ANIMAL DETECTION FOR ROAD SAFETY USING DEEP LEARNING

A PROJECT REPORT

Submitted by

VAISHNAV P (TKM21MCA-2039)

to

The APJ Abdul Kalam Technological University

In partial fulfillment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATION

Thangal Kunju Musaliar College of Engineering
Kerala

DEPARTMENT OF COMPUTER APPLICATION

MAY 2023

DECLARATION

I undersigned hereby declare that the project report on **ANIMAL DETECTION FOR ROAD SAFETY USING DEEP LEARNING**, submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Application of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of Prof. Jasmin M R. This submission represents my ideas in my own words and where ideas or words of others have been included,I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not previously served as the basis for the award of any degree, diploma, or similar title by any other University.

Kollam

16-05-2023

VAISHNAV P

DEPT. OF COMPUTER APPLICATION

TKM COLLEGE OF ENGINEERING, KOLLAM

2021 - 23

CERTIFICATE

This is to certify that the report entitled **ANIMAL DETECTION FOR ROAD SAFETY USING DEEP LEARNING** submitted by **VAISHNAV P** (TKM21MCA-2039) to the APJ Abdul Kalam Technological University in partial fulfillment of the Masters degree in Computer Application is a bonafide record of the project work carried out by him under our guidance and supervision. This report, in any form, has not been submitted to any other University or Institute for any reason.

Internal Supervisor Head of the Department External Examiner

ACKNOWLEDGEMENT

First and foremost, I thank GOD almighty and my parents for the success of this project. I owe sincere gratitude and heart full thanks to everyone who shared their precious time and knowledge for the successful completion of my project.

I am extremely grateful to **Dr. Fousia M Shamsudeen**, Head of the Department, Department of Computer Application, for providing me with the best facilities.

I would like to express my heartfelt thanks to my coordinator **Prof. Vaheetha Salam** Department of Computer Application, for her expert guidance, co-operation and immense encouragement.

I would like to thank my project guide **Prof. Jasmin M R**, Department of Computer Application, who motivated me throughout the project.

I owe my thanks to my advisor **Prof. Natheera Beevi M**, Department of Computer Application, for her continuous guidance and support.

I profusely thank all other faculty members in the department and all other members of TKM College of Engineering, for their guidance and inspiration throughout my course of study.

I owe my thanks to my friends and all others who have directly or indirectly helped me in the successful completion of this project.

VAISHNAV P

ABSTRACT

ANIMAL DETECTION FOR ROAD SAFETY USING DEEP LEARNING project aims to develop a system that can detect animals on roads to improve road safety for both drivers and animals. Animal-vehicle collisions are a major cause of road accidents worldwide and can lead to injuries, fatalities, and significant economic losses. The project proposed a deep learning-based approach for animal detection in real-time.

The system used a combination of image processing techniques and machine learning algorithms to accurately detect and classify animals in different weather conditions and lighting conditions. It also takes into account the behaviour of different animal species and adjust its detection algorithms accordingly.

The outcome of this project is a deep learning-based animal detection system that can be integrated into existing road safety systems to improve the safety of drivers and animals. The system has the potential to significantly reduce the number of animal-related accidents on our roads and protect both drivers and animals.

Contents

List of Figures **iii**

1 Introduction **1**

1.1 Existing System 2

1.2 Proposed System 2

1.3 Problem Statement 3

1.4 Objectives 4

1.4.1 Develop an advanced animal detection system 4

1.4.2 Enhance road safety 4

1.4.3 Improve animal welfare 4

1.4.4 Increase automation in animal detection 4

2 Literature Survey **5**

2.1 Purpose of the Literature Review 5

2.2 Related Works 6

2.2.1 Animal Detection Systems 6

2.2.2 YOLO Algorithm 9

2.2.3 Flask Framework 12

3 Methodology **14**

3.1 Dataset 14

3.1.1 Google Open Images V6+ Dataset 14

3.2 Data Preparation 15

3.3 Model Training and Testing 16

3.3.1 Clone YOLO Github Project 16

3.3.2 Get Weights 16

3.3.3 Data Preparation 16

3.3.4 Model Training 16

3.3.5 Testing the Model 17

3.4 YOLO Algorithm 17

3.5 Software Requirements and Specifications 18
3.5.1 Google Colab 18
3.5.2 Python 19
3.5.3 PyCharm 19
3.5.4 Flask 20
3.5.5 Hardware and experimental environment 20

4 Result and Discussion **22**
4.1 Testing Methods 23
4.1.1 Unit Testing 23
4.1.2 Integration Testing 23
4.1.3 System Testing 23
4.1.4 Performance Testing 23
4.2 Output Screens and Results 23

5 Conclusion **27**
5.1 Future Enhancement 27

References **29**

Appendix **31**

List of Figures

3.1 Google Open Images V6+ Dataset 15
3.2 Data preparation 15
3.3 System Architecture 16
3.4 Architecture of YOLO 17
3.5 The Model 18

4.1 Home Page 24
4.2 Result Page 24
4.3 Precision, Recall and mAP 25
4.4 PR Curve 25
4.5 F1 curve 26

A.1 Home Page 31
A.2 Result 1 31
A.3 Result 2 32
A.4 Result 3 32
A.5 Result 4 33
A.6 Result 5 33
A.7 Result 6(Low light) 34
A.8 Dataset images 34
A.9 Image labels 35

Chapter 1

Introduction

The rapid increase in road construction between forest and wild animal dense areas made it challenging for the driver to ride. Various animals cross the road. Sometimes the driver may not spot the animal, which sometimes leads to a fatal accident.

With the help of machine learning and a deep learning algorithm, large set of animal images can be added and let the machine predict if there is any obstacle in between also even if there is an animal in the sight of the vehicle.

ANIMAL DETECTION FOR ROAD SAFETY USING DEEP LEARNING aims to develop an innovative solution to address this pressing issue by harnessing the power of computer vision and deep learning. The project focuses on utilizing the YOLO (You Only Look Once) algorithm, a state-of-the-art object detection algorithm known for its real-time performance and accuracy. YOLO can detect multiple objects in an image and provide precise bounding box coordinates around them.

The project's main objective is to enhance road safety by providing an efficient and reliable animal detection system. This system can be integrated into vehicles, surveillance cameras, or roadside monitoring systems to alert drivers or trigger automated responses when animals are detected near roadways. Additionally, the system can assist in providing data for wildlife management initiatives, helping authorities analyze animal behavior patterns, migration routes, and hotspot areas for implementing targeted measures.

The YOLO algorithm's real-time processing capabilities make it an ideal choice for this project, as it can provide instant feedback on animal presence, enabling immediate action to prevent accidents. The trained YOLO model can then be integrated into a software application or embedded system, ensuring its practical usability and scalability.

1.1 Existing System

The existing system for animal detection in the context of road safety often relies on manual observation and human intervention. Drivers and pedestrians are expected to visually detect animals near roadways and take appropriate actions to avoid collisions. However, this approach is prone to errors and limitations. Human perception may be compromised due to factors such as poor visibility, driver distraction, or animals appearing suddenly on the road. These limitations pose a significant risk to road safety and call for a more advanced and reliable system.

Some existing systems utilize static cameras or sensors installed along roadsides to monitor animal activity. These systems detect animal movement and trigger alarms or warning signs to alert drivers. While this approach provides some level of automation, it often suffers from limited coverage and relies on pre-defined detection zones. Moreover, it may not effectively handle various animal species or rapidly changing conditions. Therefore, a more sophisticated system that can accurately identify and track animals in real-time is required to enhance road safety.

The existing systems also face challenges in terms of scalability and cost-effectiveness. Deploying and maintaining a large number of static sensors or cameras across extensive road networks can be expensive and logistically complex. Additionally, these systems may require significant human intervention for monitoring and maintenance, limiting their overall effectiveness.

Some of the existing systems are trained on relatively smaller datasets, thus failing to detect some classes of wild animals.

1.2 Proposed System

The proposed system, ”Animal Detection for Road Safety using YOLO algorithm,” aims to revolutionize the existing approach to animal detection by leveraging the power of computer vision and deep learning. The system utilized the YOLO (You Only Look Once) algorithm, known for its real-time object detection capabilities, to accurately identify and locate animals near roadways in a fast and efficient manner.

To develop an accurate and robust animal detection system, a comprehensive dataset of

animal images consisting almost 80 categories of wild animals was collected and annotated with bounding boxes. The dataset encompass various animal species, different lighting conditions, and diverse road environments to ensure the model's generalization ability. The YOLO algorithm was then be trained using this annotated dataset, employing deep learning techniques to optimize its performance and accuracy.

Once trained, the YOLO model was integrated into a software application or embedded system, enabling real-time animal detection. The system can be modified to provide instant alerts or warnings to drivers, either through visual cues or auditory signals, when animals are detected near the roadways. This early warning mechanism will allow drivers to take immediate precautionary measures, such as slowing down, changing lanes, or stopping, to avoid potential collisions with animals.

1.3 Problem Statement

The drawbacks of currently existing models are:

1. Models are trained on relatively small datasets, which will affect the accuracy of the results.

2. Many existing systems rely on static cameras or sensors placed in specific locations along roadways. This limited coverage makes it difficult to detect animals in areas without surveillance, leaving those regions vulnerable to accidents.

3. Deploying and maintaining a large number of static sensors or cameras across extensive road networks can be expensive and resource-intensive. Additionally, these systems may require regular maintenance and calibration, adding to the overall cost and complexity.

4. Existing systems may struggle to adapt to changing environmental conditions, such as variations in lighting or weather. This lack of adaptability can lead to false positives or false negatives in animal detection, reducing the overall reliability and effectiveness of the system.

1.4 Objectives

1.4.1 Develop an advanced animal detection system

The primary objective of the project is to design and develop a state-of-the-art animal detection system that utilizes the YOLO algorithm. The system should be capable of accurately detecting and identifying animals near roadways in real-time.

1.4.2 Enhance road safety

The project aims to contribute to road safety by preventing animal-related accidents. The developed system can provide early warnings or alerts to drivers when animals are detected near the roadways, enabling them to take immediate precautionary measures and avoid potential collisions.

1.4.3 Improve animal welfare

Another objective is to protect animal welfare by minimizing the risk of animal-vehicle collisions. The system enable drivers to become more aware of animals near roads and adjust their driving behavior accordingly, reducing harm to animals and their habitats.

1.4.4 Increase automation in animal detection

The project seeks to reduce reliance on manual observation and intervention by introducing an automated animal detection system. This objective aims to enhance the efficiency and accuracy of animal detection, reducing human errors and potential delays in identifying animals near roadways.

Chapter 2

Literature Survey

A literature survey, also known as a literature review, focuses on examining academic texts relevant to a certain subject. By examining the literature, it provides a complete review of the state of the subject and enables you to identify relevant theories, approaches, and knowledge gaps. When doing a literature review from an audit perspective, the evaluation of the appropriate literature should be your top priority. This process includes information published in a specific field of study as well as sporadically information produced within a specific time period. The literature review is an essential research tool that is frequently utilised as a starting point for further investigation of a certain subject area. A literature review can point forth areas that need further study, important theories and concepts, and knowledge gaps in the domain. By looking at a variety of sources, a literature study can provide a more thorough understanding of a particular topic or situation. A well-written literature review can also boost the author's authority and credibility because it demonstrates that the author is informed about the most recent studies and arguments in the field. Sometimes a literature review will include a meta-analysis, which comprises analysing the findings of numerous studies to find recurring patterns or trends. It is important to keep in mind that a literature review differs from a research paper or an argumentative essay in that it is an in-depth examination of the pertinent literature and research.

2.1 Purpose of the Literature Review

1. A literature review presents an overview and analysis of the corpus of information on a certain topic.

2. It aims to provide readers with simple access to research on a specific issue by choosing excellent articles or studies that are pertinent, significant, important, and valid, and then consolidating them into one report.

3. A literature review can give readers a more in-depth grasp of a certain topic or issue by

looking at a variety of sources.

4. The procedures and strategies used by other researchers are constructively examined.

5. As it indicates the author's acquaintance with the most recent studies and arguments in the subject, a well-written literature review can help to establish the author's credibility and authority.

6. A literature review can stand alone or be a component of a bigger research undertaking, like a thesis, dissertation, or research article.

2.2 Related Works

2.2.1 Animal Detection Systems

Accidents caused by animals unexpectedly crossing the road remain a major cause of traffic fatalities today. Due to the dark and crowded nature of the roads near the forest, vehicle drivers are unable to clearly see the animals. The model suggested in this paper effectively detects the animals and alerts the driver. With the use of a sizable open-source dataset, they are sorting the creatures using machine learning—a deep learning technique. The model forecast the object for each image frame it receives from the Live Camera using convolution neural networks. When an object is identified by the machine as an animal, the system issues a 3-second alert to warn the driver of the animal's approach. Convolution neural networks are one of the numerous subsets of machine learning techniques. A CNN is mostly used for image processing, natural language processing, and data interpretation. It consists of many hidden layers. The complexity and user determine how many layers there should be. Apart from the convolution operation, the network also performs pooling and flattening operations in-order to reduce the pixel count and convert the data to one-dimensional format. For each image frame received from the Live Camera, the images are sent into the CNN model for every frame. The system sends a 3-second alert if the machine detects an object as an animal to warn the driver of an approaching animal. The paper proposed a model that provides a low-cost, efficient system that can be really helpful to the society. The authors found the technology described in this paper as a deal-breaker since, when the camera is mounted above the dashboard and facing the road, the model provides 91% accuracy under various lighting and position. When compared to other algorithms that can

be used for the same purpose, this model requires less computation. Also the authors added that the efficiency of this model can be improved if the GPU power is upgraded and camera quality is improved. The model's efficiency depends on how slow the vehicle moves in some situations.[1]

The manual detection of animals with their names is a very tedious task. To overcome this challenge, this research work had developed a YOLOV3 model to identify the animal present in the image given by user. The algorithm used in YOLOV3 model is darknet, which has a pretrained dataset. YOLO (You Only Look Once) it is an object detector which uses the feature of deep convolutional neural network. YOLO is a fully convolutional network. YOLO consists of 75 convolutional layers with skip connections. When an input image has been considered, it goes through the feature extraction so it is divided into different scales. Its output is generated by applying a 1x1 kernel on feature map. These features are going into detector to bounding boxes information. The feature extractor for YOLO V3 model is Darknet-53. In the previous version of YOLO model there're only 19 layers so the network extends the layers from 19 to 53 layers for YOLO V3 model. For detection of input image 53 more layers are added to the YOLO V3 model. YOLO V3 is designed to be a multi-scaled detector. In this paper, when the authors were able to get right predictions, but in some cases wrong predictions occured and for some images, predictions were not happening. The authors also suggested that the model can be trained using custom dataset for own number of classes and number of images in each class to get better results.[2]

In this paper, the authors consider the animal object detection and segmentation from wildlife monitoring videos captured by motion-triggered cameras, called camera-traps. For these types of videos, existing approaches often suffer from low detection rates due to low contrast between the foreground animals and the cluttered background, as well as high false positive rates due to the dynamic background. In this work, the authors proposed to develop a new and efficient animal object proposal method using IEC which exploit animal motion and temporal correlation in the camera-trap image sequence. In this work, they use the Camera-trap and popular CDnet dataset for evaluation purposes. They also established the Camera-Trap dataset for performance evaluation of animal detection from highly cluttered natural scenes. In this paper, the authors have successfully developed an accurate method for animal object detection from highly cluttered natural scenes captured by motion-triggered cameras, called camera traps.[3]

The collision of an animal with the vehicle on the highway is one such big issue which leads to such road accidents. In this paper, a simple and a low-cost approach for automatic animal detection on highways for preventing animal-vehicle collision using computer vision techniques are proposed. A method for finding the distance of the animal in real-world units from the camera mounted vehicle is also proposed. The proposed system is trained on more than 2200 images consisting of positive and negatives images and tested on various video clips of animals on highways with varying vehicle speed. In this paper, S. Sharma and D. Shah discussed the necessity of automatic animal detection system and the algorithm for animal detection based on HOG and cascade classifier. The algorithm can detect an animal in different conditions on highways. The proposed system achieved an accuracy of almost 82.5% regarding animal (cow) detection. Authors discussed some of the limitations of the system such as, The proposed system can detect animal up to a distance of 20 meters only when a vehicle is stationary. The system can prevent collision of the vehicle with the animal when driving at a speed in between 30 to 35 kmph. Beyond this speed, though animal gets detected, time is not sufficient to prevent animal-vehicle collision. No effort has been made to detect animals during the night, which is expected to be done in the future scope of study and research.[4]

In this paper the authors addressed the task of animal detection from sub-decimeter resolution images acquired by low-cost Unmanned Aerial Vehicles. This task is of particular interest to live stock conservation, where accurate and cost-effective solutions to animal monitoring would lead to targeted counteractions to poaching. A pipeline performing animal (object) detection based on Convolutional Neural Networks (CNNs) is proposed in this paper. In this paper, the authors proposed an animal detection system, able to efficiently operate on sub-decimeter images acquired by UAVs. Experiments have shown that the proposed method to be far more precise in predicting the location of animals in images compared to the state-of-the-art Fast R-CNN model. The model was able to predict sufficiently accurate bounding boxes, while producing significantly less false positives.[5]

The significant rise in the rate of animal-vehicle collision on Indian roads in recent years underline the imperative need for effective technological interventions that help mitigate animal-vehicle collisions in real-time. This paper proposed a deep learning approach for the implementation of the Real-Time Animal Vehicle Collision Mitigation System. A comparative study of three deep learning based object detection frameworks: (1) Mask R-CNN; (2) You Only Look Once (YOLOv3); and (3) MobileNet-SSD are done in this paper. These object

detectors are implemented, evaluated and compared with each other on the basis of their mean average precision (mAP) and processing times. The results show that the MobileNet-SSD architecture gives the best trade-off between the mAP and processing time and is best suited for Real-Time Animal Detection among the three approaches considered. Using the trained model of MobileNet-SSD based object detection framework, an Animal Vehicle Collision Mitigation System is implemented and shown to perform well on several real-world test scenarios. A comparative study of three different Object Detection frameworks is made and the MobileNet-SSD Object Detection Framework is found to be best suited for Real-Time Animal-Vehicle Collision Mitigation. The single stage architecture of the SSD Network facilitates detection in the least time (0.2 s on the test system) and generally gives good results, whereas the other two frameworks are seen to fail in performing effective real time object detection on the CPU. However, the real-time detection hampers the ability of the MobileNet-SSD framework to detect small objects in the frame which decreases the detection accuracy of the network.[6]

2.2.2 YOLO Algorithm

The development of intelligent animal detection systems to improve road safety has drawn more study attention in recent years. With the increasing incidents of wildlife-vehicle collisions and the potential risks they pose to both human safety and animal conservation, researchers have been exploring various approaches to mitigate these risks. Animal detection systems utilizing computer vision techniques, such as the YOLO (You Only Look Once) algorithm, have emerged as a promising solution. The YOLO (You Only Look Once) algorithm is a popular and influential object detection algorithm that has gained significant attention in the field of computer vision.In the paper, the authors proposed a model that is simple to construct and can be trained directly on full images. Unlike classifier-based approaches, YOLO is trained on a loss function that directly corresponds to detection performance and the entire model is trained jointly. The YOLO design enables end-to-end training and real-time speeds while maintaining high average precision. The system divides the input image into an SxS grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts bounding boxes and confidence scores for those boxes. These confidence scores reflect how confident the model is that the box contains an object and also how accurate it thinks the box is that it predicts. The model is pretrained on ImageNet 1000-class competition dataset. The network was trained for approximately a week and achieved a single crop top-5 accuracy

of 88% on the ImageNet 2012 validation set, comparable to the GoogLeNet models in Caffe's Model Zoo. The paper also describes some of the limitations of the model, YOLO imposes strong spatial constraints on bounding box predictions since each grid cell only predicts two boxes and can only have one class. This spatial constraint limits the number of nearby objects that our model can predict. Model struggles with small objects that appear in groups, such as flocks of birds. Since the model learns to predict bounding boxes from data, it struggles to generalize to objects in new or unusual aspect ratios or configurations. The authors also did a comparison between YOLO and other object detection models like R-CNN. According to the authors, YOLO can be used to rescore Fast R-CNN detections and reduce the errors from background false positives, giving a significant performance boost[7].

YOLOv7 surpasses all known object detectors in both speed and accuracy in the range from 5 FPS to 160 FPS and has the highest accuracy 56.8% AP among all known real-time object detectors with 30 FPS or higher on GPU V100. In this paper, the authors proposed a new version of YOLO - YOLO v7. The contributions of this paper are summarized as follows: (1) Authors designed several trainable bag-of-freebies methods, so that real-time object detection can greatly improve the detection accuracy without increasing the inference cost; (2) for the evolution of object detection methods, they found two new issues, namely how re-parameterized module replaces original module, and how dynamic label assignment strategy deals with assignment to different output layers. In addition, they also proposed methods to address the difficulties arising from these issues; (3) they proposed "extend" and "compound scaling" methods for the real-time object detector that can effectively utilize parameters and computation; and (4) the method they proposed can effectively reduce about 40% parameters and 50% computation of state-of-the-art real-time object detector, and has faster inference speed and higher detection accuracy. In this paper, the authors had developed the YOLOv7 series of object detection systems, which received the state-of-the-art results.[8]

In this paper, the authors took the traffic signs the research object and used the YOLO neural network to analyze. The experiment was based on the Darknet-53 network structure. The contributions made in the paper are : 1) Established a new image degradation model and used different degraded images as test sets. Compared the effect of different degraded images on the standard model. 2) Modified the source network and performed different degradation processes for the training set. Also compared the accuracy of test sets on different models. Then, performed more complex degradation processes on training sets and obtained a more

generalized detection network. Afterwards, compared it with test performances. 3) Based on the above, optimized the object detection method. To conclude, the generalization ability of the model has been enhanced, and the accuracy of object detection has been improved. In this paper, the authors proposed, using the YOLO network model for object detection and states why YOLO is efficient compared to other object detection algorithms.[9]

Object detection is one of the predominant and challenging problems in computer vision. Over the decade, with the expeditious evolution of deep learning, researchers have extensively experimented and contributed in the performance enhancement of object detection and related tasks such as object classification, localization, and segmentation using underlying deep models. Broadly, object detectors are classified into two categories viz. two stage and single stage object detectors. Two stage detectors mainly focus on selective region proposals strategy via complex architecture; however, single stage detectors focus on all the spatial region proposals for the possible detection of objects via relatively simpler architecture in one shot. Performance of any object detector is evaluated through detection accuracy and inference time. Generally, the detection accuracy of two stage detectors outperforms single stage object detectors. However, the inference time of single stage detectors is better compared to its counterparts.. With the advent of YOLO (You Only Look Once) and its architectural successors, the detection accuracy have improved significantly and sometime it is better than two stage detectors. s. In this paper, the authors explored two stage object detectors viz. RCNN, FastRCNN, and Faster-RCNN along with their important applications. Single stage object detectors especially YOLOs, their architectural advancements, underlying pretrained CNN architectures, and loss function in detail are reviewed in this paper. The paper presented different aspects and optimizations carried out in the successive versions of YOLOs along with all the underlying concepts. Moreover, it presented the challenges and motivations behind the specific review of single stage object detectors. YOLOs are performing significantly better in comparison with their counterpart two stage object detectors in terms of detection accuracy and inference time.[10]

Object detection has seen many changes in algorithms to improve performance both on speed and accuracy. By the continuous effort of so many researchers, deep learning algorithms are growing rapidly with an improved object detection performance. Various popular applications like pedestrian detection, medical imaging, robotics, self-driving cars, face detection, etc. reduces the efforts of humans in many areas. Due to the vast field

and various state-of-the-art algorithms, it is a tedious task to cover all at once. This paper presented the fundamental overview of object detection methods by including two classes of object detectors. In two stage detector covered algorithms are RCNN, Fast RCNN, and Faster RCNN, whereas in one stage detector YOLO v1, v2, v3, and SSD are covered. Two stage detectors focus more on accuracy, whereas the primary concern of one stage detectors is speed. There are two types of object detection algorithms. Object detection algorithms using region proposal includes RCNN, Fast RCNN, and Faster RCNN, etc. These techniques create region proposal networks (RPN), and then the region proposals are divided into categories afterward. On the other side, object detection algorithms using regression includes SSD and YOLO, etc. These methods also generate region proposal networks (RPN) but divide these region proposals into categories at the moment of generation. All of the procedures mentioned above have significant accomplishments in object localization and recognition. YOLO consolidates labels in diverse datasets to form a tree-like arrangement, but the merged labels are not reciprocally exclusive. YOLO9000 enhances YOLO to recognize targets above 9000 categories employing hierarchical arrangement. Whereas YOLOv3 uses multilabel classification, it replaces the approach of estimating the cost function and further exhibits meaningful improvement in distinguishing small targets. This paper presents the comparison of various algorithms to identify and localize objects based on accuracy, time, and parameter values with varying sizes of the input image. The results of the comparsions done in this paper show that YOLO v3-Tiny increases the speed of object detection while ensures the accuracy of the result.[11]

2.2.3 Flask Framework

The Flask is a framework that uses Python language with easy to understand code writing. But the Flask framework still doesn't use the MVC method, so files and codes are not regular. The purpose of this paper was to design a MVC for a framework that uses the Python programming language. This system has a generator that can make MVC folder structure easily and quickly, this system is also equipped with the Bootstrap framework, and this system is open source. The results showed that the presence of MVC on the flask framework could make users easier in creating new projects and have faster fully load time. MVC is a method for making applications concise and fast, but unfortunately, in framework Flask it still doesn't use the MVC method. In this study, aim was to provide easy website creation using the MVC method in framework flask with python language. The system has been equipped with a project generator and is also

equipped with a bootstrap framework which will make it easier to design the appearance of the website. System helps developers improve the speed and quality of work, and provides a framework and work platform that is good for novice or experienced users.[12]

Python is a general purpose, high level programming language that focuses on the code readability, for web development lines of code will be fewer than other languages. Flask is a micro framework of Python which provides the basic functionality of web framework and allows more plug-ins to be added so the functionality and feature set can be extended to a new level. Flask is called as micro framework of Python because it makes the core functionality simple but extensible in terms of development. It can also be used to save time building web applications. Flask uses Jinja Template Engine and the Werkzeug WSGI Toolkit. Flask structure is categories into two parts “Static files Template files”, template file have all the Jinja templates including Html pages, where as static file have all static codes needed for website such as CSS code, JavaScript code and Image files. Jinja2 is a library for python that is designed to be flexible, fast and secure. Jinja2 is a modern and designer-friendly templating language for python, modeled after Django’s templates. It is fast, widely used and secure with the optional sandboxed template execution environment. Jinja2 is more readable because its syntax is easy to visually distinguish from HTML code. This paper described efficient way of implementing web development using python and flask. The paper also concluded that Python can be used for making web more powerful, fast and efficient with the help of Flask Template Engine.[13]

Chapter 3

Methodology

Road accidents involving animals pose a significant risk to both drivers and wildlife. To address this issue, this project focuses on developing an animal detection system using the state-of-the-art YOLO algorithm. This algorithm is chosen as it has shown promising performance in real-time object detection tasks.

3.1 Dataset

3.1.1 Google Open Images V6+ Dataset

The Google Open Images v6+ dataset is a large-scale and publicly available dataset widely utilized in the field of computer vision. It consists of millions of labeled images covering a diverse range of object categories. The dataset covers a wide array of object categories, including animals, vehicles, household items, and more. One distinctive feature of the Google Open Images v6+ dataset is its extensive annotations. Each image in the dataset is annotated with bounding boxes, providing precise object localization information. Moreover, the dataset includes object-level labels, which offer additional information about the detected objects and their corresponding categories.

In this project, animals dataset extracted from Google Open Images V6+ is used. This subset comprises a vast collection of labeled images featuring a diverse range of animals, including mammals, birds, reptiles, and more. The annotations in the dataset offer a high level of detail, enabling accurate identification and localization of animals within the images.

The dataset consists of 80 classes of wild animals. The dataset is available in Kaggle and is sub-divided into train and test directories, which eases the step of data pre-processing. Along with each class of animals, labels are provided.

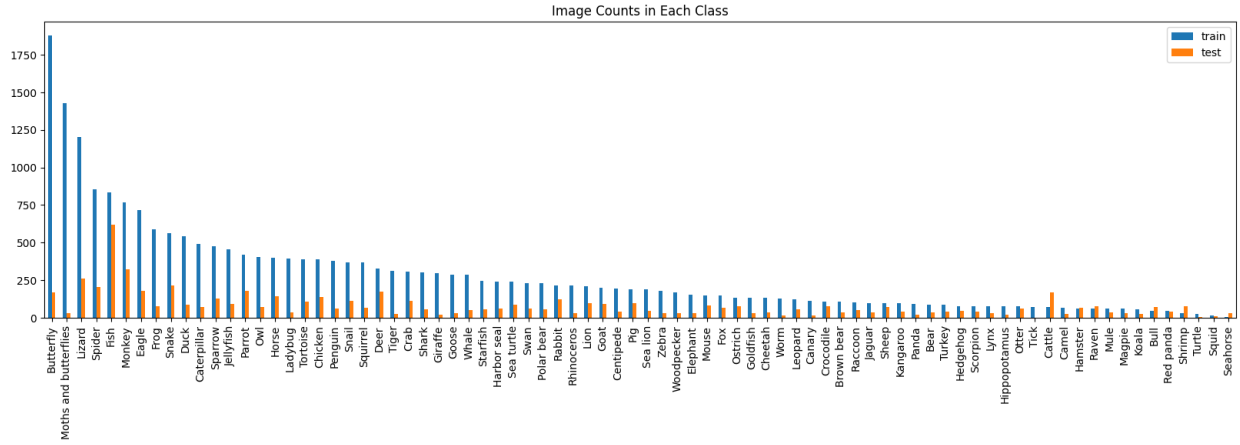

Figure 3.1: Google Open Images V6+ Dataset

3.2 Data Preparation

Since the data is divided into train and test data and labels are available for every image, a lot of data preparation steps can be reduced. However, the data needs to be prepared for working with YOLO algorithm.

The images are resized to the size 640x640.

The data need to pre-processed into the YOLO format required, where the coordinates for the center of each image in the dataset need to be supplied as depicted in figure 3.2.

For that purpose, data needs be to pre-processed, the label files associated with the animal images should be converted into a format suitable for training with the YOLO algorithm.

Figure 3.2: Data preparation

3.3 Model Training and Testing

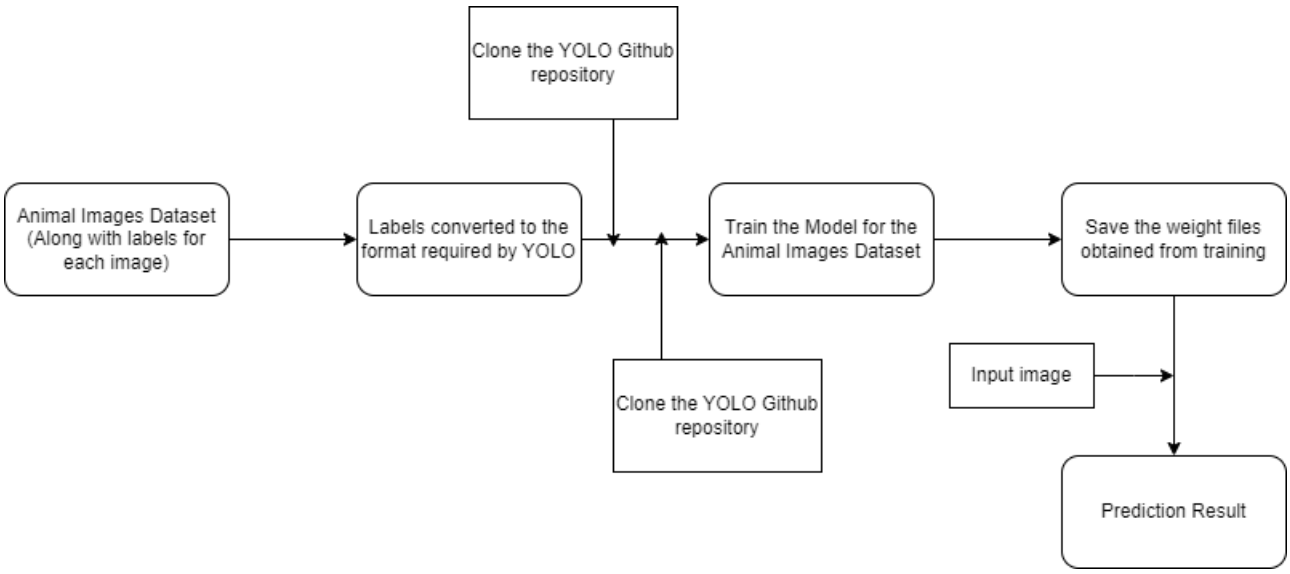

Figure 3.3: System Architecture

3.3.1 Clone YOLO Github Project

Implementation of YOLO v7 is available as open-source in Github. In-order to make use of the YOLO algorithm, the github repository should be cloned.

3.3.2 Get Weights

Download the pre-trained weights from the Github repository. YOLO algorithm have their weights pre-trained with MS-COCO (Microsoft Common Objects in Context) dataset and thus making it easy to use for object detection purposes by transfer learning.

3.3.3 Data Preparation

Convert the labels to a format supported by YOLO.

3.3.4 Model Training

Next, the model is trained by passing arguments like number of epochs, pre-trained weights and other configurations. Once the training has been completed, the resulting adapted weight-files will be saved.

3.3.5 Testing the Model

Testing is performed by using the trained moedl weight-file, the source(image) that needs to be analysed.

3.4 YOLO Algorithm

YOLO is an intelligent convolutional neural network (CNN) for real-time object detection. The system uses a single neural network to process the entire image before segmenting it into sections and predicting bounding boxes and probabilities for each one. These bounding boxes are weighted based on the probability that were predicted.

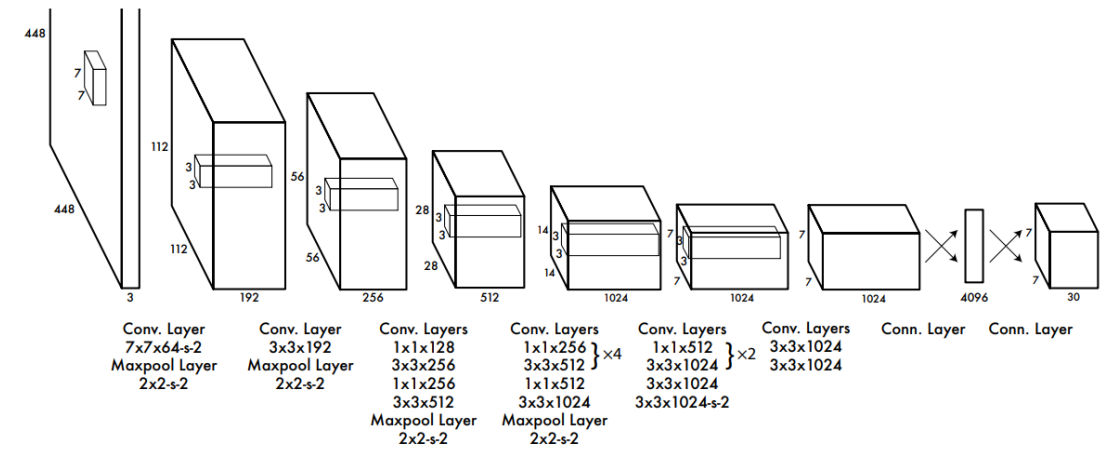

Figure 3.4: Architecture of YOLO

The algorithm works in a way that an image is divided into grids. Then for each grid bounding boxes and confidence levels are predicted as well as class probabilities. Finally the results are aggregated based on the box confidences and class probabilities.

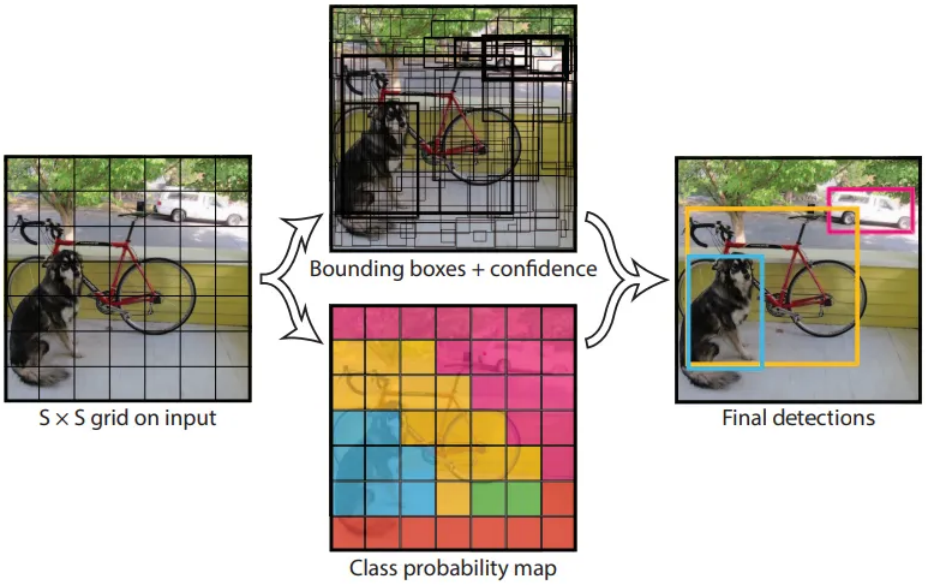

Figure 3.5: The Model

The system models detection as a regression problem. It divides the image into an S × S grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an S × S × (B * 5 + C) tensor.

3.5 Software Requirements and Specifications

The software requirements for the project are:

1. Google Colab
2. Python
3. PyCharm
4. Flask

3.5.1 Google Colab

Google Colab is an essential software requirement for the project, offering a cloud-based development environment that facilitates collaborative coding, data analysis, and model training. It provides a convenient platform for writing and executing Python code directly in the browser without requiring any local installation. Google Colab's integration with Google

Drive allows for seamless data storage and retrieval, ensuring easy access to project files and datasets. Its powerful hardware acceleration capabilities, such as GPU and TPU support, enable faster computation and training of deep learning models. With its built-in libraries and frameworks like TensorFlow and PyTorch, Google Colab simplifies the setup process and provides a ready-to-use environment for machine learning tasks. Moreover, Colab notebooks offer a combination of code cells, markdown cells, and rich text elements, allowing for the creation of comprehensive project documentation with embedded code, visualizations, and explanations. Collaboration is made easy with the ability to share and work on notebooks simultaneously with team members. Overall, Google Colab serves as an efficient and accessible software tool, enabling seamless collaboration, resource-efficient computation, and streamlined development workflow for the project.

3.5.2 Python

Python is a fundamental software requirement for the project due to its versatility, extensive libraries, and strong community support. Python's rich ecosystem of libraries and frameworks, including TensorFlow, PyTorch, and Keras, provides powerful tools for machine learning and computer vision tasks. Its simple and expressive syntax enables rapid prototyping and experimentation, facilitating efficient development iterations. Python's extensive standard library and third-party packages allow for seamless data preprocessing, image manipulation, and statistical analysis, essential for tasks such as dataset preprocessing and model evaluation. Additionally, Python's platform independence ensures compatibility across different operating systems, fostering collaboration and deployment flexibility. The language's popularity also ensures comprehensive documentation, online resources, and a thriving community for learning and troubleshooting. Overall, Python's strengths make it an indispensable software requirement, empowering us to effectively implement the YOLO algorithm and build a robust animal detection system.Version 3.9.13 of python is used here.

3.5.3 PyCharm

PyCharm is a powerful integrated development environment (IDE) specifically designed for Python programming. PyCharm offers a comprehensive set of features and tools that facilitate efficient development, debugging, and deployment of Python applications. With its intuitive user interface and intelligent code editor, PyCharm enhances productivity by providing features

like code completion, code navigation, and error detection. It also integrates seamlessly with version control systems, allowing for collaborative development and easy project management. PyCharm's built-in debugger and testing tools enable thorough testing and debugging of the project code, ensuring its robustness and reliability. Additionally, PyCharm offers support for various Python frameworks and libraries, allowing for seamless integration with external resources. Its advanced features, such as code refactoring, code inspections, and documentation generation, contribute to code quality and maintainability. Overall, PyCharm serves as an essential software requirement, providing a streamlined and efficient development environment for the successful implementation of the animal detection system using the YOLO algorithm.

3.5.4 Flask

Flask is a popular and lightweight web framework. Flask provides a solid foundation for building web applications with Python, making it an ideal choice for developing the system's user interface and backend services. With its simplicity and flexibility, Flask enables rapid development and easy integration of web functionalities into the project. It follows the Model-View-Controller (MVC) architectural pattern, allowing for clear separation of concerns and efficient code organization. Flask's extensive ecosystem of extensions provides additional features and capabilities that can be leveraged for authentication, data storage, and other web-related tasks. The microframework nature of Flask ensures minimal overhead, making it suitable for resource-constrained environments. Its lightweight design and modular structure enable scalability, allowing the system to handle increased traffic and user interactions. Flask also supports RESTful API development, facilitating smooth communication between the frontend and backend components. With its intuitive API and excellent documentation, Flask simplifies the process of building robust and secure web applications. In summary, Flask serves as a valuable software requirement, empowering the project with a flexible and efficient web framework for the implementation of the animal detection system's user interface and backend services.

3.5.5 Hardware and experimental environment

The hardware used for this experiment includes Windows 11 Home 64-bit OS, x64-based processor, AMD Ryzen 5 4600H CPU @ 3GHz, 4266 Mhz, 6 Core(s), 12 Logical Processor(s), 8 GB RAM.

The experimental environment was prepared by using Python 3.9.13 programming language. Machine learning and deep learning libraries like - NumPy, Pandas, Matplotlib, Pillow, Torch, OpenCV, Seaborn, etc were also used.

Chapter 4

Result and Discussion

When compared to other object detection models, the proposed YOLO-based system outperformed some of the traditional object detection methods in terms of detection accuracy and speed. The YOLO algorithm's real-time detection capabilities and efficient architecture contribute to its superior performance. However, there are still some challenges in animal detection, especially in scenarios with complex backgrounds or occluded animals. The model struggle to detect small animals or animals with similar color patterns to the surroundings.

Testing is the main technique for quality control in software development. Running the available computer software after the coding step serves testing objectives. Both earlier-phase defects and those introduced during development must be discovered during testing. So, the goal of testing is to identify any requirements, design, or code errors in the software.

- To test a program, it is executed with the intention of finding any errors.
- The best test cases are those that have the best potential for identifying an error that hasn't been discovered yet.
- A test is successful if it identifies a previously undetected error.

Our goal is to create tests that systematically find a variety of problems with little time and effort. According to testing, software functionalities seem to function as planned, and performance standards seem to have been satisfied. The data gathered during testing is a good measure of software quality overall and a good predictor of programme reliability. Testing does have one drawback, though: it can only show whether there are software flaws, not whether there aren't.

4.1 Testing Methods

By using criteria that the user or the organisation has set, testing assures that the system is error-free. Depending on the context in which it functions, a system may have high-end or low-end performance. The testing methods aim to assess the accuracy of animal detection, evaluate the system's ability to handle different scenarios, and validate its real-world applicability.

4.1.1 Unit Testing

Unit testing involves testing individual components or functions of the system in isolation. Here, unit testing is performed to verify the correctness and reliability of specific functions, such as processing the image, detecting animals, and handling data.

4.1.2 Integration Testing

Integration testing focuses on evaluating the interaction between different components and modules within the system. Integration testing was done to ensure the seamless integration between the image processing module, object detection algorithm, and data storage components.

4.1.3 System Testing

System testing aims to evaluate the overall functionality and performance of the entire animal detection system. Comprehensive tests that simulate real-world scenarios to assess the system's ability to accurately detect animals, handle varying environmental conditions were done to verify the robustness and reliability of the animal detection system.

4.1.4 Performance Testing

Performance testing is essential to assess the system's efficiency and responsiveness. Systme's processing speed, memory and gpu usage, response time were tested.

4.2 Output Screens and Results

1. Uploading Image

User can select the image to be tested from his/her local system.

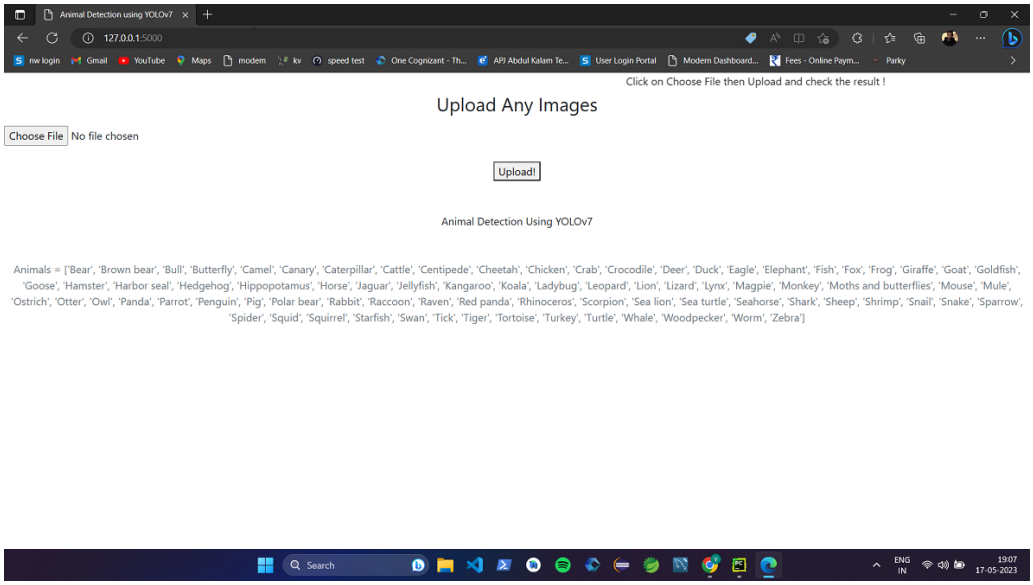

Figure 4.1: Home Page

2. Result Page

The prediction will be displayed around the image as a bounding box along with accuracy.

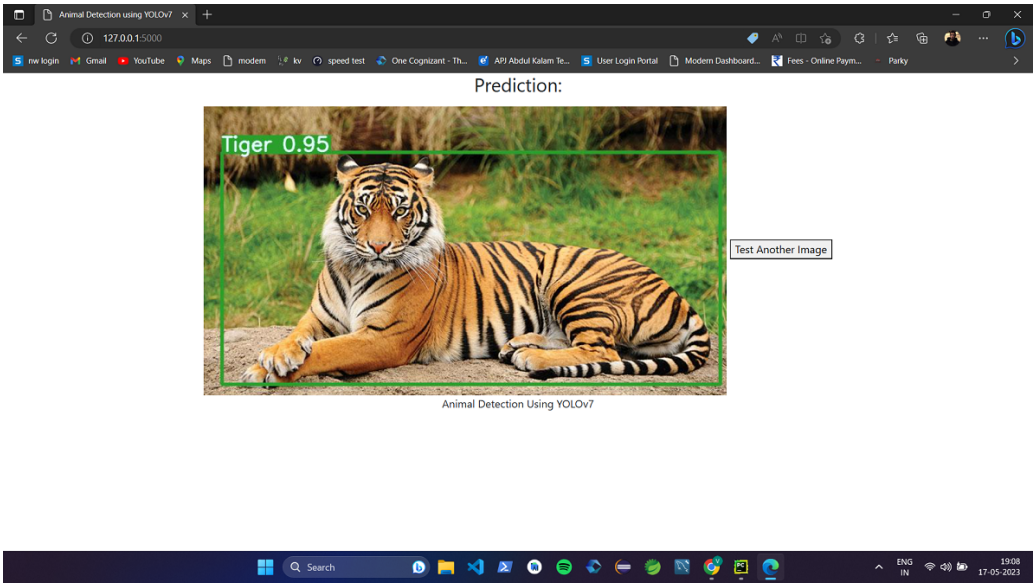

Figure 4.2: Result Page

3. Precision, Recall and Mean Average Precision

Precision is the proportion of correctly predicted positive instances out of all instances predicted as positive. Recall is the proportion of true positive instances correctly detected

out of all actual positive instances. mAP (mean average precision) measures the average precision across multiple classes or object categories.

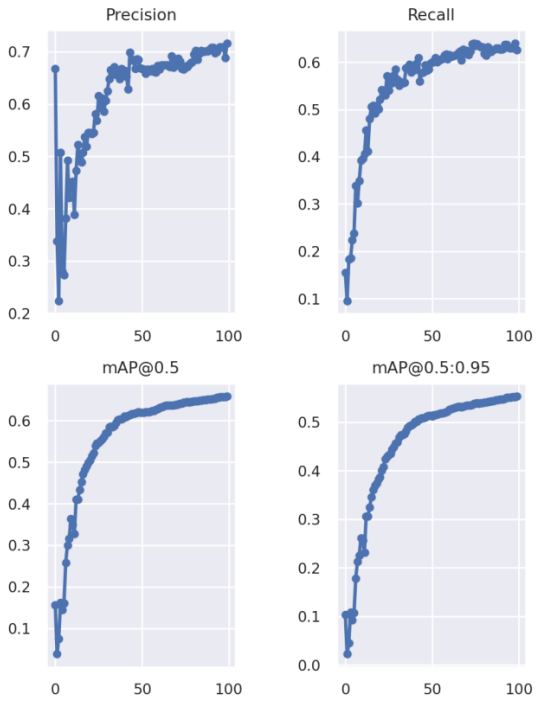

Figure 4.3: Precision, Recall and mAP

4. PR Curve

A precision-recall curve (or PR Curve) is a plot of the precision (y-axis) and the recall (x-axis) for different probability thresholds.

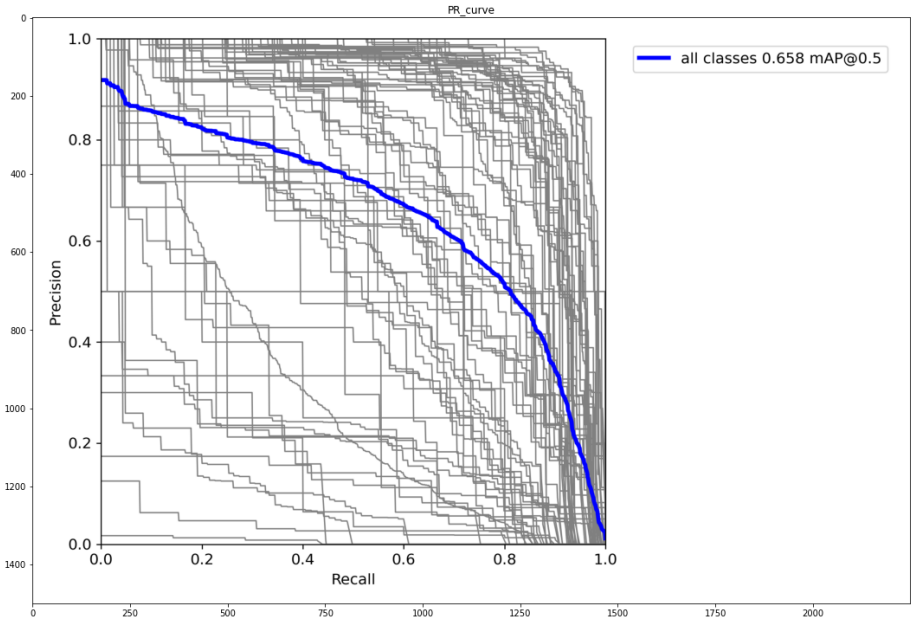

Figure 4.4: PR Curve

5. F1 Curve

   F1 score is an evaluation metric that measures a model's accuracy. It combines the precision and recall scores of a model.

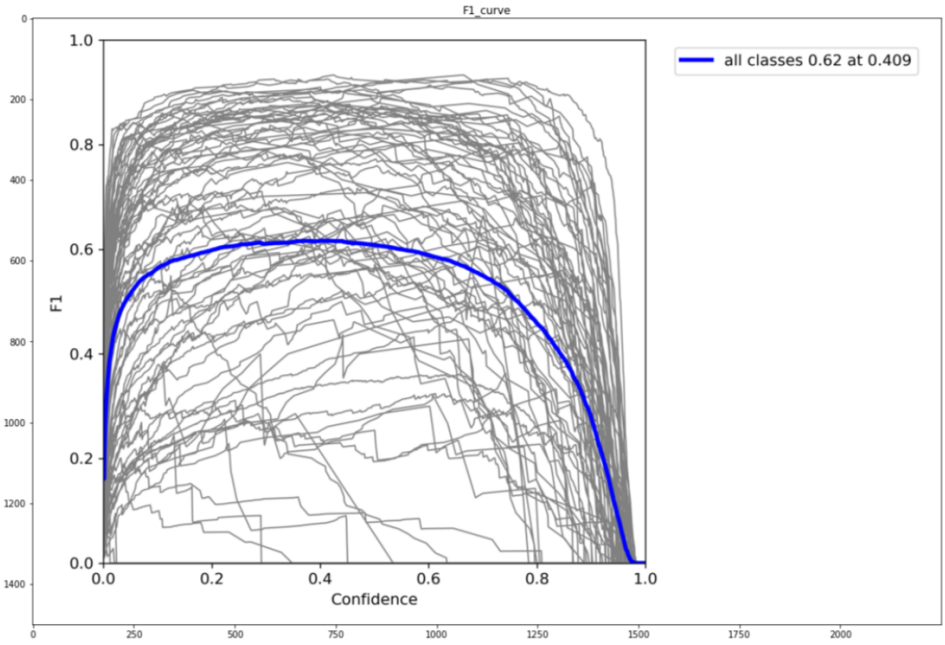

Figure 4.5: F1 curve

Chapter 5

Conclusion

The project aimed to enhance road safety by detecting animals on roads using the YOLO algorithm. The existing systems for animal detection have several limitations, such as low accuracy and efficiency. The proposed system addressed these limitations by using YOLO algorithm, which is a state-of-the-art algorithm for object detection.

By leveraging the YOLO algorithm, impressive results have been achieved in terms of object detection speed and accuracy. The algorithm's ability to process images in real-time makes it suitable for deployment in scenarios where immediate action is required. Through extensive training and fine-tuning, the YOLO model was optimized to achieve high precision and recall rates, ensuring reliable animal detection performance.

The utilization of the Google Open Images V6+ dataset has provided a diverse and comprehensive collection of animal images for training and testing.

The animal detection system developed in this project holds great potential for real-world applications. Its integration into existing road safety measures can significantly enhance driver awareness and reduce the risk of accidents caused by animal presence on roads.

In conclusion, the project has demonstrated the effectiveness of the YOLO algorithm in animal detection for road safety. It paves the way for the implementation of intelligent systems that can mitigate risks associated with animal crossings, ultimately improving road safety for both drivers and animals alike.

5.1 Future Enhancement

In addition to implementing the YOLO algorithm for animal detection, there are several enhancements that can be done in the future that can further improve the functionality and effectiveness of the project.

1. Currently, the system accepts an image as input and processes it for animal detection. By integrating real-time video streaming and implementing live object detection, users

would have the ability to capture video footage and receive immediate predictions on animal presence, enabling them to make informed decisions in real-time.

2. Expanding the animal species dataset is another important aspect. The inclusion of a broader range of animal species in the training dataset would enhance the system's ability to detect and classify various animals accurately.

3. To enhance the system's real-world applicability, integrating with existing road infrastructure and vehicle systems could be explored. This could involve integrating the animal detection system with roadside warning signs or even incorporating it into smart vehicles' advanced driver-assistance systems (ADAS). Such integration would provide real-time alerts directly to drivers, enhancing their situational awareness and allowing for proactive driving decisions.

References

[1] Sanjay Santhanam, Sudhir Sidhaarthan B, Sai Sudha Panigrahi, Suryakant Kumar Kashyap and Bhargav Krishna Duriseti, *"Animal Detection for Road safety using Deep Learning." 2021 International Conference on Computational Intelligence and Computing Applications (ICCICA) DOI: 10.1109/ICCICA52458.2021.9697287*

[2] B. Karthikeya Reddy, Shahana Bano, G. Greeshmanth Reddy, Rakesh Kommineni and P. Yaswanth Reddy, *"Convolutional Network based Animal Recognition using YOLO and Darknet." 2021 6th International Conference on Inventive Computation Technologies (ICICT). DOI: 10.1109/ICICT50816.2021.9358620*

[3] Zhi Zhang, Zhihai He, Guitao Cao and Wenming Cao, *"Animal Detection From Highly Cluttered Natural Scenes Using Spatiotemporal Object Region Proposals and Patch Verification." IEEE Transactions on Multimedia (Volume: 18, Issue: 10, October 2016).p-2079 - 2092. DOI: 10.1109/TMM.2016.2594138*

[4] Sachin Umesh Sharma and Dharmesh J. Shah, *"A Practical Animal Detection and Collision Avoidance System Using Computer Vision Technique."IEEE Access (Volume: 5). p-347 - 358. DOI: 10.1109/ACCESS.2016.2642981*

[5] Benjamin Kellenberger, Michele Volpi, and Devis Tuia, *"Fast animal detection in UAV images using convolutional neural networks" 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). DOI: 10.1109/IGARSS.2017.8127090*

[6] Mudit Goswami, V. Prem Prakash and Dhruv Goswami, *"Animal-Vehicle Collision Mitigation Using Deep Learning in Driver Assistance Systems." Advances in Computing and Data Sciences. ICACDS 2019. Communications in Computer and Information Science, vol 1045. Springer, Singapore. DOI: 10.1007/978-981-13-9939-*$8_2$6

[7] Joseph Redmon, Santosh Divvala, Ross Girshick and Ali Farhadi, *"You Only Look Once: Unified, Real-Time Object Detection." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779-788. DOI: 10.48550/arXiv.1506.02640*

[8] Chien-Yao Wang, Alexey Bochkovskiy and Hong-Yuan Mark Liao, *"YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors." 2017 IEEE International Conference on Cloud Engineering (IC2E). DOI: 10.1109/IC2E.2017.34*

[9] Chengji Liu, Yufan Tao, Jiawei Liang, Kai Li and Yihang Chen, *"Object Detection Based on YOLO Network." 2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC). DOI: 10.1109/ITOEC.2018.8740604*

[10] Tausif Diwan, G. Anirudh, Jiawei and Jitendra V. Tembhurne, *"Object detection using YOLO: challenges, architectural successors, datasets and applications." Multimed Tools Appl 82, 9243–9275 (2023). https://doi.org/10.1007/s11042-022-13644-y*

[11] Pranav Adarsh, Pratibha Rathi and Manoj Kumar, *"YOLO v3-Tiny: Object Detection and Recognition using one stage improved model." 2020 6th International Conference on Advanced Computing Communication Systems (ICACCS). DOI: 10.1109/ICACCS48705.2020.9074315*

[12] Mohammad Robihul Mufid, Arif Basofi, M. Udin Harun Al Rasyid, Indhi Farhandika Rochimansyah and Abdul rokhim, *"Design an MVC Model using Python for Flask Framework Development." 2019 International Electronics Symposium (IES). DOI: 10.1109/ELECSYM.2019.8901656*

[13] Prof. P. S. Lokhande, Fankar Armash Aslam, Hawa Nabeel Mohammed, Jummal Musab Mohd. Munir and Murade Aaraf Gulamgaus, *"Efficient Way Of Web Development Using Python And Flask." International Journal of Advanced Research in Computer Science. Volume 6, No. 2, March-April 2015. http://www.aiktcdspace.org:8080/jspui/handle/123456789/1367*

Appendix

Screenshots

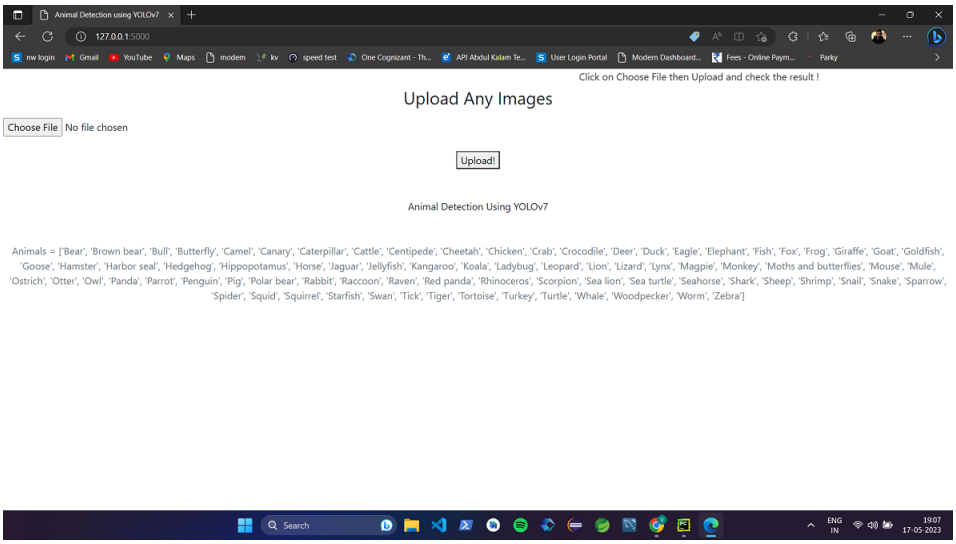

Figure A.1: Home Page

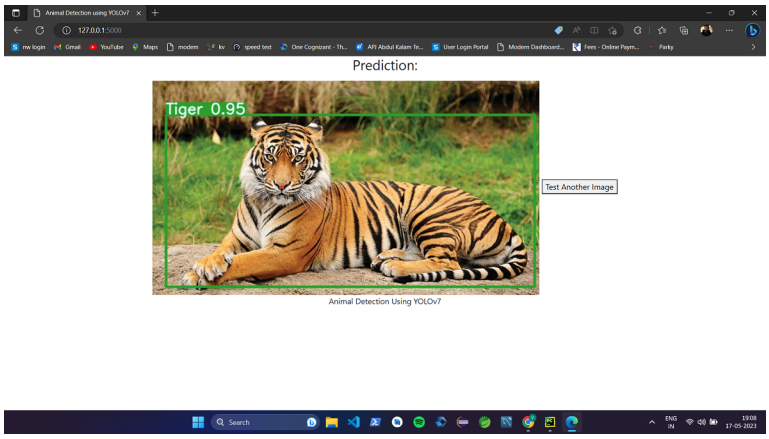

Figure A.2: Result 1

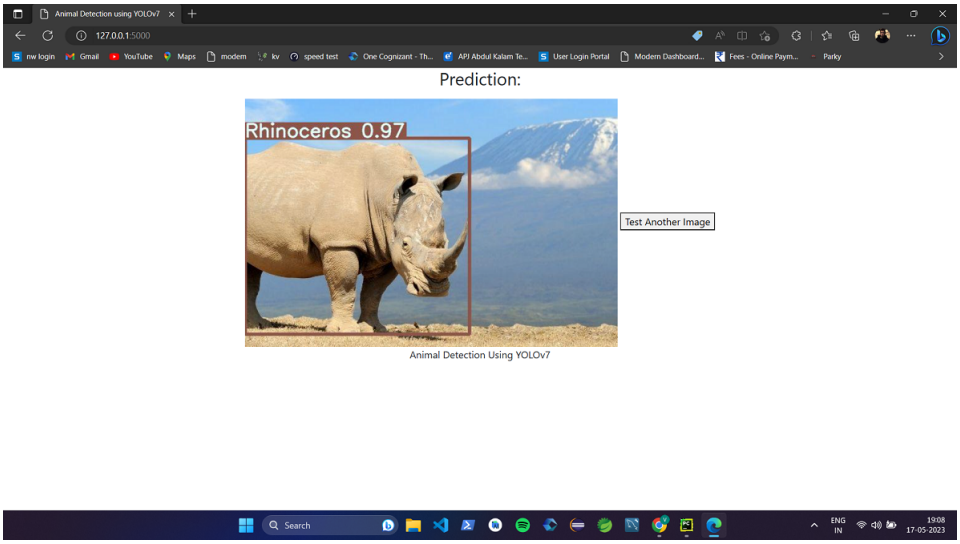

Figure A.3: Result 2

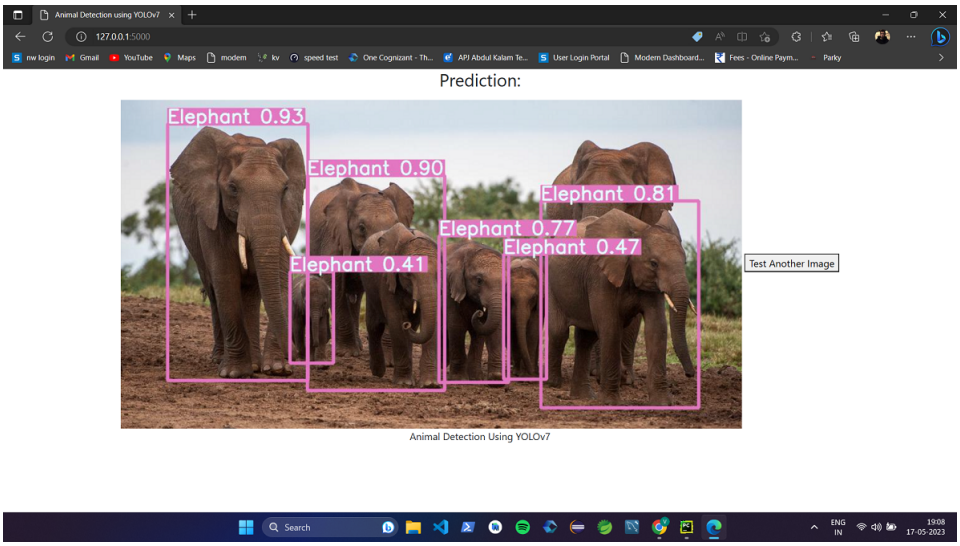

Figure A.4: Result 3

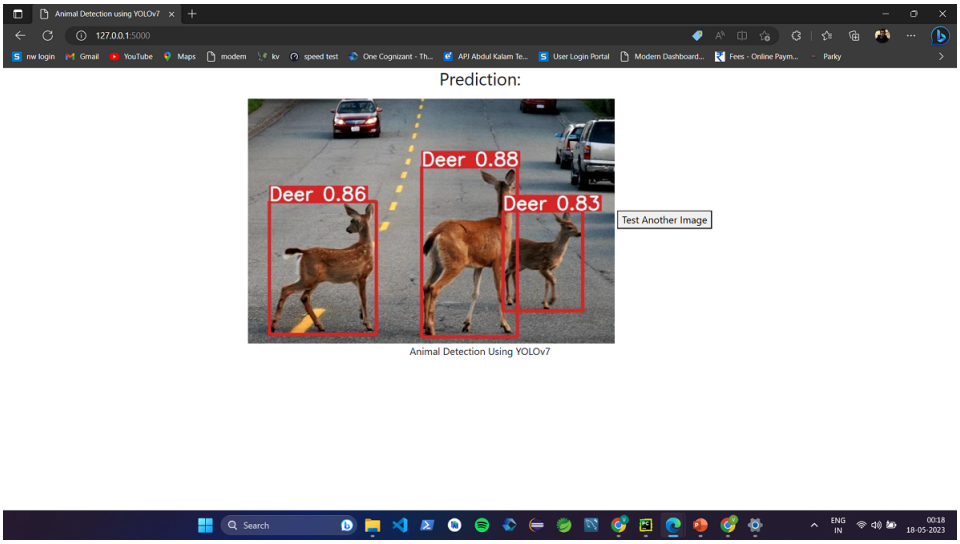

Figure A.5: Result 4

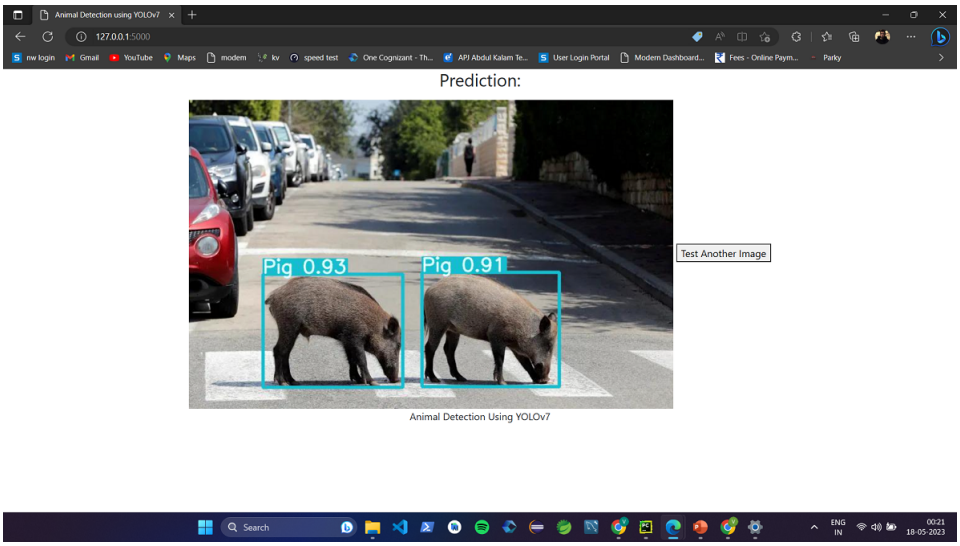

Figure A.6: Result 5

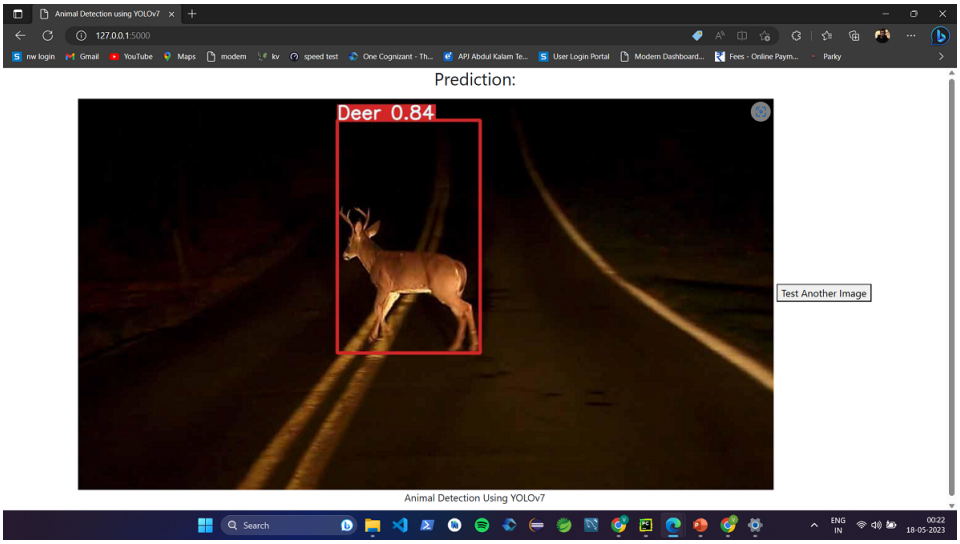

Figure A.7: Result 6(Low light)

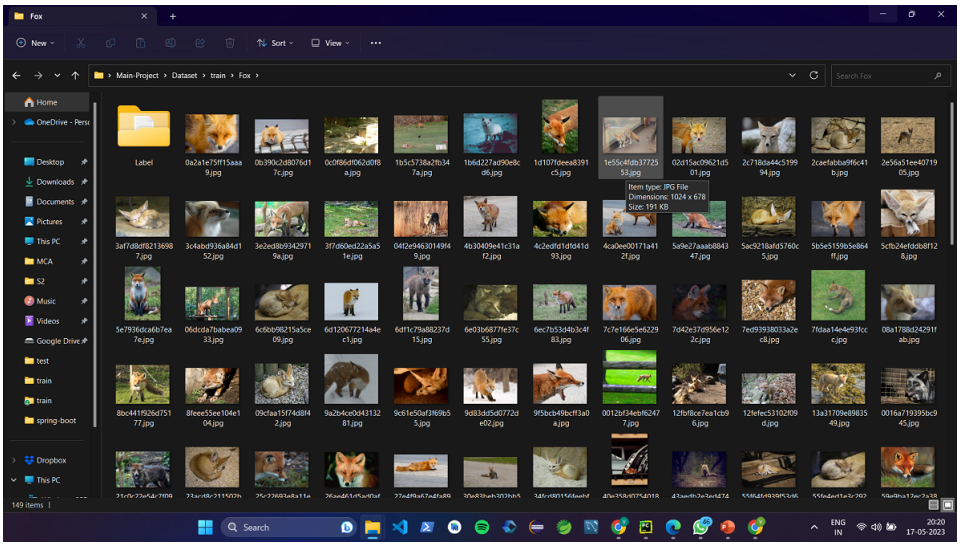

Figure A.8: Dataset images

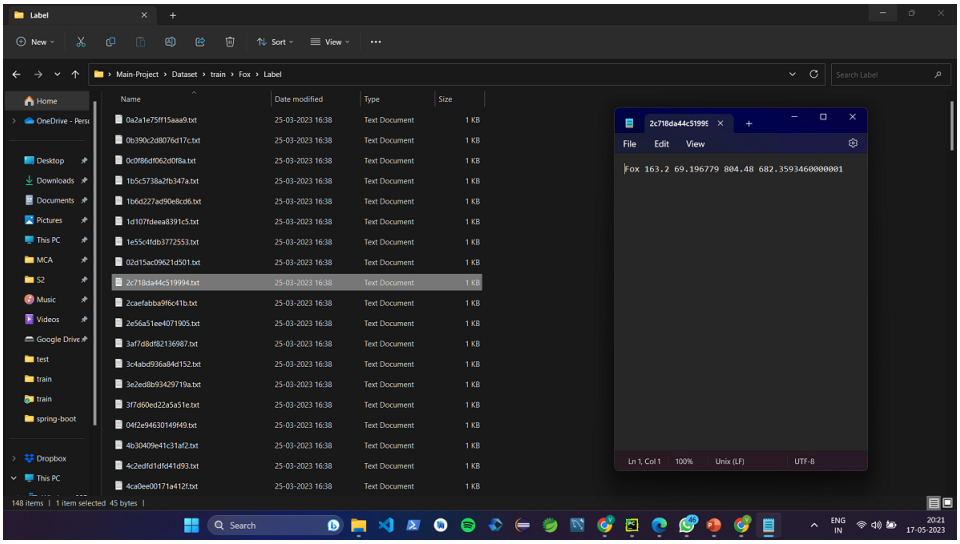

Figure A.9: Image labels