A

PROJECT REPORT ON SATELLITE IMAGE SEGMENTATION

Submitted in partial fulfillment

for the award of the degree of

Bachelor of Technology

In

Information Technology

(BIKANER TECHNICAL UNIVERSITY, BIKANER)

SESSION (2022-23)

SUBMITTED TO:	**SUBMITTED BY:**
Mrs. Meeta Sharma	Riya Soni
HOD Of CE	19EEMIT010
	Seema Gehlot
	19EEMIT011

Department of Computer Engineering

Mahila Engineering College, Ajmer

Bikaner Technical University, Bikaner

MAHILA ENGINEERING COLLEGE, AJMER
(An Autonomous Institute of Govt. of Rajasthan)
Nasirabad Road, Makhupura, Ajmer – 305002

CERTIFICATE

This is to certify that the Project Report entitle **"Satellite Image Segmentation using U-Net"** has been submitted by Ms. Riya Soni and Ms. Seema Gehlot in partial fulfillment for the requirement of the degree of Bachelor of Technology in Information Technology for the academic Session 2022-23.

She has undergone the requisite work as prescribed by Bikaner Technical University, Bikaner.

(Dr. Varun Prakash Saxena)	(Ms Meeta Sharma)
Asst. Professor of CSE	HOD
	Department of CE

Place :- MAHILA ENGINEERING COLLEGE , AJMER

Date :- 22th May, 2023

ACKNOWLEDGEMENT

I feel profound happiness in forwarding this project as an image of sincere efforts. It is almost inevitable to ensure indebtedness to all who generously helped by sharing their valuable experience and devoting their precious time with us, without whom this project report would have never been accomplished.

First & foremost I extend my thanks & gratitude to my guide **Dr. Varun Prakash Saxena**, whose guidance, teaching, and invaluable suggestions provided me a deep insight in my chosen field of technology, enhanced my knowledge and supported in widening my outlook towards the trends in technology.

I am highly indebted to our project coordinator **Pankaj Kr. Sharma** for providing me the desired platform and delivering the learning in an effective and efficient manner.

Yours Sincerely

Riya Soni

Seema Gehlot

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Image segmentation is the process of partitioning, or segmenting, a digital image into multiple Smaller segments. The goal of image segmentation is to simplify and transform the representation of an image into a format that is more meaningful to a computer and thus, easier to analyze. However, segmentation algorithms are challenged with many difficulties such as: uneven illumination, noise, and presence of foreign objects.

The aim of this project was to explore and present an optimal and efficient image segmentation method based on image processing algorithms learned throughout the course. Various types of images are used in the study. The segmentation process would have to efficiently isolate the foreground from the background and any other foreign objects present in the original image. Several algorithms can be used to fulfill the purpose. For image segmentation using CNN (Convolutional Neural Network), there are many models which are used for image semantic segmentation for example- SegNet, DenseNet, E-Net, Link-Net, Mask RCNN, PSP-Net, RefineNet etc.. In this case, U-Net has been chosen which is basically a convolutional auto-encoder and the advantage is that spatial dimensions of input data for training and inference need not be the same. It does not need much of the training data to get trained. So, it is most of the choice when working with satellite imagery data. and finally, Jupyter Notebook and Python scripts were used to serve the purpose.

CHAPTER 1

INTRODUCTION

Image segmentation is a broad and active field of research in areas such as computer vision, satellite imagery and medical field. Its purpose is to divide an image into regions which are meaningful for a particular task. Various methods and approaches are used; the choice of a particular method depends on the characteristics of the problem to be solved and its place in a wider image analysis strategy. Segmentation is an essential step prior to the description, recognition or classification of an image or its constituents. In other words, the goal of image segmentation is to group pixels together based on certain characteristics that could be its color, shape, size, etc.

The object detection in satellite imagery is a primary goal and elaborate one receiving a lot of interest in recent years and performs an essential function for a wide range of applications. After the massive fulfillment of Deep learning techniques in computer imaginative and prescient discipline, they're presently being studied in the context of satellite imagery for unique functions like object identification, object tracking, object classification, semantic segmentation of aerial/satellite images. Although diverse assessment research associated with object detection from satellite/aerial imagery is carried out, this observation provides an assessment of the latest development in the discipline of object detection from satellite imagery with the use of deep learning.

1.1 IMAGE AUGMENTATION

Image augmentation is a technique that is used to artificially expand the data-set. This is helpful when we are given a data-set with very few data samples. In the case of Deep Learning, this situation is bad as the model tends to over-fit when we train it on a limited number of data samples.

The dataset consists of aerial imagery of Dubai obtained by MBRSC satellites and annotated with pixel-wise semantic segmentation in 6 classes. The total volume of the dataset is 72 images grouped into 6 larger tiles: Buildings, Roads, Land, Water bodies, Vegetation and Unlabelled. In order to avoid the overfitting we have expanded our dataset using Image Augmentation. Our dataset have 8 tiles each having 9 images and their associated masks. The size of images in individual tiles is the same but the size of an image in a tile is different from the size of image of another tile. So to figure out this issue we have cropped them to a nearest size divisible by 256 and then divide all images into patches of 256x256x3.

1.2 SEMANTIC SEGMENTATION

Semantic segmentation is a deep learning algorithm that associates a label or category with every pixel in an image. It is used to recognize a collection of pixels that form distinct categories. It is the ability for a machine to classify features in a given image pixel-by-pixel to create a mask. The goal for this project is to input these masks into a feature detector to determine significant features in an image. A U-Net architecture is utilized to perform this semantic segmentation since it is the best performing method for aerial imagery.

U-Net is an architecture for semantic segmentation. It consists of a contracting path and an expansive path. The contracting path follows the typical architecture of a convolutional network. It was originally invented and first used for biomedical image segmentation. Its architecture can be broadly thought of as an encoder network followed by a decoder network.We have trained our model using three different optimizers: Adam, Adamax and RMSprop, to compare their results with evaluation parameters.

1.3 EVALUATION

The evaluation matrices used during training are binary accuracy and mIoU. The weights of the models with the lowest validation loss and high mIoU scores are used since these models are more likely to perform the best when predicting the final test set and because mIoU is more commonly used for semantic segmentation. The loss functions used for these experiments were the mIoU approximation and binary cross-entropy. The mIoU approximation, also known as

Jaccard loss, was implemented since the evaluation metrics for most of the datasets as well as many others in semantic segmentation use mIoU as a standard.

1.4 MOTIVATION

Satellite images are one of the most powerful and important tools we have for monitoring the earth. They track the physical environment (water, air, land, vegetation) and the changing human footprint across the globe. Satellite imagery is used to measure, identify and track human activity. Apart from this it is also used in the Archaeological department to study the land on the basis of these parameters: in different seasons, over different timestamps, soil type, etc; in order to determine whether this land can be an archaeological site for research or not.

1.5 PROBLEM STATEMENT

With the set of images and prior knowledge about the content of the images, we have to find the correct Symantec label for the pixels in the image. Labels are: Buildings, Roads, Land, Water Bodies, Vegetation, and Unlabeled. We need to go through the cycle of segmentation, feature extraction, testing, and validation.

1.6 REQUIREMENTS

Functional requirements are the high level system requirements for the above problem statement. Our functional requirements are as follows:

- Software- Jupyter Notebook 6.5.4
- Processor- GPU
- Programming Language- Python 3.10.11
- Modules and Frameworks- Tensorflow 2.12.0, Keras 3.7.2, OpenCv 4.7.0, Patchify 0.2.3, Segmentation Models 1.0.1, Numpy 1.24, Matplotlib 3.7.1, Sklearn 0.23.0
- Algorithm- U-Net
- Optimizers- Adam, Adamax, RMSprop

CHAPTER 2

PROBLEM STATEMENT

Image segmentation is one of the important trends in image processing. It is a technique which divides or partitions an image into segments. There are various application areas for image segmentation mostly are image compression, medical applications, satellite imagery, object recognition etc.. as it is not efficient to process the entire image. Image segmentation segments an image into sub regions of our interest and then those areas can be analyzed individually. There are many techniques for this which partition an image into segments based on certain features like color, texture, pixel intensity etc... and the categorization of techniques are done based on the method used. Here in this case, the problem statement is that the images received from satellite will be segmented and then roads, buildings, land, vegetation, etc. present in the image need to be detected.

2.1 DATASET

The dataset consists of aerial imagery of Dubai obtained by MBRSC satellites and annotated with pixel-wise semantic segmentation in 6 classes. The total volume of the dataset is 72 images grouped into 6 classes. The classes are:

1. Building: #3C1098
2. Land (unpaved area): #8429F6
3. Road: #6EC1E4
4. Vegetation: #FEDD3A
5. Water: #E2A929
6. Unlabeled: #9B9B9B

There are a total of 8 tiles in the directory, each tile having 9 images and their masks. The size of any image is the same with respect to the images of that particular tile, but the size of images present in different tiles have different sizes.

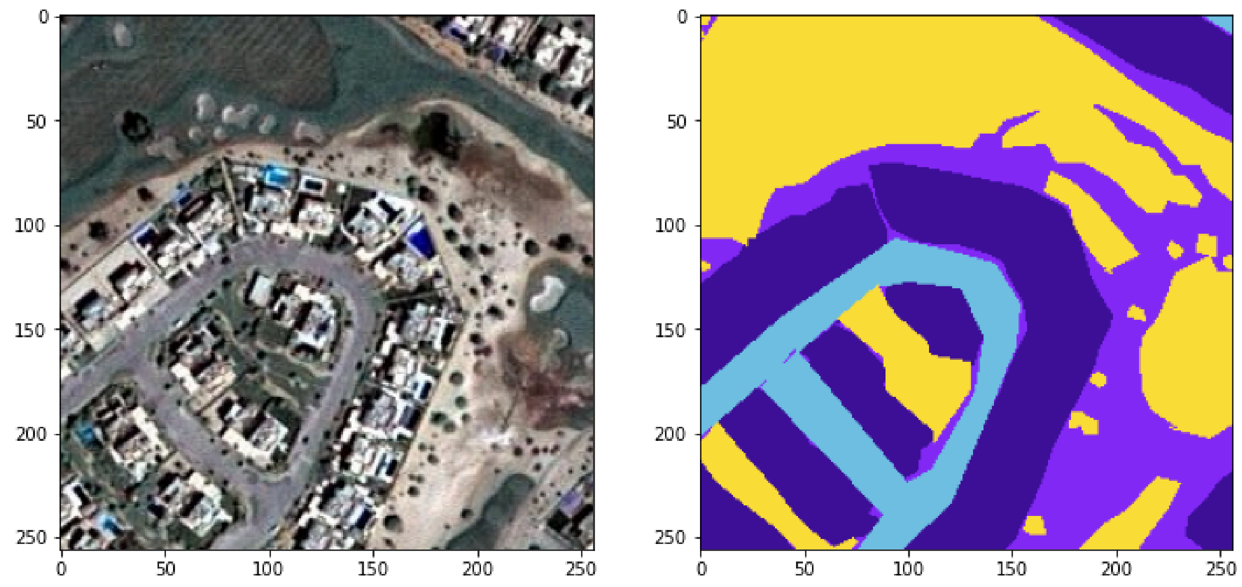


Figure 2.1 Satellite image with its Mask

2.2 SOLUTION

For image segmentation using CNN (Convolutional Neural Network), there are many models which are used for image semantic segmentation for example- SegNet, it is an architecture for image segmentation based on deep convolutional encoder-decoder. Fully Convolutional DenseNet, this is for semantic segmentation, The One Hundred Layers Tiramisu is an example of it. The other more are E-Net for real-time semantic segmentation and Link-Net, for efficient semantic segmentation it exploits the encoder representations. Few more names like Mask RCNN, PSP-Net, RefineNet etc.. In this case, U-Net has been chosen which is basically a convolutional auto-encoder and the advantage is that spatial dimensions of input data for training and inference need not be the same. It does not need much of the training data to get trained. So, it is most of the choice when working with satellite imagery data.

2.3 WORKFLOW

- Importing Libraries
- Pre-processing
  1) Getting Dataset

2) Importing Libraries
3) Importing Dataset
4) Image Augmentation
5) Replacing HEX values with RGB
6) Replacing RGB values with Labels from 0 to 5

- Model Building

1) Train Test Split
2) Model Parameters
3) Defining Model
4) Building Model using different Optimizers

- Model Evaluation

CHAPTER 3

METHODOLOGY

3.1 PRE-PROCESSING

Preprocessing of data is performed to ensure that it is in a format that the network can accept is a common first step in deep learning workflows. For example, you can resize image input to match the size of an image input layer. You can also preprocess data to enhance desired features or reduce artifacts that can bias the network.

Steps involved for pre processing in this project are:

1) Getting Dataset
2) Importing Libraries
3) Importing Dataset
4) Image Augmentation
5) Replacing HEX values with RGB
6) Replacing RGB values with Labels from 0 to 1

3.1.1 Getting Dataset: To create a machine learning model, the first thing we require is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the **dataset**.

3.1.2 Importing Libraries:

In order to perform data pre-processing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we have used for data preprocessing, which are:

- **Numpy:** Numpy Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports adding large, multidimensional arrays and matrices.

To import - import numpy is np

- **OS:** Python OS module provides the facility to establish the interaction between the user and the operating system. It offers many useful OS functions that are used to perform OS-based tasks and get related information about operating systems.
  To import - import os

- **OpenCv:** OpenCV is a Python library that allows us to perform image processing and computer vision tasks. It provides a wide range of features, including object detection, face recognition, and tracking.
  To import - import cv2

- **Matplotlib:** The second library is **matplotlib**, which is a Python 2D plotting library, and with this library, we need to import a sub-library **pyplot**. This library is used to plot any type of charts in Python for the code.
  To import - from matplotlib import pyplot as plt

- **Patchify:** It is used to split images into small overlappable patches by given patch cell size, and merge patches into the original image. This library provides two functions: patchify, unpatchify.
  To import - from patchify import patchify

- **Segmentation Models:** Segmentation models is a python library with Neural Networks for Image Segmentation based on Keras (Tensorflow) framework. The main features of this library are: High level API (just two lines to create NN) 4 models architectures for binary and multi class segmentation (including legendary U-Net)
  To import - import segmentation_models as sm

- **Tensorflow:** The TensorFlow platform helps us implement best practices for data automation, model tracking, performance monitoring, and model retraining. Here we need to import MeanIoU from this module of python.
  To import - from tensorflow.keras.metrics import MeanIoU

3.1.3 Importing Dataset:

Now we need to import the datasets which we have collected for our machine learning project. And to load the dataset we need the help of the os library.

3.1.4 Image Augmentation:

Image Augmentation is a technique that is used to artificially expand the data-set.
The size of our dataset is 32MB, it consists of only 72 images and their masks. To gain better results and avoid overfitting we need to expand our dataset. In order to do that we need to use Image Augmentation technique.
The other reason to crop images instead of resizing is to keep the sizes of objects as it is. As the size of images in a tile is different from the size of image of another tile, we have cropped them to a nearest size divisible by 256 and then divide all images into patches of 256x256x3 (see figure 3.1).

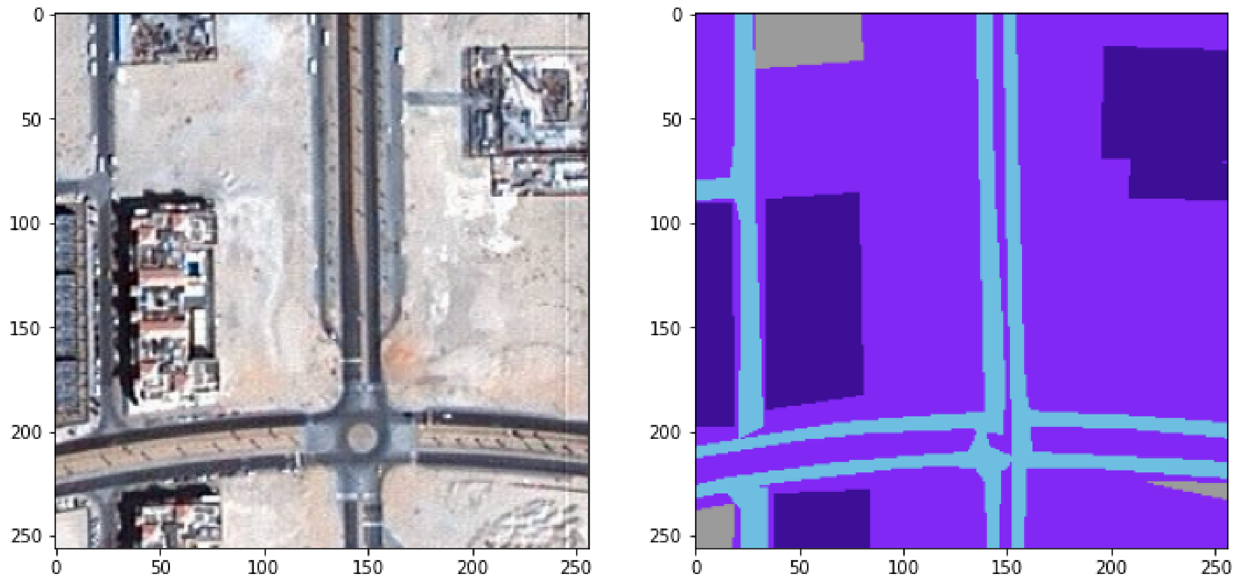


Figure 3.1 An image patch of size 256x256x3

3.1.5 Replacing HEX values with RGB: Now to move forward we need to convert the HEX values of labels into RGB values

3.1.6 **Replacing RGB values with Labels from 0 to 5:** We are now converting our RGB labels into categorical values labeling them from 0 to 5.

3.2 TRAINING MODEL

Training a model simply means learning (determining) good values for all the weights and the bias from labeled examples. In supervised learning, a machine learning algorithm builds a model by examining many examples and attempting to find a model that minimizes loss; this process is called empirical risk minimization.

Steps involved in Training the model are as follows:

1) Train Test Split
2) Model Parameters
3) Defining Model
4) Building Model using different Optimizers

3.2.1 Train Test Split:

It is a method to measure the accuracy of the model. It is used to split the data set into two sets: a training set and a testing set. 80% for training, and 20% for testing. We train the model using the training set.

3.2.2 Model Parameters:

The parameters used for the model are as follows:

- **Weights:** Weights refer to connection managements between two basic units within a neural network. To train these units to move forward in the network, weights of unit signals must be increased or decreased.

- **Dice Loss:** It is widely used in image segmentation tasks to address the data imbalance problem.

To define dice loss - dice_loss = sm.losses.DiceLoss(class_weights=weights)

- **Focal Loss:** It is an extension of the cross-entropy loss function that would down-weight easy examples and focus training on hard negatives.

  To define focal loss - focal_loss = sm.losses.CategoricalFocalLoss()

- **Total Loss:** It is the sum of Dice Loss and Focal Loss.

  It can be defined as - total_loss = dice_loss + (1 * focal_loss)

3.2.3 Defining Model:

UNET is a U-shaped encoder-decoder network architecture, which consists of four encoder blocks and four decoder blocks that are connected via a bridge. The encoder network (contracting path) halves the spatial dimensions and doubles the number of filters (feature channels) at each encoder block. Figure 3.2 shows the architectural representation of U-Net.

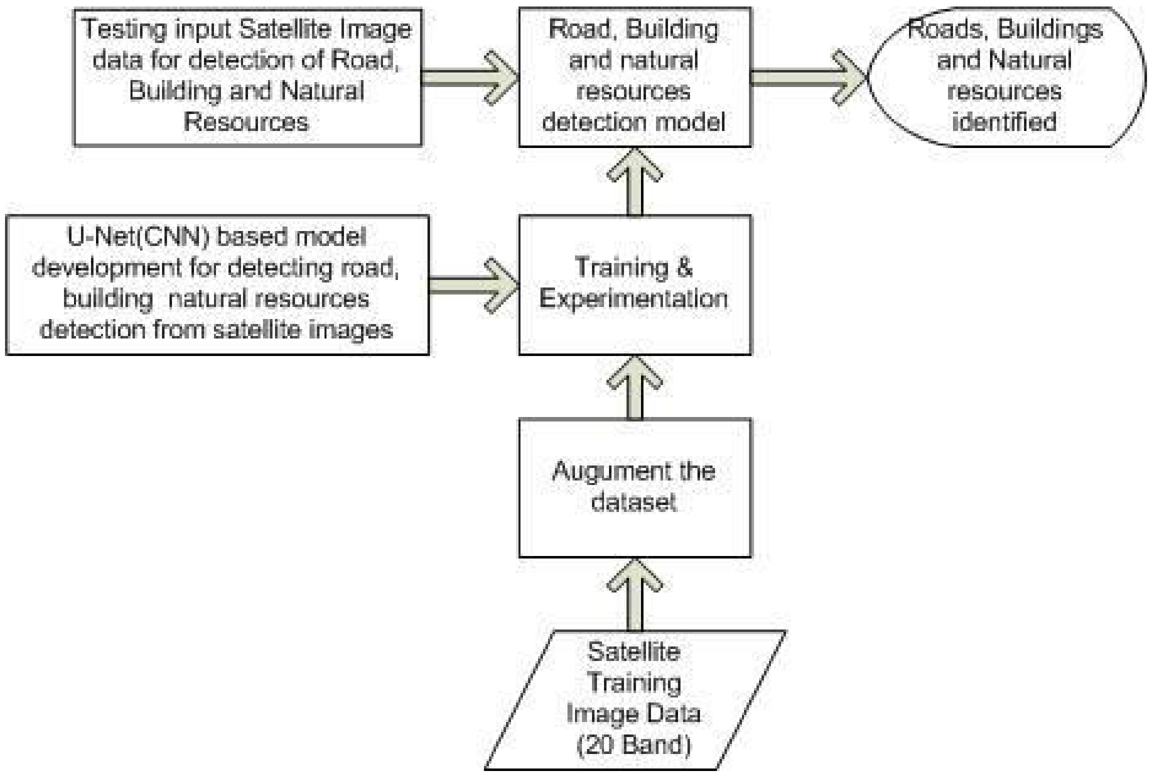

Testing input Satellite Image data for detection of Road, Building and Natural Resources
Road, Building and natural resources detection model
Roads, Buildings and Natural resources identified
U-Net(CNN) based model development for detecting road, building natural resources detection from satellite images
Training & Experimentation
Augument the dataset
Satellite Training Image Data (20 Band)

Figure 3.2 Architecture of U-Net model

3.2.4 Building Model using different Optimizers:

In this model we have used three optimizers:

- **Adam-** Adam optimization is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments.
- **Adamax-** Adamax, a variant of Adam based on the infinity norm, is a first-order gradient-based optimization method. Due to its capability of adjusting the learning rate based on data characteristics, it is suited to learn time-variant process
- **RMSprop-** The RMSprop optimizer is similar to the gradient descent algorithm with momentum. The RMSprop optimizer restricts the oscillations in the vertical direction. Therefore, we can increase our learning rate and our algorithm could take larger steps in the horizontal direction converging faster.

CHAPTER 4

IMAGE AUGMENTATION

Image augmentation is a technique that is used to artificially expand the data-set. This is helpful when we are given a data-set with very few data samples. In the case of Deep Learning, this situation is bad as the model tends to over-fit when we train it on a limited number of data samples.

Image augmentation parameters that are generally used to increase the data sample count are zoom, shear, rotation, preprocessing_function and so on. Usage of these parameters results in generation of images having these attributes during training of Deep Learning models. Image samples generated using image augmentation, in general, result in increase of existing data sample set by nearly 3x to 4x times.

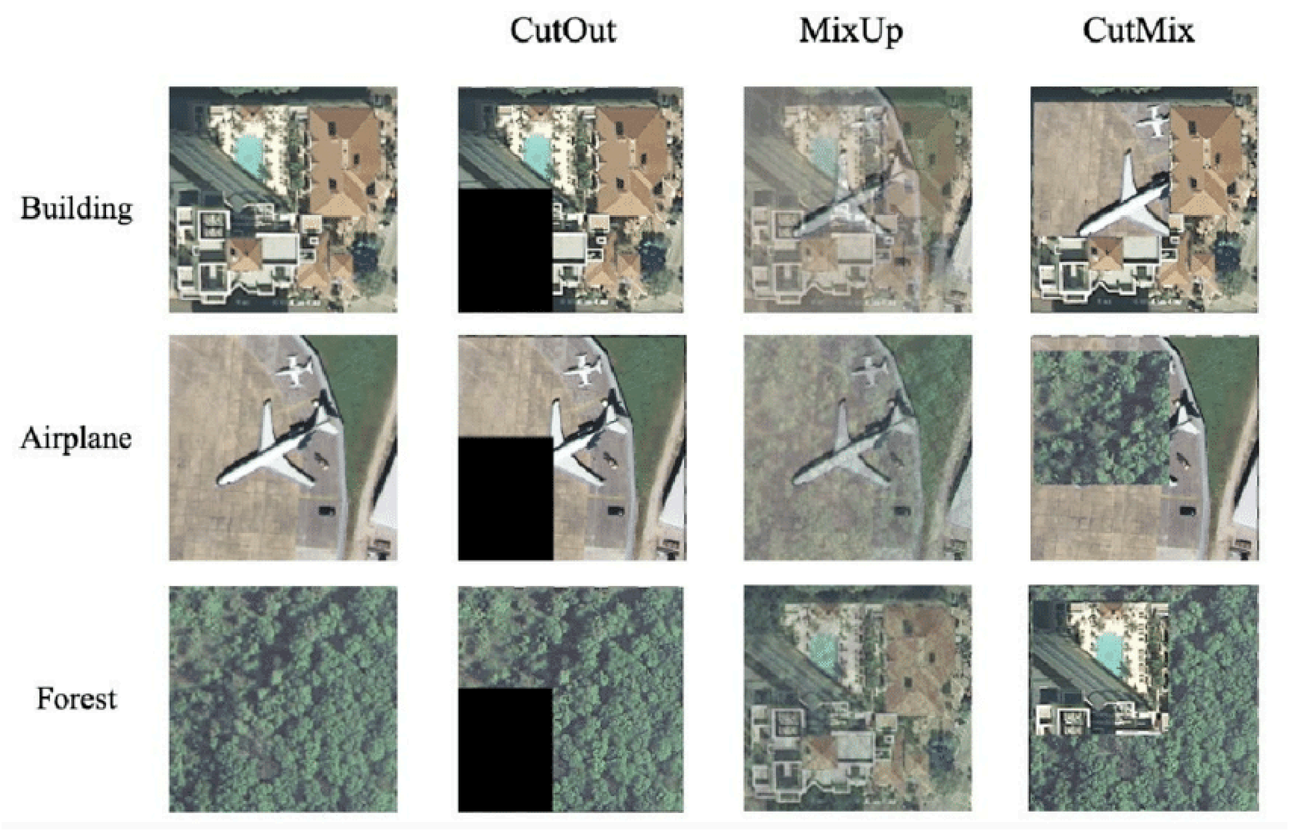


Figure 4.1 An example to demonstrate how image augmentation works

In figure 4.1 we saw how a single image can be augmented using image augmentation. This helps when the size of the dataset is less. Training a model on a very small dataset may lead to poor performance when applied to a new, never-seen-before dataset. Augmentation techniques such as random flipping, rotation, zooming and channel averaging have been successfully applied to RGB images; however these techniques are limited when applied to satellite imagery and might not have a significant impact as a result of inherent uniformity in the images. Despite the unique challenges associated with training deep learning models with images, it is widely accepted that data augmentation can lead to significant improvements in the performance of these models by making them more robust to noise and less likely to overfit.

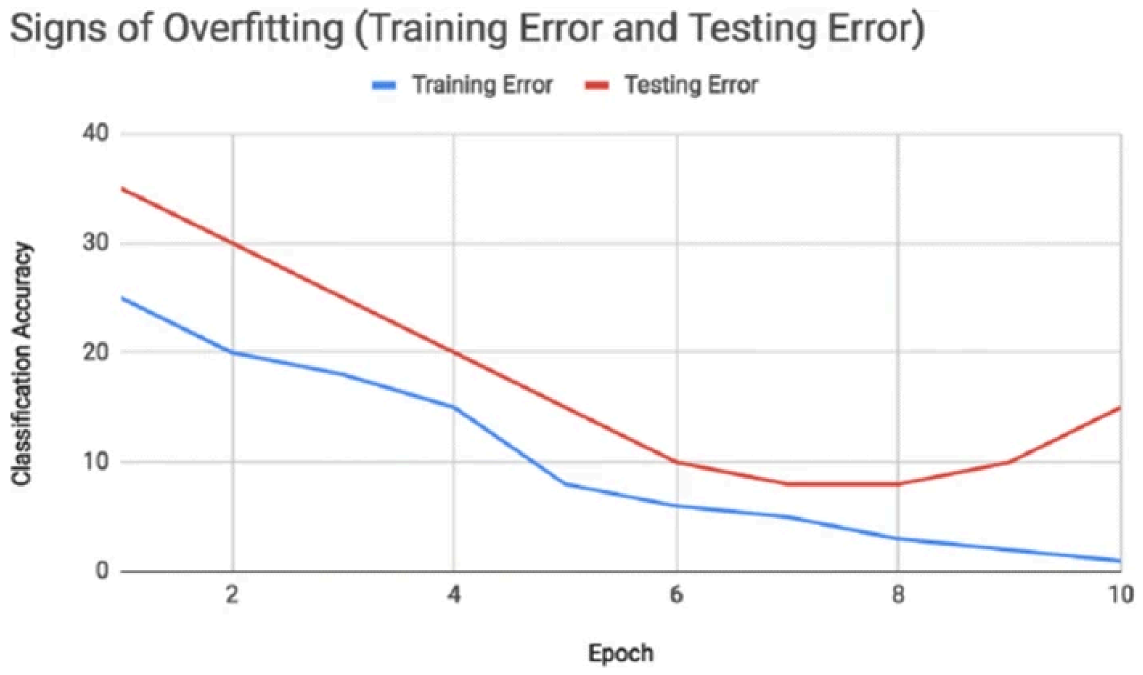


Figure 4.2 Overfitting due to less data

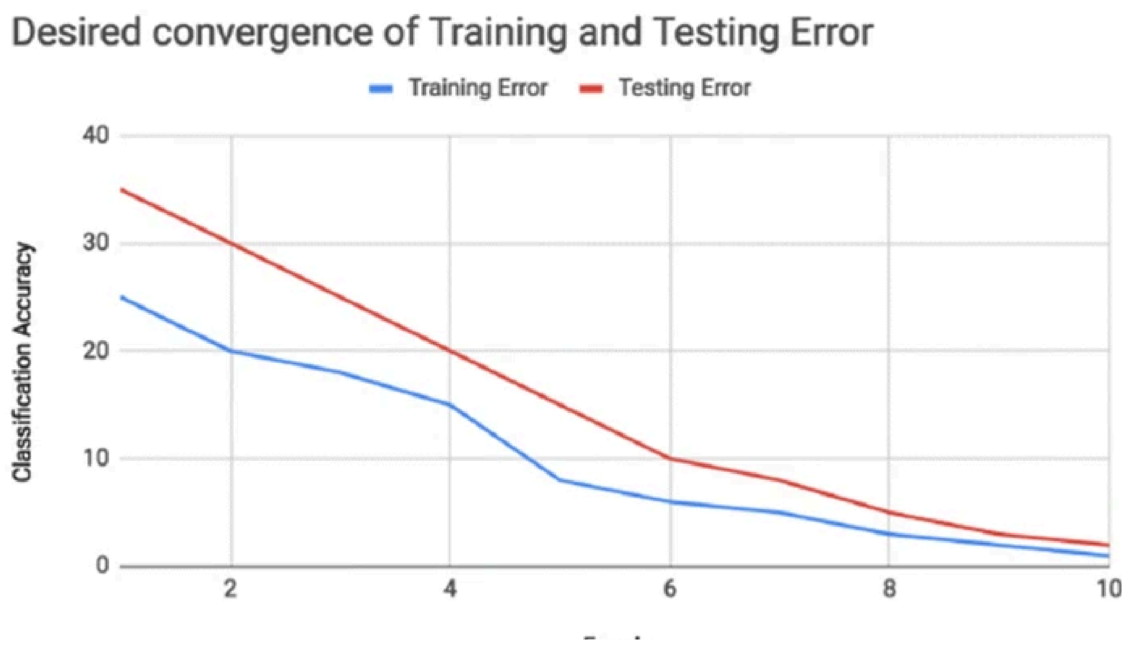


Figure 4.3 Desired convergence of Training and Testing data

The plot on the left shows an inflection point where the validation error starts to increase as the training rate continues to decrease. The increased training has caused the model to overfit to the training data and perform poorly on the testing set relative to the training set. In contrast, the plot on the right shows a model with the desired relationship between training and testing error.

4.1 TECHNIQUES USED FOR IMAGE AUGMENTATION

4.1.1 Augmentation using Geometric Techniques

Data augmentation increases the amount of data available to train a model. The primary advantage of doing this is that it makes the model more robust and less susceptible to overfitting. In addition, it can improve the model performance by mimicking the image with different features. Different augmentation techniques have been shown to improve the model performance differently, depending on the quantity and quality of these features. Small satellite image datasets have been used for training deep learning models in existing works. **Horizontal and vertical flipping** had the highest accuracy out of the techniques considered for classifying satellite images. Image zooming or scale augmentation was used when the image is zoomed in or out, depending on a rate magnitude. **Rotation augmentation** is another relevant technique in which several copies of an image are produced by rotating it through different angles (See figure 4.4).

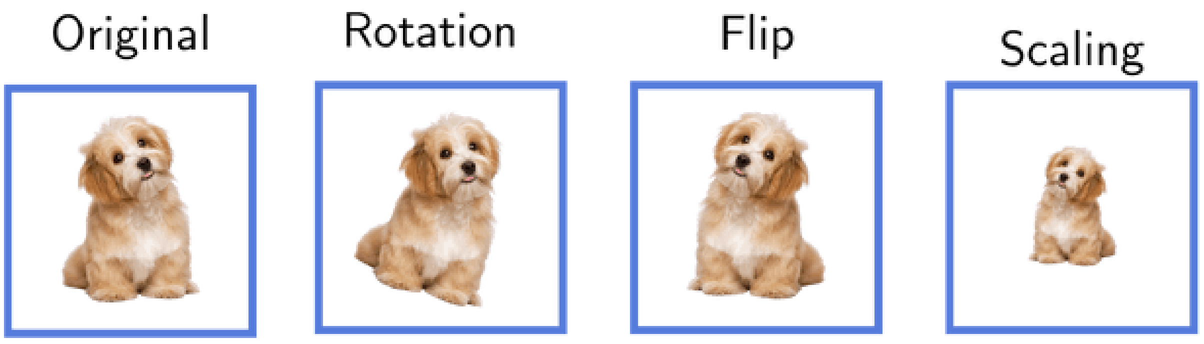


Figure 4.4 Geometric Augmentation

4.1.2 Augmentation with GANs

Generative Adversarial Networks (GANs) have also been used for data augmentation on traditional RGB images and Satellite images. This is an unsupervised way of generating data. The generative model usually comprises the generator and discriminator, which work like gaming components. The former learns to generate realistic images and trick the discriminator which attempts to differentiate between the real and synthesized images. Examples of GANs, which have been applied to satellite images are DCGAN, CycleGAN, SSSGAN. High resolution images can be generated using a progressive growing GAN. GANs can be applied to satellite images for both image generation and image style transfer with visually promising results. Conditional GAN can also be applied to satellite images. Figure 4.5 and 4.6 shows how a zero can be generated using cGan and WGAN.

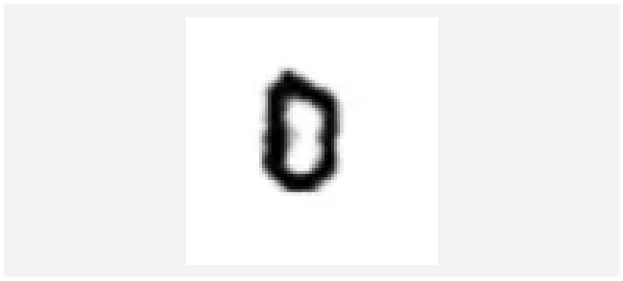


Figure 4.5 Generated "0" by the cGAN

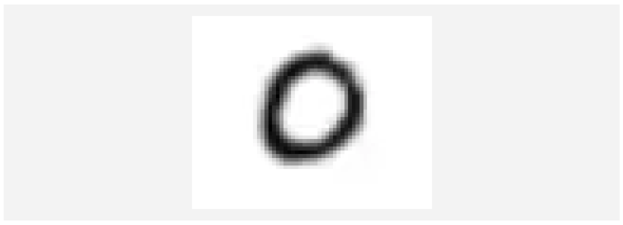


Figure 4.6 Generated "0" by the WGAN

CHAPTER 5

U-NET MODEL

Feature Extraction aims to reduce the number of features in a dataset by creating new features from the existing ones (and then discarding the original features). These new reduced sets of features should then be able to summarize most of the information contained in the original set of features. In this way, a summarized version of the original features can be created from a combination of the original set.

5.1 U-NET ALGORITHM

U-Net is a convolutional neural network architecture that expanded with few changes in the CNN architecture. It was invented to deal with biomedical images where the target is not only to classify whether there is an infection or not but also to identify the area of infection. We can build an image segmentation model using U-Net that will predict the mask of an object present in an image. The model will localize the object in the image using this method.

The UNet architecture was introduced for BioMedical Image segmentation by Olag Ronneberger et al. The introduced architecture had two main parts that were encoder and decoder. The encoder is all about the covenant layers followed by pooling operation. It is used to extract the factors in the image. The second part decoder uses transposed convolution to permit localization.

It is an architecture for semantic segmentation. It consists of a contracting path and an expansive path. The contracting path follows the typical architecture of a convolutional network. It consists of the repeated application of two 3x3 convolutions (unpadded convolutions), each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for downsampling. At each downsampling step we double the number of feature channels. Every step in the expansive path consists of an upsampling of the feature map followed by a 2x2 convolution (“up-convolution”) that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU. The cropping is necessary due to the loss of border pixels in every convolution. At the final layer a 1x1 convolution is used to map each 64-component feature vector to the desired number of classes. In total the network has 23 convolutional layers. Figure 5.1 shows an example of layered architecture of U-Net.

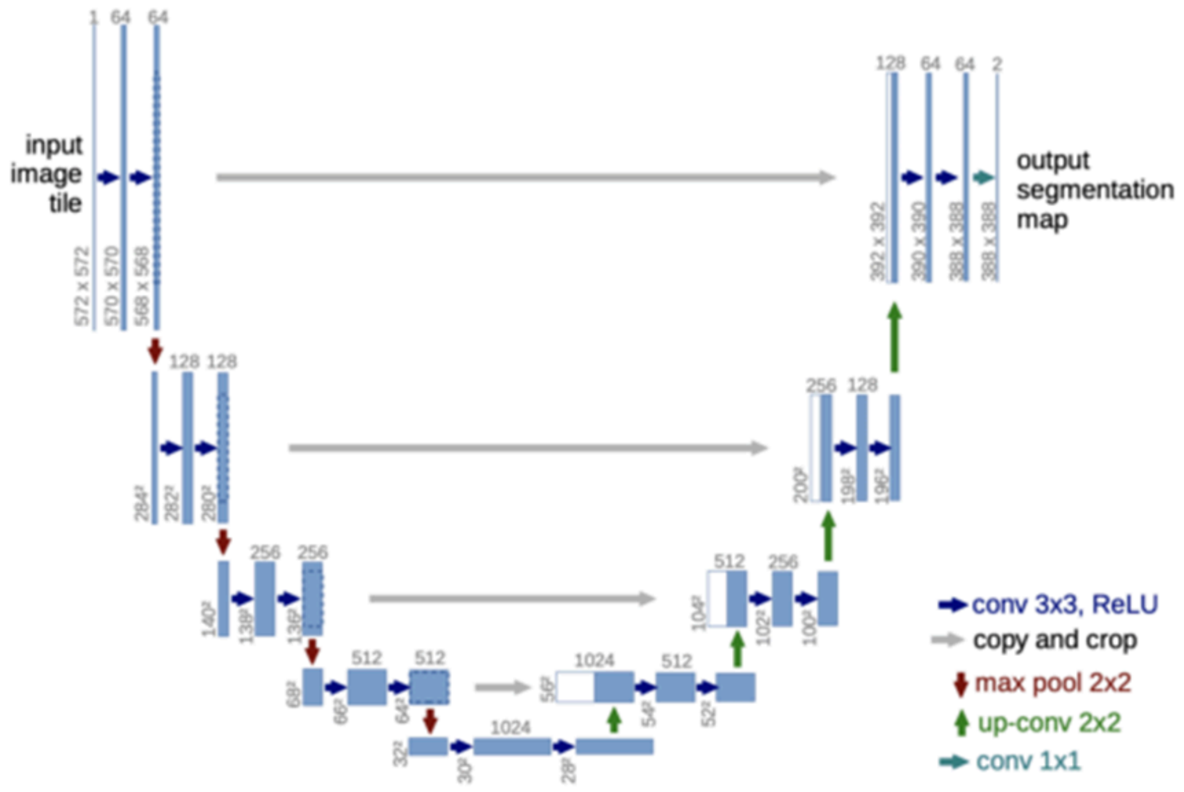


Figure 5.1 Example to show a Layered architecture of U-Net

The **Encoder** is the first half in the architecture diagram (Figure 5.1). It usually is a pre-trained classification network like VGG/ResNet where we apply convolution blocks followed by a maxpool downsampling to encode the input image into feature representations at multiple different levels.

The **Decoder** is the second half of the architecture. The goal is to semantically project the discriminative features (lower resolution) learnt by the encoder onto the pixel space (higher resolution) to get a dense classification. The decoder consists of upsampling and concatenation followed by regular convolution operations.

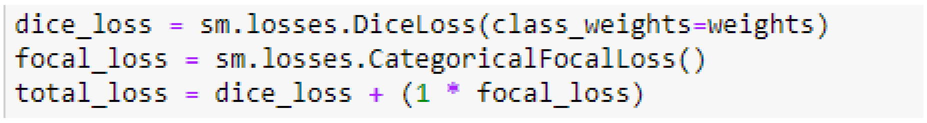
Upsampling in CNN might be new to those of you who are used to classification and object detection architecture, but the idea is fairly simple. The intuition is that we would like to restore the condensed feature map to the original size of the input image, therefore we expand the feature dimensions. Upsampling is also referred to as transposed convolution, upconvolution, or deconvolution. There are a few ways of upsampling such as Nearest Neighbor, Bilinear Interpolation, and Transposed Convolution from simplest to more complex.

Specifically, we would like to upsample it to meet the same size with the corresponding concatenation blocks from the left. You may see the gray and green arrows, where we concatenate two feature maps together. The main contribution of U-Net in this sense is that while upsampling in the network we are also concatenating the higher resolution feature maps from the encoder network with the upsampled features in order to better learn representations with following convolutions. Since upsampling is a sparse operation we need a good prior from earlier stages to better represent the localization.

5.2 MODEL ARCHITECTURE

5.2.1 Model Parameters

Model parameters that have been used are shown below in figure 5.2.



```
dice_loss = sm.losses.DiceLoss(class_weights=weights)
focal_loss = sm.losses.CategoricalFocalLoss()
total_loss = dice_loss + (1 * focal_loss)
```

Figure 5.2 Model Parameters

5.2.1.1 Dice Loss: Dice loss is the multi-class extension of Dice loss where the weight of each class is inversely proportional to the square of label frequencies. Its formula is:

$$DiceLoss(y, \overline{p}) = 1 - \frac{(2y\overline{p} + 1)}{(y + \overline{p} + 1)}$$

Figure 5.3 Dice Loss

5.2.1.2 Focal Loss: Focal loss adapts the standard CE to deal with extreme foreground-background class imbalance, where the loss assigned to well-classified examples is reduced. Its formula is:

$$\mathrm{FL}(p_t) = -(1 - p_t)^{\gamma} \log(p_t)$$

Figure 5.4 Focal Loss

5.2.1.3 Total Loss: It is the sum of Dice Loss and Focal Loss.

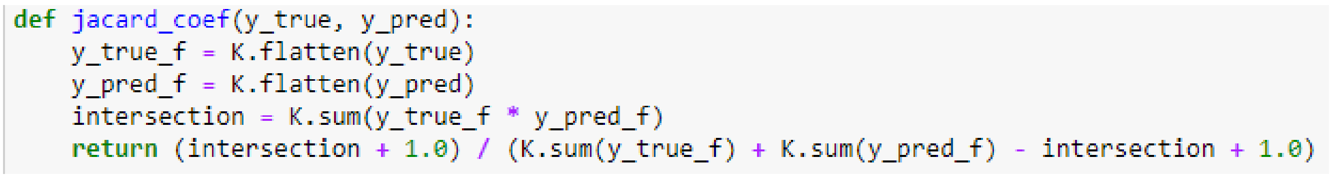
5.2.2 Jaccard Coefficient

The Jaccard similarity index (sometimes called the Jaccard similarity coefficient) compares members for two sets to see which members are shared and which are distinct. It's a measure of similarity for the two sets of data, with a range from 0% to 100%. The higher the percentage, the more similar the two populations.

Formula:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Figure 5.5 Jaccard Coefficient



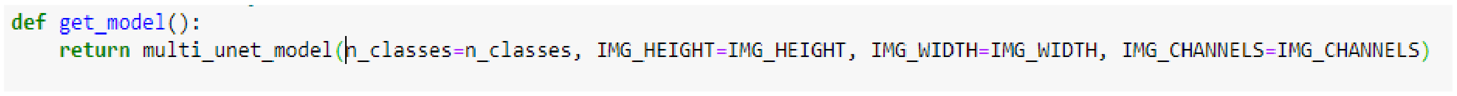
```
def jacard_coef(y_true, y_pred):
    y_true_f = K.flatten(y_true)
    y_pred_f = K.flatten(y_pred)
    intersection = K.sum(y_true_f * y_pred_f)
    return (intersection + 1.0) / (K.sum(y_true_f) + K.sum(y_pred_f) - intersection + 1.0)
```

Figure 5.6 Function to implement Jaccard Coefficient

5.2.3 Function to build U-Net Model

A function named- **multi_unet_model** has been defined to build the model.

To call this function we have defined a new function named- **get_model** (see figure 5.7**).**



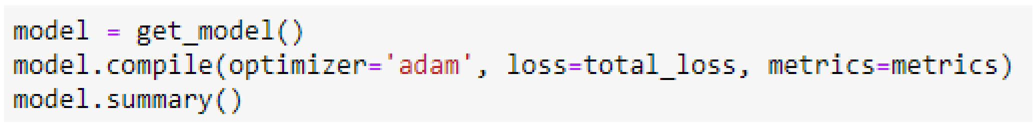
```
def get_model():
    return multi_unet_model(n_classes=n_classes, IMG_HEIGHT=IMG_HEIGHT, IMG_WIDTH=IMG_WIDTH, IMG_CHANNELS=IMG_CHANNELS)
```

Figure 5.7 Function to call multi_unet_model

5.2.4 Building Models using Different Optimizers

We have used three optimizers- Adam, Adamax, and RMSprop

5.2.4.1 Adam Optimizer: Adam optimization is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments. This method is computationally efficient, has little memory requirement, invariant to diagonal rescaling of gradients, and is well suited for problems that are large in terms of data or parameters.



```
model = get_model()
model.compile(optimizer='adam', loss=total_loss, metrics=metrics)
model.summary()
```

Figure 5.8 Model 1 using Adam Optimizer

5.2.4.2 **Adamax Optimizer:** Adamax, a variant of Adam based on the infinity norm, is a first-order gradient-based optimization method. Due to its capability of adjusting the learning rate based on data characteristics, it is suited to learn time-variant processes, e.g., speech data with dynamically changed noise conditions.

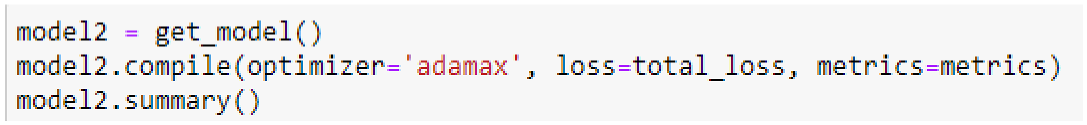

```
model2 = get_model()
model2.compile(optimizer='adamax', loss=total_loss, metrics=metrics)
model2.summary()
```

Figure 5.9 Model 2 using Adamax Optimizer

5.2.4.3 **RMSprop Optimizer:** The RMSprop optimizer is similar to the gradient descent algorithm with momentum. The RMSprop optimizer restricts the oscillations in the vertical direction. Therefore, we can increase our learning rate and our algorithm could take larger steps in the horizontal direction converging faster.

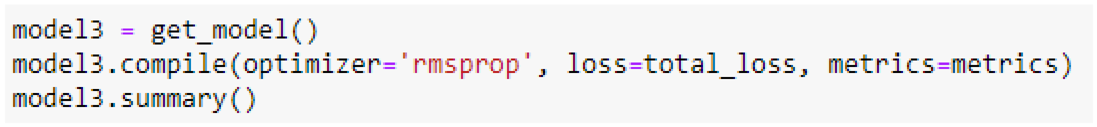

```
model3 = get_model()
model3.compile(optimizer='rmsprop', loss=total_loss, metrics=metrics)
model3.summary()
```

Figure 5.10 Model 3 using RMSprop Optimizer

CHAPTER 6

MODEL EVALUATION AND TESTING

Model evaluation is the process of using different evaluation metrics to understand a machine learning model's performance, as well as its strengths and weaknesses. Model evaluation is important to assess the efficacy of a model during initial research phases, and it also plays a role in model monitoring.

Pixel Accuracy and mIoU are the most common two ways used to evaluate how well an image segmentation model performs.

6.1 COMPARING ALL THE MODELS ON THE BASIS OF

6.1.1 Training and Validation Loss

- **Training Loss:** The training loss is a metric used to assess how a deep learning model fits the training data. That is to say, it assesses the error of the model on the training set. Note that, the training set is a portion of a dataset used to initially train the model. Computationally, the training loss is calculated by taking the sum of errors for each example in the training set. It is also important to note that the training loss is measured after each batch. This is usually visualized by plotting a curve of the training loss.

- **Validation Loss:** On the contrary, validation loss is a metric used to assess the performance of a deep learning model on the validation set. The validation set is a portion of the dataset set aside to validate the performance of the model. The validation loss is similar to the training loss and is calculated from a sum of the errors for each example in the validation set. Additionally, the validation loss is measured after each epoch. This informs us as to whether the model needs further tuning or adjustments or not. To do this, we usually plot a learning curve for the validation loss.

Table 6.1 shows the plots between Training and Validation Loss per Epoch value for all three models and optimizers

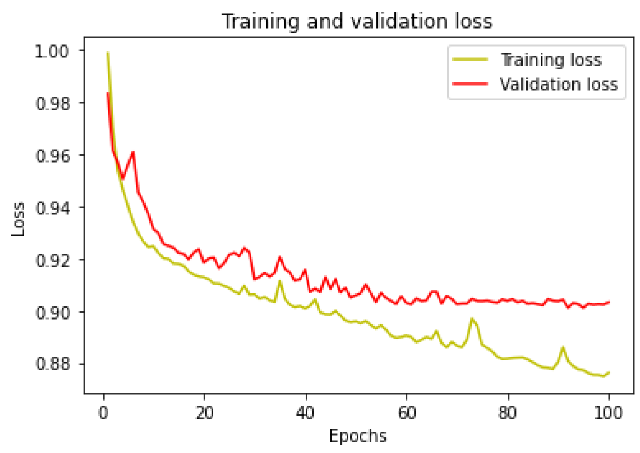
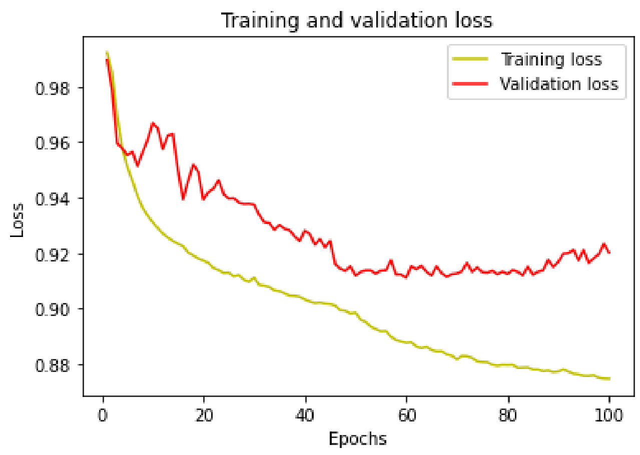
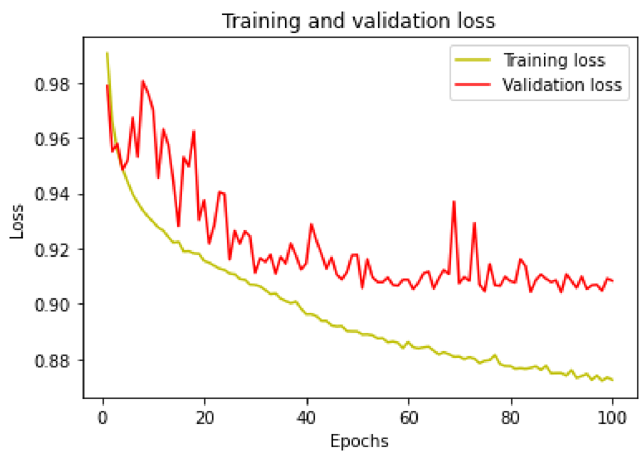
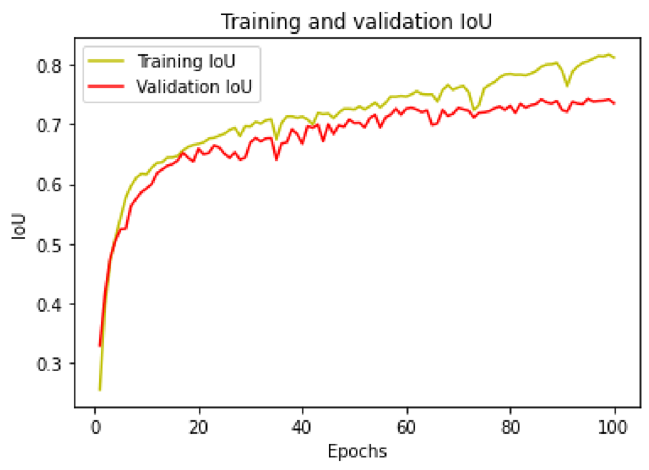
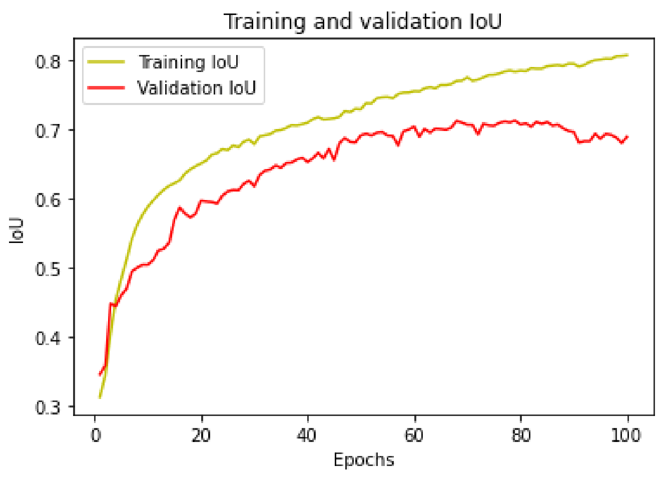
Model	Optimizer	Plot to show Training and Validation Loss Per Epoch value
1. Model_1	Adam	
2. Model_2	Adamax	
3. Model_3	RMSprop	

Table 6.1 Training and Validation Loss per Epoch value

6.1.2 Training and Validation IOU

IoU means Intersection over Union. It is a metric used to evaluate Deep Learning algorithms by estimating how well a predicted mask or bounding box matches the ground truth data. It is used when calculating mAP. It is a number from 0 to 1 that specifies the amount of overlap between the predicted and ground truth bounding box. an IoU of 0 means that there is no overlap between the boxes.

Model	Optimizer	Plot to show Training and Validation IOU Per Epoch value
4. Model_1	Adam	
5. Model_2	Adamax	

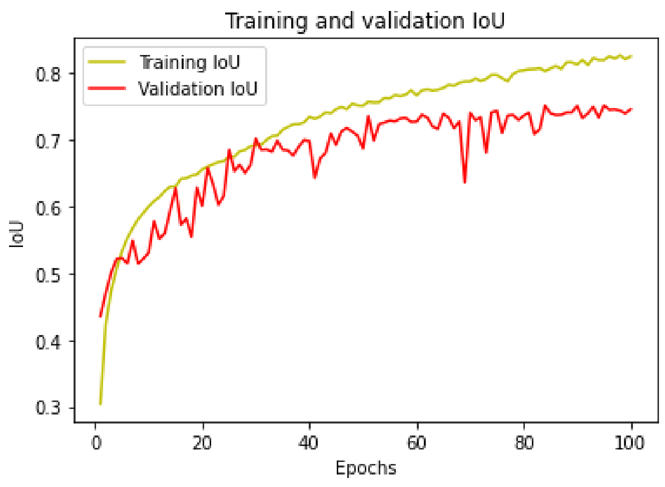
6. Model_3	**RMSprop**	

Table 6.2 Training and Validation IOU per Epoch value

6.1.3 Mean IoU Values

Mean of all the IoU values for 100 epoch values is called Mean IoU, see table 6.3 for more details.

Model	Optimizer	Mean IoU Value
1. Model_1	Adam	**0.61098385**
2. Model_2	Adamax	**0.5555609**
3. Model_3	RMSprop	**0.61034524**

Table 6.3 Mean IoU values of all three models

6.1.4 Predicting Images

6.1.4.1 Predicting images using Model_1 using Adam Optimizer

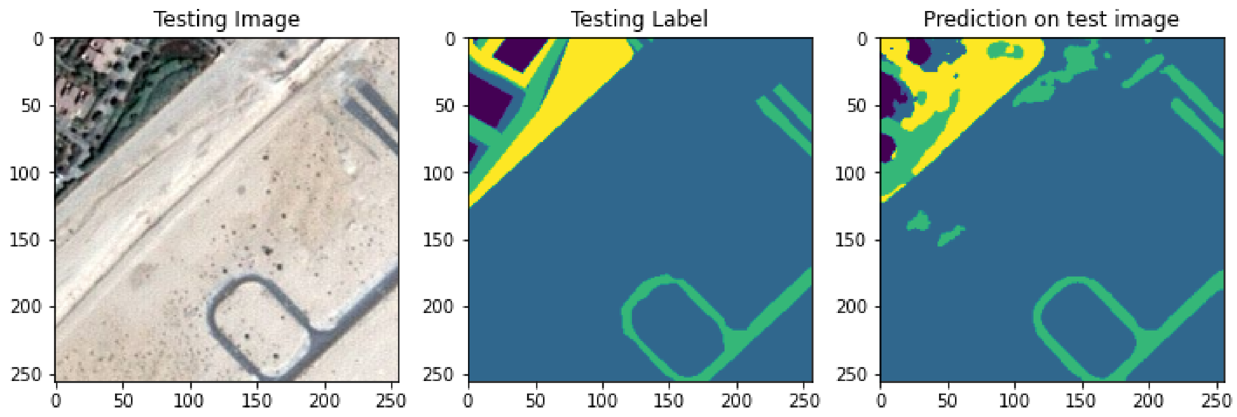


Figure 6.1 Testing using Model_1, Optimizer- Adam

6.1.4.2 Predicting images using Model_2 using Adamax Optimizer

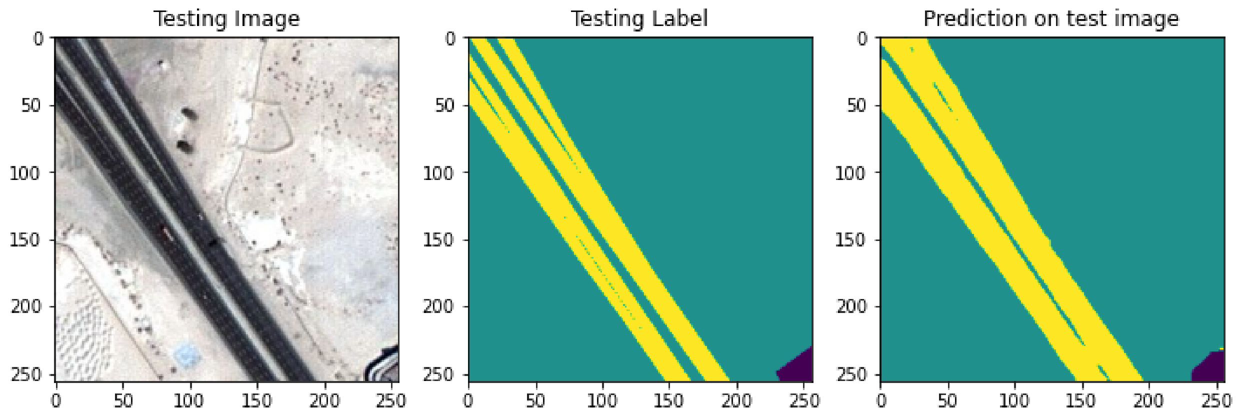


Figure 6.2 Testing using Model_2, Optimizer- Adamax

6.1.4.3 Predicting images using Model_3 using RMSprop Optimizer

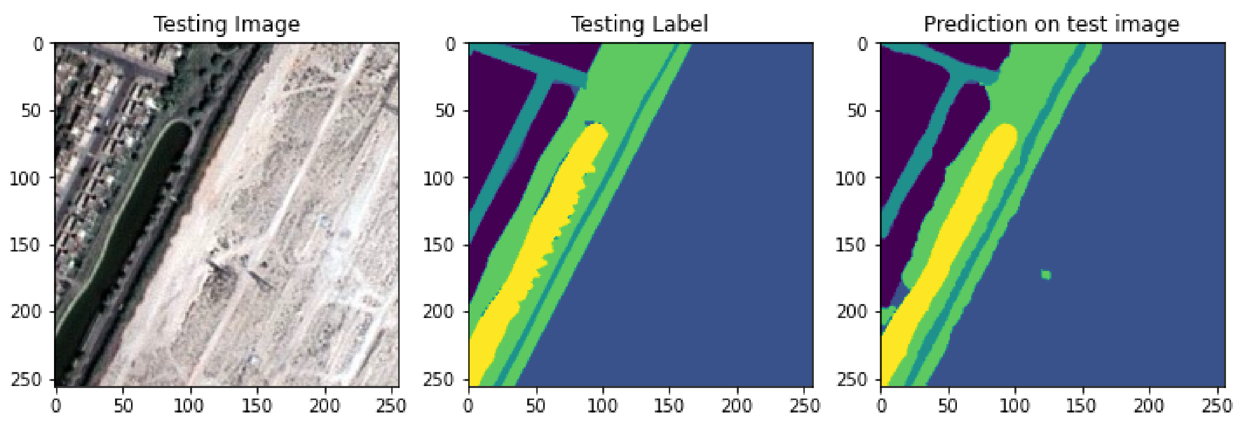


Figure 6.3 Testing using Model_3, Optimizer- RMSprop

6.2 COMPARISON ON THE BASIS OF THEIR PARAMETERS

Model	Optimizer	Number of Epochs	Batch Size	Mean IoU
Model_1	Adam	100	16	**0.61098385**
Model_2	Admax	100	16	**0.5555609**
Model_3	RMSprop	100	16	**0.61034524**

Table 6.4 Comparing Model Parameters

CONCLUSION

Image Augmentation has been successfully done using the Patchify module from Pillow. Total 1305 patches have been extracted out of 72 images.

U-Net feature extraction algorithm has been successfully implemented and the mean IoU values using three optimizers have been compared:

- Adam
- Admax
- RMSprop

Future Scope:

We can implement other algorithms like SegNet, DenseNet, E-Net, Link-Net, Mask RCNN, PSP-Net, RefineNet etc. and compare their performance for gaining better results. We can also work on smooth blending of patches to avoid unnecessary overlapping that affects the visual output.

REFERENCES

- Ming Wu, Chaung Zhang, Jiaming Liu, Lichen Zhou, and Xiaoqi Li, “Towards Accurate High Resolution Satellite Image Semantic Segmentation”, IEEE Access (Volume: 7), Electronic ISSN: 2169-3536, 26 April 2019 .
- Hadi Mansourifar, Alexander Moskovitz, Ben Klingensmith, Dino Mintas, Steven J. Simske, “GAN-Based Satellite Imaging: A Survey on Techniques and Applications”, IEEE Access (Volume: 10), Electronic ISSN: 2169-3536, 09 November 2022.
- https://www.kaggle.com/humansintheloop/semantic-segmentation-of-aerial-imagery
- https://github.com/bnsreenu/python_for_microscopists/tree/master/228_semantic_segmentation_of_aerial_imagery_using_unet
- https://www.tensorflow.org/api_docs/python/tf/keras/optimizers
- https://paperswithcode.com/method/u-net
- https://www.tensorflow.org/tutorials/images/segmentation
- https://medium.com/analytics-vidhya/image-augmentation-9b7be3972e27