Computer Vision and Image Processing

22AIE313

A Project Report

Submitted by

A.Venkata satya-CB.EN.U4AIE22005

K N Lakshmi-CB.EN.U4AIE22023

K Hemavardhan Reddy-CB.EN.U4AIE22026

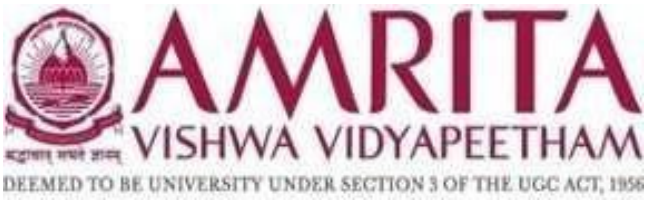
K V Vamshidhar Reddy- CB.EN.U4AIE22028

in partial fulfillment for the award of the degree of

BACHELOR OF

TECHNOLOGY IN

CSE(AIE)



Centre for Computational Engineering and Networking

AMRITA SCHOOL OF ARTIFICIAL INTELLIGENCE

AMRITA VISHWA VIDYAPEETHAM

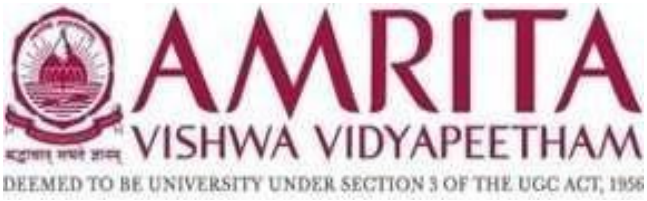
COIMBATORE - 641112 (INDIA)

APRIL – 2025

AMRITA SCHOOL OF ARTIFICIAL INTELLIGENCE

AMRITA VISHWA VIDYAPEETHAM

COIMBATORE - 641112



BONAFIDE CERTIFICATE

This is to certify that the project entitled "Satellite Imagery Super-Resolution using GAN" submitted by A.Venkata Satya (CB.EN.U4AIE22005) , K N Lakshmi (CB.EN.U4AIE22023), K Hemavardhan Reddy (CB.EN.U4AIE22026), K V Vamshidhar Reddy (CB.EN.U4AIE22028) to Dr Sajith Variyar V V , for the award of the Degree of Bachelor of Technology in the "CSE(AI)" is a bonafide record of the work carried out by her under our guidance and supervision at Amrita School of Artificial Intelligence, Coimbatore.

Dr Sajith Variyar V V
Project Guide

Dr. K.P. Soman
Professor and Head CEN

Submitted for the university examination held on 08/04/25.

AMRITA SCHOOL OF ARTIFICIAL INTELLIGENCE

AMRITA VISHWA VIDYAPEETHAM

COIMBATORE - 641112

DECLARATION

We, A.Venkata Satya- (CB.EN.U4AIE22005) , K N Lakshmi-(CB.EN.U4AIE22023), K Hemavardhan Reddy-(CB.EN.U4AIE22026), K V Vamshidhar Reddy - (CB.EN.U4AIE22028) hereby declare that this project entitled "Satellite Imagery Super-Resolution using GAN", is the record of the original work done by us under the guidance of Dr Sajith Variyar V V, Professor, Amrita School of Artificial Intelligence, Coimbatore. To the best of our knowledge, this work has not formed the basis for the award of any degree/diploma/ associate ship/fellowship/or a similar award to any candidate in any University.

Place: Coimbatore

Date: 08-04-2025 Signature of the Student

ACKNOWLEDGMENT

We would like to express our special thanks of gratitude to our teacher (Dr Sajith Variyar V V), who gave us the golden opportunity to do this wonderful project on the topic, "Satellite Imagery Super-Resolution using GAN", which also helped us in doing a lot of research and we came to know about so many new things. We are thankful for the opportunity given. We would also like to thank our group members, as without their cooperation, we would not have been able to complete the project within the prescribed time.

Contents

List of Figures:

List of Tables:

ABSTRACT

This project presents a deep learning-based approach for enhancing satellite image resolution using a Super-Resolution Generative Adversarial Network (SRGAN). Due to hardware and cost constraints, satellite images are often captured at low spatial resolutions, limiting their effectiveness in critical applications. To address this, we implement an SRGAN model trained on the DOTA-v1.0 dataset, using over 62,000 paired patches derived from 469 high-resolution images. The generator employs residual blocks and PixelShuffle layers to upscale low-resolution inputs by a factor of four, while the discriminator distinguishes real from generated outputs. The model is optimized using a combination of content loss (MSE), adversarial loss (BCE), and perceptual loss (VGG19-based). Training and evaluation were conducted on both seen and unseen data using PSNR and SSIM as metrics. Results demonstrate improved visual fidelity and structural accuracy, confirming the model's effectiveness. The proposed method offers a scalable, cost-efficient alternative to acquiring high-resolution satellite imagery.

PROBLEM STATEMENT

Satellite imagery is essential in domains such as environmental monitoring, defence surveillance, and urban planning, but the quality of these images is often limited by low spatial resolution. Due to constraints in satellite sensor hardware, imaging altitude, and high costs, acquiring high-resolution imagery remains a challenge. Low-resolution images result in the loss of critical details, making it difficult to identify small-scale objects and patterns. This limitation negatively impacts the performance of downstream applications such as object detection and scene interpretation. Therefore, there is a growing need for a deep learning-based solution capable of enhancing the resolution of satellite images in a cost-effective and scalable manner, without requiring additional hardware or reimaging.

CHALLENGES

1) Techniques like bicubic or bilinear interpolation are commonly used for upscaling images but often fail to restore fine-grained textures and edge information. This results in visually smooth but structurally unrealistic images, especially in complex satellite scenes with fine boundaries or dense object arrangements.
2) Many deep learning-based super-resolution models are trained and evaluated on datasets like DIV2K or Set5, which consist of ground-level, natural scene images. These models generalize poorly to satellite data, which contains unique characteristics such as varied scales, rotated objects, and non-uniform lighting conditions.
3) Satellite images, particularly from datasets like DOTA, include small, densely packed, and variably oriented objects (e.g., ships, vehicles, containers). General-

purpose models often struggle to preserve these intricate details due to their limited capacity to model such complexity.

4) High-resolution satellite imagery is often commercially licensed and expensive, making it difficult to gather large volumes of HR data for training. This limits the scalability of traditional supervised SR approaches and highlights the need for models that can learn efficiently from limited data using patch-based or augmented training strategies.
5) Achieving high perceptual quality i.e., the realism often comes at the cost of pixel-level accuracy, and vice versa. Many existing methods fail to strike an effective balance, leading either to overly smooth results or visually unrealistic textures.

PROJECT PROPOSAL

To overcome the limitations of existing super-resolution techniques in satellite imagery, we propose a deep learning-based framework using the Super-Resolution Generative Adversarial Network (SRGAN). The model is designed to enhance the resolution of satellite images by learning the mapping between low-resolution and high-resolution pairs. We utilized the DOTA-v1.0 dataset (469 images from the train set), which contains a variety of complex aerial scenes, and extracted over 83,000 HR–LR patch pairs using a sliding window approach with a patch size of 256×256 for HR and 64×64 for LR. Patches were generated using bicubic downsampling and stride-based extraction, ensuring comprehensive coverage of each image's spatial content.

In this project, we implement and adapt the Super-Resolution Generative Adversarial Network (SRGAN), a state-of-the-art deep learning model for single-image super-resolution. SRGAN consists of two main components i.e., a Generator and a Discriminator. The Generator is designed with 16 residual blocks and Pixel Shuffle-based upsampling layers, enabling it to reconstruct high-resolution images from low-resolution inputs by learning fine texture details. The Discriminator is a deep convolutional network that learns to differentiate real high-resolution images from the ones generated by the Generator, thereby pushing the Generator to produce more realistic outputs.

The training process is guided by a combination of three loss functions i.e., content loss (Mean Squared Error), adversarial loss (Binary Cross-Entropy), and perceptual loss, computed using high-level feature representations extracted from a pre-trained VGG19 network. This combination ensures that the generated images are not only close to the ground truth in terms of pixel values but also preserve perceptual and structural quality.

The training process was carried out on Kaggle Notebooks using dual NVIDIA Tesla T4 GPUs, which provided the necessary computational resources for deep network optimization. Given the model's complexity and the large volume of training data, each epoch took approximately 90 to 105 minutes to complete. The model was trained for 40 epochs, with generator and discriminator losses monitored alongside evaluation metrics such as Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM). The consistent improvement in these metrics over time demonstrated the effectiveness of the SRGAN in reconstructing high-quality satellite images from low-resolution inputs.

INTRODUCTION

Satellite imagery plays a crucial role in a wide range of real-world applications, including environmental monitoring, urban development, defense surveillance, and disaster management. The effectiveness of such applications often relies on the availability of high-resolution images to capture fine-grained spatial details. However, obtaining high-resolution satellite data can be both costly and limited due to constraints in satellite sensor technology, storage capabilities, and licensing policies. As a result, many publicly available satellite datasets are captured at lower resolutions, which limits their utility in critical visual analysis and detection tasks.

To address this challenge, image super-resolution techniques have gained significant attention in the field of remote sensing. These techniques aim to reconstruct high-resolution images from low-resolution counterparts by enhancing visual clarity and preserving structural details. Traditional methods such as bicubic or bilinear interpolation fall short when it comes to restoring texture and fine features. Deep learning, particularly convolutional neural networks and generative adversarial networks (GANs) has shown promising capabilities in overcoming these limitations through data-driven learning.

A GAN consists of two neural networks trained in opposition one is the Generator, which attempts to produce high-resolution images from low-resolution inputs, and other is the Discriminator, which tries to distinguish between real high-resolution images and those generated by the model. This adversarial setup enables the Generator to progressively improve its outputs, producing images that are not only structurally accurate but also

perceptually convincing. In this project, we implement the Super-Resolution Generative Adversarial Network (SRGAN), a deep learning model designed to generate photo-realistic high-resolution images.

LITERATURE REVIEW

TITLE	AUTHOR	DATASET	MODEL DESCRIPTION AND PERFORMANCE	RESEARCH GAPS
Generative Adversarial Networks for the Satellite Data Super Resolution Based on the Transformers with Attention (2023)	Mykola Lavreniuk et al.	Sentinel-2 and Landsat-9 (Kyiv region, June 2022); red, green, blue bands; cloud-free composites	Proposed a GAN model using Hybrid Attention Transformer (HAT) architecture. Generator and discriminator both built using transformer blocks. Input patches were 128x128 and output 384x384. Achieved low MAE values across bands and outperformed classical GAN and pixel-based methods. The MAE values of Red - 0.0844, Green - 0.0437, Blue - 0.0507, All - 0.05965.	Limited performance in urban or densely populated areas. Requires large-scale pretraining and more data to generalize better for complex spatial features.
A Comprehensive Study on Satellite Image Super-Resolution Using Diffusion and GAN Based Model (2024)	Abhishek Pandey et al.	Custom satellite dataset with 60 cm LR and 15 cm HR images (1024x1024 and 4096x4096 resolution)	Compared DiffIR (Diffusion) and SRGAN for 4x super-resolution. DiffIR achieved PSNR 34.44 and SSIM 0.90; SRGAN achieved PSNR 27.72 and SSIM 0.75. Object detection using YOLOv8 improved from 22% on raw LR to 52% on DiffIR output and 40% on SRGAN output.	Diffusion models give better perceptual quality but are computationally heavy. More diverse geographic datasets needed to reduce bias. Loss functions like L1 and VGG can be further tuned for better quality outputs.

Super Resolution for Single Satellite Image Using a Generative Adversarial Network	Ran Li et al.	GF-2 satellite images (GSD 1m); dataset includes resident, river, and farmland regions	Proposed a GAN with a residual-based generator and discriminator. Used ResNet blocks with 3x3 conv layers and pixel shuffle for upscaling. PSNR and SSIM (for "resident" region): 29.93 and 0.8733. Compared against nearest (25.98, 0.7332), bicubic (27.13, 0.8037), and SRCNN (29.81, 0.8409). Achieved MOS of 4.12 on average.	Model performs well but shows only slight PSNR improvement over SRCNN. Relies on pre-trained ResNet and may not generalize to varied satellite sources. Future work to improve generalizability across satellite platforms.
Enhancing the Resolution of Satellite Imagery Using a Generative Model (2021)	Mahee Tayba et al.	MODIS satellite images (NASA); spatial resolution: 250m	Implemented SRGAN using a generator with 16 residual blocks and a VGG19-based perceptual loss. Initially trained on CelebA dataset, then fine-tuned on MODIS. Downscaling factor was 4×. Reconstructed 256x256 HR images from 64x64 LR input. Achieved higher PSNR than bicubic interpolation (exact PSNR value not stated but visually demonstrated superior quality).	No numerical PSNR or SSIM benchmarks provided. Limited dataset size (1240 satellite images). Needs larger and more diverse satellite datasets across global regions to improve generalization. Model initially biased toward CelebA features before domain-specific fine-tuning.
Ultra-dense GAN for Satellite Imagery Super-Resolution (2019)	Zhongyuan Wang et al.	Kaggle Open-Source Dataset and Jilin-1 Video Satellite Imagery	Proposed udGAN, a GAN-based model with ultra-dense residual blocks (UDRB) using 2D matrix-style connections for enhanced feature learning. Achieved PSNR = 31.75, SSIM = 0.886 on Kaggle (scale x4), with AG = 4.687, NIQE = 5.641. On Jilin-1, AG = 6.223,	Dense structures increase complexity and memory cost. Though perceptual quality improved, PSNR still slightly behind top CNN methods (e.g., RDN). Further optimization

			NIQE = 6.161, outperforming SRGAN, SRCNN, and DenseNet variants in both perceptual and quantitative quality.	needed for inference speed and large-scale deployment on real-time satellite platforms.
A Review and Analysis of GAN-Based Super-Resolution Approaches for INSAT 3D/3DR Satellite Imagery using Artificial Intelligence (2024)	S P Rajamohana et al.	INSAT-3D and INSAT-3DR satellite images across 6 bands (Visible, SWIR, MIR, WV, IR1, IR2, Multispectral)	Compared Bicubic, SRCNN, SRGAN, ESRGAN, and Real-ESRGAN. Real-ESRGAN achieved highest performance: PSNR = 35.07 (WV), 32.5 (VIS), 32.1 (SWIR); SSIM = 1.0 across bands. Real-ESRGAN used U-Net discriminator and pixel-unshuffle preprocessing for better edge preservation and realism.	Despite top PSNR and SSIM, Real-ESRGAN may cause over-smoothing and loss of cloud texture, leading to possible misinterpretations in weather classification. Further validation with ground truth and derived meteorological products is needed.
Research on GAN-based Image Super-Resolution Method	Xiangyu Xue et al.	DIV2K (800 train, 100 val, 100 test), Set5, Set14, BSD100	Proposed R-SRGAN, integrating residual blocks (without batch normalization) into SRGAN and using Wasserstein loss. Achieved PSNR = 31.94, SSIM = 0.8626 on Set5; PSNR = 28.94, SSIM = 0.7866 on Set14; PSNR = 27.28, SSIM = 0.7479 on BSD100. Outperformed Bicubic, SRCNN, and SRGAN in sharpness and clarity.	Early training instability and poor colour consistency. Underperforms on very low-resolution inputs. Needs further tuning for stable colour restoration and better detail retention on extreme downscaled images.

Table 1: Literature Review

DATASET USED

In this project, we used the DOTA-v1.0 (Dataset for Object Detection in Aerial images), a large-scale benchmark dataset designed for object detection in satellite and aerial imagery. The dataset contains a diverse collection of high-resolution images captured from various sources, including Google Earth, GF-2, JL-1 satellites, and aerial photographs provided by CycloMedia B.V. It includes both RGB and grayscale images, where the RGB images are primarily obtained from Google Earth and CycloMedia, while the grayscale images originate from the panchromatic bands of GF-2 and JL-1 satellite sensors. All images are stored in PNG format to maintain high visual quality.

The dataset covers a wide range of object categories commonly found in aerial scenes, such as planes, ships, storage tanks, large and small vehicles, sports fields, bridges, and helipads, among other 15 distinct object classes. For our super-resolution task, we specifically used the training set of DOTA-v1.0, which consists of 469 high-resolution images, where each image differs in dimension.

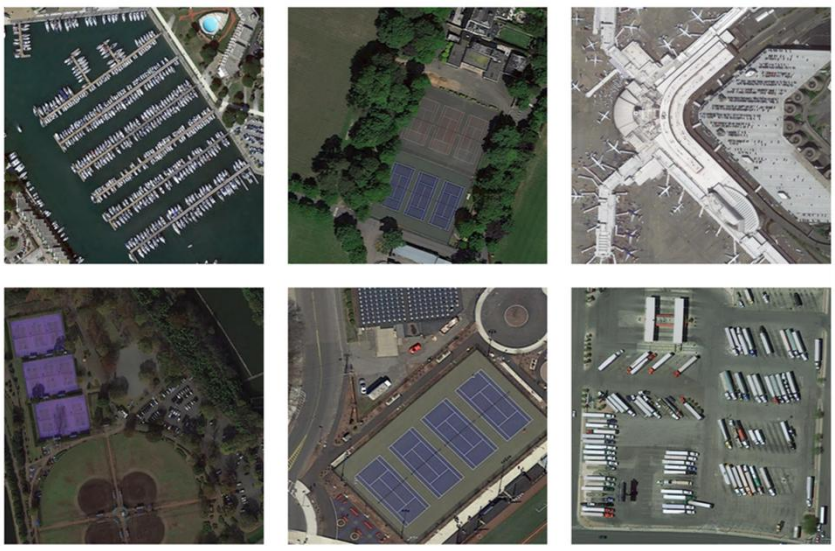


Figure 1: Sample Data from the dataset

METHODOLOGY

This project aims to enhance the resolution of satellite imagery using the Super-Resolution Generative Adversarial Network (SRGAN). The methodology involves three main phases i.e., data preparation, model architecture, and training.

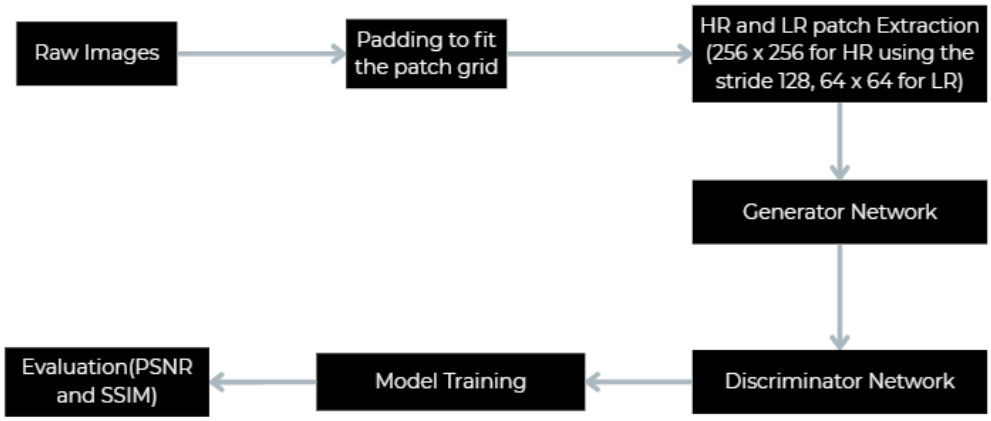


Figure 2: Overall Block Diagram

DATA PREPARATION

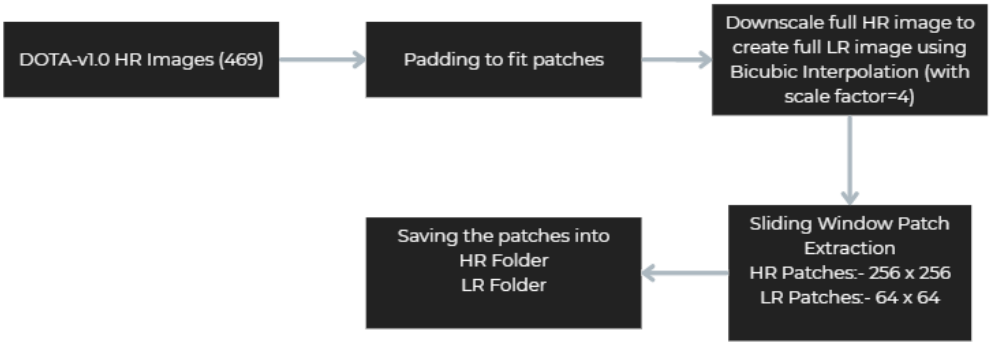


Figure 3: Block Diagram of Data Preparation

The DOTA-v1.0 training set, consisting of 469 high-resolution satellite images, was used as the source dataset. Due to GPU memory constraints and to increase training efficiency, each image was divided into smaller overlapping patches. A sliding window technique with a patch size of 256×256 pixels and a stride of 128 pixels was applied. To ensure complete coverage of image edges, reflective padding was used. The required padding for each image was calculated as:

$$Pad_{dim} = \left(\left\lceil \frac{D-P}{S} \right\rceil + 1\right).S + P - D$$

Where, D is the original image dimension (height or width), P is the patch size (256), and S is the stride (128).

After padding, over 83,000 high-resolution (HR) patches were extracted from the DOTA-v1.0 dataset using a sliding window approach. The patch extraction follows this general formula:

$$N_{patches} = (\left\lfloor \frac{H-P}{S} \right\rfloor + 1) \times (\left\lfloor \frac{W-P}{S} \right\rfloor + 1)$$

Where:

H – Height of the padded image

W – Width of the padded image

P - Patch size (256 pixels for HR)

S -Stride (128 pixels)

$N_{patches}$ – Number of patches per image

This results in overlapping HR patches of size 256×256 pixels. To create the corresponding low-resolution (LR) patches, each full HR image is downscaled using bicubic interpolation with a scale factor r=4:

$$LR_{size} = \frac{P}{r} = \frac{256}{4} = 64 \text{ x } 64$$

After downsampling the full image, the same coordinates used for HR patching are scaled down by a factor of 4 to extract perfectly aligned 64×64 LR patches.

DATA LOADING

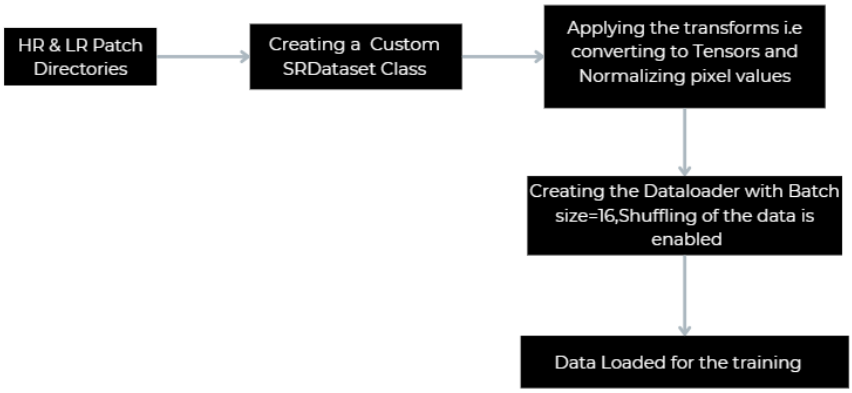


Figure 4: Block Diagram of Data Loading

Once the high-resolution (HR) and low-resolution (LR) patch pairs were generated and stored in separate folders, the data was prepared for training using a custom PyTorch Dataset class. This class, SRDataset, was designed to automatically read and load corresponding LR-HR image pairs from their respective directories. It ensured that each low-resolution image was correctly matched with its high-resolution counterpart based on file naming conventions.

To prepare the images for model input, a transformation pipeline was applied using PyTorch's transforms.Compose, which included converting the images to tensors and normalizing pixel values from the standard [0, 255] range to [0.0, 1.0]. This preprocessing step is essential for stable and efficient training in deep learning models.

The dataset was then wrapped using PyTorch's DataLoader, which allows efficient mini-batch training. The DataLoader was configured with a batch size of 16, shuffling enabled to ensure random sampling during training, and multi-threaded data loading (num_workers = 2) for improved I/O performance. This pipeline ensured that data was fed into the SRGAN model in an optimized and memory-efficient manner.

MODEL

The model architecture used in this project is based on the Super-Resolution Generative Adversarial Network (SRGAN), which consists of two neural networks:

a. Generator (G) – learns to upscale low-resolution (LR) images into high-resolution (HR) outputs.
b. Discriminator(D) - learns to distinguish between real HR images and fake (generated) ones

The generator and discriminator are trained in a minimax fashion, following the standard GAN objective:

$$min_G max_D \; E_{x\sim P_{data(x)}}[\log D(x)] + \; E_{z\sim P_{LR(z)}}\,[\log\,(1 - D\big(G(z)\big))]$$

1. Generator Network

The Generator is responsible for converting a low-resolution input (64×64) into a high-resolution output (256×256). It has the following elements in it:

i. Initial Convolution Block

The initial layer of the Generator is a 2D convolution with 3 input channels, 64 filters, a 9×9 kernel, and padding of 4, followed by a Parametric ReLU (PReLU)

activation, designed to extract low-level features such as edges and textures from the input image.

A 2D convolution is computed as:

$$Y_c(i,j) = \sum_{k=1}^{C_{in}} \sum_{m=1}^{K} \sum_{n=1}^{K} W_{c,k}\,(m,n).X_k(i+m, j+n) + b_c$$

Where:

$Y_c(i,j)$: Output feature map at position (i,j) for the output channel c

X_k: Input image/channel k∈ {1,2,3}

$W_{c,k}$: Learnable filter of size 9x9 between input channel k and output channel c

b_c: Bias term for the output channel c

K=9: Kernel size

C_{in}=3: Number of input channels

C_{out}=64: Number of output channels

After the convolution, a Parametric ReLU (PReLU) activation is applied:

$$f(x) = \begin{cases} x & if\ x \geq 0 \\ a\,.\,x & if\ x < 0 \end{cases}$$

Where a is a learnable parameter (unlike ReLU where a=0).

ii. Residual Block (Used ×16 in Generator)

Each residual block in the SRGAN Generator is designed to learn complex hierarchical image features while preserving spatial consistency through skip connections. These blocks help the network train efficiently at greater depth by addressing issues like vanishing gradients and feature degradation. A total of 16

residual blocks are used in sequence within the Generator. Each residual block consists of:

a. **Conv2d (64 → 64, kernel size = 3×3, padding = 1)**

   This layer extracts local features such as edges and textures using a small receptive field. Padding is applied to maintain the spatial dimensions of the input and output feature maps.

b. **BatchNorm2d**

   This layer normalizes the feature maps across the mini-batch, which stabilizes learning, accelerates convergence, and reduces internal covariate shift.

c. **PReLU Activation**

   The Parametric ReLU introduces non-linearity and includes a learnable negative slope. This enables the network to adaptively model complex relationships, especially in negative activation regions.

d. **Second Conv2d (64 → 64, kernel size = 3×3, padding = 1)**

   This convolution further processes the feature maps post-activation, refining and deepening the feature representation for more abstract pattern learning.

e. **Second BatchNorm2d**

   This normalization continues to stabilize training by regulating the feature distribution across layers, reducing the risk of overfitting or gradient instability.

f. Skip Connection (Residual Addition)

The input x to the block is directly added to the output of the second BatchNorm layer F(x), forming a residual connection:

$$\text{Output} = x + F(x)$$

Where F(x) is the output of the internal Conv-BN-PReLU-Conv-BN stack.

This facilitates gradient flow, encourages feature reuse, and enhances training of very deep networks.

Mathematically:

Let x be the input to the residual block. The transformation F(x) applied inside the block is:

$$F(x) = BN_2(Conv_2(PReLU(BN_1(Conv_1(x)))))$$

The final output of the residual block is:

$$\text{Output} = x + F(x)$$

iii. Bottleneck Block

The Bottleneck Block in the SRGAN Generator is designed to fuse the output of the residual block sequence with the initial low-level feature map. This fusion ensures that the network retains both high-level contextual features learned through deep residual processing and low-level structural details from the early layers. The components are as follows:

(a) Conv2d (64 → 64, kernel size = 3×3, padding = 1)

This layer compresses and refines the features after the residual blocks. It maintains the spatial resolution while integrating the global context extracted from the previous 16 residual blocks.

(b) BatchNorm2d

This layer normalizes the output feature maps to stabilize training and ensure consistent feature scaling before the upsampling stage. It also accelerates convergence by reducing internal covariate shift.

(c) Skip Connection (Residual Fusion with Initial Features)

A skip connection is applied between the output of this bottleneck and the feature map from the initial convolution layer. This helps the network retain fine-grained low-level details that might be lost during deep processing, enhancing the structural integrity of the final super-resolved image.

Mathematically:

$$Bottleneck\ Output = F_{init} + F_{residuals}$$

Where:

F_{init}: Output of the initial convolution block

$F_{residuals}$: Output of the bottleneck block after 16 residuals

iv. Upsampling Block (PixelShuffle-Based)

The upsampling block in the Generator is responsible for increasing the spatial resolution of the feature maps from 64×64 to 256×256, achieving a 4× super-resolution. This is done using a combination of learnable convolutions followed by PixelShuffle operations. The components are as follows:

a) **Conv2dd (64 → 256 filters, kernel size = 3×3, padding = 1)**

This layer expands the channel depth from 64 to 256 to prepare the feature map for upsampling via PixelShuffle. It learns to encode spatial details across additional channels that will later be spatially redistributed.

b) **PixelShuffle (scale factor = 2)**

This layer rearranges the expanded channels into higher spatial resolution. Specifically, it transforms the 256 channels into a 2×2 grid per original pixel, effectively doubling both height and width of the feature map. When applied twice consecutively, it upscales the resolution from 64×64 → 128×128 → 256×256.

Formula:

Input shape: $(C\ .r^2\ , H, W)$ ⇒ Output shape: $(C,\ rH, rW)$

Where:

C: Number of output channels (after shuffle)

r=2: Upscaling factor per step

H, W: Input height and width

Specifically, after the first step: 64 x 64 → 128 x 128. And after the second step:

128 x 128 → 256 x 256

c) **PReLU Activation**

The Parametric ReLU introduces non-linearity after each upsampling stage. It helps the model learn complex, fine-grained pixel-level patterns and adaptively handle negative activations using a learnable slope.

v. Final Output Layer

The final layer in the Generator produces the super-resolved image after all residual, bottleneck, and upsampling operations are complete. It ensures that the output has the desired format and resolution. The components are as follows:

a) Conv2d (64 → 3, kernel size = 9×9, padding = 4)

This convolutional layer reduces the number of channels from 64 to 3, corresponding to the RGB colour space. The large kernel size (9×9) allows the model to capture a broader spatial context when generating the final image. Padding ensures that the spatial resolution remains at 256×256 pixels.

b) Activation: None (Linear Output)

No activation function is applied after this layer. This allows the network to produce pixel values without constraint, enabling direct comparison with the target high-resolution image during loss computation.

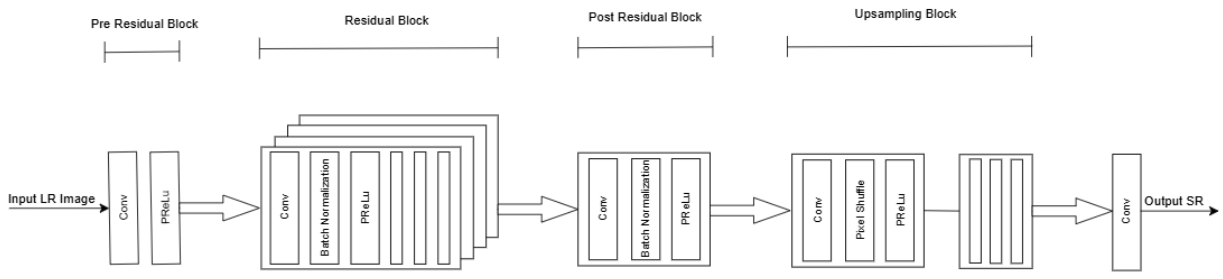


Figure 5: Architecture of Generator

2. Discriminator Network

The Discriminator in SRGAN is a binary classifier designed to distinguish between real high-resolution images and super-resolved images generated by the Generator. It guides the Generator through adversarial feedback, pushing it to produce more photo-realistic outputs. The Discriminator architecture is composed of eight convolutional blocks, progressively increasing in depth and spatial compression. It has the following components in it:

i. **Convolutional Blocks (x 8)**

The Discriminator in SRGAN consists of eight sequential convolutional blocks, each designed to progressively extract hierarchical features, reduce spatial resolution, and increase channel depth. This layered design enables the network to learn discriminative patterns to effectively distinguish between real high-resolution images and those generated by the Generator.

Structure of each block:

a) Block Format

$$\text{Conv2d} \rightarrow (\text{BatchNorm2d}) \rightarrow \text{LeakyReLU}\ (\alpha = 0.2)$$

(BatchNorm is omitted in the first block)

Each convolutional block learns spatial features using Conv2d layers, stabilizes training through Batch Normalization (from the second block onward), and introduces non-linearity with LeakyReLU ($\alpha = 0.2$), which helps retain gradients for negative inputs.

b) Downsampling

Downsampling is applied using stride = 2 in every alternate block. This progressively reduces the spatial resolution of the feature maps (e.g.,

256×256 → 128×128 → 64×64 …), allowing the network to focus on global structural features and improve classification robustness.

c) Filter Progression:

The number of convolutional filters increases progressively across blocks to allow richer feature representation:

3→64→64→128→128→256→256→512→512

ii. Adaptive Average Pooling

The Adaptive Average Pooling layer is used to ensure that the output feature map has a consistent spatial size of 6×6, regardless of the input image dimensions. This consistency is important for the following fully connected layers, which require fixed input sizes. By averaging values across flexible window sizes, the pooling operation reduces each feature map to a 512×6×6 tensor, allowing for efficient transition into the classification stage of the Discriminator.

iii. Fully Connected Classifier

After adaptive pooling, the 512 feature maps of size 6×6 are flattened into a single vector and passed through a fully connected layer with 1024 units, followed by a LeakyReLU activation. This is then passed into a second linear layer that maps to a single output neuron, followed by a Sigmoid activation, which outputs a probability indicating whether the input image is real (closer to 1) or generated (closer to 0).

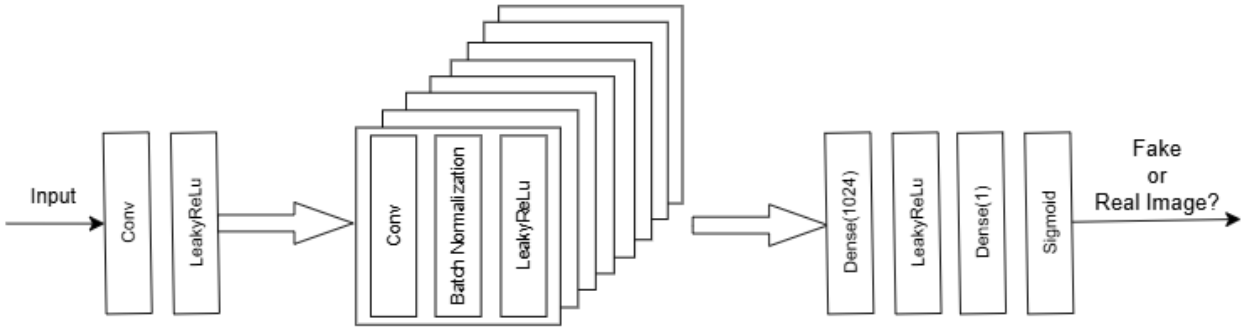


Figure 6: Architecture of Discriminator

LOSS FUNCTIONS

The SRGAN model is trained using a composite loss function that balances pixel-level accuracy, perceptual quality, and adversarial realism. The total loss for the Generator is a weighted sum of three components i.e., Content Loss, Adversarial Loss, and Perceptual Loss.

a) Content Loss (Mean Squared Error - MSE)

The content loss measures the pixel-wise difference between the generated super-resolved image and the ground truth high-resolution image. It encourages the Generator to produce outputs that are numerically close to the target.

$$\mathcal{L}_{content} = \frac{1}{N}\sum_{i=1}^{N} \left\| I_{SR}^{(i)} - I_{HR}^{(i)} \right\|^2$$

Where:

I_{SR} : Super-resolved image generated by the Generator

I_{HR} : Ground truth high-resolution image

N : Number of pixels

b) Adversarial Loss (Binary Cross Entropy – BCE)

The adversarial loss comes from the GAN framework. It encourages the Generator to produce outputs that are indistinguishable from real images by the Discriminator.

$$\mathcal{L}_{adv} = -\log D(G(I_{LR}))$$

Where:

G(I_{LR}): Generated SR image from the low-resolution input

D(.): Discriminator output probability

c) Perceptual Loss (VGG-Based Feature Loss)

Instead of comparing pixels directly, perceptual loss compares feature representations extracted from a pre-trained VGG19 network. It ensures that the texture and perceptual quality of the generated image is close to the ground truth.

$$\mathcal{L}_{perceptual} = \|\emptyset(I_{SR}) - \emptyset(I_{HR})\|^2$$

Where:

$\emptyset$(.): Feature map from an intermediate VGG layer (relu5_4)

d) Final Generator Loss

The final loss for the Generator is a weighted combination of all three:

$$\mathcal{L}_G = \lambda_{content}.\mathcal{L}_{content} + \lambda_{perceptual}.\mathcal{L}_{perceptual} + \lambda_{adv}.\mathcal{L}_{adv}$$

e) Discriminator Loss

The Discriminator is trained to maximize the probability of correctly classifying real and generated images. It uses Binary Cross Entropy (BCE) loss computed on both real and fake samples:

$$\mathcal{L}_D = \frac{1}{2}[\text{BCE}(D(I_{HR}), 1) + \text{BCE}(D(G(I_{LR})), 0)]$$

Which is equivalent to:

$$\mathcal{L}_D = -[\log D(I_{HR}) + \log\left(1 - D(G(I_{LR}))\right)]/2$$

Where,

$D(I_{HR})$: Discriminator output for real HR image

$D(G(I_{LR}))$: Discriminator output for generated SR image

TRAINING SETUP

The SRGAN model was trained on Kaggle Notebooks using dual NVIDIA Tesla T4 GPUs with 16 GB memory each. The training process was carried out over 50 epochs, with each epoch taking approximately 90–105 minutes due to the high-resolution input data and deep model complexity. The training used 83,000+ paired patches of low-resolution and high-resolution images extracted from the DOTA-v1.0 dataset.

The training setup used the Adam optimizer for both Generator and Discriminator:

i. Generator Optimizer:

$$Adam(lr = 5 \times 10^{-5}, \beta_1 = 0.9, \beta_2 = 0.999)$$

ii. Discriminator Optimizer:

$$Adam(lr = 5 \times 10^{-5}, \beta_1 = 0.9, \beta_2 = 0.999, weight\ decay = 1 \times 10^{-6})$$

A batch size of 16 was used for training, and the data was loaded using a custom PyTorch Dataset class with shuffling enabled and parallel I/O (2 workers). The Generator's loss was a weighted combination of content, adversarial, and perceptual losses, with weights:

$\lambda_{content} = 1.0$

$\lambda_{perceptual} = 1.0$

$\lambda_{adv} = 5 \times 10^{-4}$

To optimize learning, a ReduceLROnPlateau scheduler was applied on the Generator's learning rate based on validation PSNR. Models were saved after every epoch.

EVALUATION METRICS

In order to quantitatively assess the quality of the generated super-resolved images, two widely used image similarity metrics were employed: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM). These metrics provide objective insights into how closely the generated images resemble the ground truth high-resolution images.

a) Peak Signal-to-Noise Ratio (PSNR)

PSNR measures the pixel-level fidelity between the super-resolved image and the ground truth. It is based on the Mean Squared Error (MSE), and higher values indicate better reconstruction quality.

Formula:

$$PSNR = 10.log_{10}(\frac{MAX^2}{MSE})$$

Where:

MAX: Maximum possible pixel value (typically 1.0 after normalization)

MSE: Mean Squared Error between the SR and HR images

b) Structural Similarity Index Measure (SSIM)

SSIM evaluates the perceived quality of an image by considering luminance, contrast, and structural similarity. It ranges from -1 to 1, with 1 indicating perfect structural alignment.

Formula:

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

Where:

μ_x, μ_y: Mean intensities of images x and y

$\sigma_x{}^2, \sigma_y{}^2$: Variances

σ_{xy}: Covariance

C_1, C_2 : Stabilization constants

RESULTS

The SRGAN model was trained for 50 epochs on over 83,000 paired patches extracted from the DOTA-v1.0 dataset using dual NVIDIA Tesla T4 GPUs on Kaggle. Each epoch took approximately 90 –105 minutes, and the model's performance was evaluated using PSNR and SSIM metrics on both the training and unseen test sets. The best PSNR of 24.62 dB and the highest SSIM of 0.6932 were achieved at epoch 26, indicating that this was the optimal point for pixel-level fidelity and structural quality. While the Discriminator loss showed expected fluctuations due to the adversarial training process, the Generator loss remained relatively stable and consistently decreased across epochs.

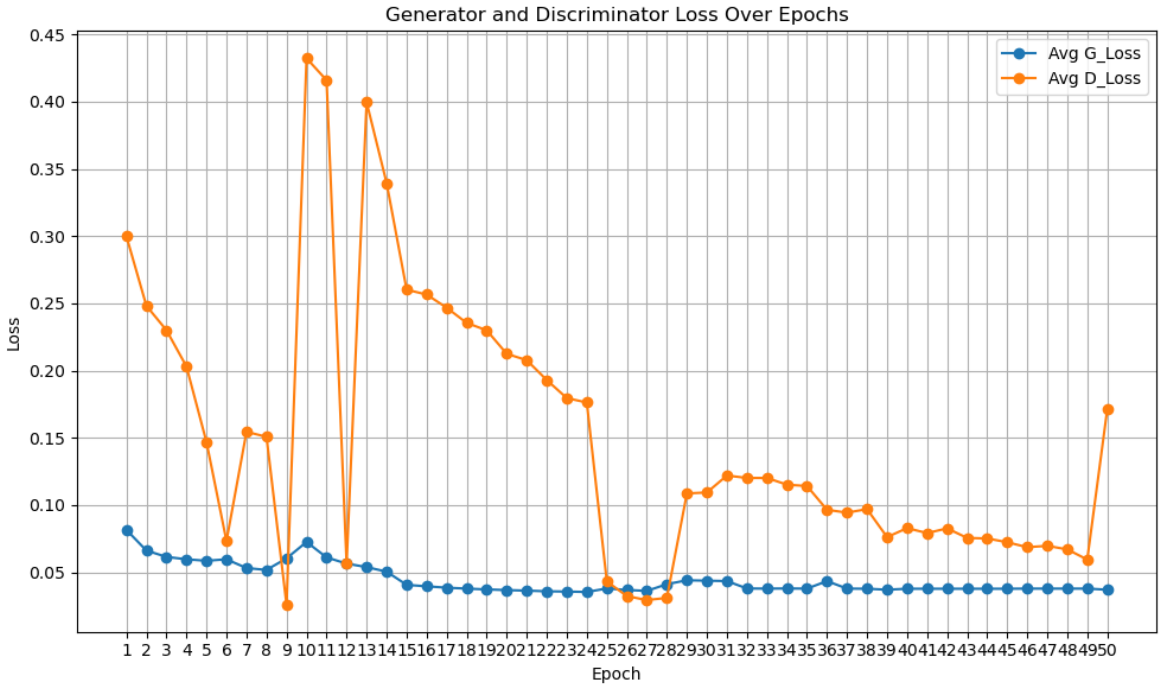


Figure 7: Average Generator Loss and Average Discriminator Loss over epochs

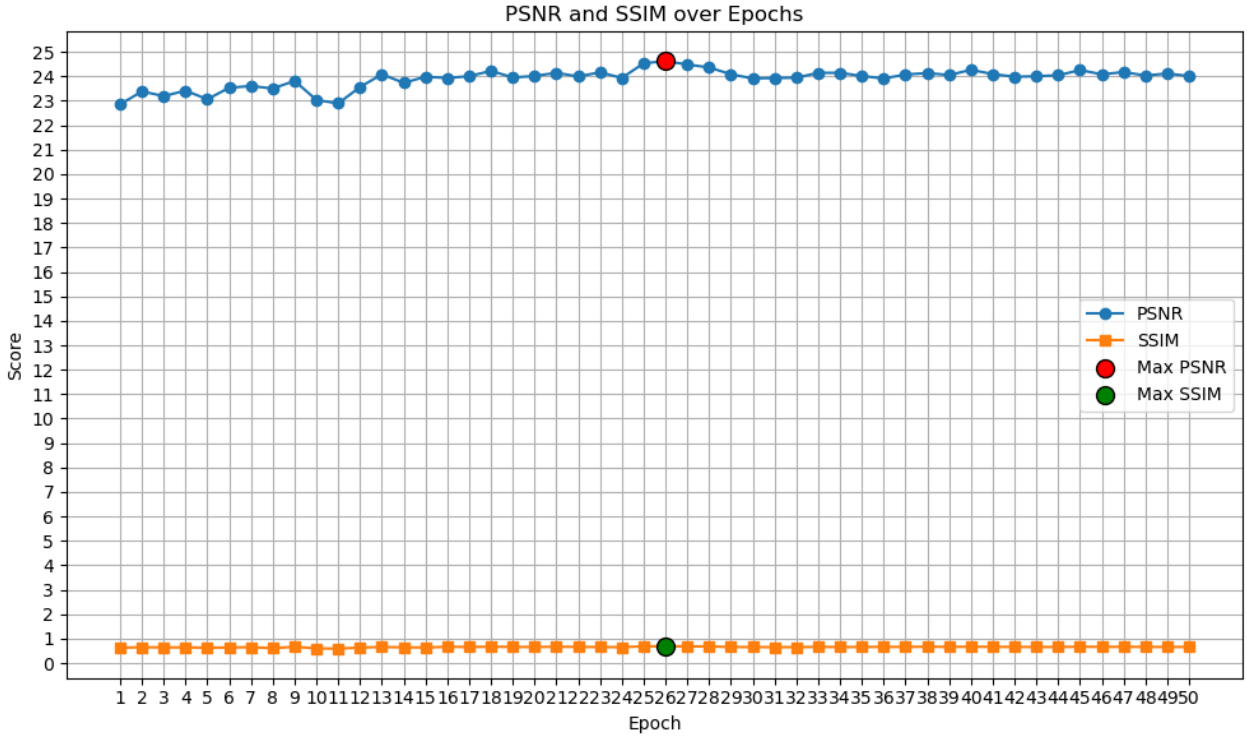


Figure 8: PSNR and SSIM over the epochs

One of the test sets consists of 40 images which were used while training the model, another test set consists of 30 images which weren't seen while training the model. The model generalized well on unseen test data, showing that the training was both stable and effective. The metrics are as follows: -

Test Set	PSNR	SSIM
Test Set -1 (40 images which were seen in the model training)	26.45	0.7101
Test Set – 2 (30 images which weren't seen in the model training)	25.39	0.6884

Table 2: Model Testing Results

Figure 9: Outputs of the model testing

REFERENCES

[1] M. Lavreniuk, L. Shumilo, and A. Lavreniuk, "Generative adversarial networks for the satellite data super resolution based on the transformers with attention," Space Research Institute NASU-SSAU, Kyiv, Ukraine; University of Maryland, College Park, MD, USA; Igor Sikorsky Kyiv Polytechnic Institut, Kyiv, Ukraine, 2024. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10281826

[2] A. Pandey, A. F. Immanuel, D. Saxena, K. Sahu, and N. Mukesh, "A comprehensive study on satellite image super resolution using diffusion and GAN based model," Asian Journal of Convergence in Technology, vol. X, no. I, pp. xx–xx, 2024. [Online]. Available: https://www.asianssr.org/index.php/ajct/article/view/1327

[3] R. Li, W. Liu, W. Gong, X. Zhu, and X. Wang, "Super resolution for single satellite image using a generative adversarial network," ISPRS Annals of the Photogrammetry,

Remote Sensing and Spatial Information Sciences, vol. V-3-2022, pp. 591–597, 2022. [Online]. Available: https://isprs-annals.copernicus.org/articles/V-3-2022/591/2022/
[4] M. Tayba and P. Rivas, “Enhancing the resolution of satellite imagery using a generative model,” in Proc. 2021 Int. Conf. on Computational Science and Computational Intelligence (CSCI), Baylor University, USA, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9799082

[5] Z. Wang, K. Jiang, P. Yi, Z. Han, and Z. He, “Ultra-dense GAN for satellite imagery super-resolution,” Neurocomputing, vol. 405, pp. 247–256, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231219314602

[6] S. P. Rajamohana, S. Thamaraiselvi, B. R. Bibraj, and S. Mitha, “A review and analysis of GAN-based super-resolution approaches for INSAT 3D/3DR satellite imagery using artificial intelligence,” Journal of Scientific and Industrial Research, vol. 83, no. 3, pp. xx–xx, 2024. [Online]. Available: https://or.niscpr.res.in/index.php/JSIR/article/view/7320

[7] X. Xue, X. Zhang, H. Li, and W. Wang, “Research on GAN-based image super-resolution method,” in Proc. 2020 IEEE Int. Conf. on Artificial Intelligence and Big Data (ICAIBD), Chengdu, China, 2020, pp. xx–xx. [Online]. Available: https://ieeexplore.ieee.org/document/9182617