iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** XI **Month of publication:** November 2024

DOI: https://doi.org/10.22214/ijraset.2024.65517

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue XI Nov 2024- Available at www.ijraset.com

DeepUrbanMapper - Satellite Image to Google Maps Translation

Zarinabegam Mundargi[1], Jay Wanjare[2], Saket Waware[3], Yash Wagh[4], Aditya Yeole[5], Yuvraj Susatkar[6]
Artifical Intelligence & Data Science, Vishwakarma Institute of Technology Pune, India

Abstract: ***The research presents DeepUrbanMapper, an innovative framework developed for the translation of satellite imagery into Google Maps-style renderings utilizing advanced image-to-image translation techniques. It is based on Generative Adversarial Networks (GANs), widely known in machine learning for their ability to yield high-quality images. To train it, the process involves a carefully curated and labeled dataset that consists of paired satellite and map images, enabling DeepUrbanMapper to learn the complex mappings between these two distinct visual domains. The GAN architecture used in DeepUrbanMapper is carefully designed and optimized to preserve spatial coherence and improve visual quality of the translated image. This helps in producing output maps that are close to the ground truth in both visual and geographical aspects. The proposed method includes few novel strategies to stabilize the GAN training and to overcome the notorious mode collapse problem and ensure a consistent output. DeepUrbanMapper has been extensively evaluated quantitatively and qualitatively for its performance. The evaluation shows that the new framework significantly outperforms the existing methods in terms of visual realism of the translated image and its ability to retain detailed features of the input satellite image which opens up many possibilities urban planning, car navigation systems, GIS.***

Keywords: Machine Learning, Deep learning, Model Accuracy, Image Processing, Object Identification and Data Processing, Generative Adversarial Networks, Generator Model, Discriminator Model.

I. INTRODUCTION

High-speed urbanization and city growth call for accurate and current mapping to support a myriad of urban planning and navigation to disaster management and GIS applications. Traditional map creation and update processes are painstaking, time-consuming, and costly, involving heavy reliance on manual digitization and interpretation of satellite imagery. This makes for a dire need for a solution that is automated, efficient, and scalable to generate quality maps from satellites. The need for this project arises from several key factors. Firstly, the sheer volume and frequency of satellite imagery being captured today demand automated solutions that can process and translate this data into useful formats rapidly. As urban landscapes evolve, having the ability to quickly update maps is crucial for maintaining accurate representations of these areas. Outdated maps can lead to inefficiencies and errors in navigation, planning, and emergency response. Secondly, there is an increasing need for high-resolution and highly detailed maps due to the demands of many businesses. Maps are essential for a variety of purposes, including the design and construction of infrastructure by city planners, the coordination of relief efforts by disaster response teams, and the optimisation of business operations and customer experiences by ride-hailing and delivery platforms, among others. Current systems used to translate satellite images into map formats are majorly based on old image processing and machine learning techniques. These are good approaches but have serious shortcomings. Most cannot capture the complexity and intricate features within satellite images, translating to maps that are neither accurate nor detailed. Other shortcomings of the said approach include challenges in maintaining spatial coherency, thereby causing artifacts and inconsistencies in maps created. The lack of adaptability to different urban environments further hampers their effectiveness.[1] Currently, geospatial data is gathered and arranged to create a comprehensive digital image (map) using satellites, GPS-equipped drones or UAVs, and a number of trustworthy organisations. The publicly accessible, human-readable maps and the real geographic conditions/street views, however, are significantly delayed. One method to lower this delay is to automate the process of turning a satellite image into a map that can be read by humans. Generative models can help accomplish this automation.

The research addresses these challenges by introducing DeepUrbanMapper, a novel framework using Generative Adversarial Networks for translation of satellite images to Google Maps style renderings. Given the powerful alternative of GANs that can generate high-fidelity images, this effort is then harnessed for this task. The GAN architecture in the DeepUrbanMapper framework is trained on a large dataset of labelled pairs of satellite and map images. The GAN learns to map the complex relationships between these two domains with high accuracy, which generates highly detailed and visually coherent map images that are very similar to the ground truth. DeepUrbanMapper aims to address the shortcomings of current systems by providing several key advantages.

The GAN-based approach ensures that spatial details and visual accuracy are preserved, producing maps with superior realism. The model's ability to generalize across different urban landscapes enhances its applicability to diverse geographic regions. Furthermore, the automated nature of this approach significantly reduces the time and labour required for map generation, offering a scalable solution that can keep pace with the rapid changes in urban environments.

The dataset, which was gathered from Google Earth Engine for satellite photos and Google Maps API for related map images, has been made freely accessible by the authors of [2]. The resolution and zoom level of the satellite photos are stated, and they are accompanied with human-readable maps. The satellite picture is fed as input to the Conditional Adversarial Network generator, which produces a bogus output image. After that, the output picture and the satellite image are matched and supplied as input to the discriminator, which further categorises images as authentic or fraudulent.

II. LITERATURE REVIEW

Generative model-based image-to-image translation has, so far, seen much important work. There are two general categories: deep learning algorithms and algorithms that are based on machine learning. The former does not serve large datasets well, while the latter enhances image-to-image translation.

The task of automating map generation from satellite imagery is of significant research interest, given its capability to make the processes of generating and updating a map easier. [1] focuses on the challenge of automating web interface design using generative deep learning models, specifically focusing on the efficacy of GANs for this task. This study brings to the fore the fact that GANs generate high-quality web designs from label maps, emphasizing their robustness and the reduction in manual overhead compared with conventional machine learning algorithms.

[2] Also based on GAN extends the utility in creating human-readable maps from satellite imagery to reduce latency between changes in the real world and map updates. The study illustrates a GAN-based model that is capable of generating Google Maps-style renderings from satellite images. It shows how this model could find applications in ride-sharing, food delivery, and national security. The research strongly puts into focus the commercial and humanitarian value of an accurate and updated map, as in the goals of automation and efficiency underlined in reference. Based on the concept of the translation from satellite to map images, the research work in [3] proposes an improved GAN model with geographic data and semantic regulation to make up for the defects of standard GANs. The authors proposed Semantic-regulated Geographic GAN, which captures the integration of GPS coordinates and semantic estimation to improve the accuracy and quality of the map generated, especially in regions of complexity and visual ambiguity. This work brings out substantial quantitative and qualitative improvements over current methods and hence stands out as a strong approach to handle challenging regions with complex road networks and visual obstructions. The integration of semantic information and other data sources, as demonstrated in [3], is a significant step towards achieving the goals that were not met in overcoming the obstacles [1] and [2] raised in the direction of dependable and effective mapping systems.

The study [4] compares Pix2Pix and CycleGAN, two popular GAN-based models for image-to-image translation. Using the Frechet Inception Distance measure, they assess them for performance against the dataset requirements, training duration, and resource utilisation. It comes to the conclusion that for picture translation tasks, Pix2Pix outperforms CycleGAN in terms of efficiency and translation quality. The research [5] introduces MapGen-GAN, a novel framework that transforms remote sensing images to maps, particularly for scenarios of disaster response. It integrates circularity-consistency and geometrical-consistency constraints to handle the absence of paired data in training and diminish semantic distortions. The framework also features BRB-Unet, an enhanced generator designed for accurate map translation. The results of the experiments show that MapGen-GAN performs better than other state-of-the-art models on datasets of New York City and Washington DC.

Deep learning has made translating remote sensing pictures into maps easier in the field of remote sensing and cartography by removing the need for conventional vectorization techniques. With the advent of GANs, particularly the Pix2Pix model, this process, referred to as rs2map translation, has made important advances. However, the inconsistency of spatial resolutions and the high cost of computation for cross-domain translation have limited the effectiveness of the existing methods in generating maps at multiple scales. A series approach for multi-scale rs2map translation was proposed by [6] to overcome these challenges. This strategy utilizes high-resolution RSIs in generating large-scale maps, which are then translated into multi-scale maps. This approach performed better compared to the parallel strategy and indicated marked improvement in metrics like intersection over union and structural similarity. Moreover, GANs have been used so much in applications of image-to-image translation; such scenarios are classified into two classes: paired and unpaired. While these pairs form the basis for traditional models like CycleGAN and Pix2Pix, their lack of semantic knowledge leads to problems like overfitting and poor generalization. To address these very issues, a semantic map infused GAN training methodology is proposed in [7].

The proposed approach improved the capability of the model for identifying and translating different object segments by introducing semantic mappings during training, which increases the overall quality and coherence of the translated images. Even though this strategy requires semantic maps during training, as compared to vanilla training approaches, it outperforms when tested on benchmark datasets.

The processing of satellite images, with applications to super-resolution tasks, has been done in the context of CNNs and GANs in order to handle the difficulties posed by air disturbances and ultradistance imaging. The inability of traditional shallow learning methods led to the design of deeper networks, including SRCNN and residual networks. However, techniques often suffer from high computing costs and noise production. A GAN-based architecture was proposed in [8] for super-resolving satellite photos to get over these restrictions. It allows multi-level mappings between low-resolution and high-resolution pictures and enhances the model's ability to recover fine features and generate photo-realistic results. It mainly comprises a feature extraction and tuning block and a mapping attention unit. Experimental results show that this strategy is better than current methods, especially when the degradation is unknown.

The research [9] compares Pix2Pix and CycleGAN in general adversarial networks for image-to-image translation. Pix2Pix is more efficient than CycleGAN, which requires fewer resources and training time while producing higher-quality images and similarity if measured by Frechet Inception Distance for performance evaluation. The authors in research methodology [10] detailed utilizing a deep learning technique that employs a modified U-Net in conjunction with a GAN framework from high-resolution satellite photos increases the capability for road segmentation. Although it suffers from issues related to the continuous extraction of roads and complicated terrain, this approach drastically increases segmentation precision and F1 scores to potentially enhance topological accuracy in the future.

Several research gaps are pointed out after conducting the literature survey. This includes problems in extracting continuous road parts from satellite imagery, the inefficiencies in the image-to-image translation task, high resource, and training time requirements, and limitations in processing complex and multiscale image sections. These challenges are prevalent in both traditional techniques and some of the modern deep learning approaches, especially in producing smooth and accurate translations. The model, described in the research study, is likely to fill these gaps efficiently using conditional GANs, which will translate the satellite photos into maps, such as Google Maps.

The conditional GANs increase the accuracy of image translation, keeping the nuances of the resultant maps and their important features, thus making it possible to condition the output for specific properties of input. They can easily deal with complicated sceneries and differences in road sections better than traditional approaches. By harnessing the capabilities of GANs to produce quality pictures and learn hierarchical features, the model can provide more accurate and resource-efficient solutions. This will finally give way to smooth and accurate map translations from satellite data.

III. METHODOLOGY

The methodology gives more information on how this approach to map generation from satellite images is realized using the Pix2Pix Maps dataset. We propose a novel use of a U-Net generator in conjunction with the Conditional GAN architecture, which leverages a family of cutting-edge deep learning algorithms for accurate image-to-image translation. Combining generative adversarial networks with the accurate localization from the U-Net architecture results in a methodology that ensures a robust solution to this difficult task.

A. Dataset Description

The Pix2Pix dataset was utilized as the primary dataset for this study. However, only a subset of the Pix2Pix dataset, consisting of the maps section, was employed in this work.

The dataset contains 1096 training images and 1098 validation images in the form of JPG images; each pair of images includes an input satellite image and a target map as illustrated in Fig. 1. These satellite photos depict a variety of locations and terrains, from urban to natural areas, that help the generator learn to generate these images. The target images show different styles of geographical views, such as street maps, aerial views, and land-use maps, which serve as ground-truth labels for satellite images. This supervised setting therefore enables robust model training and evaluation, since the generated maps will have to have a good faithfulness and accuracy level.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue XI Nov 2024- Available at www.ijraset.com

Fig. 1. Pair of Satellite Image and the target map.

B. Data Loading and Preprocessing

The first step in our methodology is to prepare our dataset for training our map generation model. A custom data loading function is implemented for efficient loading and preprocessing of the data. This uses a module that can recursively look for image files in a specified directory path. A trimming parameter is also supported, which limits the number of loaded images. It is useful when one would wish to conduct smaller-scale experiments or in the process of debugging. Images are resized to size 256 × 256 pixels and further normalized to the range [0, 1]. In addition, utility functions are implemented to visualize input images and their corresponding masks for better insights into how the data looks.

After loading and preprocessing the dataset, it is converted into a TensorFlow Dataset object using a method that creates slices from tensors. Further, this dataset is shuffled, batched, and prefetched for optimized data loading in the training phase of the model. The batch size selected is 32, balancing computational efficiency and model convergence.

C. Model Architecture

This section details the architectural design of our model for map generation, which forms the crux of our approach. Our model architecture has been carefully laid out to convert input images of a satellite into corresponding map images with high precision and detail. We have incorporated the techniques of a conditional GAN architecture and an intricate structure of the U-Net architecture.

The GANs, first proposed by Goodfellow et al. in 2014, have proven to be a method that generates images indistinguishable from real images from random noise. The conditional variant of the GANs uses this capability to condition the output of the generator on extra information—say, class labels or input images. This allows accurate image-to-image translation tasks, where the generator gains the ability to associate target images in the destination domain with input images from the source domain. Our model leverages the discriminative power of a conditional GAN architecture to ensure that generated maps are very close to real map images.

Besides the GAN framework, we use the U-Net architecture that Ronneberger et al. (2015) suggested. U-Net is renowned for its capacity to successfully capture contextual information while preserving fine-grained features. The architecture is symmetrical and has an encoder-decoder structure utilizing skip connections between the appropriate layers of the encoder and decoder. Skip connections are there to pass high-resolution information, which allows the model to keep spatial details during the upsampling process.

1) U-Net Architecture

The U-Net architecture was proposed by Olaf Ronneberger, Philipp Fischer, and Thomas Brox in their paper titled \"U-Net: Convolutional Networks for Biomedical Image Segmentation.\" It is a new approach in semantic segmentation tasks. Motivated by the necessity for precise and efficient segmentation of biomedical images, which generally poses a problem with traditional methods in terms of a limited number of labeled data and intricate image structures. The basic architecture of U-Net as shown in Fig. 2. includes:

i) Contracting Pathway (Encoder): Motivated by conventional CNNs, the network starts with a convolutional layer followed by ReLU activation and then batch normalization. It enables the hierarchical feature extraction with the subsampling performed using max-pooling.

ii) Bottleneck: The bottleneck layer serves as a repository of high-level features that are critical to semantic understanding. This is a compressed representation for efficient feature extraction and context aggregation.

iii) Expanding Pathway (Decoder): The transposed convolutions are used for upsampling, followed by reconstructing the segmented image, thus preserving the fine details with the help of skip connections. This is used to concatenate the feature maps of the contracting pathway for spatial context during upsampling.

iv) Output Layer: The segmentation mask is output by the convolutional layer, which is the final layer, which consists of a sigmoid or SoftMax activation function for pixel-wise probabilities in order to achieve precise segmentation.

It is the skip connections that allow the information to flow across different scales, improving the accuracy of localization. U-Net, with its architectural simplicity, flexibility in operating on limited data, and attentiveness to detail, has become the primary choice for segmentation tasks in a variety of applications, including cloud image segmentation. The specified loss function, typically cross-entropy, and the Adam optimizer are used for training the model while monitoring the accuracy metrics over multiple epochs for enhanced segmentation capabilities.

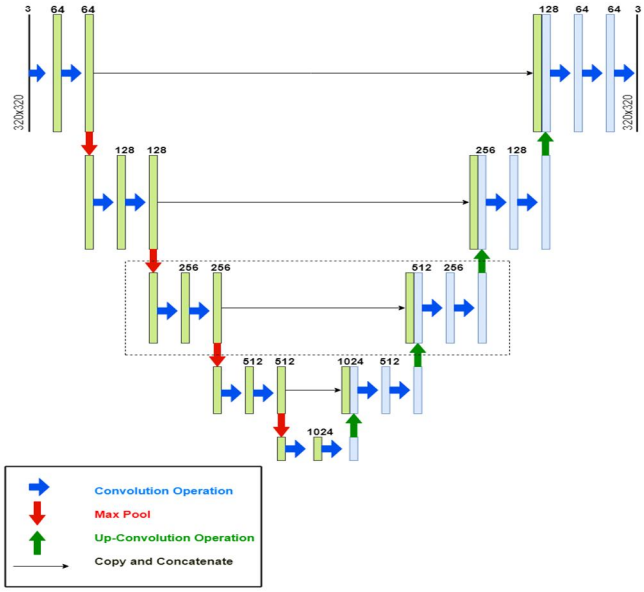


Fig. 2. Architecture of U-Net Model

Applying U-Net principles in our generator model increases its capacity to perform the duty of image-to-image translation with high accuracy, particularly in the domain of map generation from satellite images. In addition, the symmetric encoder-decoder structure with skip connections from U-Net allows efficient spatial information preservation at different scales. The encoder path extracts feature hierarchically by means of convolutional layers followed by downsampling operations. Decoder path reconstruction of the full-resolution output image comes from upsampling layers. Most importantly, skip connections allow combining features from low and high levels, hence enable the fine-grained features to be preserved at reconstruction. The integration of U-Net architecture makes our generator model perfect in its operation of producing quality maps from satellite images; it captures the intrinsic spatial relationships—a fact that makes it excellent for image translation tasks.

2) Downsampling and Upsampling

Firstly, we proceed with a downsampling operation, as implemented by the downsample function, to allow feature extraction and dimensionality reduction. The function uses strided convolutions in its convolutional layers to reduce the spatial dimensions of the input feature maps while growing the feature space in depth. Optional batch normalization is performed to stabilize and accelerate the training process, and the LeakyReLU activation function introduces non-linearity. The downsampling operation is very crucial in capturing hierarchical features from the input images and thereby enabling the model to learn meaningful representations for map generation.

The downsampling operation can be defined as follows:

a) Convolutional Layer: Applies a 2D convolution operation with specified filters and kernel size, with strides set to 2 to reduce spatial dimensions by half. The padding is set to 'same' to ensure the output has the same dimensions as the input. The convolution operation is given by:

$$Y[p,q,r] = \sum_{m=0}^{h-1}\sum_{n=0}^{w-1}\sum_{c=0}^{C-1} X[p+m,q+n,c].W[m,n,c,r]+b[r]$$

where X is the input feature map, Y is the output feature map, W is the filter kernel, b is the bias term, C is the number of input channels, h and w are the height and width of the filter, and r indexes over the number of filters.

b) Batch Normalization: Normalizes the output produced by the convolutional layer, improving training stability and performance. The batch normalization operation is given by:

$$\hat{x}^{(i)} = \frac{x^{(i)} - \mu^{(i)}}{\sqrt{\sigma^{(i)^2} + \epsilon}}$$

where $\hat{x}^{(i)}$ is the normalized value, $x^{(i)}$ is the input value, $\mu^{(i)}$ and $\sigma^{(i)^2}$ are the mean and variance of the batch for the i-th feature, and ϵ is a small constant for numerical stability.

c) LeakyReLU Activation: Introduces non-linearity with a small slope for negative inputs, preventing the dead neuron problem. The LeakyReLU activation function is defined as:

$$LeakyReLU(x) = \begin{cases} x, & if\ x > 0 \\ \alpha.x, & otherwise \end{cases}$$

where α is a small positive slope coefficient, typically set to a small value such as 0.01. This function introduces a small gradient for negative input values, allowing a small, non-zero gradient flow during backpropagation and addressing the vanishing gradient problem.

After downsampling, where we decrease spatial dimensions of the input and capture essential features at different levels of abstraction, comes the next step: reconstruction of the image details using upsampling layers. Upsampling layers play a pivotal role in bringing back the spatial details lost in the process of downsampling. They use the incorporation of batch normalization, optional dropout, transpose convolutional layers, and ReLU activation functions to increase the spatial dimensions while fine-tuning the feature maps. The upsampling function is defined as follows:

d) Transpose Convolutional Layer: This layer performs an inverse convolution operation, which increases the spatial dimensions of the input feature maps. The transpose convolution operation is mathematically given by:

$$Y[p,q,r] = \sum_{m=0}^{h-1}\sum_{n=0}^{w-1}\sum_{c=0}^{C-1} X[\lfloor {}^{p}/_{S} \rfloor+m,\lfloor {}^{q}/_{S} \rfloor+n,c].w[m,n,c,r]+b[r]$$

where X is the input feature map, Y is the output feature map, W is the filter kernel, b is the bias term, C is the number of input channels, h and w are the height and width of the filter, and r indexes over the number of filters.

e) Batch Normalization: Similar to the one mentioned and explained in downsampling.

f) Dropout: This layer is optionally included to assist prevent overfitting by randomly setting a portion of the input units to 0 during training. The dropout rate used in our study is 0.5.

g) ReLU Activation: This layer introduces non-linearity to the model, defined by the function:

$$ReLU(x) = \max(0, x)$$

3) Generator Model

Moving forward to the generator model, it lays the very foundation for our image-to-image translation framework, specifically engineered to produce quality map images given satellite images. It borrows deeply from the architecture of the U-Net and is divided into downsampling and upsampling paths, both of which are designed to capture subtle features without losing spatial information.

The downsampling path takes a set of convolutional layers followed by downsampling operations, effectively extracting higher-level features from the input satellite images. Next, the upsampling path very carefully restores spatial dimensions while incorporating fine details from the downsampling path via skip connections. These connections enable information from different abstraction levels to smoothly flow across the network, achieving faithful reconstruction of map images. More importantly, the architecture is engineered carefully to balance feature extraction and spatial fidelity so as to generate realistic and informative map images. Advanced techniques, such as batch normalization and dropout, further enhance the model stability and generalization performance, consequently improving the quality of generated outputs.

The generator architecture is as follows:

a) Input Layer: The model takes a satellite image of size 256×256×3 as input.

b) Downsampling Path: This path includes a few convolutional layers followed by downsampling. This way, it learns to capture higher-level characteristics of the input image, decreasing the spatial dimensions.
- The first layer comprises the convolutional operation using 64 filters with a kernel size of 4. Subsequent layers repeat the process with changed filter sizes to capture more abstract features.
- The final layer of the downsampling path utilizes a convolutional operation with 980 filters and a kernel size of 4, encapsulating high-level abstractions derived from the input image.

c) Upsampling Path: The upsampling path is a mirror of the downsampling path and is designed to regain the spatial dimensions of the image while keeping the features. This, too consists of a sequence of convolutional layers followed by upsampling operations.
- Every layer in the upsampling path is related to a particular layer in the downsampling path; hence, spatial details are preserved with the help of skip connections.

d) Skip Connections: These connections are an integral part of the U-Net architecture. They connect layers in the downsampling path directly to corresponding layers in the upsampling path, thus allowing the fine-grained details from the input to flow to the output.

e) Output Layer: The final layer in the generator uses a transpose convolutional operation to produce an output of the similar dimensions as the input image. The activation function used is *tanh*, ensuring the output pixel values belong in the range of [−1,1].

Skip connections provide the generator model with indispensable conduits for the easy amalgamation of intricate details, mitigating information loss through upsampling. Through direct links between corresponding layers of downsampling and upsampling paths, the latter bypasses the vanishing gradient predicament of the deeper neural networks to ensure a more effective backflow of gradients through the training process. Mathematically expressed as

$$Skip_{Connection(X,Y)} = X + Y$$

where Y represents output of a given layer, X is an input and skip connections combine low-level characteristics from the input image with high-level abstractions from the downsampling path. Such an amalgamation empowers the generator to capture fine-grained nuances and intricate spatial relationships, hence reconstructions more faithful and more realistic with respect to map images from satellite data.

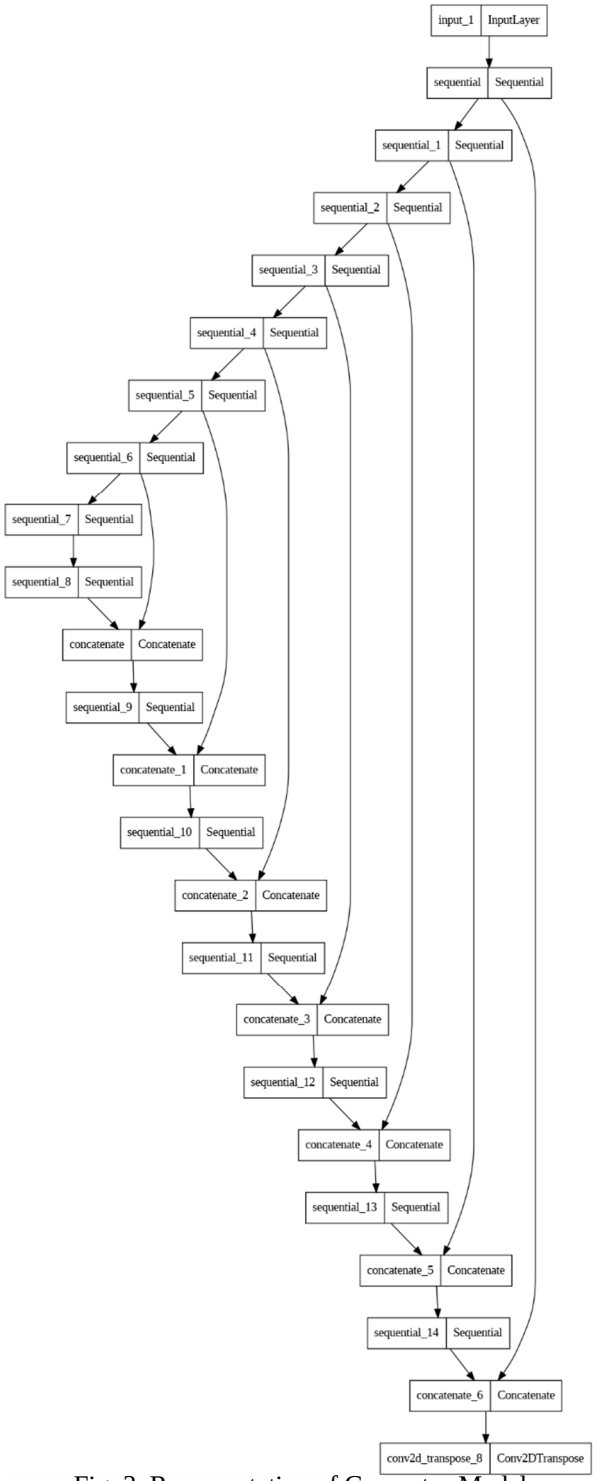


Fig. 3. Representation of Generator Model

4) Discriminator Model

The discriminator model's role is to discriminate between real and generated, in other words, fake images. It is a major component of the Conditional GAN architecture; it acts as a feedback module, helping the generator in enhancing the quality of generated images. The inputs to our model are a pair of images—that is, the real satellite image and its corresponding target map image, or the real satellite image and the generated map image—whose output is a probability indicating whether the pair of inputs fed is real or fake.

The discriminator model is developed in the following manner:

a) Initialization:
- The initialization of the weights for convolutional layers is done with a random normal initializer that has a 0.02 standard deviation and a mean of 0.

b) Input Layers:
- The discriminator model takes two inputs: inp (the input image, e.g., a satellite image) and tar (the target image, e.g., a map image).
- Both the input and target images are of shape (256, 256, 3), which is an RGB image with dimensions 256x256.

c) Concatenation:
- Along the channel axis, the input and target images are concatenated to form a single input tensor of shape (256, 256, 6). This concatenated tensor is a pair of images that the discriminator will decide upon.

d) Downsampling Layers:
- The model applies a series of downsampling operations using the downsample function. This function performs the following operations:
  - Convolution: 2D convolution with the number of filters and the kernel size specified. Strides are (2, 2) to perform downsampling, and padding=same to keep the spatial dimensions.
  - Batch Normalization: The convolutional layer output is normalized to have mean equal to zero and unit variance to speed up and stabilize training (applied if batch_norm is True).
  - Activation: Uses the LeakyReLU activation function to introduce non-linearity.
- The sequence of downsampling layers is as follows:
  - First layer: 64 filters, kernel size 4, no batch normalization.
  - Second layer: 128 filters, kernel size 4.
  - Third layer: 128 filters, kernel size 4.
  - Fourth layer: 256 filters, kernel size 4.

e) Intermediate Convolution Layer:
- Next, an intermediate convolutional layer with 512 filters and a kernel size of 4 is applied by the network. This convolutional layer has a stride of 1, which is followed by batch normalisation and an activation function called LeakyReLU.
- This convolutional layer applies zero padding both before and after it, to keep the spatial dimensions.

f) Output Layer:
- The discriminator network has an output layer that is a convolutional layer with 1 filter and kernel size 4. The output of this layer is a feature map with 1 channel, where each entry represents the probability of the corresponding patch in the input being real or fake.

g) Model Compilation:
- The discriminator model is compiled using the Keras Model class, with the concatenated input and target images as inputs, and the output probability map as the output.

By constructing the discriminator in this way, the network gains the ability to discriminate between created and actual images, and during training, gives the generator feedback. The input and target images are concatenated so that the discriminator can evaluate how well they agree with each other. In this way, the generator is not only asked to produce a plausible map given a satellite image, but also to produce a map that the discriminator thinks came from the real distribution. This step provides insightful observations about the model's functionality and helps gauge its reliability in real-world cloud segmentation scenarios. The system architecture is presented in Fig. 4 below.

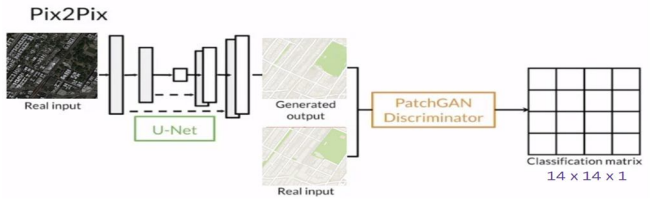


Fig. 4 System Architecture

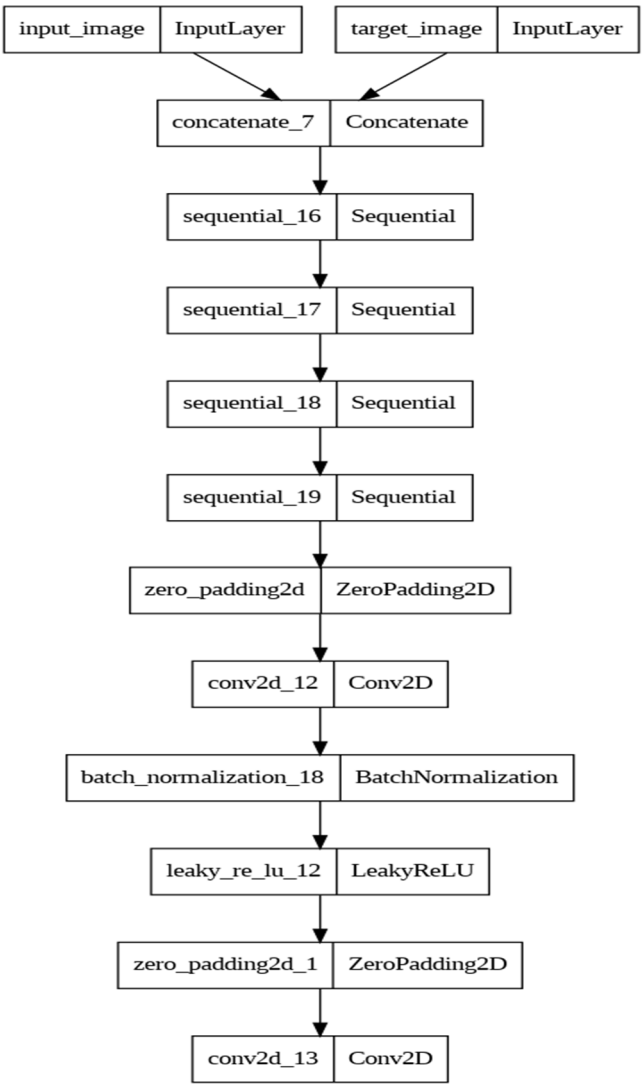


Fig. 5. Representation of Discriminator Model

5) Loss Functions and Optimizations

The training of our GAN involves the definition of the appropriate loss functions for the generator and discriminator, respectively, and choice of suitable optimization algorithms. In the following section, we outline the implementation of the loss functions and optimization.

Loss Functions

a) Binary Cross-Entropy Loss:

The binary cross-entropy loss can be seen to measure the quality of the discriminator in terms of distinguishing real from generated images, and the quality of the generator in terms of being able to successfully 'fool' the discriminator. Mathematically, binary cross-entropy loss for a single sample is defined as:

$$L(y, \hat{y}) = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y}))$$

Here, y is the true label, and $\hat{y}$ is the predicted label.

b) Generator Loss:

The generator loss is composed of two parts:

i.) Adversarial Loss: This motivates the generator model to produce images that the discriminator classifies as real. It is computed as:

$$L_{GAN} = -\frac{1}{N}\sum_{i=1}^{N} \log(D(G(z_i))$$

where G is the generator, D is the discriminator, and z is the input noise vector.

ii.) L1 Loss (Mean Absolute Error): This guarantees that, in terms of pixel values, the created photos are similar to the target images. It is computed as:

$$L_{L1} = \frac{1}{N}\sum_{i=1}^{N} \|y_i - G(x_i)\|_1$$

iii.) Total Generator Loss: The combined amount of the adversarial loss and the L1 loss is the overall generator loss scaled by a factor λ:

$$L_G = L_{GAN} + \lambda L_{L1}$$

where λ is a weighting factor (typically set to 100).

c) Discriminator Loss

The discriminator loss is also composed of two parts:

i.) Real Loss: This part of the loss evaluates the discriminator's ability to correctly identify real photos. It is computed using the binary cross-entropy loss between a tensor of ones and the discriminator's output for real images. It is computed as:

$$L_{real} = \frac{1}{N}\sum_{i=1}^{N} \log(D(y_i))$$

ii.) Generated Loss: This part of the loss measures how well the discriminator can classify generated images as fake. It is computed using the binary cross-entropy loss between a tensor of zeros and the discriminator's output for generated images. It is computed as:

$$L_{fake} = \frac{1}{N}\sum_{i=1}^{N} \log(1 - D(G(z_i))$$

iii.) Total Discriminator Loss: The total loss of the discriminator is the sum of the generated loss and the real loss:

$$L_{Total} = L_{real} + L_{fake}$$

In these equations:

- $D(y_i)$ is the discriminator's estimation for the original image y_i.
- $D(G(z_i)$ is the discriminator's estimation for the generated image $G(z_i)$.
- N is the number of samples.

d) Optimization

Both the generator and the discriminator are optimized using the Adam optimizer. This algorithm is chosen for its adaptive learning rate and momentum properties, which help in faster convergence and stable training. The learning rate α is set to 2×10^{-4} , and the first moment decay rate (β_1) is set to 0.5

The update rules for the parameters θ of the generator and discriminator are as follows:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\nabla_\theta L_t$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)(\nabla_\theta L_t)^2$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

In the given equations:

- The value m_t is the mean or first moment of the gradients of the loss function L_t concerning the model's parameters θ at a given time *t*.
- v_t represents the second moment (the uncentered variance) regard to the model's parameters θ at time t, of the gradients of the loss function L_t.
- β_1 and β_2 are constants that govern the moment estimates' exponential decay rates. They typically have values close to 1.
- $\nabla_\theta L_t$ represents the gradient of the loss function L_t with respect to the parameters θ of the model at time *t*.
- $\hat{m}_t$ and $\hat{v}_t$are estimates of the first and second moments, respectively, that have been corrected for bias.
- α is the learning rate, which controls the step size in the parameter update.
- ϵ is a small constant (typically a very small value) added to the denominator for numerical stability.

These equations describe the Adam optimization algorithm, which is commonly used for training neural networks. It adapts the learning rates for each parameter based on their first and second moments of the gradients, allowing for more effective and efficient optimization.

6) Model Training

Finally, the training process for the model involves arranging the iterative updates to the parameters through a custom fitting function and the model's predictions using a visualization function. The fitting function iterates over 100 epochs, where a single epoch involves processing batches of the training dataset. Inside this loop, a training step function is called, which is responsible for executing one training iteration. It uses TensorFlow's Autograph feature, which allows the decorator to make the code more computation-efficient.

During a training step, input images and their corresponding target masks change their data type for compatibility with the operations to be performed. It computes gradients in the context of gradient tapes, allowing the operations to be traced for both the generator and discriminator models. Then, the generator's output is evaluated by the discriminator to compute adversarial loss, GAN loss and the mean absolute error, L1 loss of the generated output compared to the target image for generator loss computation.

Simultaneously, the discriminator evaluates its loss, distinguishing between real and generated images that contribute to the overall optimization. The gradients of generator and discriminator losses are computed with respect to their trainable variables, making it possible to update the parameters. Finally, these gradients are applied through an optimization algorithm to iteratively refine the model's performance in successive epochs.

IV. RESULTS AND DISCUSSIONS

This section deals with the results by examination, using a wide range of evaluation parameters. The translation from the satellite images into representations like that of Google Maps achieves great potential in enhancing remote sensing and geographic information systems. In order to assess the model's effectiveness, we used a variety of measures, such as MAE (Mean Absolute Error), PSNR (Peak Signal-to-Noise Ratio), and SSIM (Structural Similarity Index Measure), among many others. A strong base is provided by these criteria for the visual integrity evaluation, noise reduction, and overall correctness of the developed maps. The following sections examine in greater detail the findings for these tests and allow for comment on model efficacy and possible areas for further development.

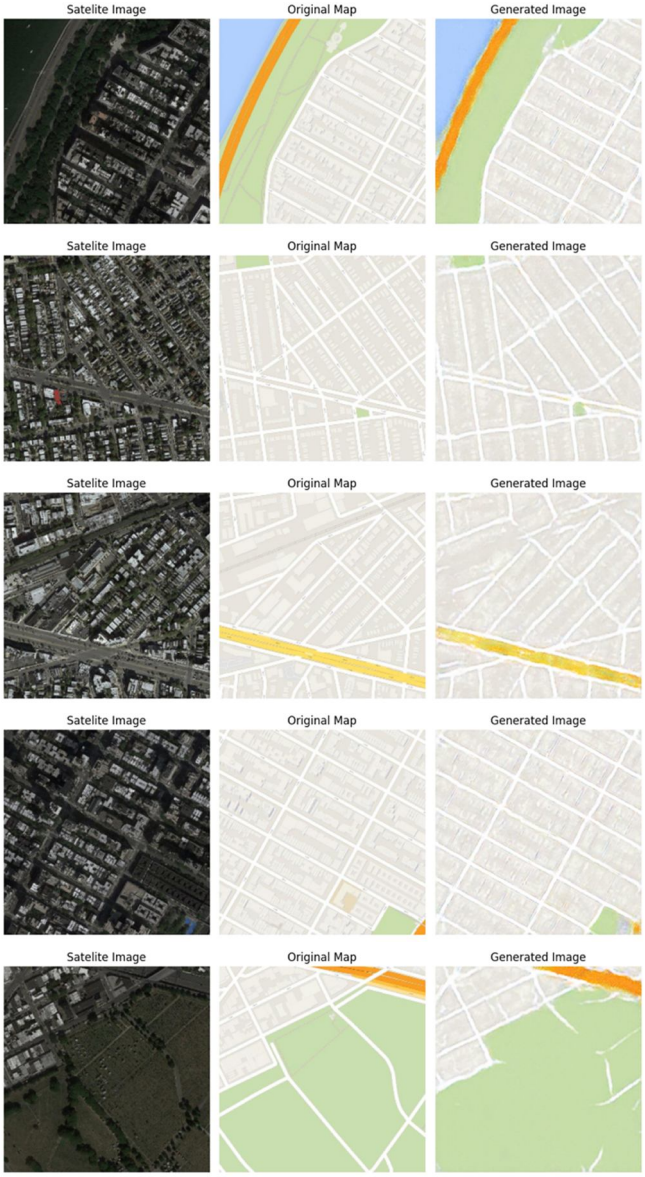


Fig. 6. Results

A. SSIM

Structural Similarity Index Measure. Using a range of established characteristics of the human visual system, more conventional, objective methods for assessing the quality of a perceived image have tried to quantify errors' visibility—the variations between a distorted and a reference image. Assuming that structural information can be extracted from a scene by human vision, we present a different supplementary paradigm for evaluating quality that is based on how structural information decreases.

Most image quality evaluation methods focus on the difference between a reference and a sample picture by measuring errors. Popular measures are differences in values of each corresponding pixel between reference and sample pictures, for example, Mean Squared Error. The visual perception system in humans is quite good at recognizing structural details from a scene, which allows it to distinguish between details taken from an example scene and a source. Because of this, a measure that mimics this behaviour will work better on tasks requiring the ability to distinguish between a reference picture and a sample. Three essential elements are extracted from a picture via the Structural Similarity Index (SSIM) metric: Structure, Contrast, and Luminance. These three qualities serve as the foundation for the comparison of the two images.

SSIM is calculated through contrasting locally distributed patterns of normalised brightness and contrast pixel intensities. The formula for SSIM between two windows x and y of common size $N \times N$ is:

$$\text{SSIM}(x, y) = ((2\mu_x\mu_y+C_1)(2\sigma_{xy}+C_2))/((\mu_x^2+\mu_y^2+C_1)(\sigma_x^2+\sigma_y^2+C_2))$$

Where:

- μ_x and μ_y are the average pixel values of windows x and y.
- σ_x^2 and σ_y^2 are the variances of x and y.
- σ_{xy} is the covariance of x and y.
- C_1 and C_2 are small constants to stabilize the division with a weak denominator.

The SSIM score of **0.6829** is indicative of a model that performs well in generating images with a good degree of similarity to the ground truth. While it demonstrates that the model is capturing important structural elements effectively, it also highlights that further tuning and improvements could enhance the level of how well-made the resulting photos are, potentially pushing the SSIM score closer to 1.

B. Peak Signal-to-Noise Ratio (PSNR)

PSNR is one measurement to determine how well the reconstructed pictures compare with the original pictures. Generally, it will be used to determine how well picture compression and reconstruction methods work. PSNR increases as the level of detail in the recreated image is better. PSNR is defined as:

$$\text{PSNR} = 20 \cdot \log_{10}(\text{MAX\_I} / \sqrt{MSE})$$

Where:

- MAX_I denotes the maximum pixel value of the image (for 8-bit images, this value is 255).
- MSE (Mean Squared Error) is the average of the squared differences between corresponding pixels of the two images.

The PSNR score of 31.3 dB suggests that the predicted images are of good quality and closely resemble the expected images. The use of PSNR as a metric provides a quantitative way to evaluate the performance of the GAN in generating high-quality images. Generally, a PSNR value above 30 dB is considered to indicate high-quality reconstruction.

C. Modified Inception Score (MIS) with KL Divergence

One popular statistic for measuring the quality and diversity of images generated by generative models is the Inception Score. Here, the version that has been changed uses the Kullback-Leibler divergence as a measure of how close conditional and marginal distributions are to each other. The marginal distribution would be the distribution of the class labels in actual photographs, or the probability over the appearance of certain classes in the real image collection. It models the class label distribution given the produced images, which is a conditional probability distribution that defines the probability of different classes given the prediction of the generative model. The value of **0.87783302** for this metric is a relatively high score, indicating that the generated images are of good quality and have a class distribution that is quite similar to the real images. In the context of the valid range of 0 to 1, where 1 represents perfect similarity, a score of 0.877 suggests that the generative model is performing well, producing images that closely match the real distribution.

D. Image Sharpness Difference

A relative metric known as image sharpness is used to describe how clear an image is. Sharpness may also be used as a metric to assess the visual quality of images generated by generative models in comparison to actual photographs. The image's Laplacian variance can be used to measure sharpness. The picture gets clearer the greater the variance. Sharpness difference represents the absolute difference in average sharpness between produced and actual pictures. A small sharpness difference indicates that the generative model is producing high-quality photos because the produced image clarity is about the same as the actual images.

A sharpness difference of 65.96148021536467 mean that the images generated and the real images have a slight difference. A difference in sharpness of about 66 means that the generated images are distinct from the real images but not different. The value is not close to zero, so the clarity and details of the produced images deviate a little from the original images. Not quite high, though, meaning the generative model does a great job, and there is room for improvement.

E. FID (Fréchet distance)

It is a metric for comparing the similarity of two image datasets. It is most frequently used to assess the quality of Generative Adversarial Network sample data since it has been demonstrated to correlate favorably with human judgement of visual quality. The location and arrangement of the points along curves are taken into consideration while calculating the Fréchet distance, a metric used in mathematics to compare curves. The value of **111.054** is a decent score for FID.

V. CONCLUSION

In this study, a GAN-based model is developed that can convert satellite images into a map-like representation; it is an architecture based on the Pix2Pix framework. The approach included the intense training process with a carefully curated dataset of paired satellite and map images. Also, the model consists of elaborately designed generator and discriminator to optimize performance by providing state-of-the-art upsampling and downsampling layers with skip connections for preserving spatial information.

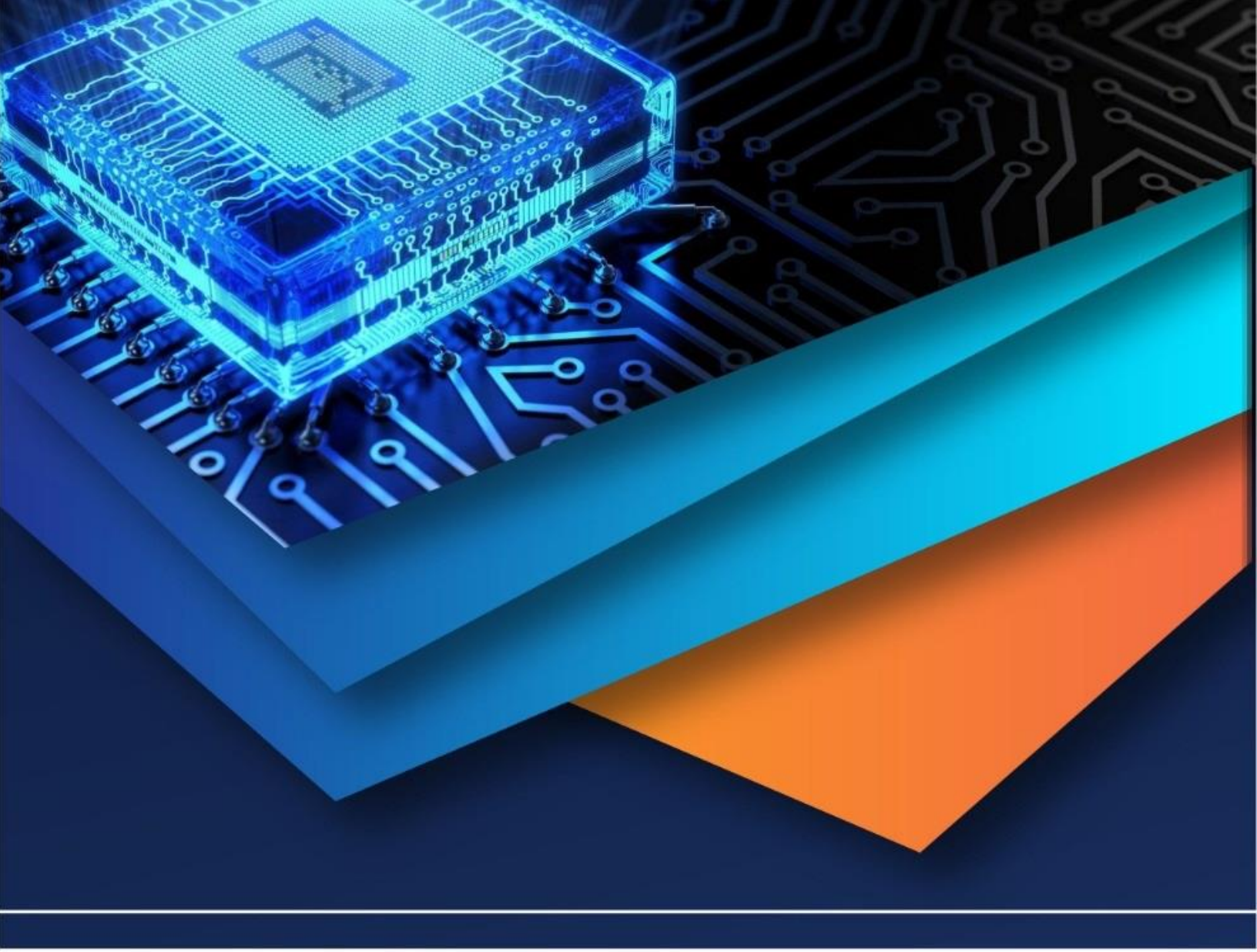
The model demonstrated an average SSIM score of 0.6829, an average PSNR of 31.30 dB, which indicates that the generated maps are of high similarity to real map images in terms of structural similarity and fidelity. Besides, the difference in sharpness, 65.96, calculated, brings to light the competence of the model in maintaining image clarity; however, there is scope for further enhancement.

One can use larger and more varied datasets to make the model more generalizable and improved in performance. It could achieve better results in sharpness and detail preservation by using the other architectures of GANs, such as StyleGAN and CycleGAN. With further tuning of hyperparameters and integration of post-processing techniques, such as sharpening, it is possible to lower the sharpness difference and improve the visual quality of the generated images. Such tailoring with respect to specific applications, such as urban planning and environmental monitoring, by the integration of domain-specific constraints and features, should be able to improve its practical utility.

In conclusion, this research presents a significant step into the automation of generating map-like images from satellite data, hence providing a foundation for future advancements in this area. The model proposed here, with a bit more refinement and adaptation, has the potential to become very useful in various geospatial applications.

REFERENCES

[1] Kamil, Anwar, and Talal Shaikh. "Literature Review of Generative models for Image-to-Image translation problems." In 2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), pp. 340-345. IEEE, 2019.
[2] Ganguli, Swetava, Pedro Garzon, and Noa Glaser. "GeoGAN: A conditional GAN with reconstruction and style loss to generate standard layer of maps from satellite images." arXiv preprint arXiv:1902.05611 (2019).
[3] Zhang, Ying, Yifang Yin, Roger Zimmermann, Guanfeng Wang, Jagannadan Varadarajan, and See-Kiong Ng. "An enhanced gan model for automatic satellite-to-map image conversion." IEEE Access 8 (2020): 176704-176716.
[4] Song, Jieqiong, Jun Li, Hao Chen, and Jiangjiang Wu. "MapGen-GAN: A fast translator for remote sensing image to map via unsupervised adversarial learning." IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 14 (2021): 2341-2357.
[5] Parekhji, Arnav, Mansi Pandya, and Pratik Kanani. "Comparing GANs for translating satellite images to maps." In 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), pp. 1267-1274. IEEE, 2021.
[6] Chen, Xu, Bangguo Yin, Songqiang Chen, Haifeng Li, and Tian Xu. "Generating multiscale maps from satellite images via series generative adversarial networks." IEEE Geoscience and Remote Sensing Letters 19 (2021): 1-5.
[7] Kshatriya, Balaram Singh, Shiv Ram Dubey, Himangshu Sarma, Kunal Chaudhary, Meva Ram Gurjar, Rahul Rai, and Sunny Manchanda. "Semantic Map Injected GAN Training for Image-to-Image Translation." In Proceedings of the Satellite Workshops of ICVGIP 2021, pp. 235-249. Singapore: Springer Nature Singapore, 2022.
[8] Kui, Zhongyuan Wang, Peng Yi, Junjun Jiang, Guangcheng Wang, Zhen Han, and Tao Lu. "GAN-based multi-level mapping network for satellite imagery super-resolution." In 2019 IEEE International Conference on Multimedia and Expo (ICME), pp. 526-531. IEEE, 2019.
[9] Parekhji, Arnav, Mansi Pandya, and Pratik Kanani. "Comparing GANs for translating satellite images to maps." In 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), pp. 1267-1274. IEEE, 2021.
[10] Abdollahi, Abolfazl, Biswajeet Pradhan, Gaurav Sharma, Khairul Nizam Abdul Maulud, and Abdullah Alamri. "Improving road semantic segmentation using generative adversarial network." IEEE Access 9 (2021): 64381-64392.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY