**Super-Resolution of Satellite Images** using SRCNN and VGG19

**1.Introduction**

Image Super-Resolution (SR) refers to the process of reconstructing high-resolution (HR) images from low-resolution (LR) inputs.

Satellite imagery plays a critical role in applications such as environmental monitoring, urban planning, disaster management, medical imaging, and video processing. However, due to hardware limitations or cost constraints, satellite images are often captured at a lower resolution. Super-resolution techniques aim to reconstruct high-resolution images from low-resolution inputs, enhancing the visual quality and enabling more accurate analysis.

In this project, we leverage a Super-Resolution Convolutional Neural Network (SRCNN) to achieve this, and fine-tune the results using perceptual loss computed via a VGG19 feature extractor.

**2.Objectives**

1. Enhance the resolution and visual quality of satellite images from the UC Merced Land Use dataset.
2. Implement the SRCNN model for initial super-resolution learning.
3. Integrate VGG19-based perceptual loss to further improve the perceptual quality of the output images.
4. Evaluate the model's performance and fine-tune it for optimal results.

**3. Dataset Overview**

**3.1 UC Merced Land Use Dataset**

The dataset used is the UC Merced Land Use dataset, which contains 2,100 aerial images classified into 21 land-use classes, such as agricultural, forest, harbor, and residential areas. Each image is:

* Resolution: 256×256 pixels
* Format: RGB
* Dataset Structure: Images are organized into class-specific subfolders.

**3.2 Dataset Preparation**

Images were read using Python's **Pillow (PIL)** library.

To train the SRCNN model, the original high-resolution images were paired with their downscaled counterparts and saved as .npy files for efficient training.

1. **Low-Resolution Images**: Created by downscaling the original 256×256 images to 128×128 using Bicubic interpolation,thrice.
2. **High-Resolution Images**: Original 256×256 images are used as ground truth targets.

**4. Model Development**

**4.1 Super-Resolution Convolutional Neural Network (SRCNN)**

The SRCNN architecture was chosen for its simplicity and effectiveness in super-resolution tasks. It consists of:

1. Three Convolutional Layers:
   * Feature extraction layer: Kernel size (9x9), 64 filters, ReLU activation.
   * Mapping layer: Kernel size (1x1), 32 filters, ReLU activation.
   * Reconstruction layer: Kernel size (5x5), 3 filters.
2. Upsampling Layer: Enlarges the low-resolution input to match the high-resolution target size.

**4.2 Loss Function**

Initially, the model was trained with **Mean Squared Error (MSE)** as the loss function, which minimizes pixel-wise differences between the predicted and ground truth images.

**5. Training the SRCNN Model**

**5.1 Dataset Splitting**

The dataset was split into **80% training** and **20% validation** sets using train\_test\_split. The images were normalized to the range [0, 1] to stabilize training.

**5.2 Training**

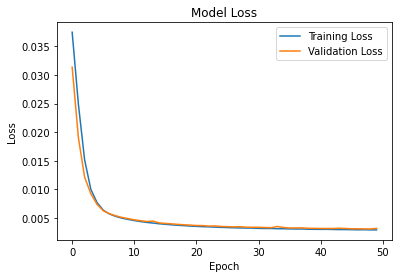
The SRCNN model was compiled with the Adam optimizer and MSE loss. It was trained for **50 epochs** with a batch size of **16**.

**6. Fine-Tuning with VGG19 and Perceptual Loss**

While SRCNN minimizes pixel-wise errors (e.g., mean squared error), this does not guarantee perceptually pleasing results. To address this, we use **VGG19** to compute perceptual loss:

* **Feature Extractor**: Outputs high-level features from the block5\_conv4 layer of VGG19.
* **Perceptual Loss**: Compares the feature maps of the predicted and ground truth HR images, encouraging the model to produce outputs with more perceptual similarity.

#### ****Fine-Tuning:**** The model was recompiled with the perceptual loss and fine-tuned for **10 epochs**.

**7. Results and Visualization**

**7.1 Qualitative Results**

**Visual comparison of original HR images and super-resolved images.**

|  |  |
| --- | --- |
|  |  |
|  |  |

**7.2 Quantitative Results**

**Report metrics like PSNR and MAE for training and validation datasets.**