception Distance is calculated over the entire testing dataset and is returned.



**Figure.13 File Browser**

- The following images provide an illustration of the intended outputs (*Refer Figure 14 and Figure 15*).



*Figure.14 Generated Aerial Map Image*



*Figure.15 Edifices Detected in the Generated Image*

## Chapter 5: RESULT ANALYSIS

### *5.1 Final System Test Case Table*

The below table has the following test cases covered:

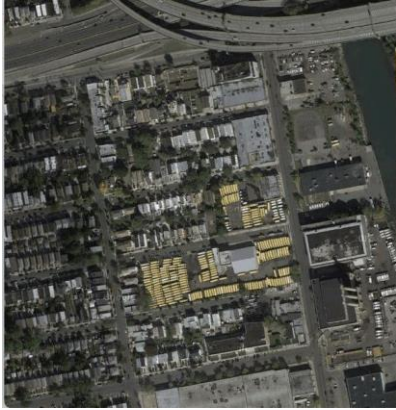


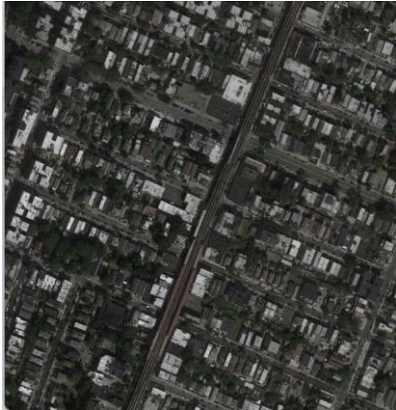
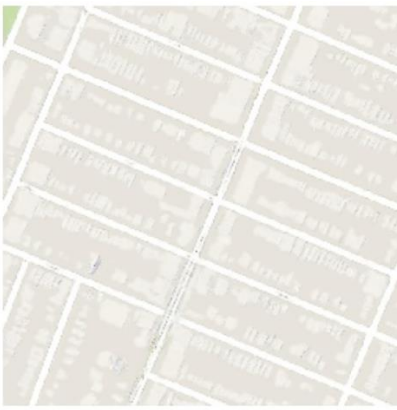
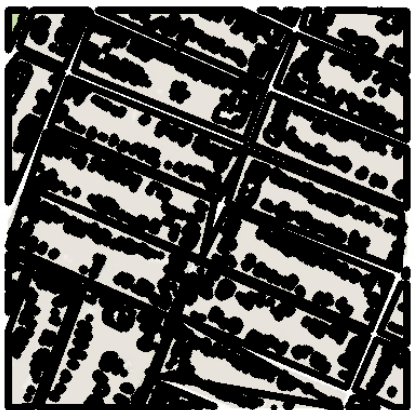






- Images consisting of only land
- Images consisting of only sea
- Images consisting of only grassland
- Images consisting of both land and grasslands
- Images consisting of both land and sea
- Images consisting of both sea and grasslands
- Images consisting of land, sea and grasslands

Observe the following final system test case table carefully

*Table.2 Final System Test Case Table*

S NO	INPUT SATELLITE IMAGE	GENERATED AERIAL MAP IMAGE	OUTPUT IMAGE WITH EDIFICES DETECTED
1			
2			
3			



4			
5			
6			
7			

## *Inferences*

As we move towards the end of the training phase of the model, the discriminator is in a position where it finds it extremely difficult to distinguish between the real and the generated images. At this stage the generator would generate highly plausible transformations of the input satellite images. This can be verified from **Table 2** (column 2). Upon feeding this as the input to the third and the final module, it discovers a wide range of edifices present in the image precisely. This can be substantiated **Table 2** (column 3).

### *5.2 Evaluation Metrics*

#### *Fréchet Inception Distance:*

$$\text{FID} = |\mu - \mu_w|^2 + \text{tr}(\Sigma + \Sigma_w - 2(\Sigma\Sigma_w)^{1/2}).$$

The Fréchet inception distance (FID) is a metric used to assess the quality of images created by the generator of a generative adversarial network (GAN). Unlike the earlier inception score (IS), which evaluates only the distribution of generated images, the FID compares the distribution of generated images with the distribution of real images that were used to train the generator. The FID metric is the Wasserstein metric between two multidimensional Gaussian distributions: the distribution of some neural network features of the images generated by the GAN and the distribution of the same neural network features from the "world" or real images used to train the GAN.

```
[ ] fid = obj1.__call__(taring,srcimg,15,128,None,False,None)
print(fid)

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6:
560.9619757832536
```

*Figure.16 Snapshot of the calculated metric – FID*

#### *Mean Square Error:*

$$MSE = \frac{1}{m \ n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i, j) - K(i, j)]^2$$

The mean squared error function takes the target image and the generated aerial map image and compares the pixel intensities. A value of 0 for MSE indicates perfect similarity. A value greater than one implies less similarity and will continue to grow as the average difference between pixel intensities increases.

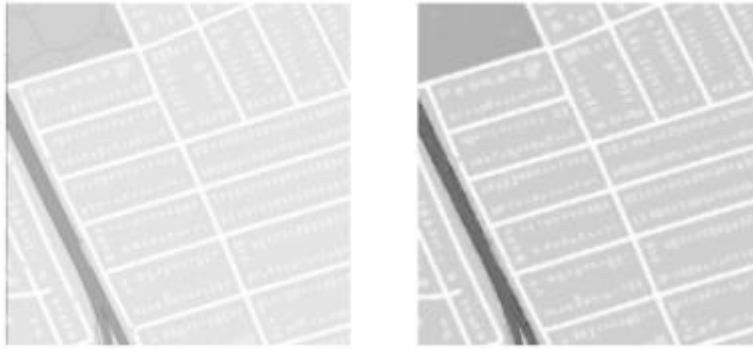
### ***Structural Similarity (SSIM):***

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

SSIM attempts to model the perceived change in the structural information of the image, whereas MSE is actually estimating the perceived errors. Doing this leads to a more robust approach that is able to account for changes in the structure of the image, rather than just the perceived change.

It takes as parameters the (x, y) location of the N x N window in each image, the mean of the pixel intensities in the x and y direction, the variance of intensities in the x and y direction, along with the covariance. The SSIM value can vary between -1 and 1, where 1 indicates perfect similarity.

Target vs. Generated      MSE: 0.002, SSIM: 0.87



***Figure.17 Snapshot of the calculated metrics – SSIM and MSE***

### ***Manhattan Norm:***

$$d_1(\mathbf{p}, \mathbf{q}) = \|\mathbf{p} - \mathbf{q}\|_1 = \sum_{i=1}^n |p_i - q_i|$$

The set of vectors whose 1-norm is a given constant form the surface of a cross polytope of dimension equivalent to that of the norm minus 1. The Manhattan norm is also called the L1 norm. The distance derived from this norm is called the Manhattan distance.



```
1 main()
```

Manhattan norm per pixel: 20.73378005324527

*Figure.18 Snapshot of the calculated metric – Manhattan Norm*

*Table.3 Comparison of the performance of the existing system and the proposed system*

<i>Evaluation Metric</i>	<i>Existing System</i>	<i>Proposed System</i>
Frechet-Inception Distance	625.25	560.961
Mean Square Error	0.0026	0.0020
Structural Similarity	0.86	0.87
Manhattan Norm	33.68	20.73

### *5.3 Epoch-wise Evaluation Metric Values*

<i>Epochs</i>	<i>Evaluation Metric</i>	<i>Proposed System</i>
50	Frechet-Inception Distance	625.173
	Mean Square Error	0.0025
	Structural Similarity	0.83
	Manhattan Norm	35.96
100	Frechet-Inception Distance	601.253
	Mean Square Error	0.0023
	Structural Similarity	0.84
	Manhattan Norm	30.42

150	Frechet-Inception Distance	580.652
	Mean Square Error	0.0022
	Structural Similarity	0.85
	Manhattan Norm	23.83

200	Frechet-Inception Distance	560.961
	Mean Square Error	0.0020
	Structural Similarity	0.87
	Manhattan Norm	20.73

#### 5.4 Graphical Analysis of Observed Evaluation Metrics:

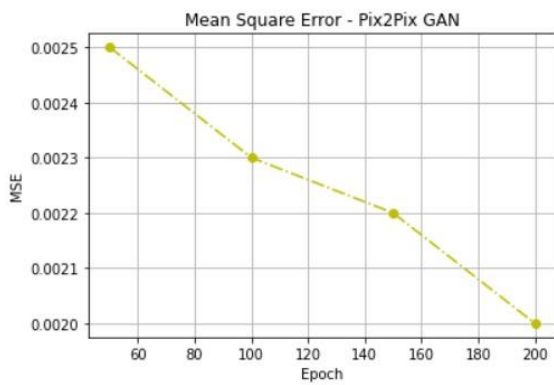


Figure.19 Mean Square Error Trend over Epochs

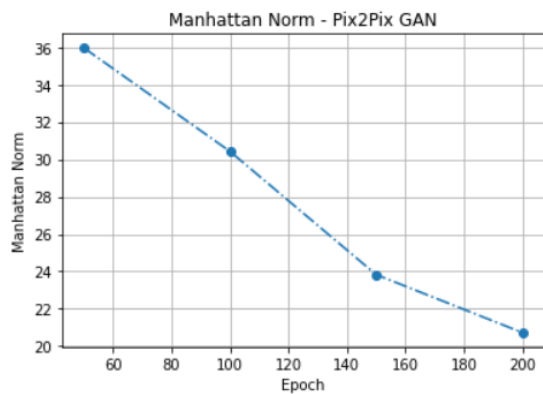
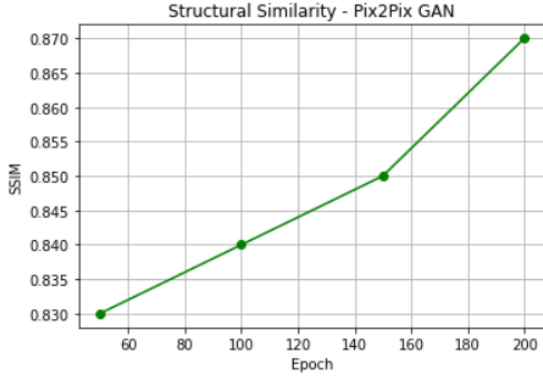
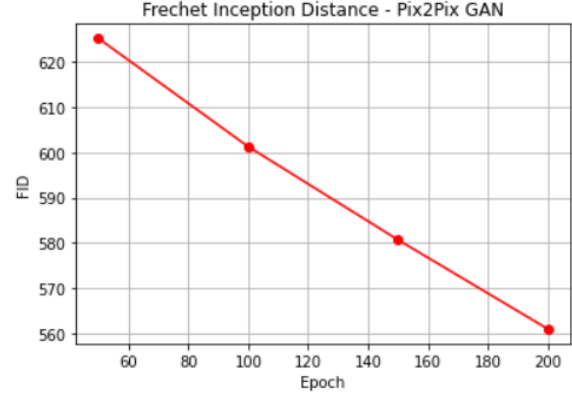


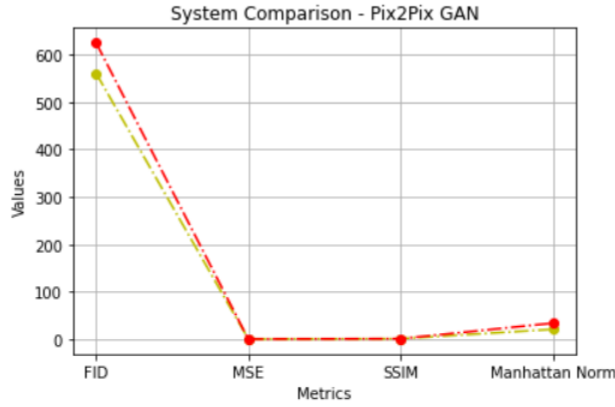
Figure.20 Manhattan Norm Trend over Epochs



**Figure.21 Structural Similarity Trend over Epochs**



**Figure.22 Fréchet Inception Distance Trend**



**Figure.23 Overall proposed system evaluation metrics trend as compared to the existing system**

### ***Inferences:***

- Since Mean Square Error compares the pixel-wise intensities of the generated and the target images, an ideal value of zero is what we intend to achieve. As the training progresses, Mean Square Error gradually keeps on falling (*Refer Figure 19*).
- Since the Manhattan norm represents the sum of the lengths of the projections of a line segment between two points onto the coordinate axes in an n-dimensional vector space, the quality of the model depends on how small the value is. This is evident from *Figure 20* where the value of the Manhattan norm falls as training progresses.
- Since the structural similarity assess the visual impact of the major three characteristics of an image namely luminance, contrast and structure, the closer the value is to 1, the more similar the arguments are. *Figure 21* corroborates the mentioned hypothesis, where the value gets closer to 1 as the model gets better.
- Since FID directly calculates the distance between the feature vectors in their respective n-dimensional vector spaces, the lower the value the better the performance (*Refer Figure 22*).

- The overall comparison between the existing and the proposed system can be comprehended from *Figure 23*.

## **Chapter 6: CONCLUSION**

### **6.1 Future Scope**

Although the proposed system works exceedingly well in tackling a multitude of hitches related to military and intelligent services related applications, the important fact to consider here is that the for the generator to generate plausible transformations of the input satellite image, the discriminator needs to be trained on a corpus of related images to an extent where it becomes extremely sensitive and can detect even pixel variations. An ambitious future scope would be to train the network on a corpus of satellite images retrieved from the exoplanets of our solar system and other extra-terrestrial celestial bodies to detect the presence of life in them by performing in-depth analysis. This would help scientists to easily add nodes to the Interplanetary Transport Network (ITN) which would eventually cut the cost of space travel by a large margin.

### **6.2 Conclusion**

We have developed an intelligent system that has learned to generate the aerial map version of any complicated satellite image with buildings, trees, grassland, rivers, sea, land, etc which was captured at any part of the day. We successfully and accurately detect all the visible as well as hidden edifices that are part of the image which helps the secret services, intelligence bureaus and other related international agencies to figure out the locations of anti - nationalists and terrorists and bring down the crime rate by a significant amount. The system also plays an extremely important role when it comes go intra - planetary transport and space exploration.

## **REFERENCES**

- [1] X. Yu, G. Yang and J. Saniie, "Face Morphing Detection using Generative Adversarial Networks," 2019 IEEE International Conference on Electro Information Technology (EIT), 2019, pp. 288-291, doi: 10.1109/EIT.2019.8834162.
- [2] A. Ukasha, E. Geepalla and M. Elbreki, "A Review: Spatial Domain Methods Comparison for Single Arabic Contours Approximation," 2020 International Conference on Computing and Information Technology (ICCIT-1441), 2020, pp. 1-5, doi: 10.1109/ICCIT-144147971.2020.9213812.
- [3] H. Tang, H. Liu and N. Sebe, "Unified Generative Adversarial Networks for Controllable Image-to-Image Translation," in IEEE Transactions on Image Processing, vol. 29, pp. 8916-8929, 2020, doi: 10.1109/TIP.2020.3021789.

- [4] N. -T. Tran, V. -H. Tran, N. -B. Nguyen, T. -K. Nguyen and N. -M. Cheung, "On Data Augmentation for GAN Training," in *IEEE Transactions on Image Processing*, vol. 30, pp. 1882-1897, 2021, doi: 10.1109/TIP.2021.3049346.
- [5] Q. -R. Han, W. -Z. Zhu and Q. Zhu, "Icon Colorization Based On Triple Conditional Generative Adversarial Networks," 2020 *IEEE International Conference on Visual Communications and Image Processing (VCIP)*, 2020, pp. 391-394, doi: 10.1109/VCIP49819.2020.9301890.
- [6] I. B. Akkaya and U. Halici, "GAIT: Gradient Adjusted Unsupervised Image-To-Image Translation," 2020 *IEEE International Conference on Image Processing (ICIP)*, Abu Dhabi, United Arab Emirates, 2020, pp. 1586-1590, doi: 10.1109/ICIP40778.2020.9191271.
- [7] T. Mukhiddin, W. Lee, S. Lee and T. Rashid, "Research Issues on Generative Adversarial Networks and Applications," 2020 *IEEE International Conference on Big Data and Smart Computing (BigComp)*, 2020, pp. 487-488, doi: 10.1109/BigComp48618.2020.00-19.
- [8] F. Xiong, Q. Wang and Q. Gao, "Consistent Embedded GAN for Image-to-Image Translation," in *IEEE Access*, vol. 7, pp. 126651-126661, 2019, doi: 10.1109/ACCESS.2019.2939654.
- [9] Kang, Taewon and Lee, Kwang Hee, "Unsupervised Image-to-Image Translation with Self-Attention Networks", 2020 *IEEE International Conference on Big Data and Smart Computing (BigComp)*, doi: 10.1109/bigcomp48618.2020.00-92
- [10] SPA-GAN: Spatial Attention GAN for Image-to-Image Translation : Hajar Emami, Majid Moradi Aliabadi, Ming Dong, and Ratna Babu Chinnam
- [11] DualGAN: Unsupervised Dual Learning for Image-to-Image Translation Zili Yi1, Hao Zhang , Ping Tan , and Minglun Gong
- [12] H. Tang, H. Liu and N. Sebe, "Unified Generative Adversarial Networks for Controllable Image-to-Image Translation," in *IEEE Transactions on Image Processing*, vol. 29, pp. 8916-8929, 2020, doi: 10.1109/TIP.2020.3021789.
- [13] X. Wang, H. Yan, C. Huo, J. Yu and C. Pant, "Enhancing Pix2Pix for Remote Sensing Image Classification," 2018 24th *International Conference on Pattern Recognition (ICPR)*, 2018, pp. 2332-2336, doi: 10.1109/ICPR.2018.8545870.
- [14] C. Li, Z. Wang and H. Qi, "Fast-Converging Conditional Generative Adversarial Networks for Image Synthesis," 2018 25th *IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 2132-2136, doi: 10.1109/ICIP.2018.8451161.
- [15] Z. Li et al., "An improvement on the CNN-based OAM Demodulator via Conditional Generative Adversarial Networks," 2019 18th *International Conference on Optical Communications and Networks (ICOON)*, 2019, pp. 1-3, doi: 10.1109/ICOON.2019.8934809.
- [16] Y. -W. Lu, K. -L. Liu and C. -Y. Hsu, "Conditional Generative Adversarial Network for Defect Classification with Class Imbalance," 2019 *IEEE International Conference on Smart Manufacturing, Industrial & Logistics Engineering (SMILE)*, 2019, pp. 146-149, doi: 10.1109/SMILE45626.2019.8965320.
- [17] P. Isola, J. Zhu, T. Zhou and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," 2017 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, pp. 5967-5976, doi: 10.1109/CVPR.2017.632.
- [18] C. Min, Y. Li, L. Fang and P. Chen, "Conditional Generative Adversarial Network on Semi-supervised Learning Task," 2019 *IEEE 5th International Conference on Computer and Communications (ICCC)*, Chengdu, China, 2019, pp. 1448-1452, doi: 10.1109/ICCC47050.2019.9064268.

- [19] M. Oza, H. Vaghela and S. Bagul, "Semi-Supervised Image-to-Image Translation," 2019 International Conference of Artificial Intelligence and Information Technology (ICAIIIT), Yogyakarta, Indonesia, 2019, pp. 16-20, doi: 10.1109/ICAIIIT.2019.8834613.
- [20] R. Yin, "Multi-Resolution Generative Adversarial Networks for Tiny-Scale Pedestrian Detection," 2019 IEEE International Conference on Image Processing (ICIP), 2019, pp. 1665-1669, doi: 10.1109/ICIP.2019.8803030.
- [21] G. Wang, G. Dong, H. Li, L. Han, X. Tao and P. Ren, "Remote Sensing Image Synthesis via Graphical Generative Adversarial Networks," IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium, 2019, pp. 10027-10030, doi: 10.1109/IGARSS.2019.8898915.
- [22] Y. Cui and W. Wang, "Colorless Video Rendering System via Generative Adversarial Networks," 2019 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), 2019, pp. 464-467, doi: 10.1109/ICAICA.2019.8873434.
- [23] M. Li, G. Liu, S. Li and Y. Wu, "Radio Classify Generative Adversarial Networks: A Semi-supervised Method for Modulation Recognition," 2018 IEEE 18th International Conference on Communication Technology (ICCT), 2018, pp. 669-672, doi: 10.1109/ICCT.2018.8600032.
- [24] Z. Zhai and J. Zhai, "Identity-preserving Conditional Generative Adversarial Network," 2018 International Joint Conference on Neural Networks (IJCNN), 2018, pp. 1-5, doi: 10.1109/IJCNN.2018.8489282.