Dr. Babasaheb Ambedkar Technological University
Lonere, Dist. Raigad, Pin 402103, Maharashtra

A
PROJECT REPORT ON

"TERRAIN RECOGNITION USING DEEP LEARNING" PART 2

Under the Guidance of

Prof. D. P. Gadhe

Submitted by :

Anil Riswal [AI4051]
Sanyukta Rajput [AI4073]
Hrushikesh Ambhore [AI4015]

For the partial fulfillment for the award of

BACHELOR OF TECHNOLOGY
IN
ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

CSMSS
CHH. SHAHU COLLEGE OF ENGINEERING
Chhatrapati Sambhajinagar – 431011
(2023-24)

CSMSS

Chh. Shahu College of Engineering,

Chhatrapati Sambhajinagar, Maharashtra - 431011

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

CERTIFICATE

This is to certify that the Project report entitled

“TERRAIN RECOGNITION USING DEEP LEARNING”

Submitted by:

Anil Riswal [AI4051

Sanyukta Rajput [AI4073]

Hrushikesh Ambhore [AI4015]

in partial fulfillment for award of the Degree **Bachelor of Technology** in **Artificial Intelligence And Data Science** of **Dr. Babasaheb Ambedkar Technological University**, Lonere, Raigad, during academic year 2023-24 Part-I

Prof. D. P. Gadhe	**Dr. S. R. Zanwar**	**Dr. U. B. Shinde**
Guide	**Head of the Department**	**Principal**

CSMSS

Chh. Shahu College of Engineering,

Chhatrapati Sambhajinagar, Maharashtra - 431011

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

PROJECT APPROVAL SHEET

Project entitled "**TERRAIN RECOGNITION USING DEEP LEARNING**" submitted by **Anil Riswal [AI4051], Sanyukta Rajput [AI4073], Hrushikesh Ambhore [AI4015]** is approved for partial fulfillment for the award of **Bachelor of Technology in Artificial Intelligence & Data Science** of **Dr. Babasaheb Ambedkar Technological University**, Lonere, Raigad (M.S.) during academic year 2023-24 Part-I.

Name and Sign	**Name and Sign**
Internal Examiners	**External Examiners**

Place: Chhatrapati Sambhajinagar

Date:

DECLARATION

We, the students enrolled in the eighth semester of the B.Tech program in **Artificial Intelligence & Data Science** at CSMSS Chh. Shahu College of Engineering, Chhatrapati Sambhajinagar, hereby assert that our project work titled "**TERRAIN RECOGNITION USING DEEP LEARNING**" submitted to Dr. Babasaheb Ambedkar Technological University, Lonere, Raigad, during the academic year 2023-24, represents original research conducted by us.

This project work is presented as a partial fulfillment of the requirements for the Bachelor of Technology degree in **Artificial Intelligence & Data Science**. The findings presented in this report have not been previously submitted to any other University or Institute for the purpose of obtaining any degree.

Roll No.	Name of the student	PRN No.	Signature
AI4051	Anil Riswal	T2125331995506	
AI4073	Sanyukta Rajput	T2125331995528	
AI4015	Hrushikesh Ambhore	T2025331995016	

Place: Chhatrapati Sambhajinagar
Date:

ACKNOWLEDGEMENT

We take immense pleasure in presenting this project report, as this page provides us with the opportunity to convey our heartfelt emotions and gratitude.

We extend our sincere thanks to our guide, **Prof. D. P. Gadhe Mam**, whose guidance was invaluable at every step of this project. Her motivation and confidence-boosting efforts were instrumental, and we acknowledge that this work would not have been possible without her support and encouragement.

Special appreciation is also extended to the project coordinator, **Prof. D. P. Gadhe Mam**, for guiding and motivating us throughout the project. We would like to express our gratitude to the Head of the Department, Dr. S. R. Zanwar, and the respected Principal, Dr. U. B. Shinde, for providing valuable resources and dedicating their valuable time to review our report.

Finally, we express our thanks to all the staff members of the **Artificial Intelligence & Data Science Department** and our friends, without whom the completion of this project would not have been possible.

Anil Riswal [AI4051]
Sanyukta Rajput [AI4073]
Hrushikesh Ambhore [AI4015]

B. Tech. (Artificial Intelligence & Data Science

CONTENTS

List of Abbreviations

SN	Symbol	Illustrations
1	CNN	Convolutional Neural Network
2	SAR	Synthetic Aperture Radar
3	InSAR	Interferometric Synthetic Aperture Radar
4	IDE	Integrated Development Environment
5	CNTK	Cognitive Toolkit

ABSTRACT

This report explores the innovative domain of deep learning applied to terrain recognition, a critical component in various fields such as autonomous vehicles, robotics, and environmental monitoring. As the demand for accurate and real-time terrain analysis continues to rise, deep learning techniques have emerged as powerful tools for extracting intricate patterns and features from diverse terrain types.

The report begins by providing an overview of the foundational concepts in deep learning and its relevance to terrain recognition. It delves into the methodologies employed, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and their hybrid architectures. A comprehensive survey of recent advancements in deep learning models tailored for terrain recognition is presented, highlighting their effectiveness in handling complex spatial and temporal dependencies inherent in diverse terrains.

Furthermore, the report investigates the practical applications of deep learning in terrain recognition, showcasing its role in enhancing the navigation capabilities of autonomous vehicles, aiding in disaster response, and facilitating environmental monitoring. Case studies and success stories from various industries demonstrate the tangible benefits and transformative impact of deploying deep learning techniques in terrain analysis.

While acknowledging the strides made in this field, the report also addresses existing challenges and potential avenues for future research. Issues such as limited labeled datasets, model interpretability, and adaptability to dynamic terrains are discussed, offering insights into areas where further research and development are needed to optimize the performance and robustness of deep learning models for terrain recognition.

CHAPTER I
INTRODUCTION

1.1 Introduction

Terrain recognition refers to the process of identifying and analysing the physical characteristics and features of the land surface. This recognition can be done manually by individuals or, more commonly in modern applications, through the use of automated systems and technology.[1] Terrain refers to the physical features and characteristics of a piece of land or a geographical area. It encompasses the natural and artificial elements that make up the Earth's surface and affects how the land can be used or traversed. Terrain includes various components such as landforms (mountains, valleys, plains), surface texture (rough, smooth), elevation, vegetation and water bodies.

Terrain recognition involves the use of technology, sensors, and algorithms to identify and understand the characteristics of different types of terrain. The primary goal is to enable systems, such as autonomous vehicles, robots, or drones, to adapt their behaviour based on the nature of the terrain they are traversing. The work of terrain recognition can be Terrain classification which involves identifying and categorizing different types of terrains, such as forests, mountains, deserts, urban areas, and water bodies. This helps in creating a map of the environment, Surface Characteristics Analysis which Assessing features like roughness, slipperiness, slope, and other properties of the ground. This information is crucial for optimizing navigation and making decisions about how a system should interact with the terrain etc.

Terrain recognition using deep learning involves leveraging advanced techniques, particularly deep neural networks, to automatically identify and understand different types of terrain. It provides a powerful and adaptive solution for systems operating in diverse landscapes, enhancing their ability to navigate and make informed decisions based on the characteristics of the terrain. Deep learning architectures are complex neural network structures designed to learn hierarchical representations of data. These architectures have been instrumental in achieving breakthroughs in various artificial intelligence tasks, including image and speech recognition, natural language processing, and more. convolutional Neural Networks (CNNs) are a class of deep neural networks specifically designed for processing structured grid data, such as images.

They have played a pivotal role in revolutionizing computer vision tasks, including image classification, object detection, and segmentation.
Xception is a deep neural network architecture that can be considered an extension of the Inception architecture. Xception was introduced to improve the efficiency and performance of deep learning models. The key innovation in Xception lies in its use of depth wise separable convolutions, which helps reduce the number of parameters and computational complexity.

This project includes the deep learning model, specifically Xception Net, for terrain recognition. The technologies employed include TensorFlow and Keras for model development, along with essential data manipulation and visualization libraries such as Pandas, NumPy, and Matplotlib.
Recognizing terrain poses significant challenges due to the diversity of landscapes, changing environmental conditions, and the need to interpret complex visual and sensor data. Traditional methods often struggle to capture the intricate features that define different terrains, leading to limitations in adaptability and robustness.

Terrain recognition using Xception Net proves to be an effective solution for accurate classification and prediction tasks. The integration of deep learning technologies, specifically Xception Net, enhances the system's ability to adapt to diverse terrains and environmental conditions.[6]

1.2 OBJECTIVE

The objective of terrain recognition using deep learning is to develop intelligent systems that can automatically identify and understand different types of terrains based on visual data. This involves training deep neural networks to recognize patterns, features, and characteristics in images or sensor data associated with diverse landscapes.

1. **Terrain Classification**: Train the deep learning model to categorize terrains into predefined classes or types, such as sandy, rocky, marshy, grassy, etc. This classification enables the system to understand and differentiate between different environmental conditions.

2. **Attribute Prediction**: Predict specific attributes of the terrain, such as roughness, slipperiness, or other relevant features. This adds a level of detail to the recognition, allowing for a more comprehensive understanding of the terrain characteristics.

3. **Real-time Terrain Analysis**: Enable the system to perform terrain recognition in real-time, allowing for immediate decision-making in applications like autonomous navigation, robotics, or other scenarios where quick responses to changing terrains are crucial.

4. **Scalability and Generalization**: Build a system that can scale to handle large datasets and generalize well to different geographic locations. This ensures that the model's performance remains effective across diverse terrains and regions.

5. **User-Friendly Interfaces**: Design user-friendly interfaces or APIs to make the terrain recognition technology accessible to end-users, whether they are developers, researchers, or professionals in specific industries. This could include interactive maps or visualizations.

1.3 Necessity

The need of creating terrain recognition using deep learning is necessary for several compelling reasons across various fields and applications.

Enhanced Safety in Autonomous Systems: Autonomous vehicles, drones, and robots require accurate terrain recognition to navigate safely. Deep learning models can analyze and interpret complex terrain features in real-time, contributing to safer operations.

Optimized Agricultural Practices: In precision agriculture, understanding the characteristics of the terrain is crucial for optimizing irrigation, planting, and harvesting. Deep learning can aid in classifying and analysing terrains, leading to improved resource efficiency and higher yields.

Environmental Monitoring and Conservation: Terrain recognition supports environmental monitoring by providing insights into changes in landscapes, vegetation, and land use. This information is vital for environmental conservation efforts and sustainable resource management.

Search and Rescue Operations: During search and rescue missions, accurate terrain recognition is essential for planning efficient routes and locating individuals in distress. Deep learning can assist in quickly assessing and navigating challenging terrains.

Military and Defence Applications: Understanding the terrain is critical for military operations. Deep learning models can analyse diverse terrains, providing valuable information for strategic planning, reconnaissance, and decision-making in defence scenarios.[5]

1.4 Challenges

1. **Diverse Terrain Conditions**: Earth's surface exhibits diverse and complex terrains, making it challenging to create a one-size-fits-all model that can accurately recognize and classify all types of terrain.
2. **Data Variability**: The availability of diverse and representative datasets is crucial for training accurate models. Obtaining comprehensive datasets that cover various terrains and environmental conditions can be challenging.
3. **Adaptability to Changing Environments**: Terrain conditions change over time due to factors such as weather, seasons, and human activities. Creating models that can adapt and generalize well to dynamic environmental changes is a significant challenge.
4. **Limited Labelling of Training Data**: Labelling terrain data for training deep learning models can be intensive and may have limitations in terms of accuracy and comprehensiveness. Limited labelled data can affect the model's performance.[2]

CHAPTER II
LITERATURE SURVEY

2.1 Literature Survey

A comprehensive literature survey on terrain recognition using deep learning involves reviewing a wide range of research papers, articles, and conference proceedings. Terrain recognition, an essential component which involves the use of advanced technologies to identify and understand different types of landscapes. This technology is essential for systems such as autonomous vehicles, drones, and robots, enabling them to adapt their behavior based on the characteristics of the terrain they encounter. The primary goal of this project is to recognize and categorize different types of terrain, allowing intelligent systems to adapt to varying environmental conditions. The system aims to identify terrain classes and predict specific characteristics, such as roughness and slipperiness.[5]

2.1 Historical context

Early Computer Vision and Image Processing (1960s-1980s):

In the early years, computer vision focused on basic image processing tasks. Traditional techniques like edge detection and texture analysis were employed for recognizing patterns in images, but these methods were limited in their ability to handle complex and diverse terrains.

Machine Learning in Remote Sensing (1990s-2000s):

Machine learning algorithms, particularly those based on supervised classification, started being applied to remote sensing data. These methods aimed to automatically categorize land cover types using features derived from satellite imagery. While these approaches showed promise, they often struggled with the complexity and variability of terrains.

Rise of Deep Learning (2010s-Present):

The breakthroughs in deep learning, especially with Convolutional Neural Networks (CNNs), marked a significant turning point. CNNs demonstrated remarkable performance in image classification tasks, inspiring researchers to apply these techniques to terrain recognition. Transfer learning became a valuable approach, allowing models pre-trained on large datasets (such as ImageNet) to be fine-tuned for specific terrain recognition tasks. This helped overcome the challenge of limited annotated datasets for diverse terrains.

Semantic Segmentation and Object Detection (2010s-Present):

Deep learning techniques evolved to address more granular tasks, such as semantic segmentation and object detection. These advancements enabled the identification and delineation of specific terrain features within an image, providing a more detailed understanding of the landscape.

Applications in Autonomous Systems (2010s-Present):

Terrain recognition using deep learning has found practical applications in autonomous systems, including self-driving cars, drones, and robotic exploration. The ability to perceive and adapt to different terrains is crucial for the safe and efficient operation of these systems.[7]

2.2 Summary of Existing Methods

Title of paper	**Preprocessing Methods**	**Methods of Feature Engineering**	**Algorithms**	**Evaluation**	**References**
Terrain Classification Using Machine Learning	noise reduction, normalizing input features	Texture analysis, elevation data transformation	Support Vector Machines (SVM) and Random Forests	accuracy, precision, and recall	International Journal of Remote Sensing[10]
Terrain Recognition Using Convolutional Neural Networks"	normalization and denoising	Convolutional Neural Networks (CNNs)	CNN architecture, with transfer learning from a pre-trained model	accuracy, precision, and F1-score	IEEE Transactions on Geoscience and Remote Sensing[8]

LiDAR-Based Terrain Analysis for Autonomous Vehicles	cloud segmentation and filtering	LiDAR intensity and elevation data	clustering algorithms and decision trees.	accuracy	LiDAR-Based Terrain Analysis for Autonomous Vehicles." Journal of Robotics and Autonomous Systems[3]
A Comprehensive Survey on Terrain Recognition	normalization and denoising	Decision Trees and Gradient Boosting	Support Vector Machines (SVM) and Random Forests	accuracy, precision, and recall	journal of Unmanned Vehicle Systems[7]

2.2.1 Existing Methods:

In contemporary terrain modeling, advanced technologies such as satellite imagery, radar, and LiDAR play a pivotal role. Recent methodologies leverage deep learning techniques to extract detailed features from the captured data. Satellite imagery benefits from resolution enhancement and multispectral analysis, enabling improved terrain characterization and change detection using convolutional neural networks (CNNs). Radar technology, specifically Synthetic Aperture Radar (SAR) and Interferometric SAR (InSAR), utilizes deep learning for terrain classification and precise elevation mapping. LiDAR data is processed with point cloud-based deep learning models like PointNet, enabling the extraction of intricate 3D terrain features. Multi-sensor fusion, combining data from various sources, further enhances terrain understanding, and sophisticated deep learning architectures facilitate feature fusion for more comprehensive and accurate terrain models.These methodologies find applications in environmental monitoring, urban planning, and disaster management, providing nuanced insights into diverse terrains.[6]

2.2.2 Proposed Methods:

In our proposed methodology for terrain recognition using deep learning, we develop a comprehensive model aimed at detecting, classifying, and predicting terrain conditions with a focus on roughness and slipperiness. Leveraging the Xception architecture for its efficiency and effectiveness, we employ Convolutional Neural Networks (CNNs) to process and analyze the features extracted from the dataset. The dataset, carefully curated to include diverse terrain types such as Grassy, Rocky, Sandy, and Marshy, facilitates robust training and validation of the model. The CNN-based model is trained to recognize distinct features associated with each terrain category, enabling accurate classification. Additionally, our model goes beyond simple classification by predicting the roughness or slipperiness of the terrain, providing valuable insights for applications like autonomous navigation or outdoor activity planning. This methodology holds promise for enhancing terrain understanding in various contexts, contributing to improved decision-making in scenarios where terrain characteristics significantly impact performance and safety.[3]

CHAPTER III
SYSTEM MODELLING

3.1 Introduction (Systematic Modeling & Architecture)

In the contemporary landscape of image processing, the amalgamation of deep learning techniques and Python programming has ushered in a new era of efficiency and accuracy. This project delves into the fascinating realm of land classification through image analysis, utilizing the power of deep learning. At the core of our endeavor lies the careful construction and deployment of a sophisticated model architecture, a pivotal element that dictates the success of image classification tasks. [1]

Deep learning, with its ability to automatically learn hierarchical representations from data, has proven instrumental in unraveling complex patterns within images. The task at hand involves the classification of land types, a challenge that demands a nuanced understanding of spatial features within diverse imagery. To address this, our project centers around the development and refinement of a Convolutional Neural Network (CNN) - a class of deep learning models well-suited for image-related tasks.

The architecture of our CNN plays a pivotal role in transforming raw input images into meaningful and actionable information. Throughout this report, we will meticulously explore the intricacies of our chosen model, elucidating the layers, parameters, and design choices that collectively contribute to its efficacy. Python, as a versatile and widely adopted programming language, forms the backbone of our implementation, facilitating a seamless integration of deep learning libraries such as TensorFlow and Keras.[1]

The systematic modeling approach involves a series of intentional decisions – from the selection of network architecture to the fine-tuning of hyperparameters – all geared towards optimizing the model's performance in classifying land types. By unravelling the layers of our CNN, we aim to provide a comprehensive understanding of how the model interprets and processes image data, showcasing the interplay between convolutional layers, pooling operations, and densely connected layers.

3.2 Block/Architecture Diagram

The Schematic Flow of the working process of the model goes as below:

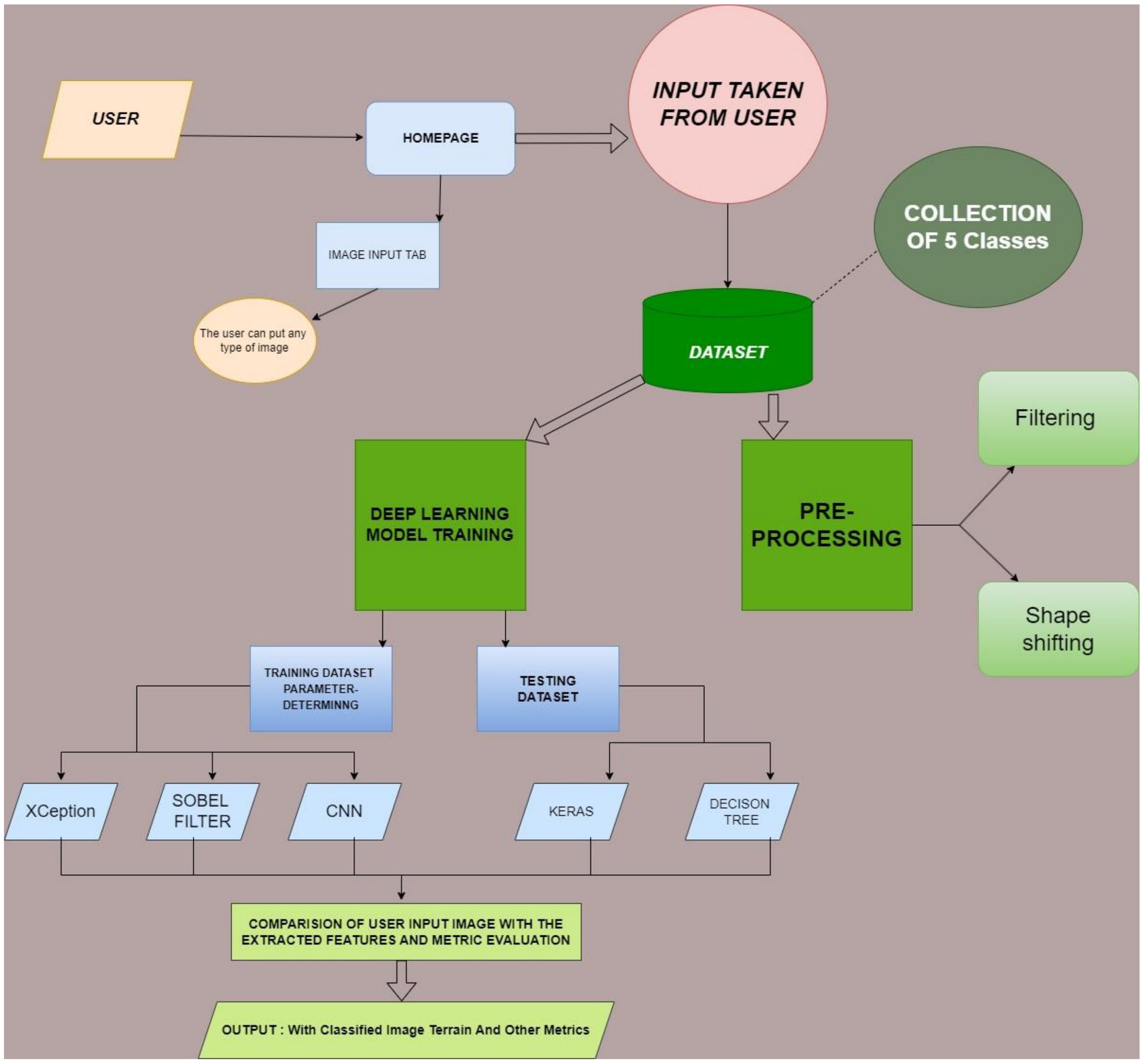

USER
HOMEPAGE
INPUT TAKEN FROM USER
COLLECTION OF 5 Classes
IMAGE INPUT TAB
The user can put any type of image
DATASET
Filtering
DEEP LEARNING MODEL TRAINING
PRE-PROCESSING
Shape shifting
TRAINING DATASET PARAMETER-DETERMINNG
TESTING DATASET
XCeption
SOBEL FILTER
CNN
KERAS
DECISON TREE
COMPARISION OF USER INPUT IMAGE WITH THE EXTRACTED FEATURES AND METRIC EVALUATION
OUTPUT : With Classified Image Terrain And Other Metrics

Figure 3.1: Flow Diagram Of The Working Process

- In the very first step of the model(workflow) we do the **collection of Dataset,** the dataset consists of all the terrain types that are there the Terrain types include-Rocky terrain, Marshy terrain, Sandy terrain & Grassy terrain. These are all the types of preset terrain. We collect these images in these 4 classes for our model to study the features of different terrains. Apart from these 4 classes we add one more class in to the dataset that is called the "Other than terrain", this data consists of all images that are not terrain, so that the model can differentiate the images that are not terrain and other than terrain. We collect the dataset in such manner.

- **Data preprocessing** is an important step in the data mining process. It refers to the cleaning, transforming, and integrating of data in order to make it ready for analysis. so in the next step we do the task of data preprocessing for the dataset to be made ready for further processing and modelling.
- Digital images are broadly made up of pixels, which are tiny boxes representing the colour and brightness values at that point in the image. Image processing involves handling
- These pixels in a desired manner to achieve what is required for the image. Most of the common operations performed on a digital image include filtering, enhancement, restoration, etc.
- **Shape Shifting** : We **Change the Input Shape** of the images before putting them through actual data processing in the model we change the input shape of the images to 223*223*3 Channels of RGB format, we change the image input and shape format.
- **Filtering** is a process of eliminating unwanted noise from an image. It is done by applying a filter that adjusts the image's pixel values. Based on the type of filter, they can be used for a wide range of applications. They can be designed to remove specific. [4]
- **Formation Of Classes** in this step the formation of the 5 classes that go by the type of: Rocky Terrain, Marshy Terrain, Sandy Terrain, Grassy Terrain and The Other than terrain dataset they are divided into these particular classes.[2]
- **Splitting The data** the splitting of the dataset into the training set, the training set is where the model get the dataset for training and it gets to study the features from the dataset, and the other is the test set where the model trains the on the unseen dataset.
- **Feature Extraction** uses an object-based approach to classify imagery, where an object (also called segment) is a group of pixels with similar spectral, spatial, and/or texture attributes. Traditional classification methods are pixel-based, meaning that spectral information in each pixel is used to classify imagery. With high-resolution panchromatic or multispectral imagery, an object-based method offers more flexibility in the types of features to extract.

1. The workflow involves the following steps:

2. Dividing an image into segments
3. Computing various attributes for the segments
4. Creating several new classes
5. Interactively assigning segments (called training samples) to each class.[8]

- The **Xception algorithm** is used on the model:

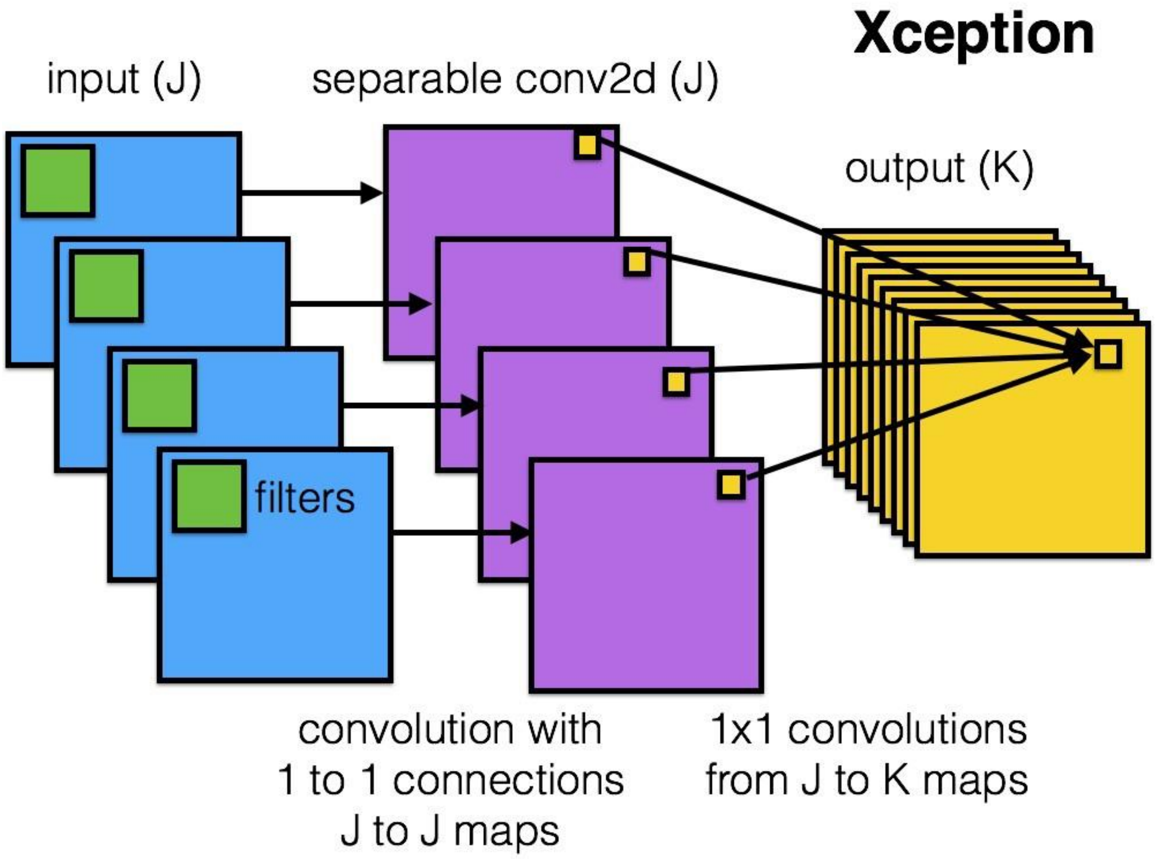


Figure 3.2 XCeption Algorithm [6]

- The **Sobel method**, or Sobel filter, is a gradient-based method that looks for strong changes in the first derivative of an image. The Sobel edge detector uses a pair of 3 × 3 convolution masks, one estimating the gradient in the x-direction and the other in the y-direction.[7]
- The testing of the model is done in this step where the model testing is done with the unseen data from the split dataset and formation that was done in the step, so in this the testing of images from unseen dataset.

- So the model classifies the terrain that is what type the terrain it is like the Rocky, Marshy, Grass, Sandy and the other terrain.
- By the **application of Sobel filter** the Parameters of Roughness And The Slipperiness is determined in metrics of low, medium and high thus the sobel filter.
- Then the output is retrieved, the output is the classification of the image and the determination of the roughness and slipperiness is detected along.[7]
- The application of **Keras Libraries** are implemented while the testing phase where these libraries run on the top of **TensorFlow Technology,** the implementation of these take place in the testing phase In Keras, every ANN is represented by Keras Models. In turn, every Keras Model is composition of Keras Layers and represents ANN layers like input, hidden layer, output layers, convolution layer, pooling layer, etc., Keras model and layer access Keras modules for activation function, loss function, regularization function, etc., Using Keras model, Keras Layer, and Keras modules, any ANN algorithm (CNN, RNN, etc.,) can be represented in a simple and efficient manner.
- **Sequential Model In Keras** − Sequential model is basically a linear composition of Keras Layers. Sequential model is easy, minimal as well as has the ability to represent nearly all available neural networks.

3.3 Model Development

3.3.1: XCEPTION ARCHITECTURE:

Xception is a deep convolutional neural network architecture that involves Depth wise Separable Convolutions. It was developed by Google researchers. Google presented an interpretation of Inception modules in convolutional neural networks as being an intermediate step in-between regular convolution and the depth wise separable convolution operation (a depth wise convolution followed by a pointwise convolution). In this light, a depth wise separable convolution can be understood as an Inception module with a maximally large number of towers. This observation leads them to propose a novel deep convolutional neural network architecture inspired by Inception, where Inception modules have been replaced with depth wise separable convolutions.[6]

I. Why XCeption network?

The data first goes through the entry flow, then through the middle flow which is repeated eight times, and finally through the exit flow.

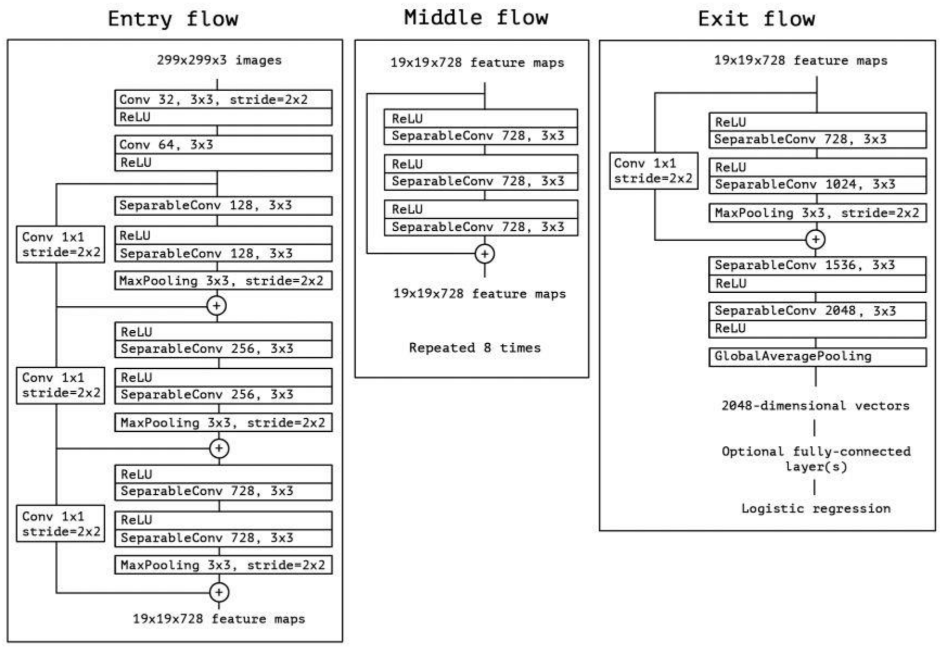


Figure 3.3 : Feature Map Flow Diagram

Xception architecture has overperformed VGG-16, ResNet and Inception V3 in most classical classification challenges.

II.How does XCeption work?

XCeption is an efficient architecture that relies on two main points:

1. wise Separable Convolution

Depth wise Separable Convolutions are alternatives to classical convolutions that are supposed to be much more efficient in terms of computation time.[6]

The limits of convolutions

First of all, let’s take a look at convolutions. Convolution is a really expensive operation. Let’s illustrate this:

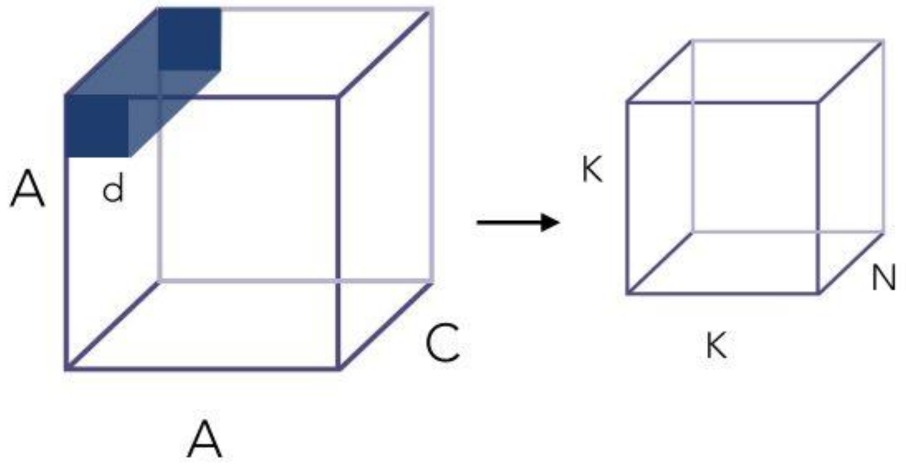


Figure 3.4 Convolution 1

The input image has a certain number of channels C, say 3 for a color image. It also has a certain dimension A, say 100 * 100. We apply on it a convolution filter of size d*d, say 3*3. Here is the convolution process illustrated:

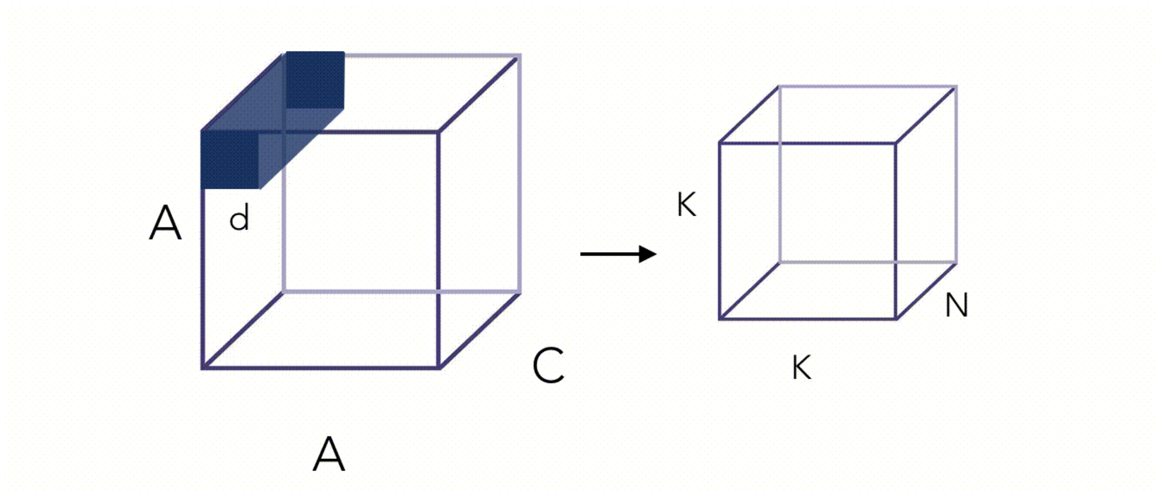


Figure 3.5 Convolution 2

Now, how many operations does that make?

Well, for 1 Kernel, that is:

$$K_2 \times d_2 \times C$$

Where K is the resulting dimension after convolution, which depends on the padding applied (e.g padding “same” would mean A = K).

Therefore, for N Kernels (depth of the convolution):

$$K_2 \times d_2 \times C \times N$$

To overcome the cost of such operations, depthwise separable convolutions have been introduced. They are themselves divided into 2 main steps:

1.The Depth wise Convolution

Depth wise Convolution is a first step in which instead of applying convolution of size d x d x C, we apply a convolution of size d x d x 1. In other words, we don't make the convolution computation over all the channels, but only 1 by 1.[6]

Here is an illustration of the Depth wise convolution process:

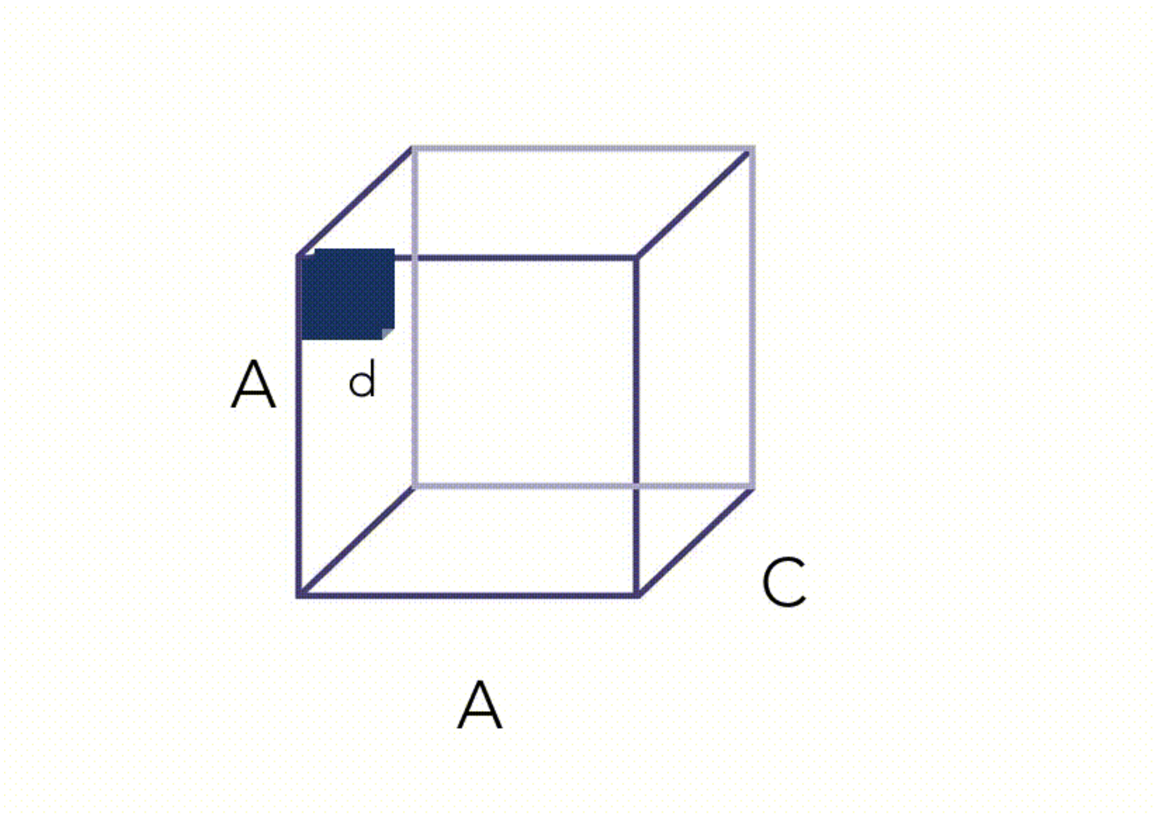


Figure 3.6 Depthwise Convolution

This creates a first volume that has size K x K x C, and not K x K x N as before. Indeed, so far, we only made the convolution operation for 1 kernel /filter of the convolution, not for N of them. This leads us to our second step.

2. Pointwise Convolution

Pointwise convolution operates a classical convolution, with size 1 x 1 x N over the

K x K x C volume. This allows creating a volume of shape K x K x , as previously:

Here is an illustration of the Pointwise Convolution :

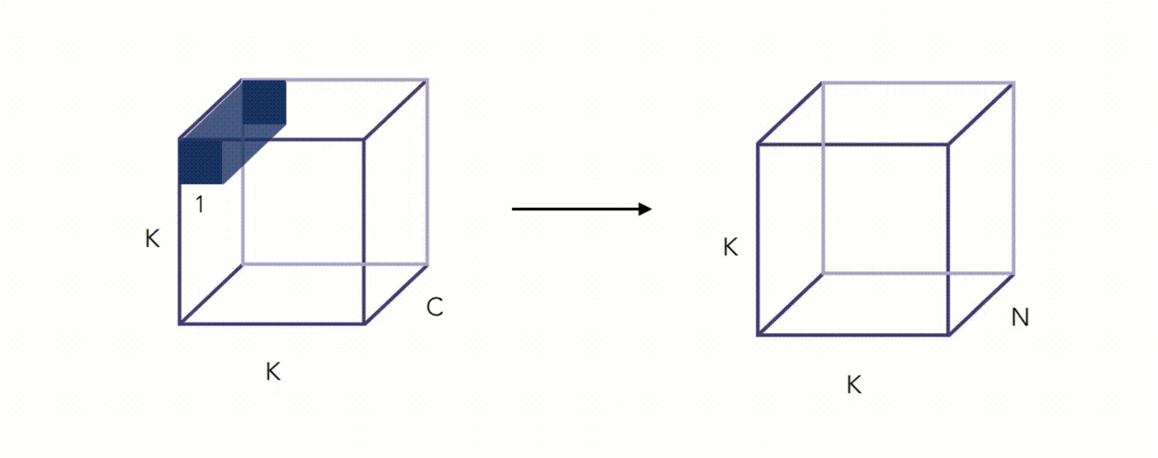


Figure 3.7 Pointwise Convolution

Alright, this whole thing looks fancy, but did we reduce the number of operations? Yes we did, by a factor proportional to 1/N.

Implementation of the XCeption

XCeption offers an architecture that is made of Depth wise Separable Convolution blocks + maxpooling, all linked with shortcuts as in ResNet implementations.The specificity of XCeption is that the Depth wise Convolution is not followed by a Pointwise Convolution, but the order is reversed, as in this example:

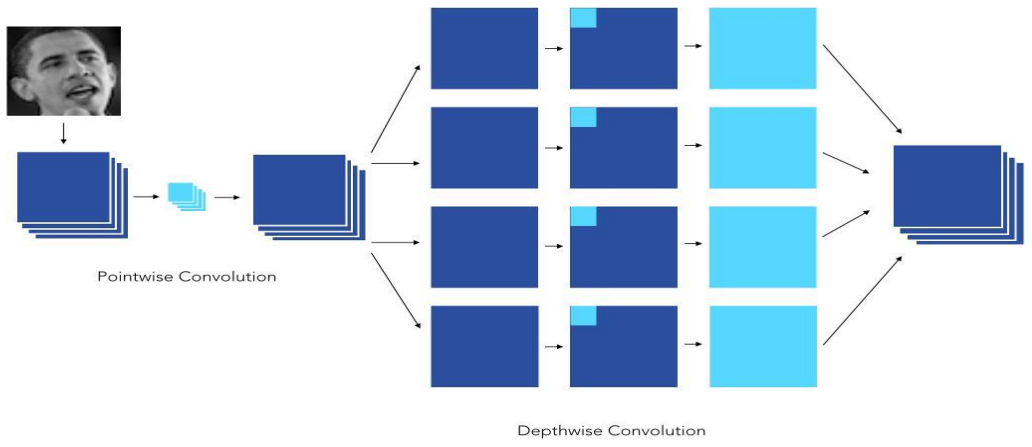


Figure 3.8 : Implementation of XCeption[6]

Conclusion: Xception models remain expensive to train, but are pretty good improvements compared to Inception. Transfer learning brings part of the solution when it comes to adapting such algorithms to your specific task.

3.4 Algorithms

The CNN Algorithm Is used in the traning of the neural layers of the model
Convolutional Neural Networks (CNN) has revolutionized computer vision tasks by enabling automated and accurate recognition of objects within images. CNN-based image classification algorithms have gained immense popularity due to their ability to learn and extract intricate features from raw image data automatically. This article will explore the principles, techniques, and applications of image classification using CNNs. We will delve into the architecture, training process, and CNN image classification evaluation metrics. By understanding the workings of CNNs for image classification, we can unlock many possibilities for object recognition, scene understanding, and visual data analysis.[7]

- **Input Layer:** The input layer of a CNN takes in the raw image data as input. The images are typically represented as matrices of pixel values. The dimensions of the input layer correspond to the size of the input images (e.g., height, width, and color channels).

- **Convolutional layers:** Convolutional layers are responsible for feature extraction. They consist of filters (also known as kernels) that are convolved with the input images to capture relevant patterns and features. These layers learn to detect edges, textures, shapes, and other important visual elements.

- **Pooling Layers:** Pooling layers reduce the spatial dimensions of the feature maps produced by the convolutional layers. They perform downsampling operations (e.g., max pooling) to retain the most salient information while discarding unnecessary details. This helps in achieving translation invariance and reducing computational complexity.

- **Fully Connected Layers:** The output of the last pooling layer is flattened and connected to one or more fully connected layers. These layers function as traditional neural network layers and classify the extracted features. The fully connected layers learn complex relationships between features and output class probabilities or predictions.

- **Output Layer:** The output layer represents the final layer of the CNN. It consists of neurons equal to the number of distinct classes in the classification task. The output layer provides each class's classification probabilities or predictions, indicating the likelihood of the input image belonging to a particular class.[7]

3.5 Technology Stack

In the project runtime environment, we use various toolkits, various different kind of libraries for the implementation of the model. These libraries and toolkits include various modules that develop the model building. The motive of the Technology Stack is where we stack different set of technologies in fusion to build the model in a efficient manner.

The different toolkits and libraries implemented in this model are :

- **Scikit-learn** is an open-source Python library that implements a range of machine learning, pre-processing, cross-validation, and visualization algorithms using a unified interface.
- **Important features of scikit-learn:**

1. Simple and efficient tools for data mining and data analysis. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means, etc.
2. Accessible to everybody and reusable in various contexts.
3. Built on the top of NumPy, SciPy, and matplotlib.
4. Open source, commercially usable – BSD license.

- **Keras** is an open-source high-level Neural Network library, which is written in Python is capable enough to run on Theano, TensorFlow, or CNTK. It was developed by one of the Google engineers, Francois Chollet. It is made user-friendly, extensible, and modular for facilitating faster experimentation with deep neural networks. It not only supports Convolutional Networks and Recurrent Networks individually but also their combination.

- **TensorFlow** is an open-source machine learning library developed by Google. TensorFlow is used to build and train deep learning models as it facilitates the creation of computational graphs and efficient execution on various hardware platforms. The article provides an comprehensive overview of tensorflow.[3]

- TensorFlow is basically a software library for numerical computation using data flow graphs where:

1. **nodes** in the graph represent mathematical operations.
2. **edges** in the graph represent the multidimensional data arrays (called tensors) communicated between them. (Please note that tensor is the central unit of data in TensorFlow).

- **Matplotlib** is a low-level graph plotting library in python that serves as a visualization utility. Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram, etc.
- As we develop the project and model in python we use the python Ide Integrated development environment the IDE is important while modelling development, and all other basic technology stacks are integrated while the development.

CHAPTER IV
RESULT AND DISCUSSION

4.1 Dataset Descriptions

- For, Dataset we gather near about 1800 images of different types of terrain and some random images which do not include terrain.
- Then we distribute these images in 5 classes namely-Grassy, Sandy, Rocky, Marshy and Other than terrain.

1. **Grassy Terrain:** Grassy terrain" typically refers to an area of land covered with grass, which is a type of green, low-growing plant with narrow leaves. Grassy terrain can vary widely in terms of its characteristics, from short and manicured lawns to tall and wild grasslands. It is a common feature in various ecosystems, including meadows, prairies, savannas, and fields. The term is often used in the context of outdoor environments and landscapes, and it can describe areas with a predominance of grass as opposed to other types of vegetation. Grassy terrain is not only visually appealing but also serves ecological functions, providing habitat for various organisms, preventing soil erosion, and contributing to the overall biodiversity of an area.

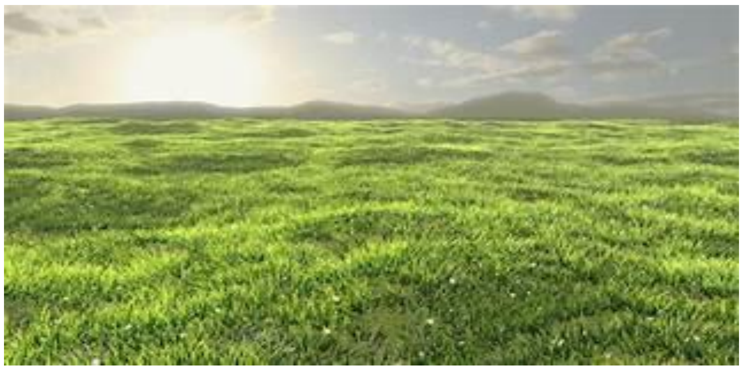


Figure 4.4: Grassy Terrain image

2. Sandy Terrain:

Sandy terrain" refers to an area of land covered with sand. Sandy terrain is a type of landscape characterized by the presence of loose, granular particles of mineral and rock that are larger than silt but smaller than gravel. Sand is typically formed by the weathering and erosion of rocks and is often associated with coastal environments, deserts, dunes, and certain inland areas.

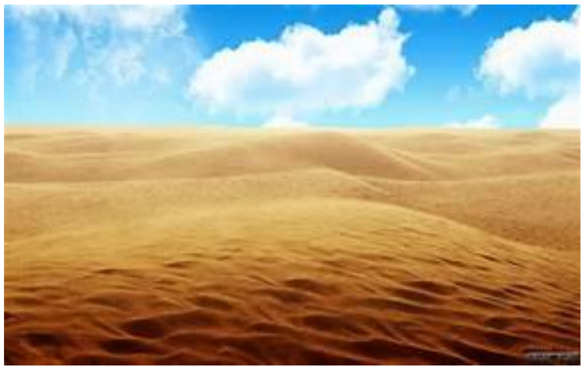


Figure 4.5: Sandy Terrain image

3. Rocky Terrain:

Rocky terrain" refers to an area of land that is covered with rocks or characterized by a significant presence of rocky features. This type of terrain can vary widely, ranging from small rocks scattered on the ground to large rock formations and rugged landscapes. Rocky terrain is found in diverse environments, including mountains, cliffs, canyons, and rocky outcrops in various ecosystems.

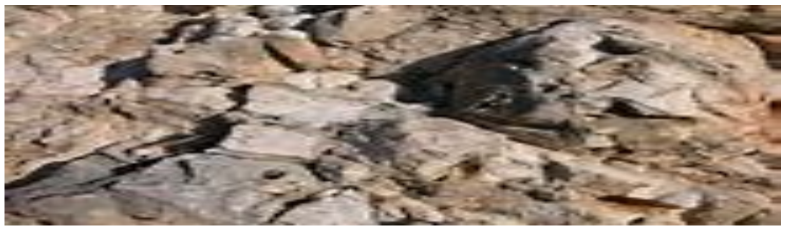


Figure 4.6: Rocky Terrain image

4. Marshy Terrain:

"Marshy terrain" refers to an area of land characterized by the presence of wet, soft, and often waterlogged ground. Marshes are a type of wetland, and their terrain is typically covered with shallow water, emergent vegetation, and a mix of soil and decaying organic matter. Marshy terrain can be found in various regions, including coastal areas, river deltas, and low-lying landscapes.

Figure 4.7: Marshy Terrain image

- We then spilt the dataset into to training and testing for model training and testing purposes.
- We gave nearly 1400 images that is 60 present of total dataset to training.
- And near about 400 images that is 40 present of total data to testing.

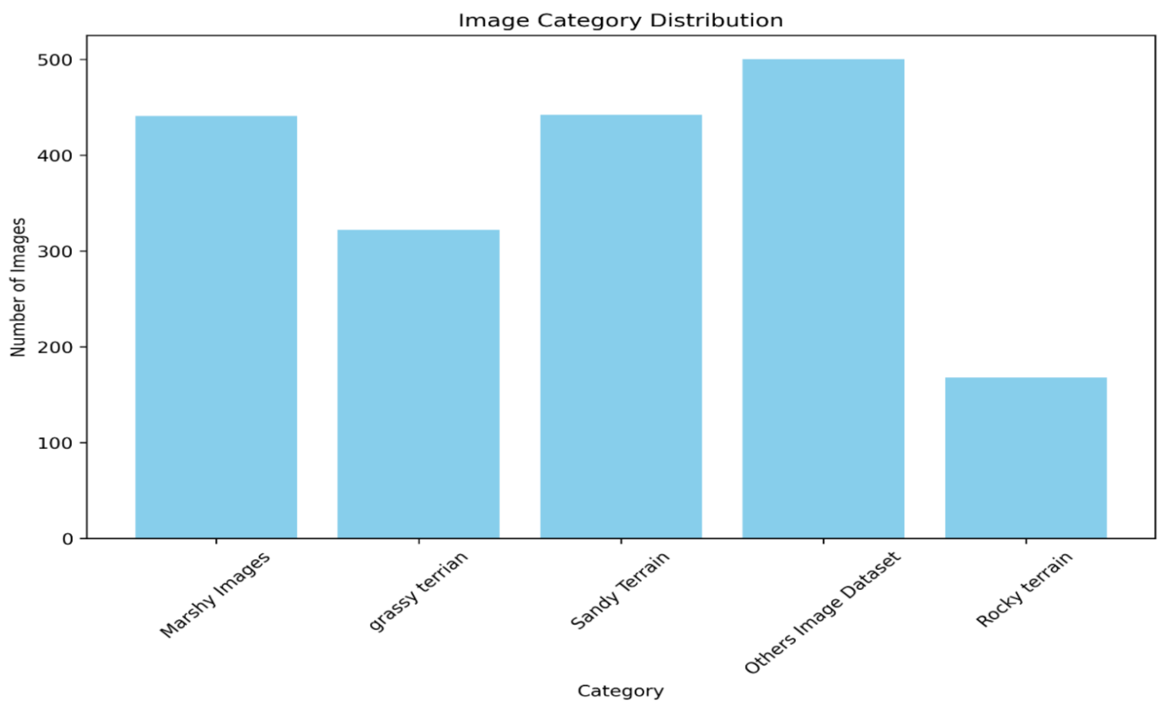


Figure 4.8: Dataset Distribution

4.1 **Performance Metric:**

Evaluating the performance of a Machine learning model is one of the important steps while building an effective ML model. To evaluate the performance or quality of the model, different metrics are used, and these metrics are known as performance metrics or evaluation metrics. These performance metrics help us understand how well our model has performed for the given data. In this way, we can improve the model's performance by tuning the hyper-parameters. Each ML model aims to generalize well on unseen/new data, and performance metrics help determine how well the model generalizes on the new dataset.

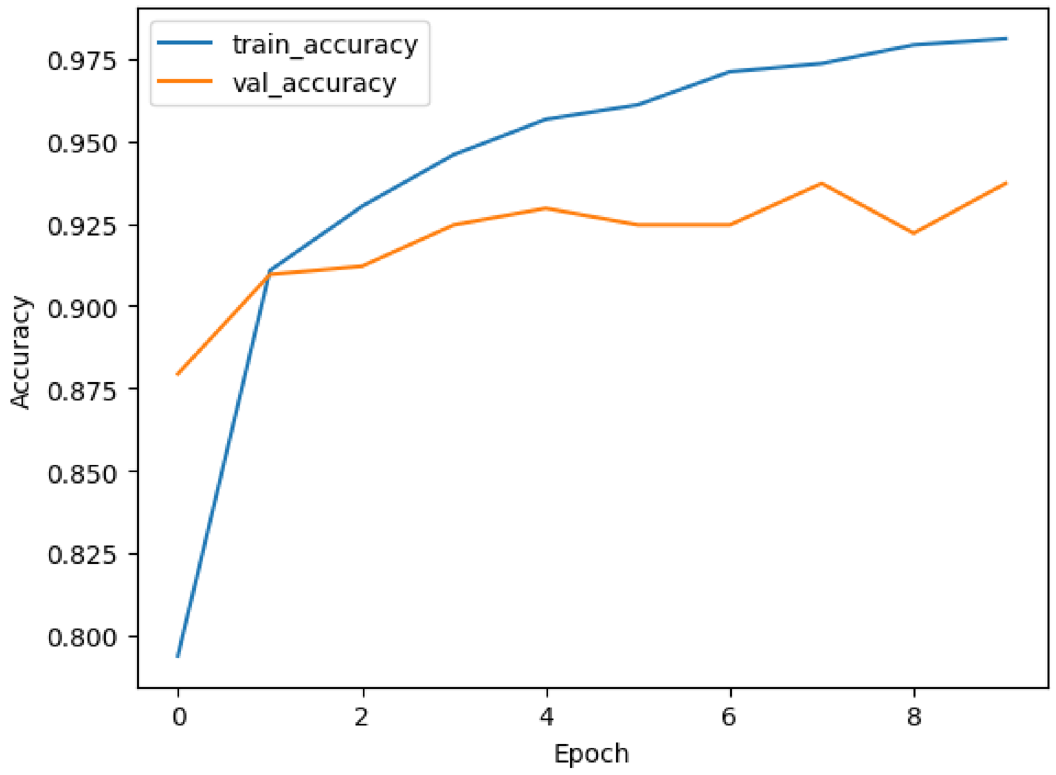


Figure 4.1 AUC-ROC CURVE

1. **Confusion Matrix:** A confusion matrix is a matrix that summarizes the performance of a machine learning model on a set of test data. It is a means of displaying number of accurate and inaccurate instances from the model's predictions. It is often used to measure the performance of classification models, which aim to predict a categorical label for each input instance.

The matrix displays, the number of instances produced by the model on the test data.

- **True positives (TP):** occurs when the model accurately predicts a positive data point.
- **True negatives (TN)**: occurs when the model accurately predicts a negative data point.
- **False positives (FP)**: occurs when the model predicts a positive data point incorrectly.
- **False negatives (FN)**: occurs when the model predicts a negative data point incorrectly.

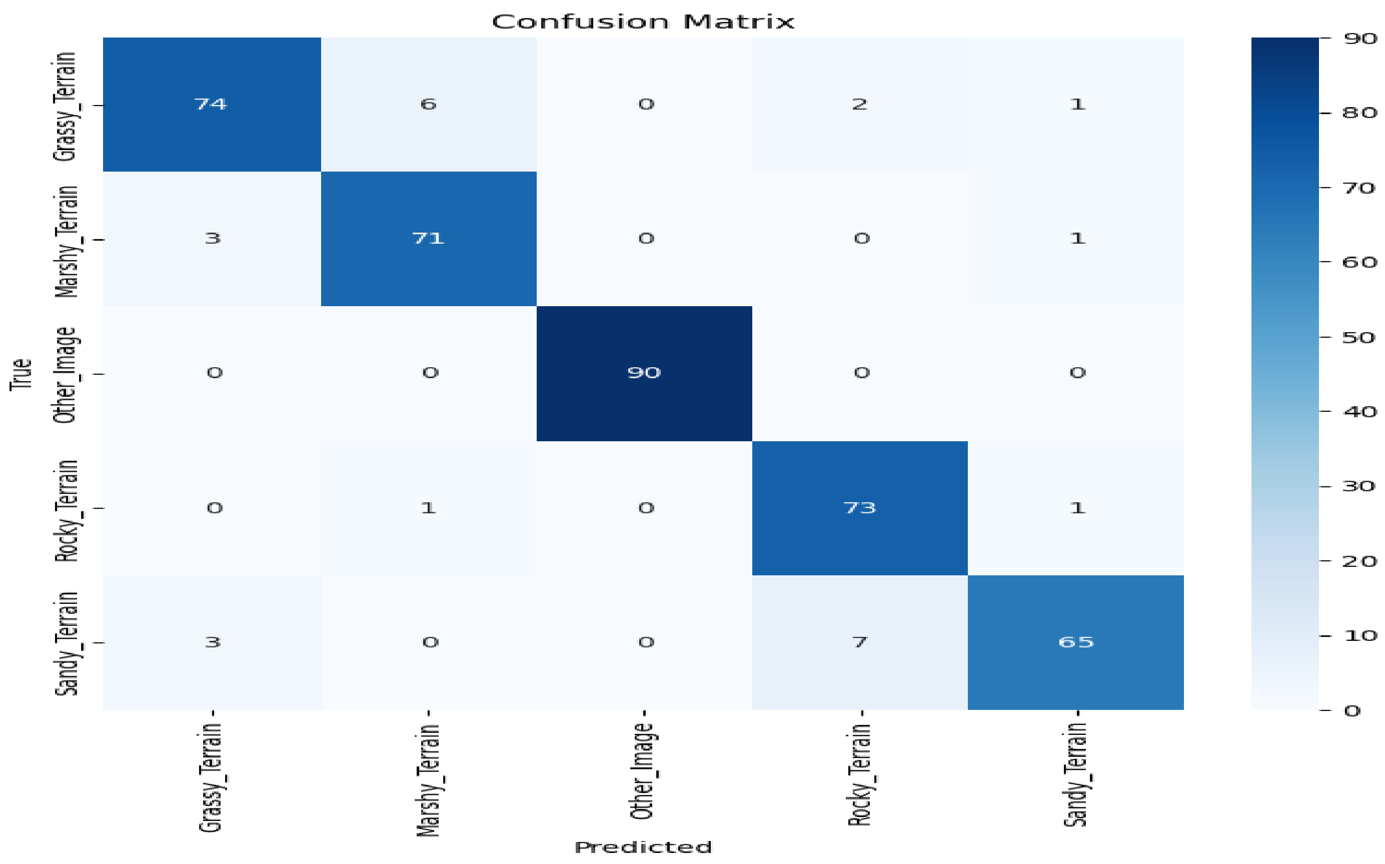


Fig 4.2: Confusion Matrix

2. **Accuracy Score:** An Accuracy score (or simply Accuracy) is a Classification measure in Machine Learning that represents a percentage of correct predictions made by a model. Due to its simplicity in calculation and interpretation, the measure has found widespread use. Additionally, the performance of the model is quantified by a single number.

If you want to use the Accuracy score to evaluate a Classification model, you will need to have: Classes that correspond to reality; Predictions made by the model.

ACCURACY SCORE FORMULA:

Accuracy = Number of correct predictions/Total number of predictions

The more formal formula is the following one.

True Negatives + True Positive

Accuracy = True Positive + False Positive + True Negative + False Negative

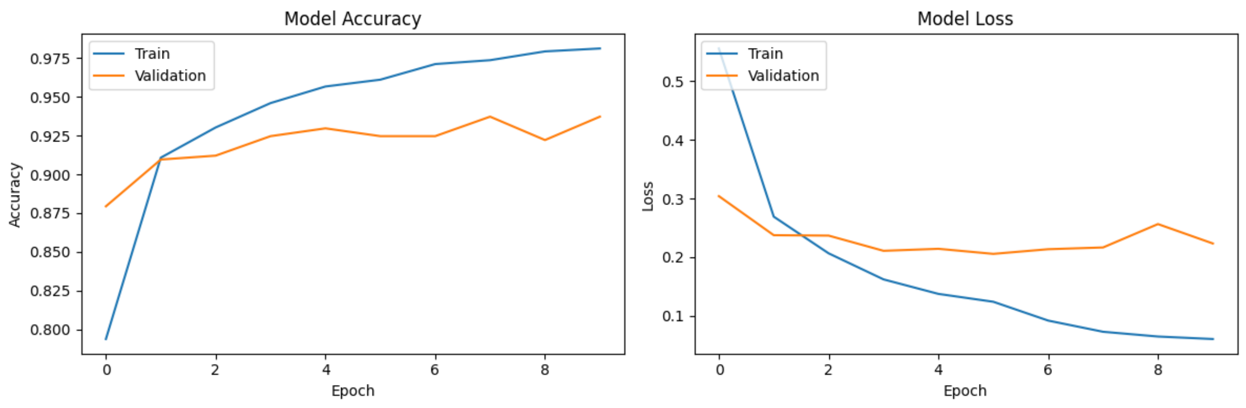


Fig 4.3: Accuracy Score

In this project we got 98% accuracy.

4.2Qualitative Results:

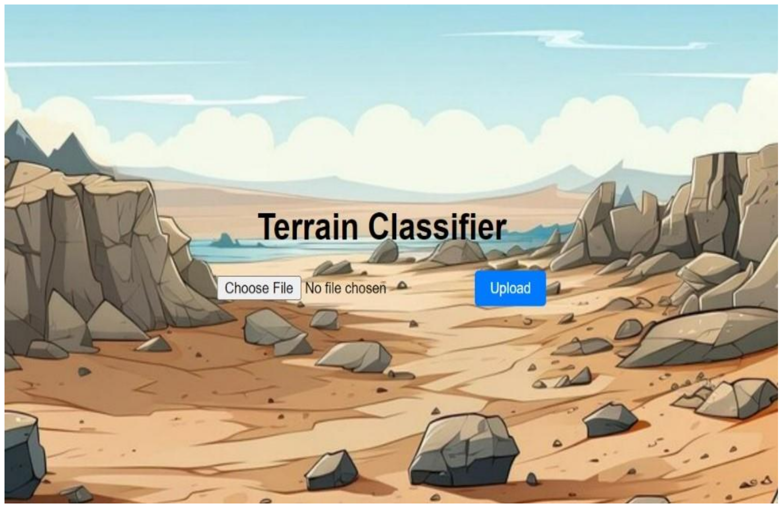
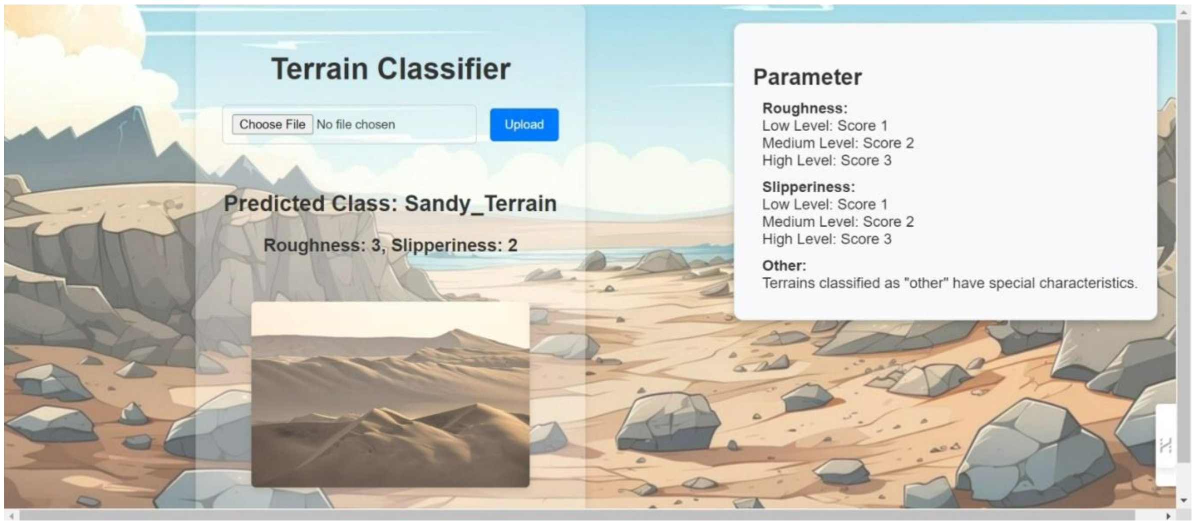


Figure 4.9: Front-end of the model

The initial step for the user involves visiting the application, followed by uploading an image. Subsequently, the website processes the uploaded image at the backend and generates the final result in the form of images. These images depict the assigned category of the uploaded image, specifically identifying whether it falls into categories such as sandy, marshy, rocky, grassy, or others. Additionally, the model provides predictions regarding the sloppiness and roughness of the image, quantifying these aspects on a scale of 0, 1,2 or 3 where, 1,2,3 are low, medium and high score respectively and 0 represents as other class which is not related to terrain. The purpose of this process is to offer users a comprehensive understanding of the terrain characteristics represented in the uploaded image**.**



Predicted Class: Sandy_Terrain

Roughness: 3, Slipperiness: 2

Parameter

Roughness:
Low Level: Score 1
Medium Level: Score 2
High Level: Score 3

Slipperiness:
Low Level: Score 1
Medium Level: Score 2
High Level: Score 3

Other:
Terrains classified as "other" have special characteristics.

Fig 4.10: Final Result

CHAPTER V
CONCLUSION

5.1 Conclusion:

Terrain recognition has led to the development of a powerful deep learning model capable of identifying, classifying, and predicting the characteristics of four primary terrain types—sandy, rocky, marshy, and grassy. This technological advancement opens avenues for enhancing the capabilities of diverse systems, enabling them to navigate and make informed decisions in varied environments. The potential applications of terrain recognition using deep learning extend to autonomous vehicles, environmental monitoring, precision agriculture, and beyond, showcasing its significance in shaping the future of intelligent systems interacting with Earth's diverse landscapes.

5.2 Limitations:

1. **Data Dependency:** Deep learning models require large amounts of labeled data for training it can be both time-consuming and expensive.
2. **Generalization Issues:** Models that perform well on specific terrain types may not necessarily generalize to others, especially if those environments were not included in the training set. This leads to a performance drop when encountering unfamiliar terrain or environmental conditions.
3. **Computational Resources:** Deep learning models, especially those with complex architectures, can be computationally intensive, requiring significant processing power for both training and inference, which can be a hurdle for real-time applications.
4. **Overfitting:** There is a risk that a deep learning model may become too tailored to the training data, capturing noise and anomalies that do not generalize well to real-world scenarios.
5. **Algorithmic Bias:** If the training datasets are not sufficiently diverse, there's a risk that these biases will propagate into the model's performance, leading to biased or uneven terrain recognition capabilities.

5.3 Future Scope:

Terrain recognition using deep learning will improve how self-driving cars and drones navigate, especially in emergencies. We can expect better monitoring of the environment, helping us understand changes in landscapes. Connecting with smart devices and sensors will provide more comprehensive data. The technology will become more adaptable to changes like seasons and weather, benefitting areas like precision farming. Collaboration and using different sensors will make the system even more accurate. Easy-to-use apps for navigation using terrain recognition are in the works. Security and surveillance will also get a boost. In short, terrain recognition is set to make our smart systems even smarter in dealing with Earth's different terrains.

References:

1. http://ieeexplore.ieee.org/document/8406736/ - **Pawel Kozlowski** [Institute of Control, Poznan University of Technology, Poznan, Polan], **Krzysztof Walas**
2. https://drive.google.com/drive/folders/1An6lRkFisNPp3KO1T1vXFxpQI7bOVKMq?usp=sharing - link of the dataset that we collected it contains classes such as sandy, rocky, grassy etc.
3. https://www.sciencedirect.com/science/article/abs/pii/0031320370900191- **J.M.Idelsohn [**Bendix Research Laboratories, Southfield, Michigan 48075, U.S.A].
4. https://www.researchgate.net/publication/320968382_Xception_Deep_Learning_with_Depthwise_Separable_Convolutions - Francois Chollet [DOI:10.1109/CVPR.2017.195, Conference: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)].
5. https://ieeexplore.ieee.org/abstract/document/8406736/ - Jia Xue, Hang Zhang, Kristin Dana
6. https://maelfabien.github.io/deeplearning/xception/ -
7. https://ieeexplore.ieee.org/abstract/document/996 Sobel Operator Image Edge Detection
8. https://iopscience.iop.org/article/10.1088/1742-6596/1601/3/032056 - Xiao-Nan Wang1, Qiang Zhang1, Feng-Yu Wang1 and Yang Gao [2020]