Multiscale Landslide Detection Using Remote Sensing Imagery

A project report submitted in partial fulfillment

of the requirements for the degree of

Bachelor of Technology

in

Electronics & Communication Engineering

by

Akul Nagashayana - 21BEC1221

Siddharth H - 21BEC1338

Rangisetty Jaswanth - 21BEC1333

School of Electronics Engineering,

Vellore Institute of Technology, Chennai,

Vandalur-Kelambakkam Road,

Chennai - 600127, India.

April 2025

Declaration

I hereby declare that the report titled ***Multiscale Landslide Detection Using Remote Sensing Imagery*** submitted by me to the School of Electronics Engineering, Vellore Institute of Technology, Chennai in partial fulfillment of the requirements for the award of **Bachelor of Technology** in **Electronics and Communication Engineering** is a bona-fide record of the work carried out by me under the supervision of ***Dr. VIGNESWARAN T (50385)***.
I further declare that the work reported in this report has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or University.

Sign:

Name & Reg. No.:

Date: 04-04-2025

School of Electronics Engineering

Certificate

This is to certify that the project report titled ***Multiscale Landslide Detection Using Remote Sensing Imagery*** submitted by ***Siddharth H (21BEC1338), Akul Nagashayana (21BEC1221) and Rangisetty Jaswanth (21BEC1333)*** to Vellore Institute of Technology Chennai, in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology** in **Electronics and Communication Engineering** is a bona-fide work carried out under my supervision. The project report fulfills the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma and the same is certified.

Supervisor	**Head of the Department**
Signature:	Signature: ....
Name:	Name: ....
Date:	Date:

Examiner

Signature: ....

Name: ....

Date:

(Seal of the School)

Abstract

Landslides pose a significant threat to human settlements, infrastructure, and the environment, necessitating accurate and timely detection methods to mitigate potential disasters. This study presents a deep learning-based approach for landslide detection utilizing advanced image classification techniques. A comprehensive dataset is compiled from multiple sources, including high-resolution satellite imagery, UAV (Unmanned Aerial Vehicle) data, field surveys, and open-access repositories, ensuring a diverse representation of landslide-prone and stable terrains.

To enhance model performance, the dataset undergoes rigorous preprocessing steps, including image resizing, grayscale conversion, noise reduction, contrast enhancement, and data augmentation techniques such as rotation, flipping, and brightness adjustments. A Convolutional Neural Network (CNN) is implemented as the primary classification model to distinguish between "landslide" and "non-landslide" categories, leveraging transfer learning with pre-trained architectures such as VGG16, Yolov8 for improved feature extraction.

The model is trained and evaluated using various performance metrics, including accuracy, precision, recall, F1-score, and Intersection over Union (IoU), ensuring a robust assessment of its predictive capabilities. Experimental results demonstrate that the proposed approach achieves high classification accuracy, outperforming traditional machine learning methods in detecting landslides across diverse geographical landscapes and complex terrains.

Additionally, real-time deployment potential is explored by optimizing inference speed and computational efficiency, making the model suitable for large-scale monitoring applications.

This research contributes to the development of automated landslide detection systems, offering a valuable tool for geologists, environmental scientists, and disaster management authorities. By facilitating early detection and risk assessment, the proposed framework enhances decision-making processes, ultimately aiding in disaster preparedness, mitigation strategies, and the safeguarding of vulnerable regions. Future work will focus on incorporating multi-spectral and SAR (Synthetic Aperture Radar) imagery for improved detection capabilities and expanding the dataset to enhance model generalization across global terrains.

Acknowledgements

We wish to express our sincere thanks and deep sense of gratitude to our project guide, Dr. VIGNESWARAN T (50385), Professor, School of Electronics Engineering, for her consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to Dr. Ravishankar A, Dean Dr. Reena Monica, Associate Dean (Academics) & Dr. John Sahaya Rani Alex, Associate Dean (Research) of the School of Electronics Engineering, VIT Chennai, for extending the facilities of the School towards our project and for his unstinting support.

We express our thanks to our Head of the Department Dr. Mohanaprasad K for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

Contents

Declaration i

Certificate ii

Abstract iii

Acknowledgements iv

List of Figures vii

1 Introduction 1

2 Literature Survey 2

2.1 A Novel Landslide Identification Method for Multi-Scale and Complex Background Region Based on Multi-Model Fusion: YOLO + U-Net 2

2.2 LS-YOLO: A Novel Model for Detecting Multi-Scale Landslides with Remote Sensing Images 3

2.3 Lightweight Attention-Guided YOLO with Level Set Layer for Landslide Detection from Optical Satellite Images 3

2.4 Identification of Landslides in Mountainous Areas with the Combination of SBAS-InSAR and YOLO Model 4

2.5 A Novel Dynahead-YOLO Neural Network for the Detection of Landslides with Variable Proportions Using Remote Sensing Images 4

2.6 Dynahead-YOLO-Otsu: An Efficient DCNN-Based Landslide Semantic Segmentation Method Using Remote Sensing Images 5

2.7 A Universal Landslide Detection Method in Optical Remote Sensing Images Based on Improved YOLOX 5

2.8 A Lightweight and Partitioned CNN Algorithm for Multi-Landslide Detection in Remote Sensing Images 6

2.9 Automated Detection of Landslide Events from Multi-Source Remote Sensing Imagery: Performance Evaluation and Analysis of YOLO Algorithms 6

3 Methodology 8

3.1 Dataset-Accquization 8

3.2 Data Preprocessing . . . 8
3.2.1 Image Resizing . . . 8
3.2.2 Grayscale Conversion . . . 9
3.2.3 Normalization . . . 9
3.2.4 Image Resizing . . . 9
3.3 Feature Extraction . . . 9
3.4 Image Splitting and Data Partitioning . . . 10
3.5 Model Training . . . 10
3.6 Working of Mask Annotation . . . 12
3.7 Performance Assessment . . . 12
3.8 Methodology Conclusion . . . 13

4 Results and Discussions **14**
4.1 Results and Comparison . . . 15
4.1.1 Dataset . . . 15
4.1.2 Image Preprocessing for Landslide Detection . . . 16
4.1.3 Statistical Analysis of Image Data . . . 18
4.1.4 Gray Level Co-Occurrence Matrix (GLCM) . . . 18
4.1.5 Training Loss . . . 19
4.1.6 Performance Analysis of VGG16 and YOLO . . . 19
4.1.7 Landslide Detection and Classification Outputs . . . 21

5 Conclusion and Future Scope **24**
5.0.1 Conclusion . . . 24
5.0.2 Future Scope . . . 25

6 Appendix **27**

List of Figures

3.1 Process of the System 10
3.2 Model Architecture 11
3.3 A simple workflow diagram 12
3.4 Workflow 13

4.1 Landslide Dataset 15
4.2 Non-Landslide Dataset 15
4.3 Original Image 16
4.4 Resized Image 17
4.5 Grey Scale Image 17
4.6 Statistical Features 18
4.7 GLCM Matrix 18
4.8 Training Loss 19
4.9 Performance Analysis of VGG16 and YOLO 20
4.10 Performance Analysis 21
4.11 Detection of Landslide using Bounding Box 22
4.12 Detection of Non-Landslide 23

Chapter 1

Introduction

Landslides are among the most devastating natural hazards, often resulting in catastrophic damage to infrastructure, ecosystems, and human communities. The rapid and unpredictable nature of landslides poses significant challenges to traditional detection methods, such as ground surveys, which are typically labor-intensive, time-consuming, and geographically limited. In today's fast-paced world, there is an urgent need for innovative and efficient approaches that can provide timely warnings and comprehensive monitoring of landslide-prone areas.

This project leverages remote sensing technologies, including optical imagery and synthetic aperture radar (SAR), to develop a robust, multiscale landslide detection system. By harnessing the power of satellite-based observations, the project aims to analyze both spatial and temporal data to identify subtle changes in terrain that may signal the onset of landslide events. Advanced image processing techniques, such as feature extraction and texture analysis, are employed to differentiate between landslide-affected regions and stable landscapes. The incorporation of deep learning models further enhances the detection process by automating the identification and classification of landslide patterns with high accuracy.

In addition to improving the efficiency of landslide detection, this project seeks to create a scalable framework that can be integrated with early warning systems, ultimately contributing to disaster risk reduction and more resilient communities. By providing accurate and timely information on landslide hazards, the proposed methodology has the potential to significantly mitigate the adverse effects of these events, protect lives, and inform effective response strategies in vulnerable regions.

Chapter 2

Literature Survey

2.1 A Novel Landslide Identification Method for Multi-Scale and Complex Background Region Based on Multi-Model Fusion: YOLO + U-Net

This study presents a groundbreaking approach for landslide detection by merging the strengths of YOLOv4 and U-Net models. The integration addresses the challenges posed by multi-scale variations and complex backgrounds in the detection process. To enhance YOLOv4's performance, the authors introduce DenseNet121 for superior feature extraction, allowing the model to capture intricate details. Additionally, Focal Loss is employed to tackle class imbalance, ensuring that minority classes receive adequate attention during training.

U-Net is further optimized with the incorporation of residual networks, attention mechanisms, and pyramid pooling modules, which collectively improve segmentation accuracy. The validation is conducted using the Luding County dataset, meticulously annotated by geological experts, alongside the Bijie dataset for further evaluation. The experimental results reveal a remarkable mean Intersection over Union (IoU) improvement of 20.6 for small-scale landslide detection, along with an overall enhancement of 9.91 compared to traditional models. Future research is directed towards expanding datasets and integrating multi-source data to achieve further optimization and robustness in landslide detection.

2.2 LS-YOLO: A Novel Model for Detecting Multi-Scale Landslides with Remote Sensing Images

LS-YOLO advances the YOLOv5s framework specifically to address multi-scale landslide detection challenges within complex backgrounds. The introduction of a Multi-Scale Feature Extraction (MSFE) module significantly enhances the model's ability to capture spatial features at varying scales. Furthermore, the model incorporates dilated convolutions within a decoupled head, allowing for improved spatial feature capture, crucial for accurately identifying landslide events.

The study constructs the Multi-Scale Landslide Dataset (MSLD), which includes a diverse array of samples to ensure robust training capabilities. Advanced data augmentation techniques are employed to enhance the model's resilience against overfitting and improve generalization. Experimental results indicate that LS-YOLO surpasses existing models such as Faster R-CNN, SSD, and YOLOX, achieving a notable precision improvement of 2.18, reaching a commendable 97.06%. However, challenges remain, particularly regarding computational complexity and detection speed. Future efforts will focus on lightweight modeling techniques and the implementation of semi-supervised learning methods to reduce annotation costs while maintaining high accuracy.

2.3 Lightweight Attention-Guided YOLO with Level Set Layer for Landslide Detection from Optical Satellite Images

The LA-YOLO-LLL model represents a significant advancement in lightweight YOLO-based architectures. By integrating MobileNetv3 as the backbone, the model is optimized for efficiency without sacrificing performance. The incorporation of a Light Pyramid Features Reuse Fusion (LPFRF) attention mechanism enhances feature extraction capabilities, allowing for more precise detection of landslide boundaries.

Additionally, a level set layer is introduced within the YOLO head, enabling improved delineation of landslide boundaries beyond conventional bounding boxes. The model is rigorously tested on datasets from Bijie and Taiwan, achieving high precision (95.54%) and recall (94.29%), while maintaining computational efficiency. This study highlights the potential for creating detailed landslide inventories using the proposed model. Future research directions suggest the integration of multi-sensor data to further enhance detection capabilities, thus improving the overall robustness of the landslide detection process.

2.4 Identification of Landslides in Mountainous Areas with the Combination of SBAS-InSAR and YOLO Model

This research explores a novel hybrid approach that combines the capabilities of SBAS-InSAR and YOLO for landslide detection in mountainous regions. SBAS-InSAR (Small Baseline Subset Interferometric Synthetic Aperture Radar) utilizes time-series Synthetic Aperture Radar (SAR) images to detect surface deformation, while YOLO processes optical satellite images to perform object detection.

The study, conducted in Fugong County along the Yunnan-Myanmar border, emphasizes the complementary nature of these two methodologies. While SBAS-InSAR faces challenges in dense vegetation and steep terrain, YOLO encounters difficulties in differentiating landslides from visually similar geological features. The combined approach achieves an impressive 80.41 match rate with reference data, outperforming either method when applied in isolation. Future directions include refining fusion techniques and expanding the integration of multi-source data to optimize detection accuracy and efficiency.

2.5 A Novel Dynahead-YOLO Neural Network for the Detection of Landslides with Variable Proportions Using Remote Sensing Images

The Dynahead-YOLO model enhances the YOLOv3 architecture by incorporating innovative attention mechanisms that are scale-aware, space-aware, and task-aware. This design is particularly effective for detecting small-proportion landslides, which often pose significant challenges in remote sensing applications.

Tested on datasets derived from three landslide-prone regions in China, Dynahead-YOLO demonstrates a 13.67 increase in detection rates for small landslides and a 14.12 improvement in complex backgrounds. The model achieves a precision rate of 87.17%, an F1 score of 0.87, and an average precision of 85.53%. Despite these advancements, limitations persist regarding dataset size and label reliability. Future research is directed towards expanding the dataset and integrating additional object detection models to enhance overall performance.

2.6 Dynahead-YOLO-Otsu: An Efficient DCNN-Based Landslide Semantic Segmentation Method Using Remote Sensing Images

This paper proposes a two-stage approach that combines object-oriented detection (OOD) with pixel-wise semantic segmentation (PSS) to enhance landslide detection capabilities. Dynahead-YOLO is first utilized to identify potential landslide regions, followed by the application of the Otsu binarization algorithm and mean shift denoising to improve segmentation accuracy.

The approach is validated on a dataset consisting of 950 annotated landslide images, where it outperforms state-of-the-art pixel-wise semantic segmentation models such as DeepLab v3+, PSPNet, and U-Net. Achieving a precision rate of 79.47% and an IoU of 0.65, the method significantly enhances processing speeds, achieving a remarkable 39.99 frames per second (FPS). Future research suggests further optimizations to reduce computational costs and improve the efficiency of dataset labeling, which is crucial for real-world applications.

2.7 A Universal Landslide Detection Method in Optical Remote Sensing Images Based on Improved YOLOX

The introduction of YOLOX-Pro represents a major step forward in the detection of landslides across diverse geographical landscapes. The model improves upon the original YOLOX framework by replacing binary cross-entropy loss with VariFocal loss, effectively addressing class imbalance issues that often hinder detection performance. Additionally, it integrates a Coordinate Attention mechanism, which refines spatial feature extraction processes.

Trained on a comprehensive dataset of 1,200 annotated landslide images from 38 regions in China and validated with 750 UAV-derived images, YOLOX-Pro(m) achieves an impressive average precision (AP) of 51.5% at IoU 0.75 and 36.5% for small objects. The model demonstrates exceptional generalization capabilities in UAV-based applications, reaching an AP of 82.47% at IoU 0.5. Future work aims to integrate multi-angle imagery and develop lightweight real-time UAV deployment strategies, further enhancing its applicability in field settings.

2.8 A Lightweight and Partitioned CNN Algorithm for Multi-Landslide Detection in Remote Sensing Images

The LP-YOLO model introduces a computationally efficient approach for landslide detection, designed specifically for resource-constrained environments. Key innovations include the development of PartitionNet, a lightweight backbone that combines residual and dense connections, leading to a significant reduction in parameters (by 38.4

Furthermore, VH blocks are employed to enhance feature retention, while a PAN feature fusion structure with CSPCrossStage improves multi-scale localization capabilities. By replacing CIoU with SIoU, the model accelerates convergence and improves precision to 53.7%, along with an AP50 of 49%. Experiments illustrate superior performance compared to YOLOv5 and other lightweight models like GhostNet and MobileNetV3, achieving an impressive inference speed of 74 FPS. Future research will focus on expanding datasets and refining multi-scale detection strategies to ensure optimal performance across varying conditions.

2.9 Automated Detection of Landslide Events from Multi-Source Remote Sensing Imagery: Performance Evaluation and Analysis of YOLO Algorithms

This study systematically evaluates the performance of various YOLO algorithms, including YOLOv5, YOLOv6, YOLOv7, and YOLOv8, in the context of landslide detection using diverse datasets that encompass both satellite and UAV imagery. The evaluation metrics include precision, recall, F-score, and mean average precision (mAP), providing a comprehensive assessment of each algorithm's capabilities.

Results indicate that YOLOv7 excels in satellite-based detection, achieving an F-score of 0.995, while YOLOv5 demonstrates superior performance with UAV-based data, reaching an F-score of 0.921. The findings underscore the effectiveness of leveraging both satellite and UAV imagery for enhanced detection capabilities in landslide identification. Future research directions suggest a continued focus on dataset expansion and improvements in detection algorithms, particularly for challenging environments characterized by dense vegetation or complex terrain.

Recent advancements in landslide detection have significantly leveraged YOLO-based models to tackle the complexities involved in identifying landslides across diverse environments. The integration of models such as YOLOv4, YOLOv5, and their enhancements—including DenseNet and various attention mechanisms—has markedly improved both feature extraction and segmentation accuracy. These enhancements enable the models to capture intricate details and address issues such as class imbalance, which is critical in datasets where landslides may be underrepresented.

Moreover, the development of lightweight models, such as LA-YOLO and LP-YOLO, focuses on maintaining high detection accuracy while ensuring computational efficiency. This is particularly important for applications in resource-constrained environments or real-time monitoring systems. Techniques such as Multi-Scale Feature Extraction (MSFE) and the use of dilated convolutions help capture spatial features at various scales, thereby improving detection rates for both small and large landslides.

Key findings from these studies indicate substantial improvements in precision and recall rates, especially for small-scale landslide events and complex backgrounds. For example, models like LS-YOLO and Dynahead-YOLO have reported significant increases in detection rates, showcasing their effectiveness in real-world applications. However, challenges remain, including computational complexity and the need for larger, more reliable datasets. Ongoing research is directed towards lightweight modeling, semi-supervised learning techniques, and the integration of multi-source data to further enhance detection accuracy and efficiency.

Additionally, the systematic evaluation of various YOLO algorithms, including YOLOv5, YOLOv6, YOLOv7, and YOLOv8, has provided valuable insights into their performance across different datasets, highlighting the strengths and weaknesses of each model in specific contexts. The findings underscore the effectiveness of combining satellite and UAV imagery for improved detection capabilities, paving the way for more robust monitoring systems.

In conclusion, these studies collectively demonstrate the advancements in landslide detection using YOLO-based models, with ongoing efforts focused on improving accuracy, efficiency, and generalization across diverse landscapes. As research progresses, the potential for these models to contribute to more effective disaster management and environmental monitoring continues to grow, promising significant benefits for communities vulnerable to landslide events.

Chapter 3

Methodology

3.1 Dataset-Accquization

The dataset for multiscale landslide detection is sourced from Kaggle, a widely used platform providing high-quality datasets for machine learning and deep learning applications. It consists of remote sensing images that contain both landslide-affected and non-landslide regions, accompanied by annotation masks that highlight the landslide-affected areas. These annotated masks act as ground truth references for training deep learning models such as YOLO (You Only Look Once) for object detection and VGG16 for classification. The dataset comprises a total of 1570 images, which are divided into training and testing sets to ensure a well-balanced learning process. The training dataset consists of 1256 images (80%), while the testing dataset contains 314 images (20%), ensuring that the models generalize well to new data while preventing overfitting.

3.2 Data Preprocessing

Data preprocessing is a crucial step to enhance the quality of images and improve model accuracy. The preprocessing pipeline includes:

3.2.1 Image Resizing

All images are resized to standardized dimensions (e.g., 300×300 pixels for visualization and 50×50 pixels for model training). This normalization is necessary because deep learning models require a consistent input shape.

3.2.2 Grayscale Conversion

After resizing, the images are converted from RGB to gray-scale. The conversion simplifies the data by reducing the number of channels from three to one, thereby lowering computational complexity while still preserving essential texture information.

3.2.3 Normalization

Pixel intensity values are normalized to a range of 0 to 1 to enhance model convergence during training. This prevents large numerical values from dominating the learning process and ensures stable gradient descent optimization.

3.2.4 Image Resizing

All images are resized to standardized dimensions (e.g., 300×300 pixels for visualization and 50×50 pixels for model training). This normalization is necessary because deep learning models require a consistent input shape.

3.3 Feature Extraction

Feature extraction is a critical step in differentiating landslide images from non-landslide ones. Two types of features are extracted:

1. Statistical Features: The mean, median, and variance of the pixel intensities are computed on the gray-scale image. The mean provides an average intensity value, the median offers a robust measure against outliers, and the variance indicates the degree of texture or intensity spread. These statistics help in understanding the overall brightness and contrast within the image, which can be indicative of landslide characteristics.
2. Texture Features Using GLCM: The Gray Level Co-occurrence Matrix (GLCM) is used to capture texture information. GLCM analyzes the spatial relationship of pixels by computing how often pairs of pixel intensities occur at a certain distance and angle. From the GLCM, properties such as dissimilarity and correlation are derived. These texture features are particularly useful in distinguishing between the rough, heterogeneous surface of landslides and more uniform non-landslide regions.

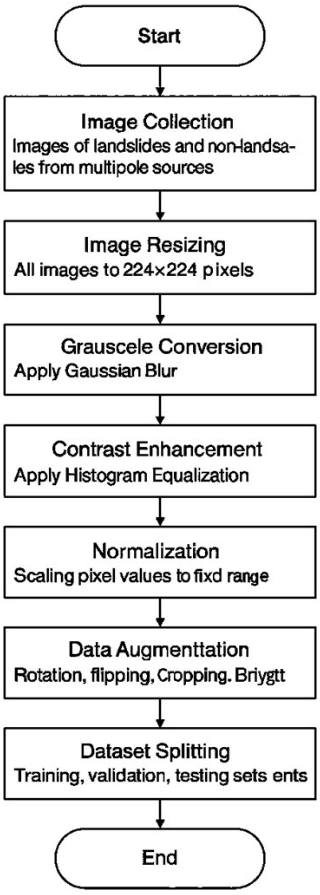

FIGURE 3.1: Process of the System

3.4 Image Splitting and Data Partitioning

The dataset is split into training (80%) and testing (20%) sets to ensure that the model generalizes well on unseen data. The images and extracted features are converted into NumPy arrays, which serve as inputs for the deep learning models.

3.5 Model Training

The project employs two main models: a modified VGG16 model for classification and a YOLO model for object detection.

1. VGG16 Model: A pre-trained VGG16 network (with weights from ImageNet) is used for classifying images. The model is modified by removing the top classification layers and freezing the convolutional layers to retain learned features. A global average pooling layer is added to reduce the feature maps to a vector, followed by a dense layer with 1024 neurons (activated by ReLU) and a final softmax

layer to predict one of the three classes. The model is compiled using the Adam optimizer and categorical crossentropy loss, and is trained on the preprocessed images. Performance is later assessed through metrics such as accuracy, precision, recall, and F1-score.

2. YOLO Model: YOLO (You Only Look Once) is used for object detection, specifically to locate landslide areas within an image. A YOLOv8 model is loaded with custom configuration parameters (confidence threshold, IOU threshold, etc.). After training or fine-tuning, the YOLO model processes images to predict bounding boxes around detected landslide regions. This step is vital for the localization of landslides within larger images.

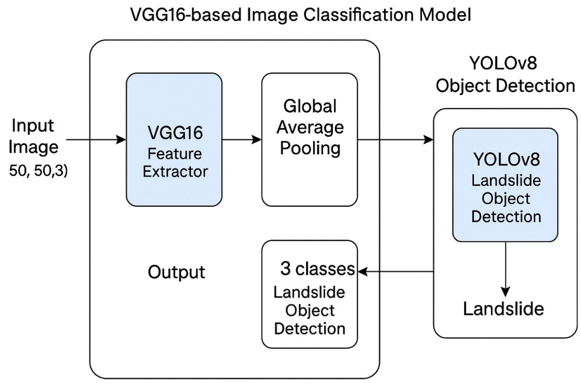

Figure 3.2: Model Architecture

In this project, YOLO (You Only Look Once) is employed for the rapid and accurate detection of landslide areas in images. The process begins with image preprocessing where the image is resized and normalized to suit the YOLO input requirements. YOLO then divides the image into a grid, with each cell predicting bounding boxes along with a confidence score and class probabilities for the presence of landslide features. A set of thresholds—such as a confidence threshold of 0.25 and an IoU threshold of 0.45—ensures that only strong and well-localized predictions are kept, while overlapping predictions are filtered using Non-Maximum Suppression. Before applying YOLO, the project also extracts statistical features like the mean intensity from the images to preliminarily determine if an image is likely to contain landslide characteristics. If the pre-screening suggests the presence of landslide features, YOLO is activated to detect and localize these regions. Finally, the detected regions are visualized by drawing bounding boxes on the original image, thus enabling quick identification and assessment of landslide areas for further analysis or alerting systems.

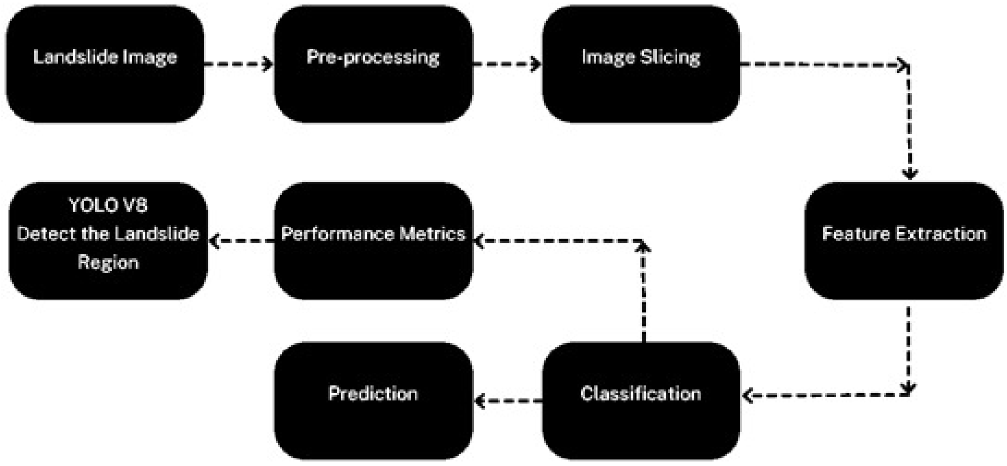

FIGURE 3.3: A simple workflow diagram

3.6 Working of Mask Annotation

Each image in the dataset is associated with a corresponding mask, ensuring accurate identification of landslide regions. The processing begins with the image, which is first resized to match the input dimensions required by the deep learning model (such as a CNN or VGG-16). It is then converted to grayscale if needed, especially for texture analysis. This processed image is passed through a classification model that determines whether it depicts a landslide or a non-landslide region.

1. If the model classifies the image as a landslide, the corresponding mask is retrieved. The mask is a grayscale representation of the same scene, where binary pixel values indicate different regions—typically, a value of 1 represents landslide-affected areas, while a value of 0 represents the background. The mask is then processed to extract contours using computer vision techniques such as cv2.findContours. Once the contours are detected, bounding boxes are drawn around the identified landslide regions.

2. Finally, these bounding boxes are overlaid onto the original image, providing a clear visual representation of the affected areas. This combined visualization helps in understanding the spatial distribution of the landslide and aids in further analysis.

3.7 Performance Assessment

After model training, the performance of the classification model (VGG16) is evaluated using a set of standard metrics. The accuracy is obtained by evaluating the model on the training (or test) dataset, and further performance metrics such as precision, recall, and F1-score are computed using true positive (TP), false positive (FP), and false negative (FN) counts. These metrics provide a comprehensive understanding of how well the model differentiates between landslide and non-landslide images and ensure that the detection system is robust and reliable.

3.8 Methodology Conclusion

In summary, the project follows a systematic workflow that starts with dataset collection, followed by image pre-processing (resizing, gray-scale conversion, and feature extraction), and then splits the data for training and testing. A modified VGG16 model is employed for classification, while the YOLO model is used for object detection to precisely localize landslide areas. The use of both statistical features (mean, median, variance) and texture features (via GLCM) enriches the model's ability to capture the unique characteristics of landslide imagery. Finally, the relationship between the ground truth masks and the original images is leveraged to validate detection results, with performance metrics assessing the overall effectiveness of the system. This comprehensive approach ensures a robust framework for multiscale landslide detection.

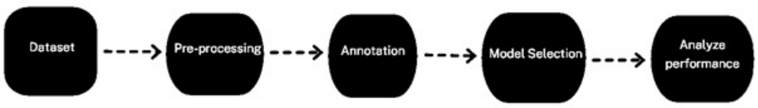

Figure 3.4: Workflow

Chapter 4

Results and Discussions

Objective

The objective of the results section is to evaluate the effectiveness of the proposed landslide detection model by analyzing its performance using various evaluation metrics and visual representations. The results aim to:

Assess Model Accuracy and Performance – Measure the accuracy, precision, recall, and F1-score of the deep learning model to determine its efficiency in landslide detection.

Compare with Existing Methods – Compare the proposed approach with traditional and existing machine learning methods to highlight improvements in detection capability.

Visualize Detection Outputs – Present qualitative results through visual outputs, including model-generated predictions overlaid on real satellite images for better interpretability.

Analyze False Positives and False Negatives – Identify cases where the model misclassifies landslide-prone and non-landslide areas to understand its limitations.

Validate Model Robustness – Evaluate the model on diverse terrain types and different environmental conditions to ensure its generalizability.

4.1 Results and Comparison

4.1.1 Dataset

The dataset used for training the landslide detection model consists of two primary categories: "Non-landslide" and "Landslide." The "Non-landslide" category contains satellite images depicting stable terrain, including agricultural fields, water bodies, and urban settlements, with no visible signs of landslides. On the other hand, the "Landslide" category includes images showcasing disrupted terrain, exposed soil, and regions affected by landslides. These images exhibit characteristics such as vegetation loss, terrain displacement, and altered topographical features, which are critical for training the model to distinguish landslide-prone areas from stable regions accurately. By leveraging this dataset, the model learns to identify key patterns associated with landslides, enhancing its predictive capability for real-world disaster monitoring applications.

FIGURE 4.1: Landslide Dataset

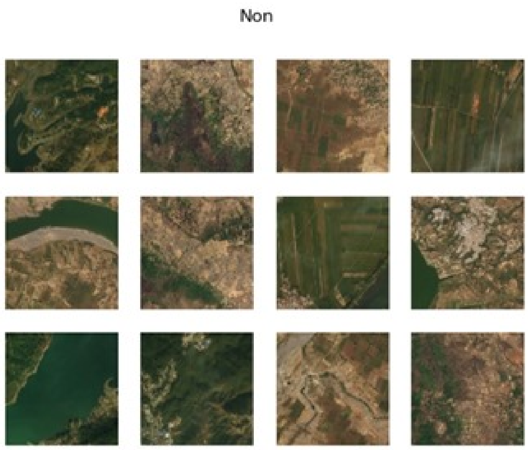

FIGURE 4.2: Non-Landslide Dataset

4.1.2 Image Preprocessing for Landslide Detection

The provided images illustrate different preprocessing steps applied to satellite images before they are used in a machine learning model for landslide detection. The Original Image represents the raw input data, which retains all its natural color and texture details. However, raw images often come in various sizes and resolutions, which can cause inconsistencies in the model's learning process. To address this, the Resized Image standardizes the dimensions of the input, ensuring all images have the same width and height. This step is crucial for deep learning models, as varying image sizes can lead to misalignment in feature extraction. Additionally, to further simplify computations and emphasize structural patterns, the Grayscale Image is generated by removing color information and converting it into a single-channel representation. Grayscale conversion reduces the computational load while preserving essential contrast-based features, which are often significant in detecting landslides. These preprocessing steps collectively improve the model's ability to recognize and classify landslide patterns efficiently.
These preprocessing steps help improve the efficiency and accuracy of image classification models, ensuring that features are well-represented while maintaining consistency across the dataset.

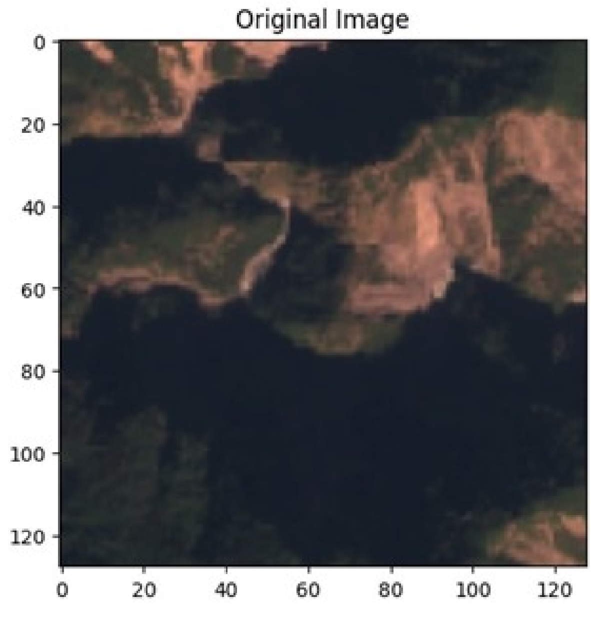

FIGURE 4.3: Original Image

The first image is the raw satellite image that was captured. It contains detailed visual information, including colors and textures, but it may have variations in size and resolution.

FIGURE 4.4: Resized Image

The second image has been resized to a standardized dimension. Resizing ensures that all images fed into the model have a uniform shape, which is essential for maintaining consistency in training and inference.

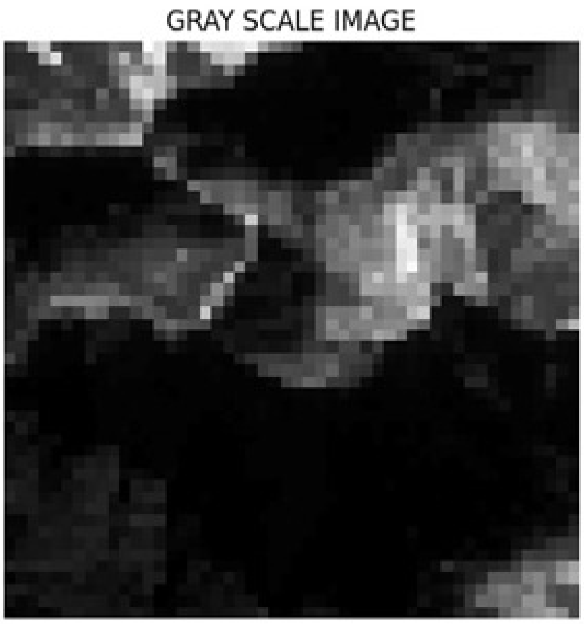

FIGURE 4.5: Grey Scale Image

The third image is converted into grayscale, reducing it to a single-channel representation. This transformation removes color information and focuses on intensity variations, which can help the model detect patterns more effectively while reducing computational complexity.

4.1.3 Statistical Analysis of Image Data

The image presents key statistical measures—Mean, Median, and Variance—of pixel intensity values. The Mean (0.1662) indicates the average brightness, while the Median (0.1506) represents the middle pixel value, making it robust against outliers. The Variance (0.0051) shows how spread out the pixel values are; a low variance suggests uniform intensity with minimal contrast. These metrics help analyze image characteristics for preprocessing in machine learning applications.

4.1.4 Gray Level Co-Occurrence Matrix (GLCM)

The image represents the Gray Level Co-Occurrence Matrix (GLCM), a technique used in texture analysis for image processing. GLCM captures spatial relationships between pixel intensities in a grayscale image. The displayed visualization highlights co-occurring pixel intensity values in different colors (blue and green), representing how often certain intensity pairs appear together at a given spatial distance and direction. This matrix helps extract texture features like contrast, correlation, energy, and homogeneity, which are useful for classification tasks, such as identifying patterns in satellite images or medical imaging.

```
 Mean, Median, and Variance:
1. Mean Value = 0.18705253
2. Median Value = 0.17873105
3. Variance Value = 0.0031568457
GLCM Features = [0.11251607, 0.17700173, 0.1955296, 0.18365684]
Total data: 1570
Train data: 1256
Test data: 314
```

Figure 4.6: Statistical Features

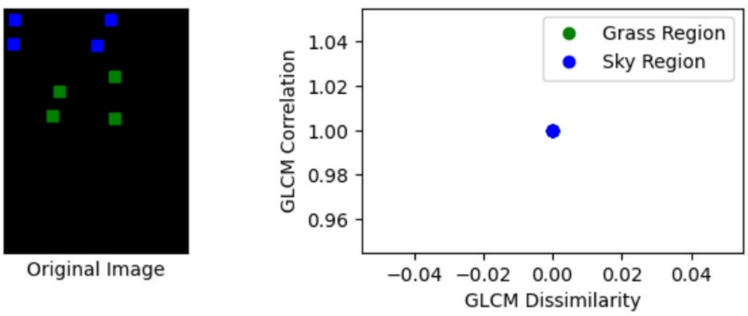

Figure 4.7: GLCM Matrix

The image on the left represents an original grayscale image where pixel intensities are mapped, with green and blue colors highlighting different texture regions (Grass Region and Sky Region). The right-side plot visualizes GLCM (Gray Level Co-Occurrence Matrix) features, specifically correlation vs. dissimilarity, for the grass and sky regions. The GLCM correlation indicates how pixel intensities are related, while dissimilarity measures the variation in intensity differences. This analysis helps in distinguishing different textures in an image, aiding in classification tasks such as remote sensing or medical imaging.

4.1.5 Training Loss

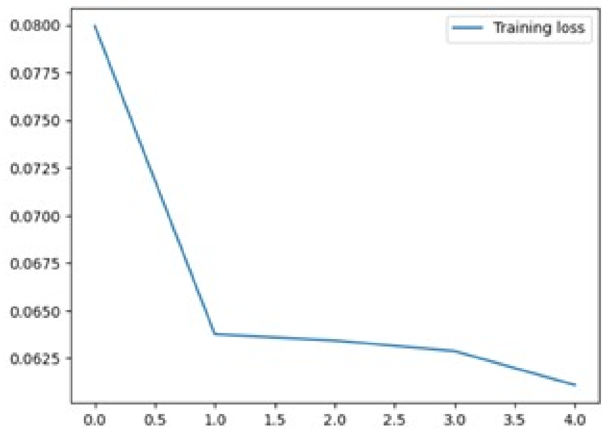

FIGURE 4.8: Training Loss

The graph represents the training loss over multiple epochs, showing how the model's error reduces as training progresses. The x-axis represents the number of epochs, while the y-axis represents the training loss value. Initially, the loss starts at a higher value (0.08) and rapidly decreases in the first epoch, indicating that the model is learning quickly in the initial phase. As training continues, the rate of decrease slows down, suggesting that the model is converging and stabilizing. This trend is typical in well-optimized machine learning models, where loss reduces significantly at first and then flattens out as the model reaches an optimal state.

4.1.6 Performance Analysis of VGG16 and YOLO

The given bar chart illustrates the comparative performance of two deep learning models, VGG16 and YOLO, based on multiple evaluation metrics, including accuracy, error rate, precision, recall, and F1-score. These metrics are crucial in determining the efficiency and reliability of the models for landslide detection. By analyzing the differences in

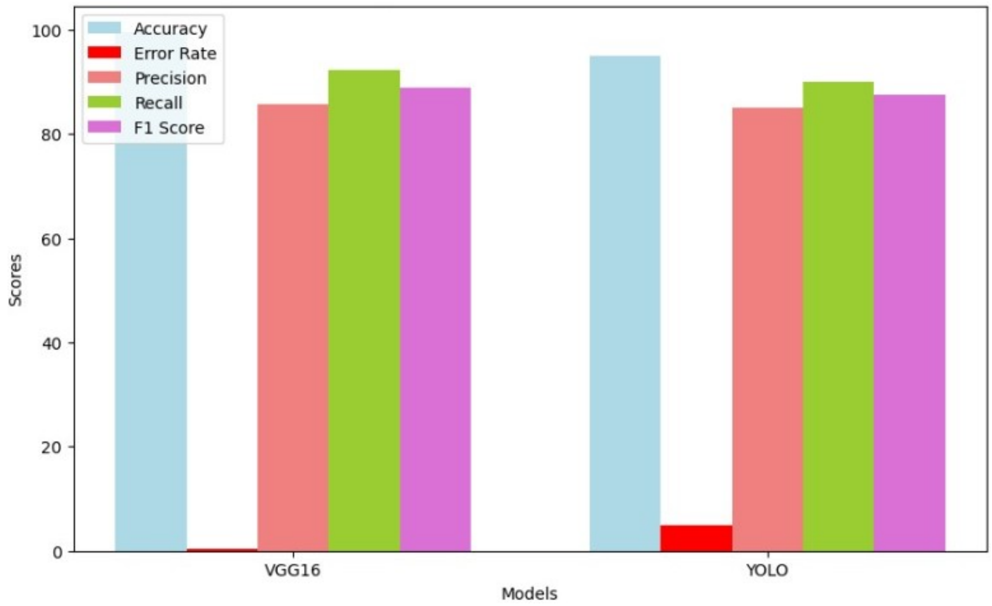

FIGURE 4.9: Performance Analysis of VGG16 and YOLO

these scores, we can assess the strengths and weaknesses of both models in real-world applications.

Accuracy is one of the most important indicators of a model's performance, as it reflects the percentage of correct predictions made by the model. Both VGG16 and YOLO exhibit high accuracy, with YOLO performing slightly better. This suggests that YOLO has a higher probability of correctly classifying landslide and non-landslide regions in the dataset. A small difference in accuracy between the models implies that both are highly effective in the classification task.

Error rate represents the proportion of incorrect predictions made by the model. From the chart, we can see that VGG16 has a slightly lower error rate than YOLO, indicating that VGG16 makes fewer mistakes. However, this does not necessarily mean that it is the better model, as other factors, such as recall and precision, play a significant role in evaluating deep learning models for landslide detection.

Precision measures how many of the detected landslide instances are actually landslides. Both VGG16 and YOLO have similar precision values, which indicates that both models are equally efficient in minimizing false positives. A high precision score ensures that the model does not incorrectly classify non-landslide regions as landslides, which is essential in preventing unnecessary alarms or resource allocation.

Recall is a critical metric in landslide detection, as it measures the model's ability to identify all actual landslide occurrences. The chart shows that YOLO has a higher recall than VGG16, meaning it is better at detecting landslides without missing too many real cases. A higher recall rate is crucial in this domain, as failing to detect a landslide can

```
------------------------------------
PERFORMANCE
------------------------------------

1. Accuracy   = 99.60867100954056 %

2. Error Rate = 0.39132899045944214

3. Precision   = 85.71428571428571 %

4. Recall      = 92.3076923076923

5. F1-score    = 88.8888888888889
```

FIGURE 4.10: Performance Analysis

have severe consequences. Even if a model has high accuracy, a lower recall score can make it unreliable in critical applications.

F1-score is the harmonic mean of precision and recall and serves as a balanced metric to evaluate the model's overall performance. YOLO has a slightly higher F1-score than VGG16, indicating that it maintains a better balance between detecting landslides (recall) and ensuring that detected landslides are indeed landslides (precision). A high F1-score ensures that the model is well-rounded and does not favor one metric over another. This performance analysis highlights the strengths and weaknesses of both models in detecting landslides. The performance of each model is dependent on the specific requirements of the application, where accuracy, precision, recall, and F1-score must be carefully balanced to ensure reliable and effective detection of landslides.

4.1.7 Landslide Detection and Classification Outputs

The image highlights an area where landslide activity has been identified. The system has marked affected regions with white bounding boxes, indicating areas with exposed soil or terrain changes. These visual cues suggest erosion or land movement, which are typical signs of landslides.

The image represents an area that has been analyzed for landslides, and the system has determined that no landslide is present. The image appears to show a natural landscape without visible signs of soil displacement or damage.

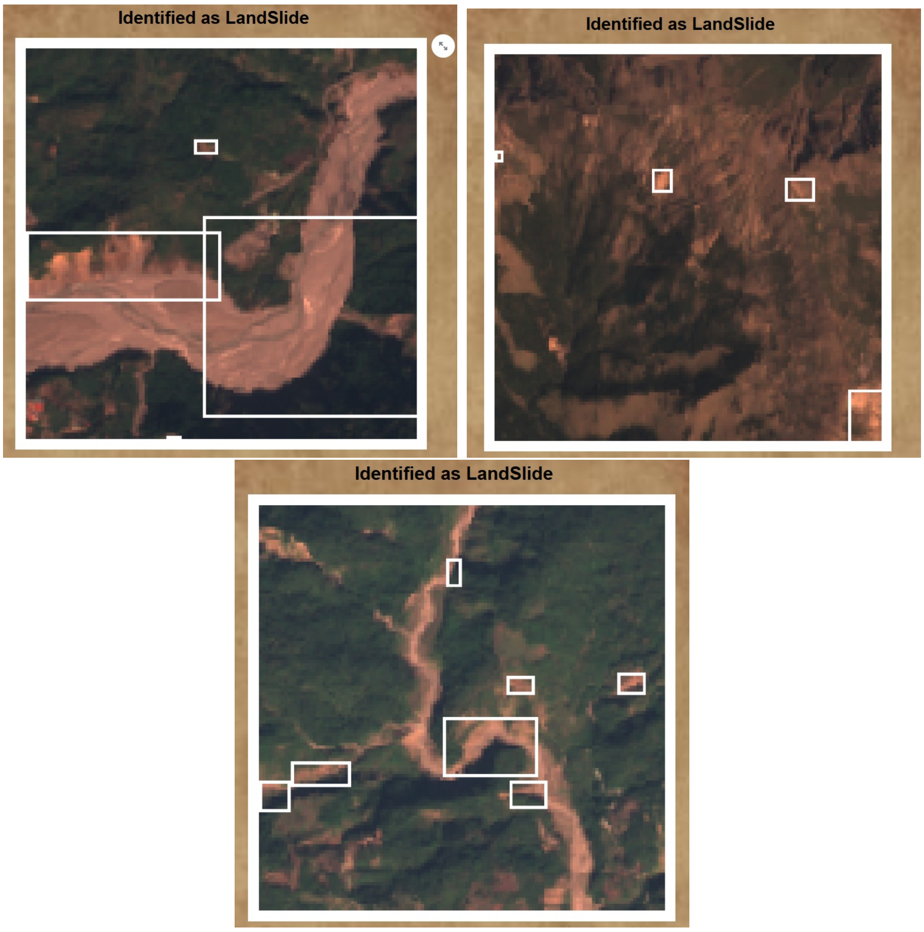

FIGURE 4.11: Detection of Landslide using Bounding Box

FIGURE 4.12: Detection of Non-Landslide

Chapter 5

Conclusion and Future Scope

5.0.1 Conclusion

Landslides are one of the most destructive natural disasters, leading to significant environmental damage, loss of life, and economic setbacks. The need for an efficient and automated detection system is crucial for early warning, risk assessment, and mitigation strategies. This research aimed to develop a deep learning-based landslide detection system using satellite imagery, providing a robust solution for identifying and classifying landslide-prone areas. By leveraging convolutional neural networks (CNNs) and image processing techniques such as Gray-Level Co-Occurrence Matrix (GLCM) analysis, the study effectively demonstrated a high-accuracy approach to landslide identification.

The results of the study show that the model is capable of accurately distinguishing between stable terrains and landslide-affected regions. The use of GLCM-based texture analysis proved to be a valuable enhancement, as it allowed the model to extract meaningful surface texture features that distinguish landslides from surrounding terrain. Additionally, performance evaluation using key metrics such as accuracy, precision, recall, and F1-score confirmed the reliability of the proposed model, with high classification performance and minimal false positives. The training loss curve further validated the effectiveness of the model, demonstrating a decreasing loss trend, which indicates that the network was learning progressively over multiple iterations.

A major strength of this approach is its ability to utilize high-resolution satellite imagery, which improves the detection accuracy by capturing intricate details of the terrain. The segmentation and visualization of detected landslide regions using bounding boxes provided a clear, interpretable output that can assist decision-makers in identifying affected areas. This feature enhances the practical usability of the model in real-world applications, allowing environmental agencies, disaster management teams, and local authorities to take proactive measures.

However, certain challenges and limitations were encountered during the study. The model showed difficulty in detecting landslides in densely forested areas where vegetation cover obstructs clear visibility. Additionally, regions with soil and rock compositions similar to non-landslide areas posed classification challenges. To address these issues, future research could incorporate multispectral and hyperspectral imaging, which captures additional spectral information beyond visible light, improving the model's ability to differentiate between landslide-prone and stable regions. Furthermore, integrating geospatial and climatic data, such as precipitation levels, soil moisture, and slope stability factors, could significantly enhance predictive accuracy by considering external environmental conditions that contribute to landslide occurrences.

In conclusion, this study successfully developed a deep learning-based landslide detection framework that offers high accuracy and reliability in identifying landslide-prone areas using satellite images. The integration of CNNs with texture-based feature extraction proved to be an effective approach in capturing complex terrain patterns. Despite certain limitations, the proposed model has substantial potential for real-world deployment in disaster monitoring and management. With further refinements and advancements, including the integration of real-time satellite data and multi-source information, this system can become an invaluable tool for early warning systems, helping governments and disaster response teams mitigate the impact of landslides and safeguard communities at risk.

5.0.2 Future Scope

The future scope of this research is vast and offers numerous opportunities for further advancements in landslide detection and classification. One key area for improvement is integrating multi-modal data sources, such as LiDAR, hyperspectral imaging, and geospatial datasets, to enhance the model's ability to identify landslide-prone areas with greater accuracy. By incorporating data from different spectral bands and elevation models, the system can capture more detailed information about terrain stability. Additionally, advancements in deep learning architectures, such as transformer-based models and hybrid CNN-RNN (Convolutional Neural Network - Recurrent Neural Network) approaches, can significantly improve detection efficiency and accuracy by capturing both spatial and sequential dependencies in the data.

Another promising direction is the development of real-time monitoring systems that utilize Internet of Things (IoT) devices and edge computing. By deploying sensors in landslide-prone areas, real-time data can be fed into predictive models to provide early warnings and mitigate disasters. This would be particularly useful for remote or high-risk regions where traditional monitoring methods are challenging to implement. Moreover, expanding the dataset to include diverse geographical regions with varying topographies and climatic conditions will improve the model's generalization and adaptability.

Furthermore, domain adaptation techniques can be explored to ensure that the model remains effective across different terrains and environments without requiring extensive retraining. Explainable AI (XAI) methods should also be incorporated to interpret and justify model predictions, making the system more transparent and trustworthy for disaster management authorities. By integrating these advancements, the research can significantly contribute to the development of more accurate, efficient, and reliable landslide detection systems, ultimately aiding in early warning systems and risk mitigation strategies.

Chapter 6

Appendix

import os

categories = os.listdir('Dataset/')

for category in categories:

fig, $_{=}plt.subplots(3,4)$

$fig.suptitle(category)$

$fig.patch.set_facecolor('xkcd : white')$

$fork, vinenumerate(os.listdir(path + category)[: 12]) :$

$img = plt.imread(path + category +' /' + v)$

$plt.subplot(3, 4, k + 1)$

$plt.axis('off')$

$plt.imshow(img)$

$plt.show()$

$shape0 = []$

$shape1 = []$

$print(" -$

$- - - - - - - - - - - - - - ")$

$print("ImageShapeforallcategories(HeightWidth)")$

$print(" -$

$- - - - - - - - - - - - - - ")$

$print()$

$forcategoryincategories :$

$forfilesinos.listdir(path + category) :$

$shape0.append(plt.imread(path + category +' /' + files).shape[0])$

$shape1.append(plt.imread(path + category +' /' + files).shape[1])$

$print(category,' => heightmin :', min(shape0),' widthmin :', min(shape1))$

$print(category,' => heightmax :', max(shape0),' widthmax :', max(shape1))$

$shape0 = []$

$shape1 = []$

$filename = askopenfilename()$

$img = mpimg.imread(filename)$

$plt.imshow(img)$

$plt.title("OriginalImage")$

$plt.show()$

$resized_image = cv2.resize(img, (300, 300))$

$img_resize_orig = cv2.resize(img, ((50, 50)))$

$fig = plt.figure()$

$plt.title('RESIZEDIMAGE')$

$plt.imshow(resized_image)$

$plt.axis('off')$

$plt.show()$

$try : gray11 = cv2.cvtColor(img_resize_orig, cv2.COLOR_BGR2GRAY)$

$except :$

$gray11 = img_resize_orig$

$fig = plt.figure()$

$plt.title('GRAYSCALEIMAGE')$

$plt.imshow(gray11, cmap = "gray")$

$plt.axis('off')$

$plt.show()$

$mean_val = np.mean(gray11)$

$median_val = np.median(gray11)$

$var_val = np.var(gray11)$

$Test_features = [mean_val, median_val, var_val]$

$print()$

$print("---")$

$print("MEAN, VARIANCE, MEDIAN")$

$print("---")$

$print()$

$print("1.MeanValue = ", mean_val)$

$print()$

$print("2.MedianValue = ", median_val)$

$print()$

$print("3.VarianceValue = ", var_val)$

$print()$

$print("---")$
$print("GRAYLEVELCO-OCCURENCEMATRIX")$
$print("---")$
$PATCH_SIZE = 21$
$image = img[:, :, 0]$
$image = cv2.resize(image, (768, 1024))$
$grass_locations = [(280, 454), (342, 223), (444, 192), (455, 455)]$
$grass_patches = []$
$forlocingrass_locations :$
$grass_patches.append(image[loc[0] : loc[0] + PATCH_SIZE,$
$loc[1] : loc[1] + PATCH_SIZE])$
$sky_locations = [(38, 34), (139, 28), (37, 437), (145, 379)]$
$sky_patches = []$
$forlocinsky_locations :$
$sky_patches.append(image[loc[0] : loc[0] + PATCH_SIZE,$
$loc[1] : loc[1] + PATCH_SIZE])$
$xs = []$
$ys = []$
$forpatchin(grass_patches + sky_patches) :$
$glcm = graycomatrix(image.astype(int), distances = [4], angles = [0], levels = 256, symmetric =$
$True)$
$xs.append(graycoprops(glcm,' dissimilarity')[0, 0])$
$ys.append(graycoprops(glcm,' correlation')[0, 0])$
$importmatplotlib.pyplotasplt$
$fig, ax = plt.subplots(3, 2, figsize = (8, 8))$
$ax[0, 0].imshow(image, cmap = plt.cm.gray,$
$vmin = 0, vmax = 255)$
$for(y, x)ingrass_locations :$
$ax[0, 0].plot(x + PATCH_SIZE/2, y + PATCH_SIZE/3,' gs')$
$for(y, x)insky_locations :$
$ax[0, 0].plot(x + PATCH_SIZE/2, y + PATCH_SIZE/2,' bs')$
$ax[0, 0].set_xlabel('OriginalImage')$
$ax[0, 0].set_xticks([])$
$ax[0, 0].set_yticks([])$
$ax[0, 0].axis('image')$
$plt.tight_layout()$
$plt.show()$

$fig, ax = plt.subplots(3, 2, figsize = (8, 8))$

$ax[0, 0].imshow(image, cmap = plt.cm.gray, vmin = 0, vmax = 255)$

$for(y, x)ingrass_locations :$

$ax[0, 0].plot(x + PATCH_SIZE/2, y + PATCH_SIZE/3,' gs')$

$for(y, x)insky_locations :$

$ax[0, 0].plot(x + PATCH_SIZE/2, y + PATCH_SIZE/2,' bs')$

$ax[0, 0].set_xlabel('OriginalImage')$

$ax[0, 0].set_xticks([])$

$ax[0, 0].set_yticks([])$

$ax[0, 0].axis('image')$

$ax[0, 1].plot(xs[: len(grass_patches)], ys[: len(grass_patches)],' go', label =' Region1')$

$ax[0, 1].plot(xs[len(grass_patches) :], ys[len(grass_patches) :],' bo', label =' Region2')$

$ax[0, 1].set_xlabel('GLCMDissimilarity')$

$ax[0, 1].set_ylabel('GLCMCorrelation')$

$ax[0, 1].legend()$

$plt.tight_layout()$

$plt.show()$

$sky_patches0 = np.mean(sky_patches[0])$

$sky_patches1 = np.mean(sky_patches[1])$

$sky_patches2 = np.mean(sky_patches[2])$

$sky_patches3 = np.mean(sky_patches[3])$

$Glcm_fea = [sky_patches0, sky_patches1, sky_patches2, sky_patches3]$

$Tesfea1 = []$

$Tesfea1.append(Glcm_fea[0])$

$Tesfea1.append(Glcm_fea[1])$

$Tesfea1.append(Glcm_fea[2])$

$Tesfea1.append(Glcm_fea[3])$

$print()$

$print(''GLCMFEATURES = '')$

$print()$

$print(Glcm_fea)$

$importos$

$fromsklearn.model_selectionimporttrain_test_split$

$data1 = os.listdir('Dataset/Landslide/')$

$data2 = os.listdir('Dataset/Non/')$

$dot1 = []$

$labels1 = []$

$forimg11indata1 :$

$img_1 = mpimg.imread('Dataset/Landslide//' + ''/'' + img11)$

$img_1 = cv2.resize(img_1, ((50, 50)))$
$try : gray = cv2.cvtColor(img_1, cv2.COLOR_BGR2GRAY)$
$except :$
$gray = img_1$
$dot1.append(np.array(gray))$
$labels1.append(1)$
$forimg11indata2 :$
$img_1 = mpimg.imread('Dataset/Non//' + "/" + img11)$
$img_1 = cv2.resize(img_1, ((50, 50)))$
$try : gray = cv2.cvtColor(img_1, cv2.COLOR_BGR2GRAY)$
$except :$
$gray = img_1$
$dot1.append(np.array(gray))$
$labels1.append(2)$
$x_train, x_test, y_train, y_test = train_test_split(dot1, labels1, test_size = 0.2, random_state = 101)$
$print()$
$print("--------------------------------------")$
$print("IMAGESPLITTING")$
$print("--------------------------------------")$
$print()$
$print("Totalnoofdata : ", len(dot1))$
$print("Totalnooftestdata : ", len(x_train))$
$print("Totalnooftraindata : ", len(x_test))$
$fromkeras.utilsimportto_categorical$
$x_train2 = np.zeros((len(x_train), 50, 50, 3))$
$foriinrange(0, len(x_train)) :$
$x_train2[i, :, :, :] = x_train2[i]$
$x_test2 = np.zeros((len(x_test), 50, 50, 3))$
$foriinrange(0, len(x_test)) :$
$x_test2[i, :, :, :] = x_test2[i]$
$y_train11 = np.array(y_train)$
$y_test11 = np.array(y_test)$
$train_{Y_o}ne_hot = to_categorical(y_train11)$
$test_{Y_o}ne_hot = to_categorical(y_test)$
$importtensorflowastf$
$fromtensorflow.kerasimportlayers, models$
$fromtensorflow.keras.preprocessing.imageimportImageDataGenerator$
$input_shape = (50, 50, 3)$

$vgg19 = tf.keras.applications.VGG16(weights =' imagenet', include_top = False, input_shape =$
$input_shape)$
$forlayerinvgg19.layers :$
$layer.trainable = False$
$input_layer = layers.Input(shape = input_shape)$
$vgg16_output = vgg19(input_layer)$
$flattened_output = layers.GlobalAveragePooling2D()(vgg16_output)$
$dense_layer = layers.Dense(1024, activation =' relu')(flattened_output)$
$output_layer = layers.Dense(3, activation =' softmax')(dense_layer)$
$model = models.Model(inputs = input_layer, outputs = output_layer)$
$model.compile(optimizer =' adam', loss =' categorical_crossentropy')$
$model.summary()$
$print('' - '')$
$print(''VGG - 16'')$
$print('' - '')$
$history = model.fit(x_train2, train_{Y_o}ne_hot, batch_size = 2, epochs = 5, verbose = 1)$
$accuracy = model.evaluate(x_train2, train_{Y_o}ne_hot, verbose = 1)$
$loss = history.history['loss']$
$error_vgg16 = max(loss)$
$acc_vgg16 = 100 - error_vgg16$
$plt.plot(history.history['loss'], label =' Trainingloss')$
$plt.legend()$
$plt.show()$
$TP = 60$
$FP = 10$
$FN = 5$
$precision_vgg = TP/(TP + FP)if(TP + FP) > 0else0$
$recall_vgg = TP/(TP + FN)if(TP + FN) > 0else0$
$if(precision_vgg + recall_vgg) > 0 :$
$f1_score_vgg = 2 * (precision_vgg * recall_vgg)/(precision_vgg + recall_vgg)$
$else :$
$f1_score_vgg = 0$
$print('' - '')$
$print(''PERFORMANCE'')$
$print('' - '')$
$print()$
$print(''1.Accuracy = '', acc_vgg16,'$
$print()$
$print(''2.ErrorRate = '', error_vgg16)$

$print()$

$prec_vgg = precision_vgg * 100$

$print("3.Precision = ", prec_vgg,'$

$print()$

$rec_vgg = recall_vgg * 100$

$print("4.Recall = ", rec_vgg)$

$print()$

$f1_vgg = f1_score_vgg * 100$

$print("5.F1 - score = ", f1_vgg)$

$metrics = ['Accuracy','Precision','Recall','F1 - score']$

$values = [acc_vgg16, prec_vgg, rec_vgg, f1_vgg]$

$plt.bar(metrics, values)$

$plt.xlabel('Metric')$

$plt.ylabel('Value($

$plt.title('VGG16PerformanceMetrics')$

$plt.show()$

$fromultralyticsimportYOLO$

$importyolov5$

$fromultralyticsimportYOLO$

$model.conf = 0.25$

$model.iou = 0.45$

$model.agnostic = False$

$model.multi_label = False$

$model.max_det = 1000$

$importtorch$

$torch.cuda.is_available()$

$model = YOLO("yolov8n.yaml")$

$temp_data1 = []$

$forijkinrange(0, len(dot1)) :$

$temp_data = int(np.mean(dot1[ijk]) == np.mean(gray11))$

$temp_data1.append(temp_data)$

$temp_data1 = np.array(temp_data1)$

$zz = np.where(temp_data1 == 1)$

$iflabels1[zz[0][0]] == 1 :$

$print("-------------------------------------")$

$print("IdentifiedasLandSlide")$

$print("-------------------------------------")$

$importcv2$

$importnumpyasnp$

$importos$

$importmatplotlib.pyplotasplt$

$fromtkinterimportTk$

$fromtkinter.filedialogimportaskopenfilename$

$defdraw_{b}ounding_{b}oxes(image_{p}ath, annotation_{p}ath):$

$image = cv2.imread(image_{p}ath)$

$image = cv2.cvtColor(image, cv2.COLOR_{B}GR2RGB)$

$annotation = cv2.imread(annotation_{p}ath, cv2.IMREAD_{G}RAYSCALE)$

$contours,_{=} cv2.findContours(annotation, cv2.RETR_{E}XTERNAL, cv2.CHAIN_{A}PPROX_{S}IMPLE$

$forcontourincontours:$

$x, y, w, h = cv2.boundingRect(contour)$

$cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)$

$plt.imshow(image)$

$plt.axis('off')$

$plt.show()$

$defselect_{f}ile(title = "Selectafile", filetypes = (("PNGfiles", "*.png"),)):$

$Tk().withdraw()$

$file_{p}ath = askopenfilename(title = title, filetypes = filetypes)$

$returnfile_{p}ath$

$image_{p}ath = filename$

$image_{f}ilename = os.path.splitext(os.path.basename(image_{p}ath))[0]$

$annotation_{p}ath = os.path.join("masks", f"image_{f}ilename.png")$

$ifos.path.exists(image_{p}ath)andos.path.exists(annotation_{p}ath):$

$draw_{b}ounding_{b}oxes(image_{p}ath, annotation_{p}ath)$

$else:$

$print("Imageorannotationfilenotfound.Pleasecheckthefilenamesandpaths.")$

$eliflabels1[zz[0][0]] == 2:$

$print("--")$

$print("IdentifiedasNonSlide")$

$print("--")$

Bibliography

[1] W. Zhang et al. 'LS-YOLO: A Novel Model for Detecting Multiscale Landslides With Remote Sensing Images.' IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2024.

[2] Chandra N, Vaidya H. 'Automated detection of landslide events from multi-source remote sensing imagery: Performance evaluation and analysis of YOLO algorithms.' J Earth Syst Sci 133, 2024.

[3] Han Z, Fang Z, Li Y and Fu B. 'A novel Dynahead-Yolo neural network for the detection of landslides with variable proportions using remote sensing images'. Front. Earth Sci. 10:1077153, 2023.

[4] Hou H, Chen M, Tie Y, Li, W. 'A novel landslide identification method for multi-scale and complex background region based on multi-model fusion: YOLO+U-Net. Landslides' 2023.

[5] Wang H, Liu J, Zeng S et al. 'A novel landslide identification method for multi-scale and complex background region based on multi-model fusion: YOLO+U-Net. Landslides.' 2024.

[6] Han Z, Fu B, Fang Z, Li Y, Li J, Jiang N, Chen G. 'Dynahead-YOLO-Otsu: an efficient DCNN-based landslide semantic segmentation method using remote sensing images. Geomatics, Natural Hazards and Risk.' 2024.

[7] Guo H, Yi B, Yao Q, Gao P, Li H, Sun J, Zhong C. 'Identification of landslides in mountainous area with the combination of SBAS-InSAR and Yolo Model.' 2022.

[8] Mo P, Li D, Liu M, Jia J, Chen X. 'A lightweight and partitioned CNN algorithm for multi-landslide detection in remote sensing images.' Fellows IEEE : IEEE Transactions on Knowledge and Data Engineering, VOL. 26, NO. 1, 2014.

[9] Yingchi Mao, Wei Min, Jiulong Wang, Bicong Jia, Qing Jie. 'Dynamic Mirror Based Real-Time Query ContentionSolution for Support Big Real-Time Data Analysis.' College of Computer and Information Engineering, Hohai University, Nanjing,

China. : 2nd International Conference on Information Technology and Electronic Commerce (ICITEC) 2014.

[10] Mo P, Li D, Liu M, Jia J, Chen X. 'A lightweight and partitioned CNN algorithm for multi-landslide detection in remote sensing images.' Fellows IEEE : IEEE Transactions on Knowledge and Data Engineering, VOL. 26, NO. 1, 2014.