Aircraft Detection and Classification by Using Satellite Imagery with AI/Ml techniques

A project report submitted in the internship

by

Raj Rohit (1905071)

BACHELOR OF TECHNOLOGY in Civil Engineering

2022

Department of Civil Engineering

Birsa Institute of Technology Sindri

Aircraft Detection and Classification by Using Satellite Imagery with AI/Ml techniques

A project report submitted in the internship

by

Raj Rohit (1905071)

Guide: ARAVINDA KUMAR P

SCIENTIST/ENGINEER-SG

Head,Bhuvan Content Generation Group

2022

Bhuvan Geo-portal & Web Services Area

National Remote Sensing Centre, Hyderabad

Candidate's Declaration

I declare that this internship report titled Aircraft Detection and Classification by Using Satellite Images for Hyderabad District submitted for the award of the degree Bachelor of Technology in Civil Engineering is a record of original work carried out by me under the supervision of Aravinda Kumar P (Head BCGD/CGVAG/BGWSA, Scientist/Engineer 'SG', NRSC) and has not formed the basis for the award of any degree, diploma, associateship, fellowship, or other titles in this or any other Institution or University of higher learning. In keeping with the ethical practice in reporting scientific information, due acknowledgments have been made wherever the findings of others have been cited

RAJ ROHIT
(1905071)

Place: Hyderabad
Date: November 2022

DEPT. OF CIVIL ENGINEERING

Birsa Institute of Technology Sindri Dhanbad, Jharkhand

Certificate

I hereby certify that the Project titled Aircraft Detection and Classification by Using Satellite Images for Hyderabad District which is submitted by Raj Rohit (1905071) for fulfillment of the requirements for summer internship is a record of the project work carried out by the student under guidance and supervision of Head BCGD/CGVAG/BGWSA, Scientist/Engineer 'SG', NRSC).

The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

ARAVINDA KUMAR P
Head BCGD/CGVAG/BGWSA
SCIENTIST/ENGINEER-SG

Place: Hyderabad
Date: October 2022

Acknowledgments

The successful completion of any task is incomplete and meaningless without giving any due credit to the people who made it possible without which the project would not have been successful and would have existed in theory.

First and foremost, I am beholden to Dr. T Ravisankar, Deputy Director, BGWSA, NRSC to giving me this opportunity.I am grateful to Dr. Ranvijay Kumar Singh, HOD, Department of Civil Engineering, Birsa Institute Of Technology, Sindri, and all other faculty members of our department to giving me this opportunity to carry out this internship project.I am greatful to Srinivasa Rao S, GD CGVAG, Scientist/Engineer 'G', NRSC to giving me this opportunity. I owe a lot of thanks to my NRSC supervisor, Aravind Kumar P, Head BCGD/CGVAG/BGWSA, Scientist/Engineer 'SG' National Remote Sensing Centre (NRSC), for her continuous guidance, motivation and support during this project.

I would also like to show my thanks and gratitude to Amitesh Anand Sci./Engr,'SC' NRSC who directly have given his support and help for this challenging task. Last, but never least, I thank my well-wishers and parents for always being with me, in every sense and constantly supporting me in every possible sense whenever possible.

Abstract

Airplane Classification is one of the major problem in point object classification from High Resolution satellite imagery. Satellite images are optical and non optical. Feature extraction from satellite image is more feasible with line features like road,railways and polygon feature like rivers, solar farms etc. At the same time point features like boat,airplanes are difficult to extract due to less information to noise ration in sample dataset. Simple model will learn less and put lot of bias for point feature extraction whereas complex model will over train and perform poorly for new dataset. Hence careful analysis of sample images has to be performed before putting the dataset for training. Conventional machine learning models and cnn based deep learning models have been explored. Dataset has been gathered from Kaggle which is similar to Bhuvan HR data. Labelling has been done and provided in Json format.

A classifier is presented as the result of this exercise. This will act as a framework for point feature classification from High Resolution Satellite imagery.

Table of Contents

List of Tables and Figures

Chapter 1

Introduction

1.1 About the Organization

National Remote Sensing Centre (NRSC) is one of the primary centres of Indian Space Research Organisation (ISRO), Department of Space (DOS). NRSC at Hyderabad has been converted into a full-fledged centre of ISRO since September 1, 2008. The Centre is responsible for remote sensing satellite data acquisition and processing, data dissem-ination, aerial remote sensing and decision support for disaster management NRSC has the mandate for establishment of ground stations for receiving satellite data, generation of data products, dissemination to the users, development of techniques for remote sensing applications including disaster management support, geospatial services for good gover-nance and capacity building for professionals, faculty and students. National Remote Sensing Centre (NRSC), a constituent of the Indian Space Research Organization (ISRO), located at Hyderabad with four regional remote sensing centres located at Bengaluru, Nagpur, Kolkata Jodhpur and is responsible for:

- Acquisition planning, data reception, processing and dissemination of satellite data from various Indian and foreign satellites
- Carrying out aerial surveys and extending large scale mapping services cartographic mapping and digital data services to the civilian sector.
- Providing geospatial solutions to various user agencies.
- Extending a 24X7 service to support disaster management during floods, agricultural drought, forest fires, cyclones, landslides earthquakes.

- Outreach through training, workshops, student projects promotional activities re-lated to remote sensing applications at regional national level. These allotments are strictly on requirement, depending on the projects/ programs at NRSC.

- Extending services to set up ground stations and processing facilities.

1.2 Bhuvan

Bhuvan (is a Sanskrit word, meaning Earth) is a Geoportal of Indian Space Research Organisation (ISRO), hosted through URL https://bhuvan.nrsc.gov.in. It Provides services and applications related to satellite remote sensing data for public use. Bhuvan Services are offered by NRSC. Striving to realize the Indian Space Vision, is a key player in ISRO's Earth Observation Programme and Disaster Management Support Programme. NRSC is responsible for the acquisition, processing, supply of aerial and Satellite remote sensing data and for continuously exploring the practical uses of remote sensing Technology for multi level (global to local) applications. Bhuvan Cell at NRSC is responsible for the maintenance and offered services by the Bhuvan.

There are several services offered by Bhuvan for central ministries, state ministries and also in application sector.Satellite Data Service and Thematic services are provided by Bhuvan. Satellite images can be visualized on Bhuvan 2d and available as different country wise layers whereas it can be downloaded through NOEDA portal. In Bhuvan various type of data input is available which can be use for feature extraction using AI/ML. These features can be of large size like water bodies, Solar park, agricultural body etc. Or of small size like aeroplane, boat, power grid pole etc.

As the most commonly used metric for classifying optical satellite imagery, spatial resolution refers to the distance represented by a pixel in an image. For example, NASA's Landsat collects imagery at 15-meter resolution—so every pixel in one

of its images represents a 15 m by 15 m square on the ground.There are many layers of satellite imagery available on Bhuvan such as AWifs: Advanced Wide Field Sensor (AWiFS) data products of 56m resolution, LISS III: Linear Imaging Self Scanning sensor data products, 5m data, 2.5m data, High Resolution Data 1m resolution.

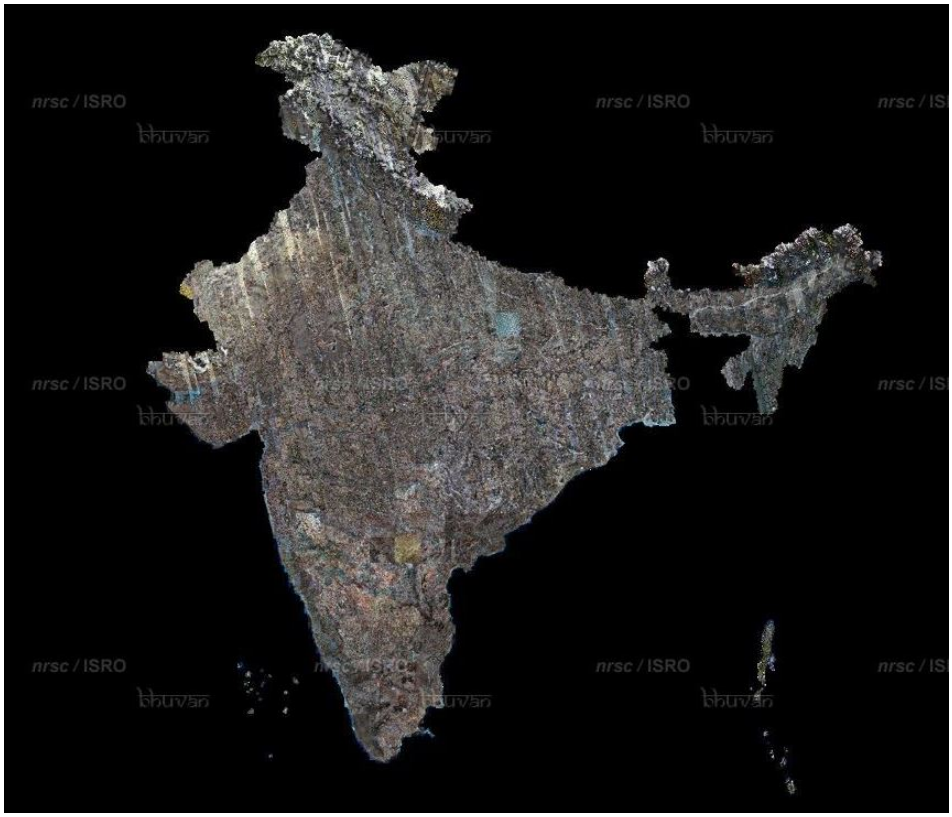

Figure 1.1: HR Data

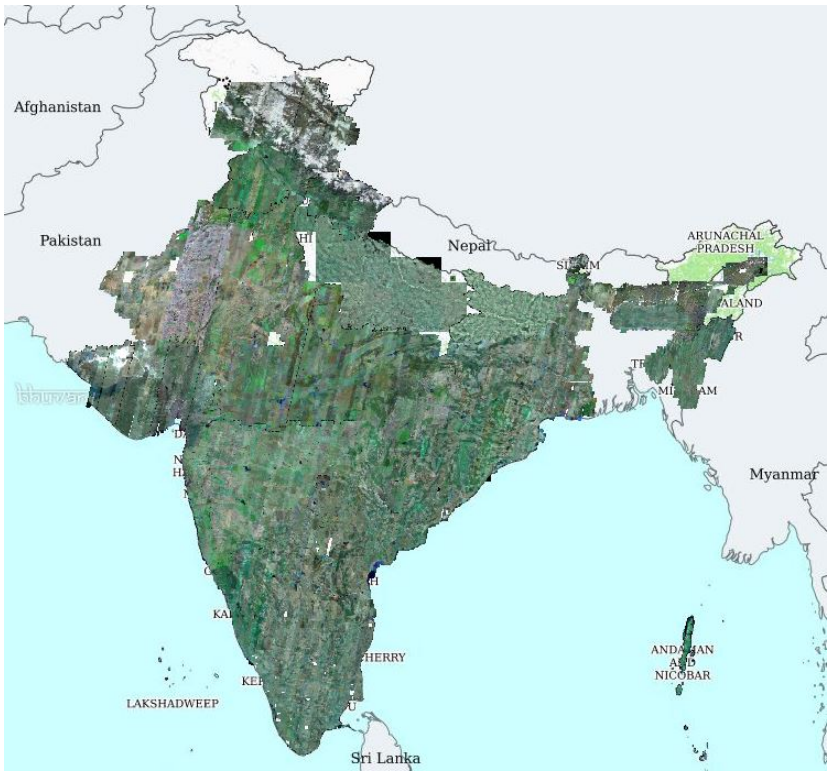

Figure 1.2: 2.5m Data

1.3 Problem Statement and Motivation

Object detection and classification is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos.

It is one of those fields that have witnessed great success.It is used in many areas like face detection(used by Facebook to recognize people), tumor detection(used in medical fields), etc.. It gives frame work to point object in satellite imagery. The major problem we face during analysis is to detect the point object. In satellite imagery it is very difficult to detect the point object due to which a small size object would lose the detailed information. In the larger image also we face challenges during analysis as current GPU memory capacity is insufficient to process large images. Downsampling a large image to a small size would lose the detailed information. To solve this problem the large images can be simply

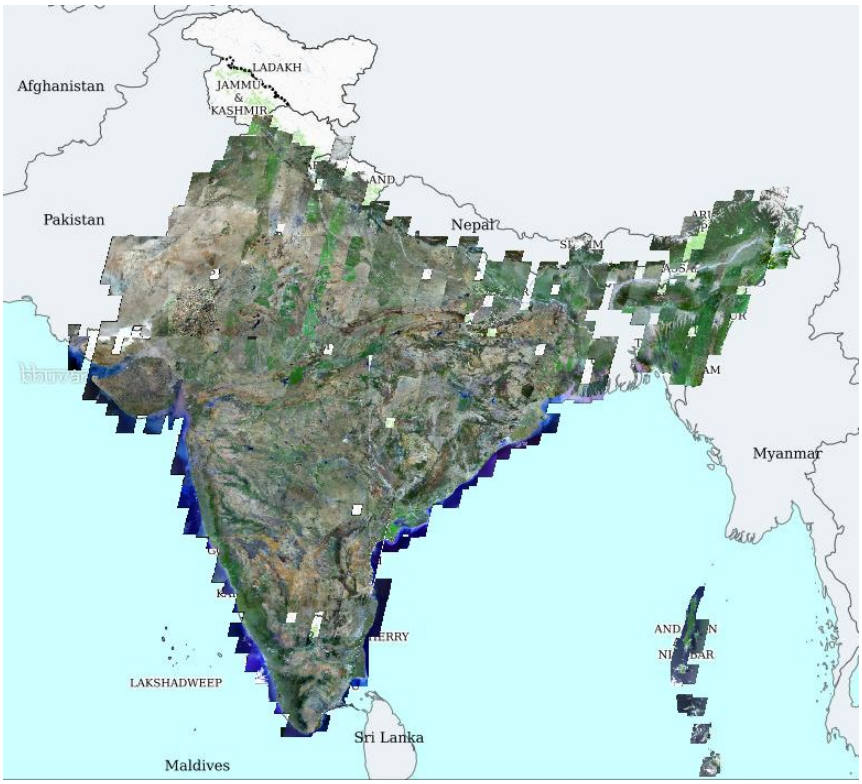

Figure 1.3: 5m Data

split into small patches. After obtaining the results on these patches, the results are integrated into large images. To speed up inference on large images, these methods first find regions that are likely to contain instances in the large images and then detect objects in the regions.

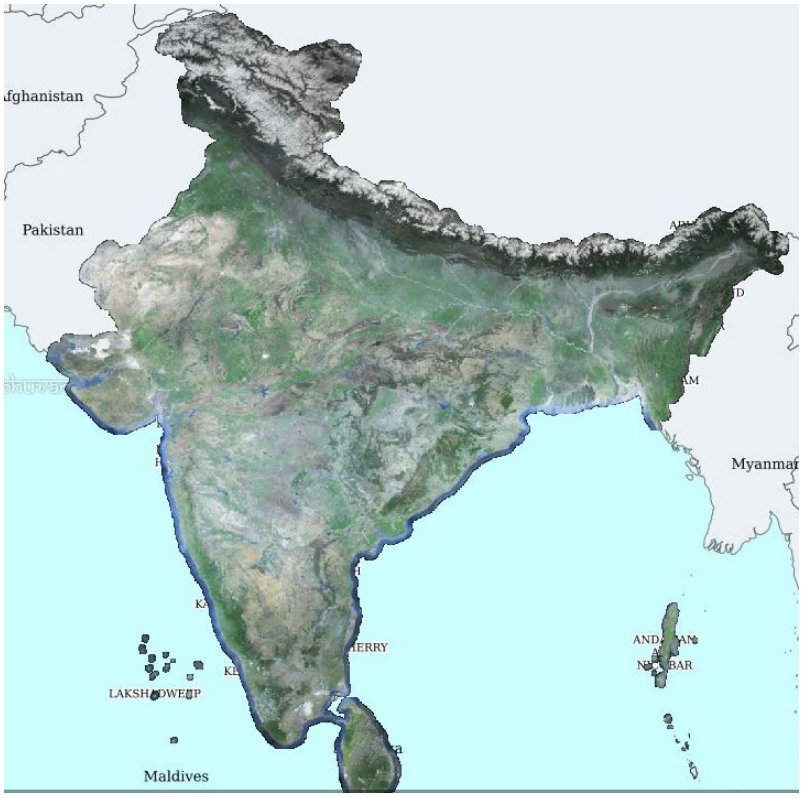

Figure 1.4: LISS III Data

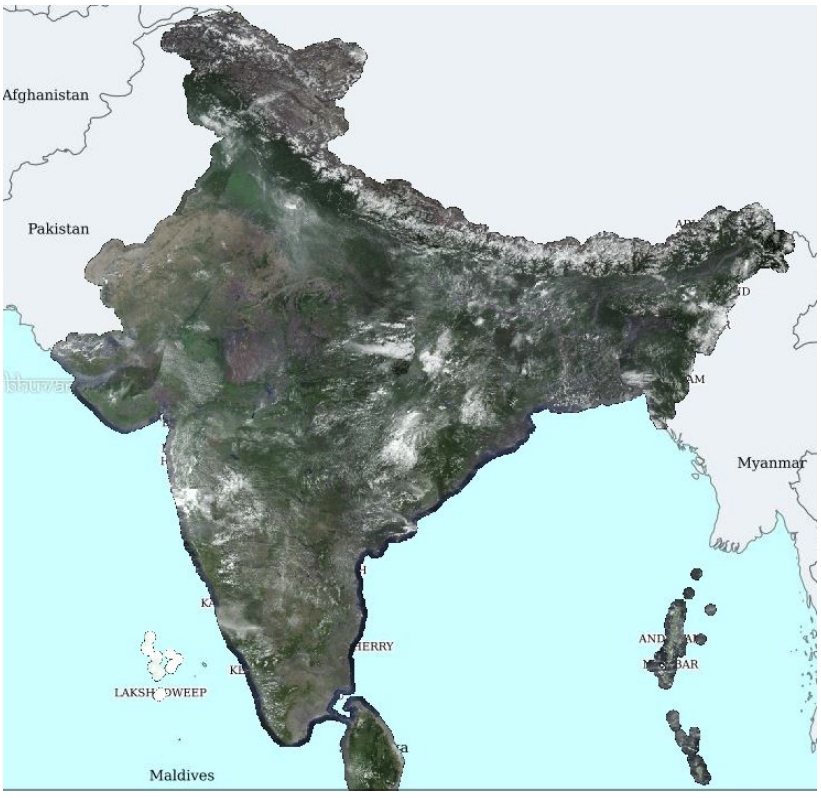

Figure 1.5: AWIfs Data

Chapter 2

Literature Survey

Feature extraction methods from satellite images ranges from classical machine learning methods to deep neural networks. Pre-processing and Post-processing methods can contribute more to inference accuracy depending upon the type of dataset.Most papers have followed manual and semi-automated methods for labelling the dataset. Edge filters and basic machine learning methods have been replaced with deep neural networks. Accuracy varies greatly with spatial characteristics and quality of dataset acquired. Occlusions like cloud, aerosols also affect the inference capability of model Here few papers have been discussed to make a solid foundation for the problem statement.

2.1 Airplanes Detection for Satellite using Faster RCNN

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos. Ever since the inception of deep learning in computer vision, tasks like object detection have become comparatively easier and efficient. The deep learning models provide better accuracy, less time consumption, less complexity, overall better performance than the earlier computer vision approaches. Deep learning provided outstanding results over the traditional computer vision methods for object detection, leading to the wide use of deep learning models.In this method the datasets is taken from kaggle which contains computer-generated satellite images of planes as shown in

figure(x) and then we use data augmentation, It is a technique to artificially create new training data from existing training data. It also generates new images for training and hence improving the model performance .At last by using faster RCNN we detect the aircrafts very easily, It is one of the best performing object detection. The original goal of R-CNN is to take an input image and produce a set of bounding boxes as a output as shown in figure(x), where each bounding box contains an object and also the category (eg. Car, or pedestrian) of the object. More recently, R-CNN has been extended to perform other computer vision tasks.

2.2 Aircraft detection in satellite imagery using deep Learning-based object detectors

Identifying the precise location of an object in a larger scene, known as object detection, is a fundamental step in many high level vision tasks. As one of the primary computer vision problems, object detection provides key information for a better understanding of images and videos and is being widely used to resolve tasks such as image classification, face recognition and autonomous driving. Detecting potential targets in high-resolution satellite images has a wide range of applications in military and homeland surveillance. Several studies have addressed this problem using machine learning methods. However, due to the high complexity of satellite imagery, accurate and efficient detection of aircraft remains a challenging task. In satellite images, objects occur in multiple orientations and have large appearance variations thus creating performance limitations for most of the previous approaches.

To recognize different objects, initially it is required to extract visual features that provide semantic representation of the object. The extracted features help distinguish a target object from other categories using a classifier. In most of the traditional object detection approaches, detectors are trained using various

hand-crafted features including Haar-like features.

Over the past few years, Convolutional Neural Network (CNN)-based object detectors have gained popularity in research community due to its ability to automatically compute complex contextual image features . The current forefront methods for object detection based on CNN can be broadly categorized as region-based two-stage detectors or regression-based single stage detectors. Examples of region-based detectors include Region-based CNN (RCNN) , Fast RCNN and Faster RCNN , while detectors such as You Only Look Once (YOLO) and Single Shot MultiBox Detection (SSD) are examples of regression-based detectors. Regression-based models are generally less accurate as compared to the region-based detectors, however, regression-based detectors are significantly faster as compared to region-based detectors. Efforts have been made by researchers to develop new CNN-based models for improved performance and efficiency . These models have deeper architectures and are capable of learning more coarse features automatically. Our work is motivated by the recent visual object detection approaches rooted in rich features extracted using CNN.

This paper provides a comparative analysis of some of the well-known object detectors based on CNN when applied to the problem of aircraft detection in satellite imagery.

2.3 Stationary aircraft detection from satellite images

The Detection and recognition of objects from satellite images is one of the important research areas in the field. Satellite image analysis can be useful in many applications such as water and climate observation, land cover classification and energy exploration. Especially, the detection of stationary aircrafts can be strategically important in military applications. This type of information provides more robust and successful decision mechanism in dynamically chang-

ing military operations. Detection of aircrafts from satellite images using image processing algorithms highly depends on learning the geometrical structure of aircrafts. Making this structure more evident plays important role for the detection of these aircrafts from satellite images.

The objective of this study is to develop a learning based system that detects stationary aircrafts in satellite images obtained from Google Earth. Google Earth is a virtual globe, map and geographical information program that maps the Earth by the superimposition of images obtained from satellite imagery. The features that emphasize the geometric structure of an aircraft are determined using Gabor filter. A set of Gabor filters with different frequencies and orientations is useful for extracting useful image features from an image. Frequency and orientation representations of Gabor filters are similar to representations of the human visual system, and they are widely used in many preprocessing steps of image processing systems that include texture representation and discrimination. In the spatial domain, a 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave. In this study, a learning-based aircraft detection system is developed using 2D Gabor features obtained from satellite images. The detection is performed by using Support Vector Machines (SVM) classification method. The SVM is a supervised learning method that analyzes data and recognizes patterns for classification. The SVM is used in many applications including optical character recognition, data mining face recognition, facial expression analysis and biomedical.

As evident from above discussions CNN based models are more successful as far as satellite imagery classification is considered.

Chapter 3

Dataset Info and Data visualization

3.1 Context

Satellite imagery provides unique insights into various markets, including agriculture, defense and intelligence, energy, and finance. New commercial imagery providers, such as Planet and BlackSky, are using constellations of small satellites to exponentially increase the amount of images of the earth captured every day. This flood of new imagery is outgrowing the ability for organizations to manually look at each image that gets captured, and there is a need for machine learning and computer vision algorithms to help automate the analysis process.

The aim of this dataset is to help address the difficult task of detecting the location of airplanes in satellite images. Automating this process can be applied to many issues including monitoring airports for activity and traffic patterns, and defense intelligence.

Continuously updates will be made to this dataset as new Planet imagery released. Current images were collected as late as July 2017.

3.2 Content

Provided is a zipped directory planesnet.zip that contains the entire dataset as .png image chips. Each individual image filename follows a specific format: label_scene id longitude_latitude.png

- label: Valued 1 or 0, representing the "plane" class and "no-plane" class, respectively
- scene id: The unique identifier of the PlanetScope visual scene the image chip was extracted from. The scene id can be used with the Planet API to discover and download the entire scene
- longitude_latitude: The longitude and latitude coordinates of the image center point, with values separated by a single underscore.

The dataset is also distributed as a JSON formatted text file planesnet.json. The loaded object contains data, label, scene_ids, and location lists.
The pixel value data for each 20x20 RGB image is stored as a list of 1200 integers within the data list. The first 400 entries contain the red channel values, the next 400 the green, and the final 400 the blue. The image is stored in row-major order, so that the first 20 entries of the array are the red channel values of the first row of the image.
The list values at index i in labels, scene_ids, and locations each correspond to the i-th image in the data list.

3.3 Class Labels

The "plane" class includes 8000 images. Images in this class are near-centered on the body of a single airplane, with the majority of the plane's wings, tail, and nose also visible. Examples of different aircraft sizes, orientations, and atmospheric collection conditions are included. Example images from this class are shown below.

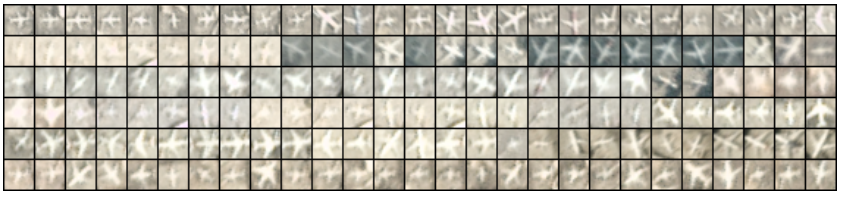

Figure 3.1: Plane class Labelled images

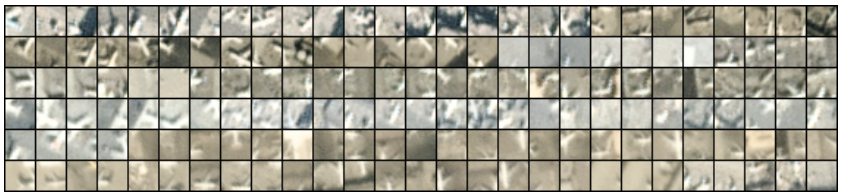

Figure 3.2: Plane class Labelled images

The "no-plane" class includes 24000 images. A third of these are a random sampling of different landcover features - water, vegetion, bare earth, buildings, etc. - that do not include any portion of an airplane. The next third are "partial planes" that contain a portion of an airplane, but not enough to meet the full definition of the "plane" class. The last third are "confusers" - chips with bright objects or strong linear features that resemble a plane - that have previously been mislabeled by machine learning models. Example images from this class are shown below.

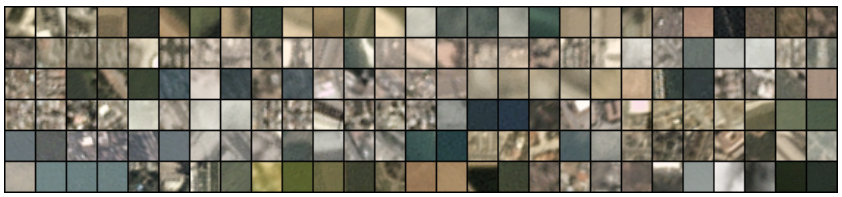

Figure 3.3: No Plane class Labelled images

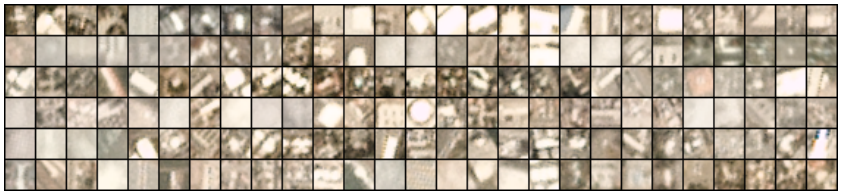

Figure 3.4: No Plane class Labelled images

Chapter 4

EDA - Exploratory Data Analysis

4.1 Displaying and Saving Image Chips

The dataset is normally given as image and corresponding mask. Although modern standards dictates that JSON can be given to improve interoperability.Here we have demonstrated an overview on how to load the JSON version of the dataset and access individual images chips using PIL,numpy and json Python packages.The dataset can be loaded into a Python dictionary object.Each individual image is contained as a 1200 value integer list within the **data** list. Images can be recreated by reforming this list into a 20x20x3 numpy array.

4.2 Data preprocessing techniques

Data preprocessing is a Data Mining method that entails converting raw data into a format that can be understood. Real-world data is frequently inadequate, inconsistent, and/or lacking in specific activities or trends, as well as including numerous inaccuracies. This might result in low-quality data collection and, as a result, low-quality models based on that data. Preprocessing data is a method of resolving such problems.

Machines do not comprehend free text, image, or video data; instead, they comprehend 1s and 0s. So putting on a slideshow of all our photographs and expecting our machine learning model to learn from it is probably not going to be adequate.Data Preprocessing is the step in any Machine Learning process in which the data is changed, or encoded, to make it easier for the machine to parse

it. In other words, the algorithm can now easily interpret the data's features.Data Preprocessing can be done in four different ways. Data cleaning/cleaning, data integration, data transformation, and data reduction are the four categories.

- Data Cleaning
- Data Integration
- Data Transformation
- Data Reduction

4.2.1 Data Cleaning

Data in the real world is frequently incomplete, noisy, and inconsistent. Many bits of the data may be irrelevant or missing. Data cleaning is carried out to handle this aspect. Data cleaning methods aim to fill in missing values, smooth out noise while identifying outliers, and fix data discrepancies. Unclean data can confuse data and the model. Therefore, running the data through various Data Cleaning/Cleansing methods is an important Data Preprocessing step.

- Missing Data: It's fairly common for your dataset to contain missing values. It could have happened during data collection or as a result of a data validation rule, but missing values must be considered anyway.

    1. Dropping rows/columns: If the complete row is having NaN values then it doesn't make any value out of it. So such rows/columns are to be dropped immediately. Or if the % of row/column is mostly missing say about more than 65% then also one can choose to drop.
    2. Checking for duplicates: If the same row or column is repeated then also you can drop it by keeping the first instance. So that while running machine learning algorithms, so as not to offer that particular data object an advantage or bias.

3. Estimate missing values: If only a small percentage of the values are missing, basic interpolation methods can be used to fill in the gaps. However, the most typical approach of dealing with missing data is to fill them in with the feature's mean, median, or mode value.

- Noisy Data: Noisy data is meaningless data that machines cannot interpret. It can be caused by poor data collecting, data input problems, and so on. It can be dealt with in the following ways:

  1. Binning Method: This method smooths data that has been sorted. The data is divided into equal-sized parts, and the process is completed using a variety of approaches. Each segment is dealt with independently. All data in a segment can be replaced by its mean, or boundary values can be used to complete the task.

  2. Clustering: In this method, related data is grouped in a cluster. Outliers may go unnoticed, or they may fall outside of clusters.

4.2.2 Data Integration

It is involved in a data analysis task that combines data from multiple sources into a coherent data store. These sources may include multiple databases. Do you think how data can be matched up ?? For a data analyst in one database, he finds Customer_ID and in another he finds cust_id, How can he sure about them and say these two belong to the same entity. Databases and Data warehouses have Metadata (It is the data about data) it helps in avoiding errors.

4.2.3 Data Transformation

This stage is used to convert the data into a format that can be used in the mining process. This is done in the following ways:

- Normalization: It is done to scale the data values in a specified range (-1.0

to 1.0 or 0.0 to 1.0). It also helps in reducing processing load while training and testing.

- Concept Hierarchy Generation: Using concept hierarchies, low-level or primitive/raw data is substituted with higher-level concepts in data generalization. Categorical qualities, for example, are generalized to higher-level notions such as street, city, and nation. Similarly, numeric attribute values can be translated to higher-level concepts like age, such as youthful, middle-aged, or elderly.

- Smoothing: Smoothing works to remove the noise from the data. Such techniques include binning, clustering, and regression.

- Aggregation: Aggregation is the process of applying summary or aggregation operations on data. Daily sales data, for example, might be combined to calculate monthly and annual totals. Feature Aggregation — If the features are highly correlated or if the features can be aggregated into another single feature then it is worth doing it. For example in the dataset you have the height and width of an object then they can be featured into a single feature area. This decreases dimensionality. These types of features are highly correlated in nature as a result it also decreases multicollinearity.

4.2.4 Data Reduction

When dealing with large amounts of data, analysis becomes more difficult. We employ a data reduction technique to get rid of this. Its goal is to improve storage efficiency while lowering data storage and analysis expenses.

- Dimensionality Reduction: A huge number of features may be found in most real-world datasets. Consider an image processing problem: there could be hundreds of features, also known as dimensions, to deal with. As the name suggests, dimensionality reduction seeks to minimize the number of features — but not just by selecting a sample of features from the feature

set, which is something else entirely — Feature Subset Selection or feature selection.

- Numerosity Reduction: Data is replaced or estimated using alternative and smaller data representations such as parametric models (which store only the model parameters rather than the actual data, such as Regression and Log-Linear Models) or non-parametric approaches (e.g. Clustering, Sampling, and the use of histograms).

4.2.5 Image preprocessing

While basic concept remains same Image processing is little bit different from text and speech processing.The term "image pre-processing" refers to actions on images at the most basic level. If entropy(degree of randomness) is an information metric, these methods do not improve image information content, but rather decrease it. Pre-processing aims to improve image data by suppressing unwanted distortions or enhancing particular visual properties that are important for subsequent processing and analysis.
Steps to perform for image pre-processing are as follows:

- Read image - Read the images

- Resize image - Resize the images because the image size captured and fed to the model is different. So it is good to establish a base size and resize the images

- Remove noise(Denoise) - Using Gaussian blur inside the function processing() we can smooth the image to remove unwanted noise.

- Segmentation & Morphology(smoothing edges)— We will segment the image in this stage, separating the background from foreground objects, and then we will refine our segmentation with more noise removal.

There are 4 different types of Image Pre-Processing techniques and they are listed below.

- Pixel brightness transformations/ Brightness corrections like Gamma correction or Power Law Transform,Histogram equalization,Sigmoid stretching
- Geometric Transformations
- Image Filtering and Segmentation
- Fourier transform and Image restauration

Chapter 5

Terminologies & Background for ML/DL

Before, we start working on our machine learning models there are few terminologies that we need to know to better understand the model. This chapter will contain all the essential knowledge required before we go to the model building task.

5.1 What is Machine Learning and Model?

The most basic, question that arises is what is this machine learning and model that is being used. Essentially, machine learning is a core sub-area of Artificial Intelligence (AI). ML applications learn from experience (or to be accurate, data) like humans do without direct programming. When exposed to new data, these applications learn, grow, change, and develop by themselves. In other words, machine learning involves computers finding insightful information without being told where to look. Instead, they do this by leveraging algorithms that learn from data in an iterative process. Machine Learning Model is a program that can find patterns or make decisions from a previously unseen dataset. It uses mathematical method to find patterns in a set of data called algorithms.

There are three type of machine learning techniques supervised, unsupervised and reinforcement learning. We will focus on supervised learning as this is what we will do. Supervised learning is a technique where we train our machine learning model with labelled data. Labelled data means that we explicitly tell our model

what is the output for a given set of features/inputs. When we tell our model the input and output, it learns the underlying pattern and use that to predict output on unseen data.

5.2 Classical Machine Learning Models

It refers to the process of building algorithms that can learn from existing observations (or data sets), and leverage that learning to predict new observations, or determine the output of new input. The basic idea of these algorithms is to learn from it's own mistakes and try to model the output feature given the input features.

5.3 Supported Vector Machine

The original SVM model was invented by Vladimir Vapnik and Alexey Chervonenkis in 1963. In 1992, a more powerful model was proposed by Bernhard Boser, Isabelle Guyon and Vladimir Vapnik, which can create nonlinear classifiers by applying the kernel trick to maximum-margin hyperplanes. In this report, we will focus on the linear SVM in binary classification.

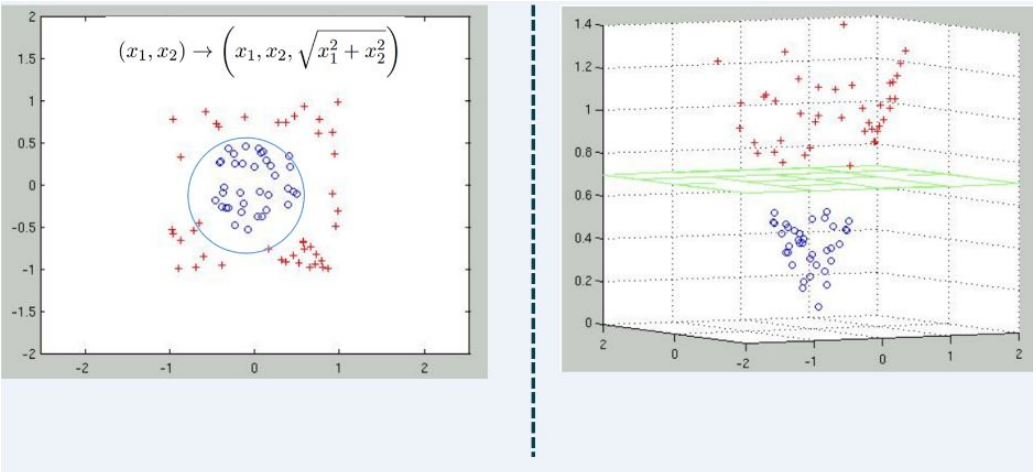

Figure 5.1: Mapping data into higher dimensional space

5.3.1 Linear SVM

In most cases it is impossible for all the points in the training data to follow the rules. To solve this problem, people add this constraint as a regularization part in the object function which we want to minimize. First of all, consider the training set of n points $\{\mathbf{x}_i, y_i\}$ $i = 1, \ldots, n$ where $y_i \in \{-1, 1\}$ is the class of the point x_i. Here, we see the two classes as $y_i = 1$ and $y_i = -1$. We want to find a classifier (a hyperplane) which can map x_i's into higher dimensional space so that the two different classes of points can be divided. Also we want the hyperplane to have the maximum-margin, which can maximize the distance of the hyperplane and nearest points from both groups.

In linear cases, the hyperplane can be written as:

$$\mathbf{x}_i\mathbf{w} + b = 0$$

We want to find two parallel hyperplanes that can also separate the data and we want their distance to be as large as possible. Here's a way to describe them:

$$\mathbf{x}_i\mathbf{w} + b = +1$$

and

$$\mathbf{x}_i\mathbf{w} + b = -1$$

It's not difficult to prove that the distance between the two hyperplanes is $\frac{2}{||w||}$:

$$d_+ + d_- = \frac{|1 - b|}{\|w\|} + \frac{|-1 - b|}{\|w\|} = \frac{2}{\|w\|} \tag{5.1}$$

which means we want to minimize $\|w\|$ since it is always positive. Besides, we want for every $i \in (1, n)$, x_i and y_i follows the constraints:

$$\mathbf{x}_i\mathbf{w} + b \geq +1, \quad y_i = +1 \tag{5.2}$$

$$\mathbf{x}_i\mathbf{w} + b \leq -1, \quad y_i = -1 \tag{5.3}$$

$$\equiv \tag{5.4}$$

$$y_i(\mathbf{x}_i\mathbf{w} + b) - 1 \geq 0, \qquad \forall i \tag{5.5}$$

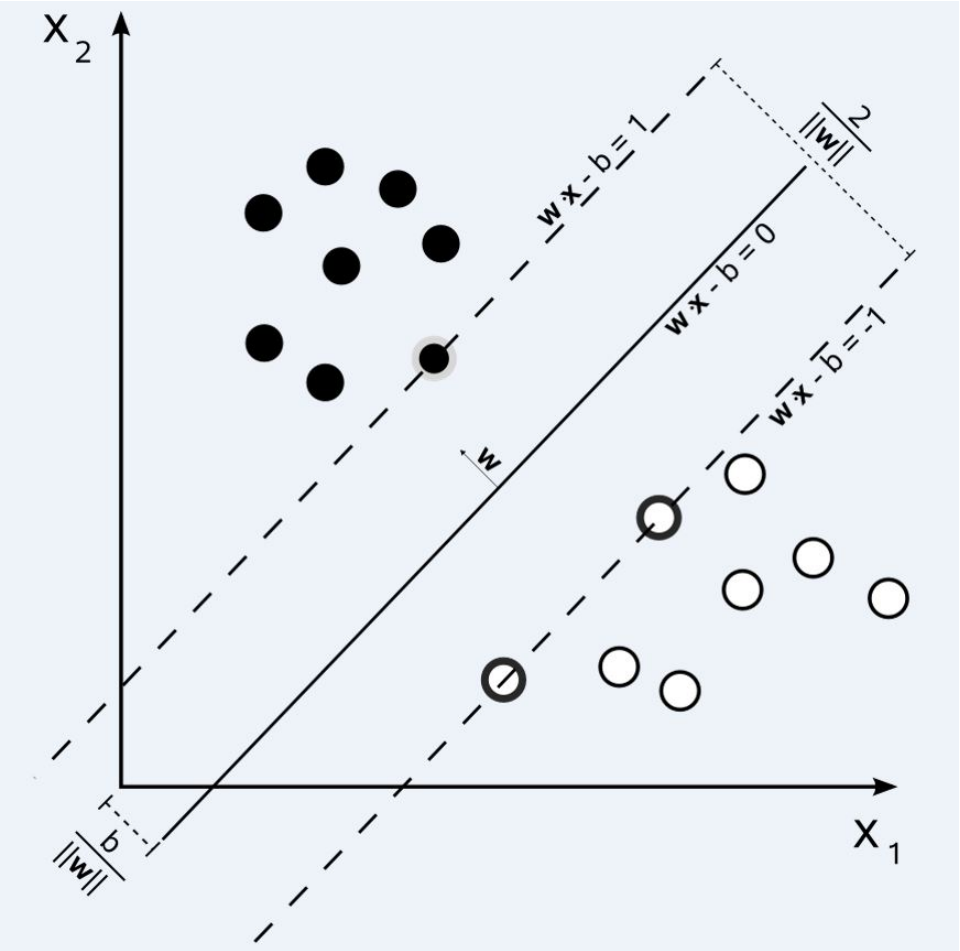

Figure 5.2: Two hyperplanes

5.3.2 Lagrange Multiplier

As mentioned before, we want to maximize the margin, which is the same as minimize $\|w\|$. Also we want them data points to be properly classified.

Thus we want to solve the following optimization problem:

$$\text{Minimize } \frac{\|w\|^2}{2}$$
$$\text{s.t } y_i(\mathbf{x}_i\mathbf{w} + b) - 1 \geq 0$$

What we want is to find the w and b that can make this statement true. However, it is almost impossible to solve this equation directly. In order to solve this optimization problem, we need to introduce the Lagrange Multiplier.

In mathematical optimization, the method of Lagrange multipliers is a strategy for finding the local maxima and minima of a function subject to equality constraints.

For the case of only one constraint and only two choice variables, consider the optimization problem:

$$\text{Minimize } f(x, y)$$
$$\text{s.t } g(x, y) = 0$$

This looks exactly like the problem that we want to solve and we can switch the problem into a lagrangian problem. For convenience, I will skip the mathematical proof part and therefore we can change the problem into:

$$\text{Minimize } W_p(\alpha) = \frac{\|w\|^2}{2} - \sum_{i=1}^{l} \alpha_i y_i(\mathbf{x}_i\mathbf{w} + b) + \sum_{i=1}^{l} \alpha_i$$

We can see this as a convex quadratic programming problem with the dual:

$$\text{Maximize } W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \mathbf{x}_j)$$

The reason that we make this transformation is that the solution of dual problem can be easily approximate with computer. Although there will always be a small gap between the approximated result and the real answer, it is still very powerful and useful.

5.3.3 Finding Classifier

Usually, most α_i's are 0 and the points with $\alpha_i > 0$ are called "supported vectors". After we find the α_i's, we need to use them to compute the vector w and the scalar

b. But how? In the dual problem, we want to:

$$\text{maximize } W(\alpha)$$

where,

$$W(\alpha) = \sum_{j=1}^{n} \alpha_j - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j$$

and the $\alpha \geq 0$ satisfying:

$$\sum_{j=1}^{n} \alpha_j y_j = 0$$

Using KKT-conditions for optimality,

$$\begin{aligned} \nabla_w L &= 0, \text{i.e., } w = \sum_{j=1}^{n} \alpha_j y_j x_j \\ \nabla_b L &= 0, \text{i.e., } \sum_{j=1}^{n} \alpha_j y_j = 0 \end{aligned}$$

Once an optimal α is obtained we find w as the linear classifier of the x_i's and we can also find b ,

$$b = \frac{-\sum_{i,j} \alpha_i \alpha_j y_i y_j < x_i, x_j >}{\sum_j \alpha_j}$$

5.4 Random Forest Classifier

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on

the majority votes of predictions, and it predicts the final output.
The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

5.4.1 Advantages of Random Forest Classifier

- Random Forest is capable of performing both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

5.4.2 Disadvantages of Random Forest Classifier

- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

5.5 Deep Learning Metods

5.5.1 Single Neuron Perceptron

A Neural Network is combinations of basic Neurons — also called perceptrons (arranged in multiple layers as a network . To understand the working and power of a large network, first we need to understand the working and power of a single unit. The input layer has n nodes and each node has weight w_i attached to it. Single layer perceptron take these information and add some bias to it before giving output. The difference between predicted output and original output is called cost/error.Now objective is to adjust weights and bias to minimize this error. Gradient descent is used for this task.First we take derviatives of cost with respect to weights.These derivative are called Gradients. Basically a Derivative of a function at some point is a tangent or Gradient of that function at that point.Now to reach at the minima we will start taking small steps towards the

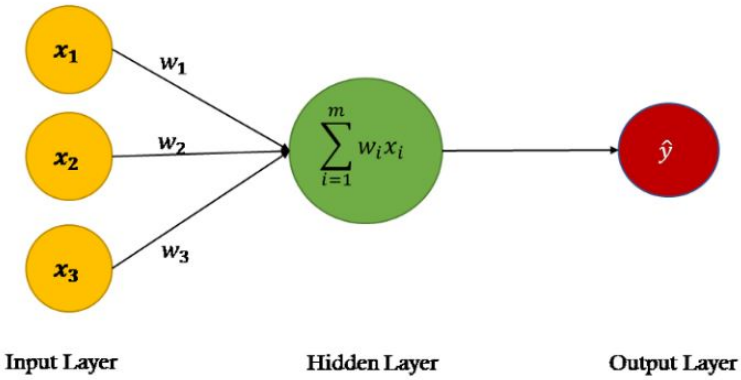

Figure 5.3: Single Neuron Perceptron

direction of minima (which means opposite the direction of Gradient) by updating weights by a small amount each time. In short we are descending in opposite direction of tangent or Gradient — that is why the name "Gradient Descent".

5.5.2 Activation Functions

The activation function is an important part of an artificial neural network. They basically decide whether a neuron should be activated or not. Thus it bounds the value of the net input. The activation function is a non-linear transformation that we do over the input before sending it to the next layer of neurons or finalizing it as output.

- Sigmoid function: A weighted sum of inputs is passed through an activation function and this output serves as an input to the next layer. When the activation function for a neuron is a sigmoid function it is a guarantee that the output of this unit will always be between 0 and 1

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

- Tanh function: Tanh Activation is an activation function used for neural networks.Historically, the tanh function became preferred over the sigmoid

function as it gave better performance for multi-layer neural networks.

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

- Softmax function: The softmax function, also known as softargmax or normalized exponential function, converts a vector of K real numbers into a probability distribution of K possible outcomes. It is a generalization of the logistic function to multiple dimensions, and used in multinomial logistic regression.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for\ i = 1, 2, \ldots, K$$

- ReLu function: A general problem with both the sigmoid and tanh functions is that they saturate. This means that large values snap to 1.0 and small values snap to -1 or 0 for tanh and sigmoid respectively. Further, the functions are only really sensitive to changes around their mid-point of their input, such as 0.5 for sigmoid and 0.0 for tanh.The limited sensitivity and saturation of the function happen regardless of whether the summed activation from the node provided as input contains useful information or not. Once saturated, it becomes challenging for the learning algorithm to continue to adapt the weights to improve the performance of the model. Layers deep in large networks using these nonlinear activation functions fail to receive useful gradient information. Error is back propagated through the network and used to update the weights. The amount of error decreases dramatically with each additional layer through which it is propagated, given the derivative of the chosen activation function. This is called the vanishing gradient problem and prevents deep (multi-layered) networks from learning effectively.

$$Relu(z) = max(0, z)$$

5.5.3 Backpropagation Algorithm

To train a neural network, there are 2 passes (phases): Forward pass and Backward Pass.In the forward pass, we start by propagating the data inputs to the input layer, go through the hidden layer(s), measure the network's predictions from the output layer, and finally calculate the network error based on the predictions the network made.Note that the process of propagating the inputs from the input layer to the output layer is called forward propagation. Once the network error is calculated, then the forward propagation phase has ended, and backward pass starts.In the backward pass, the flow is reversed so that we start by propagating the error to the output layer until reaching the input layer passing through the hidden layer(s). The process of propagating the network error from the output layer to the input layer is called backward propagation, or simple backpropagation. The backpropagation algorithm is the set of steps used to update network weights to reduce the network error.Backpropagation algorithms are used extensively to train feedforward neural networks in areas such as deep learning. They efficiently compute the gradient of the loss function with respect to the network weights. This approach eliminates the inefficient process of directly computing the gradient with respect to each individual weight. It enables the use of gradient methods, like gradient descent or stochastic gradient descent, to train multilayer networks and update weights to minimize loss.It does not have any parameters to tune except for the number of inputs.It is highly adaptable and efficient and does not require any prior knowledge about the network.It is user-friendly, fast and easy to program.

5.5.4 Convolutional Neural Networks

A Convolutional Neural Network, also known as CNN or ConvNet, is a class of neural networks that specializes in processing data that has a grid-like topology, such as an image. A digital image is a binary representation of visual data. It

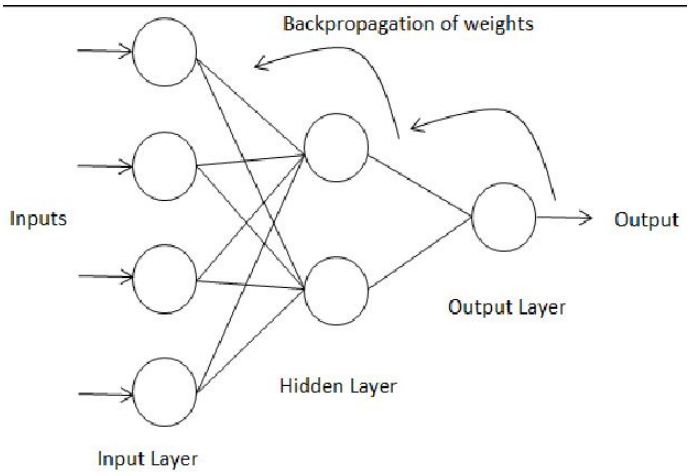

Figure 5.4: Backpropagation Algorithm

contains a series of pixels arranged in a grid-like fashion that contains pixel values to denote how bright and what color each pixel should be.

The human brain processes a huge amount of information the second we see an image. Each neuron works in its own receptive field and is connected to other neurons in a way that they cover the entire visual field. Just as each neuron responds to stimuli only in the restricted region of the visual field called the receptive field in the biological vision system, each neuron in a CNN processes data only in its receptive field as well. The layers are arranged in such a way so that they detect simpler patterns first (lines, curves, etc.) and more complex patterns (faces, objects, etc.) further along. By using a CNN, one can enable sight to computers.

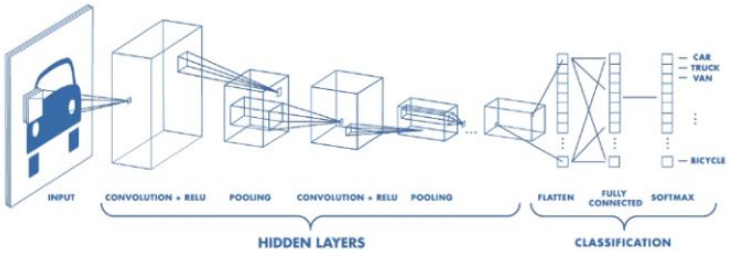

Figure 5.5: Architecture of a CNN

A CNN typically has three layers: a convolutional layer, a pooling layer, and a fully connected layer.

- Convolutional layer: The convolution layer is the core building block of the

CNN. It carries the main portion of the network's computational load.This layer performs a dot product between two matrices, where one matrix is the set of learnable parameters otherwise known as a kernel, and the other matrix is the restricted portion of the receptive field. The kernel is spatially smaller than an image but is more in-depth. This means that, if the image is composed of three (RGB) channels, the kernel height and width will be spatially small, but the depth extends up to all three channels.

During the forward pass, the kernel slides across the height and width of the image-producing the image representation of that receptive region. This produces a two-dimensional representation of the image known as an activation map that gives the response of the kernel at each spatial position of the image. The sliding size of the kernel is called a stride. Convolution leverages three important ideas that motivated computer vision researchers: sparse interaction, parameter sharing, and equivariant representation. Trivial neural network layers use matrix multiplication by a matrix of parameters describing the interaction between the input and output unit. This means that every output unit interacts with every input unit. However, convolution neural networks have sparse interaction. This is achieved by making kernel smaller than the input e.g., an image can have millions or thousands of pixels, but while processing it using kernel we can detect meaningful information that is of tens or hundreds of pixels. This means that we need to store fewer parameters that not only reduces the memory requirement of the model but also improves the statistical efficiency of the model.

In a traditional neural network, each element of the weight matrix is used once and then never revisited, while convolution network has shared parameters i.e., for getting output, weights applied to one input are the same as the weight applied elsewhere.Due to parameter sharing, the layers of convolution neural network will have a property of equivariance to translation. It says that if we changed the input in a way, the output will also get changed

in the same way.

- Pooling layer: The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually.

  There are several pooling functions such as the average of the rectangular neighborhood, L2 norm of the rectangular neighborhood, and a weighted average based on the distance from the central pixel. However, the most popular process is max pooling, which reports the maximum output from the neighborhood.In all cases, pooling provides some translation invariance which means that an object would be recognizable regardless of where it appears on the frame.

- Fully Connected Layers: Since convolution is a linear operation and images are far from linear, non-linearity layers are often placed directly after the convolutional layer to introduce non-linearity to the activation map.The variations of layers depend upon activation function as discussed above.

Chapter 6

Model Design and Training

We have used Tensorflow for model design and training TFLearn: Deep learning library featuring a higher-level API for TensorFlow.TFlearn is a modular and transparent deep learning library built on top of Tensorflow. It was designed to provide a higher-level API to TensorFlow in order to facilitate and speed-up experimentations, while remaining fully transparent and compatible with it.

6.1 Model Design

We will start by importing necessary packages as shown in code snippet.

```
from __future__ import division, print_function, absolute_import

import json
import numpy as np
import tflearn
from tflearn.data_utils import shuffle, to_categorical
from tflearn.layers.core import input_data, dropout, fully_connected
from tflearn.layers.conv import conv_2d, max_pool_2d
from tflearn.layers.estimator import regression
from tflearn.data_preprocessing import ImagePreprocessing
from tflearn.data_augmentation import ImageAugmentation
```

Figure 6.1: Import Necessary packages

Now we will load our datasets and reshape image as 20 X 20 X 3 and convert output to categorical labels.

```
# Load planesnet data
f = open('../input/planesnet.json')
planesnet = json.load(f)
f.close()

# Preprocess image data and labels
X = np.array(planesnet['data']) / 255.
X = X.reshape([-1,3,20,20]).transpose([0,2,3,1])
Y = np.array(planesnet['labels'])
Y = to_categorical(Y, 2)
```

Figure 6.2: DataLoader and Preprocessing

We will add some augmentation to increase dataset size and variation by rotation and flipping. Simultaneously we will calculate mean and zero center every sample with this mean.We will also add some standard normalization.

```
# Real-time data preprocessing
img_prep = ImagePreprocessing()
img_prep.add_featurewise_zero_center()
img_prep.add_featurewise_stdnorm()

# Real-time data augmentation
img_aug = ImageAugmentation()
img_aug.add_random_flip_leftright()
img_aug.add_random_rotation(max_angle=25.)
```

Figure 6.3: Preprocessing and Augmentation

We will add input layer as per our dataset. We will add convolutional layer followed by max pool layer,followed by convolutional layer followed by convolutional layer followed by max pool layer.We will add full convolutional layer at the end. We will add droppout layers to prevent overfitting.We will use Adam optimizer and categorical cross entropy loss function.

$$-(y \log(p) + (1 - y) \log(1 - p))$$

Finally we will use DNN class to get details of model during training itself with verbose option.

```
# Convolutional network building
network = input_data(shape=[None, 20, 20, 3],
                     data_preprocessing=img_prep,
                     data_augmentation=img_aug)
network = conv_2d(network, 32, 3, activation='relu')
network = max_pool_2d(network, 2)
network = conv_2d(network, 64, 3, activation='relu')
network = conv_2d(network, 64, 3, activation='relu')
network = max_pool_2d(network, 2)
network = fully_connected(network, 512, activation='relu')
network = dropout(network, 0.5)
network = fully_connected(network, 2, activation='softmax')
network = regression(network, optimizer='adam',
                     loss='categorical_crossentropy',
                     learning_rate=0.001)
model = tflearn.DNN(network, tensorboard_verbose=0)
```

Figure 6.4: Model Design

6.2 Model Training

We will use fit function for training. The training dataset is divided into train and validation in 80:20 ratio respectively. The model shows an accuracy of 95

```
# Train the model
model.fit(X, Y, n_epoch=10, shuffle=True, validation_set=.2,
          show_metric=True, batch_size=128, run_id='planesnet')
```

```
Training Step: 1999  | total loss: 0.09137 | time: 33.498s
| Adam | epoch: 010 | loss: 0.09137 - acc: 0.9772 -- iter: 25472/25600
Training Step: 2000  | total loss: 0.08747 | time: 36.416s
| Adam | epoch: 010 | loss: 0.08747 - acc: 0.9771 | val_loss: 0.05365 - val_acc: 0.9812 -- iter: 25600/25600
--
```

Figure 6.5: Model Training and accuracy

6.3 Hyperparameter Tuning

There are two types of parameters in model. First is learnable parameter like weight and bias. Second is non learnable parameters which remains constant throughout the training process. Those are called hyperparameters. For our model hyperparameters are learning rate,epochs,optimizer,loss function etc. RandomsearchCV and GridsearchCV are the two popular techniques for hyperparameter tuning.

- GridSearchCV: Grid Search uses a different combination of all the specified hyperparameters and their values and calculates the performance for each combination and selects the best value for the hyperparameters. This makes the processing time-consuming and expensive based on the number of hyperparameters involved.In GridSearchCV, along with Grid Search, cross-validation is also performed. Cross-Validation is used while training the model.The most popular type of Cross-validation is K-fold Cross-Validation. It is an iterative process that divides the train data into k partitions. Each iteration keeps one partition for testing and the remaining k-1 partitions for training the model. The next iteration will set the next partition as test

data and the remaining k-1 as train data and so on. In each iteration, it will record the performance of the model and at the end give the average of all the performance. Thus, it is also a time-consuming process.

- RandomsearchCV: GridSearchCV can be computationally expensive, especially if you are searching over a large hyperparameter space and dealing with multiple hyperparameters. A solution to this is to use RandomizedSearchCV, in which not all hyperparameter values are tried out. Instead, a fixed number of hyperparameter settings is sampled from specified probability distributions. You'll practice using RandomizedSearchCV.

We have used RandomsearchCV which gives optimial parameter as follows:

- learning rate: 0.002
- optimizer: Adam
- loss: categorical crossentropy
- epochs: 13
- droput: 0.6
- maxpool : 2

Chapter 7

Results & Analysis

After hyperparameter tuning the model shows an accuracy of 97We have 2 labels Plane and No plane.Hence, If model predicts a plane label correctly then it is True Positive and if model predicts a no plane label incorrectly for no plane data then it is False Positive. Here by positive we means plane label and by negative we mean no plane label. The confusion matrix and other model validation parameters are given below for 166 random samples.

	True Positive	True Negative
Predicted Positive	32	4
Predicted Negative	2	128

Figure 7.1: Confusion Matrix

Measure	Value	Derivations
Sensitivity	0.9412	TPR = TP / (TP + FN)
Specificity	0.9697	SPC = TN / (FP + TN)
Precision	0.8889	PPV = TP / (TP + FP)
Negative Predictive Value	0.9846	NPV = TN / (TN + FN)
False Positive Rate	0.0303	FPR = FP / (FP + TN)
False Discovery Rate	0.1111	FDR = FP / (FP + TP)
False Negative Rate	0.0588	FNR = FN / (FN + TP)
Accuracy	0.9639	ACC = (TP + TN) / (P + N)
F1 Score	0.9143	F1 = 2TP / (2TP + FP + FN)
Matthews Correlation Coefficient	0.8920	TP*TN - FP*FN / sqrt((TP+FP)*(TP+FN)*(TN+FP)*(TN+FN))

Figure 7.2: Various model validation parameters

Chapter 8

Conclusion & Future Work

This activity provides a model which can classify plane and no plane class with 97% accuracy.Machine learning and deep learning terminologies are discussed with respect to the problem statement. Challenges of overfitting of point objects are addressed and possible trade-offs are discussed.

Future scope of work could be to make it robust for on chip computation and through stream of video for real time image processing. If these two tasks can be achieved this model can be deployed for search,rescue and monitoring of planes with satellite imagery. Auxiliary information can be provided to model with some attention features to avoid noise and improve learning rate. Non optical data can be also explored for post processing stage.This activity can be replicated with other point features like boat,aquaculture, power grid poles with the same framework. Deep neural network can be more dense to identify more fine features.

Bibliography

- https://www.researchgate.net/figure/Satellite-image-from-NRSC-ISROs-Bhuvan-showing-key-structural-features-along-inlet-and_fig2_352766014
- https://bhuvan-app3.nrsc.gov.in/data/download/index.php
- https://www.researchgate.net/publication/221621494_Support_Vector_Machines_Theory_
- https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00444-8
- https://arxiv.org/abs/2003.05689
- http://tflearn.org/
- https://www.researchgate.net/publication/355096788_Confusion_Matrix
- https://developers.planet.com/docs/api/