ANIMAL DETECTION USING MACHINE LEARNING

A Project Report

submitted

by

ADARSH.V (MBT20EC011/B20EC1105)

ADITH.V (MBT20EC012/B20EC1106)

MOHAMMED FINAZ.A (MBT20EC062/B20EC1138)

MRIDUL.M (MBT20EC063/B20EC1139)

MRUDHUL SREEKUMAR (MBT20EC064/B20EC1140)

to

the APJ Abdul Kalam Technological University

in partial fulfillment of the requirements for the award of the Degree

of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

Department of Electronics and Communication Engineering
Mar Baselios College of Engineering and Technology

(Autonomous)

Mar Ivanios Vidya Nagar, Nalanchira, Thiruvananthapuram- 695015

April 2024

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

MAR BASELIOS COLLEGE OF ENGINEERING AND TECHNOLOGY(Autonomous)

Mar Ivanios Vidyanagar, Nalanchira

Thiruvananthapuram-695015

CERTIFICATE

*This is to certify that the report entitled "**ANIMAL DETECTION USING MACHINE LEARNING**" by **ADITH.V (MBT20EC012/B20EC1106)** to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Electronics and Communication Engineering is a bonafide record of the project work carried out by him/her under my/our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.*

Dr. Luxy Mathews	**Ms. Deepa P L**	**Dr. Jayakumari J**
(Project Co-ordinator)	(Internal Supervisor)	(Head of the Department)
Associate Professor	Assistant Professor	Professor
Department of ECE	Department of ECE	Department of ECE

Place: Thiruvananthapuram

Date:

External Examiner

ACKNOWLEDGEMENT

With great enthusiasm and pleasure, we are bringing out this projector report here. We use this opportunity to express our heartiest gratitude to the support and guidance offered to us from various sources during the course of completion of our project.

We are also grateful to **Dr. Jayakumari J**, Professor, Head of the Department, Electronics and Communication Engineering, for her valuable suggestions.

It is our pleasant duty to acknowledge our Project Co-coordinator, **Dr. Luxy Mathews** in the Department of Electronics and Communication Engineering, who has guided and given supervision for the project work.

Let us express our heartfelt gratitude to our guide, **Ms. Deepa P L,** Assistant professor ECE dept. for helping us all throughout the project.

Above all, we owe our gratitude to the **Almighty** for showering abundant blessing upon us.

We also express our wholehearted gratefulness to all our classmates who have expressed their views and suggestions about our projects and have helped us during the course of the project.

We extend our sincere thanks and gratitude once again to all those who helped us make this undertaking a success.

ABSTRACT

In rural areas, the increasing frequency and severity of animal attacks pose a significant threat to the safety and well-being of residents, livestock, and local ecosystems. Our idea is to develop a comprehensive system designed to detect wildlife using the YOLOv8 object detection model integrated with Closed-Circuit Television (CCTV) cameras, with a focus on notifying officials in real-time. Leveraging the efficiency of YOLOv8, the system processes live camera feeds, identifying and localizing wildlife within the monitored environment. The integration ensures seamless communication with a notification system, triggering alerts to relevant officials when specific wildlife species are detected. The system encompasses key steps, including YOLOv8 installation, pre-trained weight download, dataset preparation, model training, real-time object detection, and the establishment of trigger conditions for alerting. The proposed system holds promising applications in wildlife conservation, addressing human-wildlife conflicts, and enhancing the efficiency of monitoring efforts in natural habitats.

CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

Detecting and identifying animal species in their natural habitats is a critical aspect of wildlife conservation and the mitigation of wildlife-human conflicts. However, implementing such tasks, particularly on embedded devices with limited resources, poses significant challenges.

One of the fundamental requirements in this domain is the need for real-time detection capabilities. Wildlife encounters, whether involving humans or vehicles, can lead to conflicts and hazards. Therefore, there is a pressing demand for a detection system that is not only fast and accurate but also lightweight enough to operate in real-time. Portable embedded platforms are particularly relevant for wildlife monitoring in remote or challenging environments.

To address these challenges, a modified YOLOv8 model is proposed. This modification involves merging multi-level features, removing unnecessary convolutional layers, and integrating deformable convolutional layers (DCLs). The primary objective is to enhance both accuracy and efficiency in the detection of animal species. By tailoring the YOLOv8 model to the specific requirements of wildlife detection, the aim is to create a system that can effectively identify and classify animals in their natural habitats while operating seamlessly on resource-constrained embedded platforms.

The YOLOv8 algorithm, an evolution of the YOLO series, excels in real-time object detection tasks, making it particularly well-suited for wildlife monitoring applications. With its ability to process images and videos in a single pass, YOLOv8 delivers impressive speed and accuracy, enabling conservationists to quickly identify and track wildlife across vast and challenging terrains. The algorithm's versatility allows it to detect multiple animal species simultaneously, providing a comprehensive and holistic approach to wildlife monitoring.

This endeavor not only aligns with the broader goals of wildlife conservation but also acknowledges the unique constraints of the deployment environment. The integration of

advanced technologies and modifications to existing models showcases a commitment to leveraging cutting-edge approaches for the benefit of both wildlife monitoring initiatives and the safety of human populations.

Chapter 2

LITERATURE REVIEW

The following are the base paper and reference papers that we have taken as reference for our main project:

Mai Ibraheam; Kin Fun Li; Fayez Gebali, " An Accurate and Fast Animal Species Detection System for Embedded Devices ", IEEE Access, 13 March 2023.

Encounters between humans and wildlife often lead to injuries, especially in remote wilderness regions, and highways. Therefore, animal detection is a vital safety and wildlife conservation component that can mitigate the negative impacts of these encounters. Deep learning techniques have achieved the best results compared to other object detection techniques; however, they require many computations and parameters.

A lightweight animal species detection model based on YOLOv2 was proposed. It was designed as a proof of concept of and as a first step to build a real-time mitigation system with embedded devices. Multi-level features merging is employed by adding a new pass-through layer to improve the feature extraction ability and accuracy of YOLOv2. Moreover, the two repeated 3 × 3 convolutional layers in the seventh block of the YOLOv2 architecture are removed to reduce computational complexity, and thus increase detection speed without reducing accuracy. Animal species detection methods based on regular

Convolutional Neural Networks (CNNs) have been widely applied; however, these methods are difficult to adapt to geometric variations of animals in images. Thus, a modified YOLOv2 with the addition of deformable convolutional layers (DCLs) was proposed to resolve this issue. Our experimental results show that the proposed model outperforms the original YOLOv2 by 5.0% in accuracy and 12.0% in speed. Furthermore, our analysis shows that the modified YOLOv2 model is more suitable for deployment than YOLOv3 and YOLOv4 on embedded platform devices.

Ms.V.Megala ; Dr.S.K.Jayanthi,"An object detection method using the modified YOLO V4 algorithm- An abnormal activity detection in roads and railways".

One of the most extensively used techniques in computer vision is object detection. It enables us to recognize and find an object in an image or video. The main aim is to detect the object which was involved in the abnormal activity in public places like Railway Tracks and National Highway Roads. The abnormal activity denotes the harassment, robbery, suicide and accidents which mostly happen during the night time in Roadways and Railways. This detection methodology uses the modified YOLO v4 algorithm to recognize the object, with the kind of identification and localization, it may be used to count the items in a given image and monitor their suitable place. Object detection distinguishes the sorts of objects present in the picture before locating the precise manifestation of the object.

For object labelling, a bounding box surrounding the item can be used to accomplish object localization. As a result, the object detection alerts the control room of the appropriate department about the abnormal activity and helps the officers to take immediate action to prevent the illegal activity occurring in the area. This object detection methodology not only detects human beings but also detects the trapped animals in the railway tracks.

J. K. Vishwas1; S. M. Prajwal Raj ; Bhagya, M. Anand ,"CNN Based Animals Recognition using Advanced YOLO V5 and Darknet", International Journal of Research in Engineering, Science and Management, 6 June 2022.

In general, the manual detection of animals with their names is a very tedious task. To overcome this challenge, this research work has developed a YOLOV3 model to identify the animal present in the image given by the user. The algorithm used in the YOLOV3 model is darknet, which has a pre- trained dataset. The overall performance of the model is based on different training images and testing images of the dataset. The main goal of this research work is to build an animal recognition methodology using the YOLOV3 model. The image of the

animal will be given as input, then it will display the name of the animal as output by using the YOLOV3 model. The detection is done by using a pre-trained coco dataset from darknet.

Wild animal monitoring is of great significance to population discovery and research on animal behavior and habits. Early wild animal monitoring mainly relied on human effort, which is time-consuming and contains safety risks. In recent years, with the continuous development of pattern recognition techniques, automated wildlife detection algorithms based on image content analysis have also made progress thanks to these achievements. However, due to the complexity of field scenes, the recognition accuracy and robustness of existing methods can not meet the practical application requirements. Based on considerations, we suggest a field animal detection method based on YOLOv8, which aims to localize and recognize the wild animal. We analyzed the change of recognition accuracy for different scenes in detail, especially for the scene containing multiple targets, small targets or occluded targets. We have used a large number of experiments to verify the feasibility of this method.

Chapter 3

PROPOSED METHODOLOGY

In rural areas, the rising frequency and severity of animal attacks present a substantial threat to the safety and well-being of residents, livestock, and local ecosystems. These encounters between humans and wildlife can result in various challenges and potential hazards, ranging from property damage to direct threats to human life.

3.1 PROBLEM STATEMENT

Human-wildlife conflicts in rural settings often arise due to factors such as habitat loss, encroachment on natural habitats, and changes in animal behavior patterns. As human populations expand into wildlife habitats, there is an increased likelihood of interactions that may lead to confrontations or attacks.

According to reports from the State Forest Ministry, a total of 637 people were killed due to wild animal attacks in the last 5 years in Kerala alone. India's cost of human–elephant conflict is estimated at 1 million (destroyed crops), 10,000 to 15,000 damaged properties, 400 human deaths, and 100 dead elephants per year.

Agricultural communities often bear the brunt of wildlife attacks, resulting in significant financial losses and damage to crops. This ongoing issue poses a substantial threat to the livelihoods of farmers and the sustainability of agriculture in affected regions.

3.2 PROPOSED SOLUTION

The proposed solution involves the creation of an animal detection system utilizing machine learning algorithms, specifically employing deep learning techniques. Deep learning, a subset of machine learning, involves the use of neural networks with multiple layers (deep neural networks) to automatically learn and extract features from data. In the context of wildlife detection, this approach aims to leverage advanced algorithms to detect and identify various animal species.

This model can be implemented in the existing CCTV cameras across the major roads and railways routes where wildlife encroachment is usually seen and can provide assistance for

reducing the number of fatalities caused by human and wildlife interactions.

Integrating the animal detection model with existing CCTV cameras presents a pragmatic and impactful solution for mitigating human-wildlife conflicts. By providing real-time information and alerts, the system contributes to the safety of both humans and wildlife, fostering a more harmonious coexistence.

Strategically placed cameras analyze real-time video feeds, distinguishing between harmless and potentially dangerous animals. Upon detection, the system triggers immediate alerts to local authorities and communities, enabling swift response and intervention.

CHAPTER 4

DESIGN STEPS

The basic design steps include identifying routes, training a model, incorporating CNN algorithm, categorizing the identified animal, alert system etc.

4.1 IDENTIFYING ROUTES

Identifying routes associated with animal reserves, national parks, and common animal encroachment areas is a crucial step in establishing a YOLO-based CCTV system for effective wildlife monitoring and management. This process involves leveraging geospatial data and conservation insights to map out transportation routes that intersect with or run in close proximity to these key wildlife habitats. By strategically situating YOLO-based CCTV cameras along these identified routes, authorities can enhance their capacity to detect and track animal movements, fostering early intervention in potential human-wildlife conflict scenarios. This proactive approach not only aids in safeguarding both human populations and wildlife but also provides valuable data for ecological research and conservation efforts. The YOLO-based CCTV system, renowned for its real-time object detection capabilities, becomes a powerful tool in ensuring the coexistence of human activities and wildlife, contributing to biodiversity preservation and sustainable wildlife management practices.

4.2 MODEL TRAINING

Training a model to identify an animal based on visual data involves a multifaceted process. Initially, a diverse dataset containing annotated images of the target animal is curated, specifying bounding boxes around the animals and assigning corresponding class labels. This dataset is then used to train a deep learning model, such as YOLO (You Only Look Once), employing transfer learning with pre-trained weights for efficiency. During training, the model learns to recognize distinctive features and patterns associated with the specified animal classes. The training process involves adjusting model parameters through iterations to optimize performance. Post-training, the model is fine-tuned and evaluated on validation datasets to ensure generalization to unseen data. This trained model can subsequently be

integrated into a real-time system, enabling the automatic identification of animals in visual data, with applications ranging from wildlife monitoring and conservation to automated animal detection in surveillance and ecological research.

4.3 INCORPORATING CNN ALGORITHM

Incorporating a Convolutional Neural Network (CNN) algorithm alongside YOLOv8 serves as a strategic approach to enhance the efficiency of object detection. YOLOv8, known for its real-time capabilities, can benefit from the feature extraction prowess of a CNN to further refine and augment its understanding of complex visual patterns. By integrating a CNN as part of the YOLOv8 architecture, the model gains the capacity to capture intricate hierarchical features in the input data. This synergy allows the CNN to perform sophisticated feature extraction, providing more nuanced representations to YOLOv8 during the object detection process. The combined model harnesses the strengths of both YOLOv8's speed and CNN's feature extraction capabilities, resulting in improved accuracy and robustness in identifying objects, especially in scenarios with intricate visual details or challenging environmental conditions. This collaborative integration optimizes the overall efficiency of the object detection system, making it well-suited for applications requiring both speed and high precision.

4.4 CATEGORIZATION

Categorizing identified animals based on their nature represents a critical phase in wildlife monitoring and management, extending beyond the initial detection provided by models like YOLOv8. This process involves a meticulous analysis of the detected species, taking into account a myriad of factors such as behavior, physical characteristics, and ecological role. Once an animal is identified through the object detection model, additional layers of data, including species information, habitat preferences, and behavioral patterns, are integrated to classify it according to its nature. This comprehensive approach allows for a nuanced understanding of the wildlife population, distinguishing between predatory and herbivorous

species, identifying migratory patterns, and elucidating the animal's ecological significance.

The value of categorization lies in its multifaceted applications across ecological research, wildlife conservation, and management efforts. By gaining insights into the behavioral and ecological attributes of identified animals, tailored strategies can be developed for habitat preservation, mitigating potential human-wildlife conflicts, and fostering sustainable coexistence. The knowledge derived from detailed categorization contributes to informed decision-making in conservation initiatives, facilitating a more holistic and proactive approach to wildlife management. As a result, the combination of real-time object detection and comprehensive categorization becomes a powerful tool for not only monitoring wildlife but also actively contributing to the preservation of biodiversity and the delicate balance of ecosystems.

4.5 ALERT SYSTEM

Implementing an effective wildlife animal detection system involves a comprehensive notification infrastructure that ensures timely alerts and responses. The first step in this process is setting up a robust notification system, which includes determining the channels through which alerts will be disseminated. This can encompass various communication mediums such as email, SMS, and push notifications, ensuring that alerts reach relevant stakeholders promptly. Integrating geolocation services further enhances the system's capabilities by providing precise information about the location of wildlife activity. This geospatial data not only enables rapid responses in cases of potential human-wildlife conflicts but also contributes to the creation of detailed maps and analytics, aiding researchers and conservationists in understanding wildlife behavior and movement patterns.

The deployment of such a notification system goes beyond immediate threat mitigation; it plays a crucial role in empowering communities to coexist safely with wildlife. By receiving real-time alerts through accessible channels, community members can take precautionary measures to avoid conflicts and safeguard their livelihoods. This proactive approach fosters a

sense of shared responsibility, encouraging communities to actively participate in conservation efforts. Additionally, the integration of geolocation services ensures that responders can reach the precise location of wildlife activity swiftly, minimizing response times and optimizing resource allocation. This collaborative and technology-driven approach not only enhances human safety but also promotes a harmonious coexistence between communities and wildlife, ultimately contributing to long-term biodiversity conservation efforts.

CHAPTER 5

NETWORK DESIGN USING YOLO

YOLO, which stands for "You Only Look Once," is a real-time object detection system in computer vision. The key idea behind YOLO is to detect objects in an image or a video frame by dividing the input image into a grid and predicting bounding boxes and class probabilities directly for each grid cell.

Unlike traditional object detection methods that involve multiple stages, YOLO performs object detection in a single pass. It divides the input image into a grid and assigns bounding boxes to different grid cells. Each bounding box prediction includes the coordinates of the box, the confidence score of the prediction, and the class probabilities for the object contained in the box. This makes YOLO computationally efficient and well-suited for real-time applications.

There have been several versions of YOLO, with improvements and optimizations introduced in each iteration. Some notable versions include YOLOv2, YOLOv3, and YOLOv4, etc. each bringing enhancements in terms of accuracy, speed, and capabilities. YOLO has been widely used in various applications, such as autonomous vehicles, surveillance, and object tracking, due to its ability to provide fast and accurate object detection in real-time.

5.1 YOLOv8

YOLO version 5, or YOLOv8, represents a state-of-the-art real-time object detection model in the field of computer vision. YOLO is renowned for its unique single-shot detection approach, allowing the model to process an entire image in a single forward pass through the neural network. Developed by Ultralytics, YOLOv8 builds upon its predecessors, introducing advancements in terms of both accuracy and speed. The architecture incorporates anchor boxes during training, facilitating precise bounding box predictions. This makes YOLOv8 particularly effective for scenarios where rapid and accurate object detection is paramount, such as in video surveillance, autonomous vehicles, and robotics.

YOLOv8 has a modular and flexible design, enabling users to easily adapt the model to different tasks and datasets. The framework is open-source, fostering collaboration and

widespread adoption in the computer vision community. YOLOv8 is trained on diverse datasets, often utilizing the Common Objects in Context (COCO) dataset, enabling it to generalize well to various object categories and scenes. Its evolution through different versions, from YOLOv1 to the latest YOLOv8, reflects a continuous effort to push the boundaries of real-time object detection, making it a versatile and widely-used tool in both research and practical applications.

5.1.1 WORKING

Step 1: Image Preprocessing

In the initial step of the YOLOv8 algorithm, known as Image Preprocessing, the input image undergoes essential transformations to standardize its characteristics for effective processing. First, the image is resized to a specific dimension, often set to 640 x 640 pixels, which establishes a consistent scale for subsequent analysis. This resizing step is crucial for maintaining computational efficiency and ensuring uniform handling of images with varying resolutions. Additionally, color normalization is performed by adjusting pixel values based on mean and standard deviation values. This normalization process helps the model operate effectively across diverse datasets by standardizing color representations, reducing the impact of variations in lighting conditions, and enhancing the overall stability of the algorithm.

Furthermore, the normalized and resized images serve as the input for subsequent stages of the YOLOv8 pipeline. These preprocessing steps lay the foundation for the neural network to extract meaningful features during the subsequent backbone network phase. The emphasis on standardization and normalization in this initial step contributes to the model's ability to generalize across different datasets and effectively process images of varying scales and lighting conditions.

Step 2: Feature Extraction

In Step 2 of the YOLOv8 algorithm, the feature extraction process takes place within the backbone network. This neural network is responsible for analyzing the preprocessed image and extracting hierarchical features that represent both low-level details and high-level patterns essential for object detection. YOLOv8 employs a custom architecture that consists of several types of layers:

Convolutional Layers (Conv): These layers play a fundamental role in feature extraction by capturing low-level features such as edges, textures, and basic shapes within the image. Convolutional operations involve sliding filters across the input image to identify local patterns.

C3 Blocks: YOLOv8 introduces C3 blocks, which are stacked sets of convolutional layers. These blocks facilitate the extraction of more complex and abstract features, combining information from different receptive fields within the image. The stacking of convolutional layers allows the model to capture increasingly sophisticated representations.

Spatial Pyramid Pooling (SPP): YOLOv8 incorporates SPP to aggregate features from multiple scales within the image. This helps the model maintain context and effectively capture objects of varying sizes. SPP enables the network to consider both fine-grained and coarse-grained information, contributing to improved object detection across different spatial resolutions.

The feature extraction stage enhances the model's ability to understand the hierarchical structure of the input image, enabling it to capture relevant patterns and representations for subsequent stages of the YOLOv8 algorithm. The result is a set of feature maps that contain increasingly abstract and informative representations of the input image, laying the foundation for precise object detection in the later stages of the algorithm.

Step 3: Prediction and Detection

In Step 3 of the YOLOv8 algorithm, the feature fusion process in the neck network plays a

pivotal role in consolidating information from different levels of the backbone network. The backbone extracts feature at various scales, capturing both detailed and high-level representations of the input image. The Path Aggregation Network (PAN) is employed to facilitate efficient interconnections between these feature levels. By creating paths that link higher and lower-level features, PAN enables the model to merge fine-grained details with broader contextual information. This fusion of features from multiple scales enhances the model's ability to detect objects across a diverse range of sizes, ensuring a comprehensive understanding of the input image.The feature fusion mechanism in YOLOv8's neck network is designed to improve the model's versatility and adaptability. By synthesizing information from different levels, the model achieves a holistic representation of the scene, allowing it to accurately identify objects of varying scales and complexities. This step is instrumental in the overall success of YOLOv8's object detection capabilities, contributing to its effectiveness in real-world scenarios with diverse and dynamic visual environments.

Step 4: Non-Maximal Suppression (NMS)

refine the output of the object detection algorithm. After the model predicts bounding boxes for potential objects, it's common to have multiple bounding boxes that overlap or cover the same object. NMS addresses this issue by eliminating redundant and less confident detections, ensuring that only the most relevant and accurate predictions are retained. The process involves sorting the detected bounding boxes based on their associated confidence scores and iteratively selecting the box with the highest confidence. Subsequently, boxes with significant overlap (measured by intersection over union, or IoU) with the selected box are suppressed, leaving only the most confident and non-overlapping predictions for a given object. This results in a cleaner and more accurate set of bounding boxes, mitigating the problem of duplicate detections and improving the precision of the overall object detection system.

NMS is crucial in scenarios where the model may generate multiple bounding boxes for the same object, as it helps streamline the final output for downstream tasks. By selectively retaining the most confident and non-overlapping predictions, NMS enhances the

interpretability of the detection results, providing a more concise and reliable set of bounding boxes for the objects present in the input image.

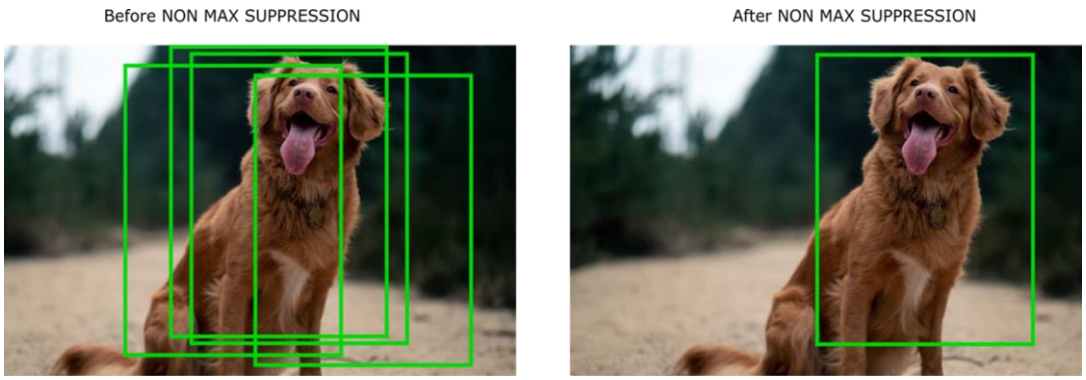

Fig 5.1-Before and After non max suppression

Step 5: Output and Visualization

In Step 5 of the YOLOv8 algorithm, the output and visualization process is a critical stage where the model's predictions are interpreted and presented in a human-readable format. The output typically includes a set of bounding boxes, each associated with an object's class label, confidence score, and spatial coordinates. These bounding boxes encapsulate the predicted locations of detected objects within the input image. Following this, the visualization step involves overlaying these bounding boxes onto the original image, often using different colors to distinguish between object classes. The resulting visual output provides an intuitive representation of the model's object detection performance, allowing users to identify and understand the recognized objects, their positions, and associated confidence levels within the context of the input image. This visualization step is instrumental in both evaluating the model's accuracy and communicating the detection results to end-users or downstream applications in scenarios such as surveillance, autonomous vehicles, or image analysis.

Fig 5.2 - Bounding boxes and confidence score

CHAPTER 6

PROPOSED METHOD

The basic workflow of the model is as follows: Dataset is taken as input and is preprocessed. Then this data is given to a YOLOv8 model which trains and validates the model using the dataset. This model is then integrated to a CCTV which analyses live footage and detects wild animals and notifies the authorities about sightings of dangerous wild animals.

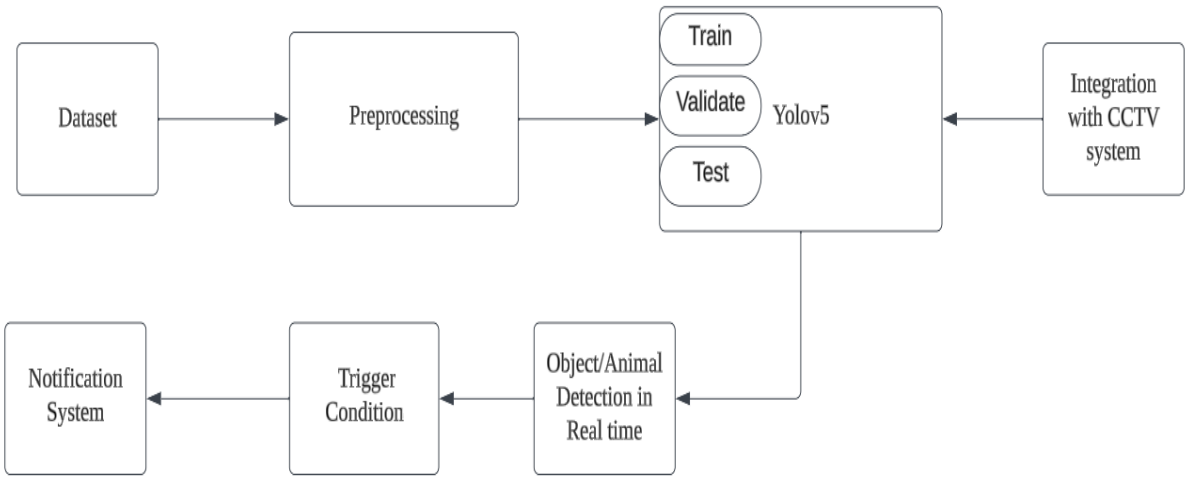

Fig 6.1 - Bounding boxes and confidence score

Dataset: A dataset comprises images and corresponding annotation files defining object locations and class labels within those images. The dataset is essential for training and evaluating the YOLO model, with each image having an associated annotation in YOLO format specifying object classes and normalized bounding box coordinates. The dataset includes a list of classes, and it is typically divided into training, validation, and testing sets. The quality and diversity of the dataset are critical for the model's ability to generalize to new scenarios. Preparation involves careful organization, annotation, and preprocessing to ensure accurate training and evaluation of the YOLO model.

Preprocessing: To preprocess a custom dataset for YOLOv8, organize the data with images and corresponding annotation files, creating label files in YOLO format. Split the dataset into

training, validation, and testing sets, ensuring class balance. Generate a YAML configuration file describing dataset details such as paths, class names, and image dimensions. Resize images to the model's input size, normalize pixel values, and optionally apply data augmentation. Verify annotations, class names, IDs, and file paths for correctness. Once preprocessed, use the YOLOv8 training script with the provided configuration file to train the model, monitoring performance and adjusting parameters as needed during training.

Training YOLO with custom dataset: YOLOv8 model is trained on the prepared dataset using the collected and annotated images. Training involves adjusting the model's parameters based on the dataset, optimizing its ability to recognize specific objects. Hyperparameters such as learning rate, batch size, and epoch count are fine-tuned to achieve optimal performance. The model iteratively refines its internal weights through backpropagation, minimizing the difference between predicted and ground truth bounding boxes. Regular evaluations on a separate validation dataset ensure that the model is not overfitting to the training data, and adjustments are made as needed. The goal of this step is to enable the YOLOv8 model to accurately and efficiently detect specified objects in real-world scenarios.

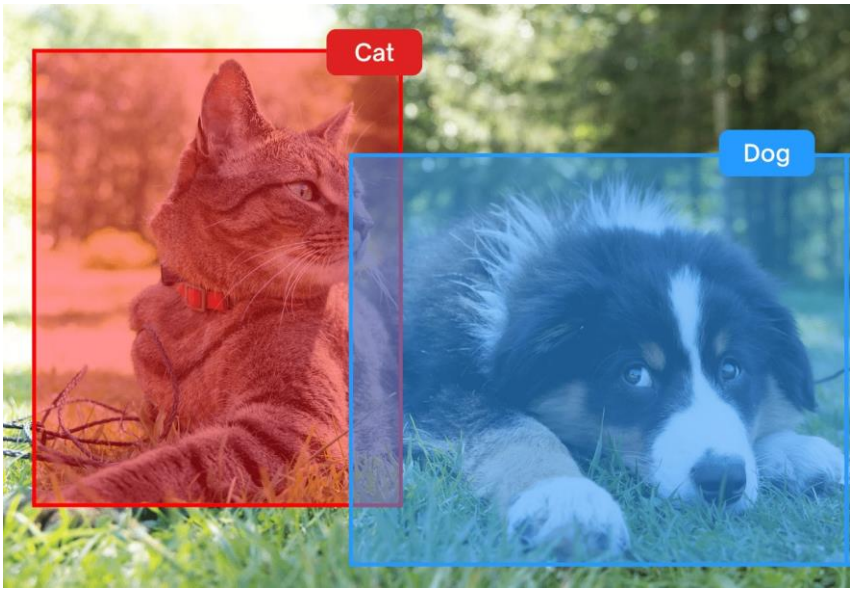

Fig 6.2 - Labeling custom data for training

Validation And Test: After training the YOLOv8 model, a rigorous evaluation is conducted on a separate validation dataset to thoroughly assess its ability to generalize and detect objects accurately in diverse scenarios. The validation process involves comparing the model's predictions with the ground truth annotations present in the validation set. Key evaluation

metrics, such as precision, recall, and mean average precision (mAP), are computed to quantitatively measure the model's performance. Precision reflects the accuracy of positive predictions, recall measures the ability to capture all relevant instances, and mAP provides an aggregate metric of detection accuracy across different object classes. By scrutinizing these metrics, practitioners gain insights into the model's strengths and weaknesses. Adjustments to the model architecture, training parameters, or the dataset may be iteratively refined based on the evaluation results to enhance overall performance and mitigate potential issues, ensuring that the YOLOv8 model can reliably and accurately identify objects in a variety of real-world scenarios.

Integration with CCTV: Integrating YOLO with CCTV for animal detection involves configuring CCTV cameras to stream video feeds to a system where YOLO is applied for real-time object detection. The video feed is preprocessed to align with YOLO's input requirements, and the model is integrated to identify and locate animals within the monitored area. Detection thresholds, post-processing, and custom training may be applied to improve accuracy, and alerts are triggered upon animal detection. Logging systems record events for analysis, and continuous monitoring ensures real-time responsiveness. The feedback loop allows for model updates, adjustments, and system optimization, making the integrated solution effective for applications like wildlife monitoring, agricultural protection, and areas where automated animal detection is crucial.

Object/Animal detection in real time: Real-time object detection with YOLOv8 is implemented by deploying the trained model to continuously process incoming data, such as live camera feeds or video streams. The system is configured to efficiently process each frame, predicting bounding boxes and class labels for detected objects in real-time. The integration ensures seamless communication with the notification system, triggering alerts to officials when specified conditions are met. Continuous monitoring, error handling, and performance optimization strategies are employed to maintain robust and responsive object detection capabilities. This step is essential for applications requiring timely responses, such as surveillance and security systems, where the immediate identification of objects is crucial for effective decision-making.

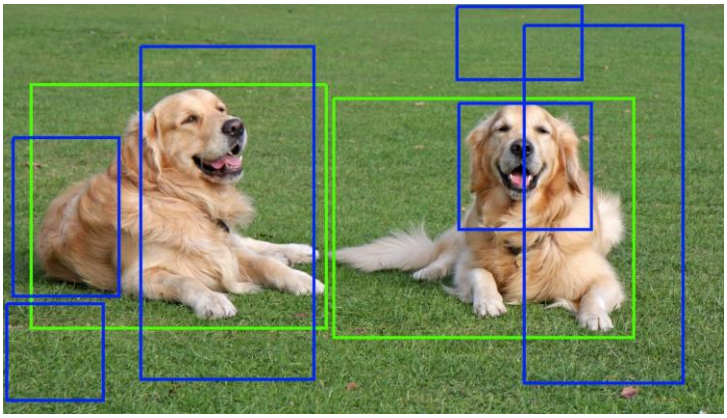

Fig 6.3 - Selection of bounding boxes

Trigger Condition: In the context of YOLOv8-based object detection and notification, trigger conditions encompass predetermined criteria that, when satisfied during real-time detection, activate the alerting mechanism to notify officials. These conditions may encompass various parameters, such as identifying objects belonging to specific classes like elephants, tigers, or lions, with a confidence score surpassing a predefined threshold. Furthermore, spatial considerations within the frame and patterns of movement can also be integral trigger conditions.

By delineating trigger conditions, the system ensures that notifications are dispatched selectively, aligning precisely with the desired objectives of the surveillance or monitoring application. This tailored approach enhances adaptability to diverse scenarios, optimizing the system's responsiveness and mitigating the occurrence of false alerts. The ability to customize trigger conditions offers a level of flexibility that proves crucial in refining the system's performance, ensuring that alerts are triggered only under circumstances deemed relevant and significant for effective surveillance and monitoring.

Notification System: The system incorporates an effective notification mechanism designed to promptly alert officials upon the real-time detection of specific objects by the YOLOv8 model. The implementation of this mechanism involves the establishment of a robust notification system, including the configuration of channels for alert delivery—such as email, SMS, or push notifications—and the seamless integration between the object detection component and the notification infrastructure. This integration ensures that the moment the YOLOv8 model identifies objects meeting predefined conditions, the notification system is triggered to instantly notify officials. This swift notification process enhances situational

awareness, facilitating timely and responsive actions concerning the identified objects.

By relying on the signals from the YOLO-installed CCTV system, the system can precisely determine the location of the detected objects. This feature is pivotal in expediting response times, as officials can quickly and accurately assess the situation based on the real-time information provided by the YOLOv8-based object detection system. The synergy between object detection and notification mechanisms plays a critical role in maximizing the practical utility of the YOLOv8 system. The seamless coordination between these components ensures that the system not only identifies objects efficiently but also facilitates a rapid and informed response, contributing to the overall effectiveness of the surveillance and security infrastructure.

Screen capture and transfer for manual verification : This involves several steps.

To implement the feature of saving and sending images via Gmail when detections are made using YOLO, we first need to set up the object detection functionality. This includes loading the YOLO model by configuring its architecture and loading the necessary weights and configuration files. With the model set up, we can then process images to detect objects within them. This involves defining the detection process, where the model identifies and localizes objects present in the image.

Once the detection process is established, we proceed to modify our script to save images when detections occur. After each detection, we use OpenCV's `cv2.imwrite()` function to save the processed images. This step ensures that images containing detected objects are stored locally for future reference or for further analysis.

The final step involves integrating email sending functionality into our script. To achieve this, we utilize a library like `smtplib` in Python, which enables us to connect to an SMTP server, such as Gmail's SMTP server. With proper authentication using Gmail credentials, we compose an email message that includes the detected image as an attachment. This email is then sent to the desired recipient using the `smtplib` library. It's crucial to ensure that the Gmail account used for sending emails allows access to less secure apps or uses OAuth for authentication to send emails programmatically.

By combining these steps, our model becomes capable of detecting objects in images using

YOLO, saving the annotated images locally, and subsequently sending them via Gmail with the detected objects highlighted. This feature proves to be valuable in various applications, such as surveillance systems, automated monitoring, or object recognition tasks. Now an authorized individual can analyze the given image and take necessary action accordingly.

Chapter 7

SOFTWARES USED

Google Colab, is a cloud-based platform provided by Google that offers a collaborative environment for writing and executing Python code. Hosted on Google Drive, Colab provides a Jupyter notebook interface, making it particularly popular for tasks like machine learning and data analysis. It offers free access to GPUs and TPUs, accelerating computations, and integrates seamlessly with other Google services. With the ability to share and collaborate on notebooks in real-time, Colab facilitates collaborative coding projects, and its integration with Google Drive ensures easy storage and retrieval of work. Overall, Google Colab is a versatile and accessible tool that combines the convenience of cloud computing with the power of Python for diverse computational tasks.

7.1 PYTHON

Machine learning has emerged as a transformative force across various industries, revolutionizing the way we analyze data, make predictions, and automate decision-making processes. At the heart of this revolution lies Python, a versatile and powerful programming language that has become the de facto standard in the field of machine learning. In this two-page description, we explore the key aspects of Python's role in machine learning, its popularity, and the rich ecosystem of libraries and frameworks that make it a preferred choice for researchers, data scientists, and developers.

Python's popularity in machine learning can be attributed to its simplicity, readability, and a vast community of developers actively contributing to its ecosystem. The language's clean syntax allows for easy expression of complex ideas, making it accessible for both beginners and experienced programmers. Its versatility enables seamless integration with other languages and technologies, fostering collaboration and innovation within the machine learning community.

Python's success in machine learning is closely tied to its rich ecosystem of libraries and frameworks. NumPy and pandas provide essential tools for data manipulation and analysis, while scikit-learn offers a comprehensive suite of machine learning algorithms and tools. TensorFlow and PyTorch, two of the most popular deep learning frameworks, have Python interfaces that facilitate the development of sophisticated neural network models. The ease with which these libraries can be combined allows developers to build end-to-end machine learning pipelines efficiently.

The open-source nature of Python has fostered a vibrant and collaborative community, driving continuous improvement and innovation in machine learning. Online platforms such as GitHub host countless repositories where developers share code, models, and best practices. The collaborative nature of the Python community accelerates the development of new algorithms, techniques, and applications, making it easier for practitioners to stay at the forefront of advancements in the field.

Python's ecosystem includes powerful libraries for data visualization and exploration, such as Matplotlib, Seaborn, and Plotly. These tools enable researchers and data scientists to gain insights from their datasets, visualize model performance, and communicate results effectively. The seamless integration of these libraries with machine learning workflows enhances the interpretability and communication of complex models and results.

In conclusion, Python has firmly established itself as the go-to language for machine learning, offering a winning combination of simplicity, versatility, and a thriving ecosystem. Its widespread adoption across academia, industry, and research institutions highlights its pivotal role in the advancement of machine learning applications. As the field continues to evolve, Python's adaptability and community support position it as a foundational tool for unlocking the full potential of machine learning.

7.2 OPEN LABELING SOFTWARE

Open labeling software has become an indispensable component in the realm of object detection, particularly in the context of the highly efficient YOLO (You Only Look Once) algorithm. The success of YOLO in real-time object detection hinges on the quality and diversity of annotated datasets used for model training. Open labeling software addresses the challenges associated with annotating large datasets, providing annotators with tools to label objects, define bounding boxes, and ensure standardized, accurate annotations. As a result, it streamlines the annotation process, contributing to the robustness and effectiveness of YOLO-based object detection models.

An essential aspect of open labeling software for YOLO is its ability to export annotations in the format required by YOLO models. The YOLO format includes class labels, normalized coordinates, and object dimensions. This compatibility ensures a smooth transition from annotation to training, allowing developers to seamlessly integrate annotated datasets into their YOLO-based object detection pipelines. It streamlines the workflow, making the entire process more accessible and efficient for researchers, data scientists, and developers working with YOLO.

The role of open labeling software in the success of YOLO-based object detection is pivotal. By expediting the annotation process, these tools contribute to the creation of high-quality datasets, enabling YOLO models to accurately recognize and localize diverse object classes. The efficient annotation facilitated by these tools enhances the generalization and real-world performance of YOLO models, making them robust and applicable across various domains. As YOLO evolves and continues to be at the forefront of object detection, open labeling software remains a crucial ally, driving advancements and ensuring the continued success of YOLO-based solutions in real-time object detection scenarios.

7.3 CLOUD COMMUNICATION PLATFORM

Twilio api offers a comprehensive suite of communication tools, with its SMS and MMS messaging functionality standing out as a cornerstone feature. Through Twilio's APIs, developers gain the ability to seamlessly integrate programmable SMS and MMS capabilities into their applications. This empowers businesses to engage with their customers through text-based communications, facilitating a wide range of use cases.

With Twilio's SMS and MMS messaging capabilities, developers can send and receive text messages and multimedia messages programmatically, enabling efficient communication channels for notifications, alerts, verification codes, and marketing messages. This feature is instrumental in enhancing customer engagement, improving service delivery, and driving business growth.

Twilio's SMS functionality extends beyond simple text messaging, allowing for the integration of rich media content, such as images, videos, and audio files, through MMS. This enables businesses to create more immersive and engaging messaging experiences for their customers, enhancing the effectiveness of their communication strategies.

Overall, Twilio's SMS messaging feature serves as a powerful tool for businesses to deliver timely, personalized, and impactful communications to their customers. By leveraging Twilio's robust platform, developers can build scalable and reliable communication solutions that drive engagement, build loyalty, and drive business success.

CHAPTER 8

RESULTS

We were able to collect different datasets of various wild animals and to preprocess the data by adding labels to the data and train a YOLOv8 model with this data to successfully identify 8 different species of wild animals which are commonly known to create interruptions in the state of Kerala in India.

8.1 MODEL TRAINING OUTPUT

1. **mAP (mean Average Precision)**: It's the mean of the AP values across all classes. mAP provides an overall performance measure of the model across different object classes. It measures the accuracy of a model in locating objects within an image**.**

2. **Precision:** It measures the accuracy of the positive predictions made by the model. Precision is the ratio of true positive predictions to the total number of positive predictions made by the model.

3. **Recall:** It measures how well the model captures all the positive samples. Recall is the ratio of true positive predictions to the total number of actual positive samples

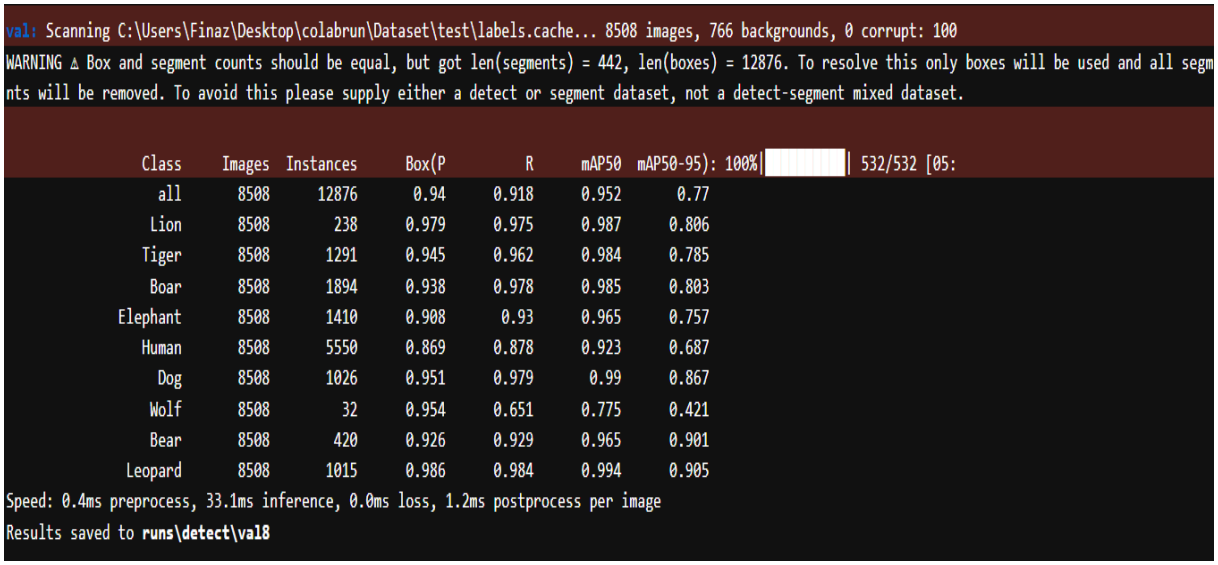

Fig 8.1 - Model training output

mAP value averages at about 0.77

8.2 CONFUSION MATRIX

The confusion matrix serves as a cornerstone in assessing the performance of classification models, offering a comprehensive view of model predictions by juxtaposing actual and predicted class labels. Through its depiction of true positives, true negatives, false positives, and false negatives, it furnishes a nuanced understanding of a model's strengths and weaknesses, facilitating targeted error analysis and model refinement. This matrix not only quantifies overall model accuracy but also dissects precision, recall, specificity, and other performance metrics for individual classes, ensuring a granular evaluation of classification efficacy. Its interpretability and accessibility make it a versatile tool, enabling stakeholders to gauge model performance and make informed decisions regarding model deployment and improvement strategies.

With its ability to quantify model performance and diagnose specific error types, the confusion matrix empowers practitioners in the realm of machine learning. By delineating true and false predictions across classes, it illuminates potential areas for enhancement, guiding adjustments to decision thresholds and feature engineering endeavors. Moreover, its utility extends beyond model evaluation, as it aids in detecting class imbalances within datasets and facilitates comparative analyses between different classification algorithms. As an indispensable instrument in the evaluation toolkit, the confusion matrix underpins the iterative process of model development, fostering continuous improvement and refinement in machine learning applications.

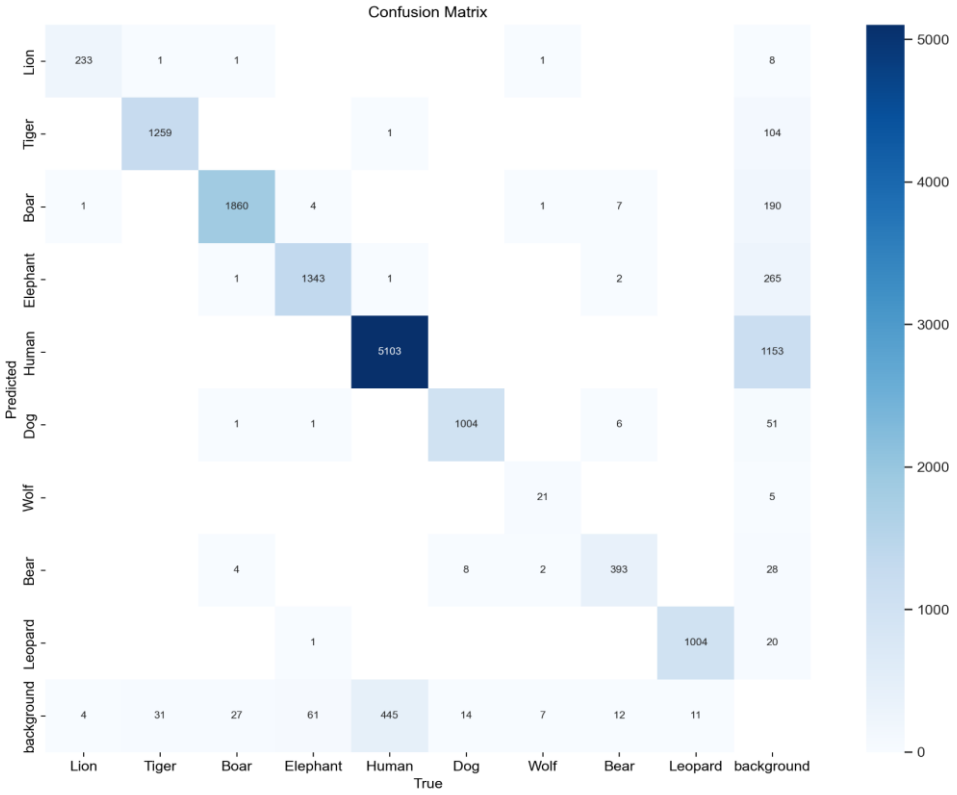

Fig 8.2 - Confusion matrix

A confusion matrix is a tabular representation used in machine learning to assess the performance of a classification model. It summarizes the results of classification predictions by comparing predicted labels with actual labels. The matrix is organized into four quadrants: true positives, true negatives, false positives, and false negatives. True positives are instances where the model correctly predicts positive outcomes, true negatives are instances where negative outcomes are correctly predicted, false positives occur when the model incorrectly predicts positive outcomes, and false negatives occur when negative outcomes are incorrectly predicted. This matrix provides a clear view of the model's strengths and weaknesses, enabling the evaluation of performance metrics such as accuracy, precision, recall, and F1 score, which are crucial for understanding the model's effectiveness in differentiating between classes.

8.3 NORMALISED CONFUSION MATRIX

The confusion matrix stands as a cornerstone in evaluating classification models, offering a comprehensive view of model predictions by juxtaposing actual and predicted class labels. Through its depiction of true positives, true negatives, false positives, and false negatives, it furnishes a nuanced understanding of a model's strengths and weaknesses, facilitating targeted error analysis and model refinement. This matrix not only quantifies overall model accuracy but also dissects precision, recall, specificity, and other performance metrics for individual classes, ensuring a granular evaluation of classification efficacy. Its interpretability and accessibility make it a versatile tool, enabling stakeholders to gauge model performance and make informed decisions regarding model deployment and improvement strategies.

However, advancements in evaluation techniques have introduced nuanced variations, such as the confidence curve, which provides additional insights beyond the conventional confusion matrix. Unlike the traditional matrix, the confidence curve plots model confidence levels against prediction accuracy, allowing for a more nuanced understanding of model performance across different confidence thresholds. By visualizing the relationship between confidence and accuracy, it helps identify the trade-offs between false positives and false negatives at varying confidence levels, enabling practitioners to optimize decision thresholds for specific use cases. The confidence curve thus complements the traditional confusion matrix by offering deeper insights into model behavior and aiding in fine-tuning classification models for enhanced performance.

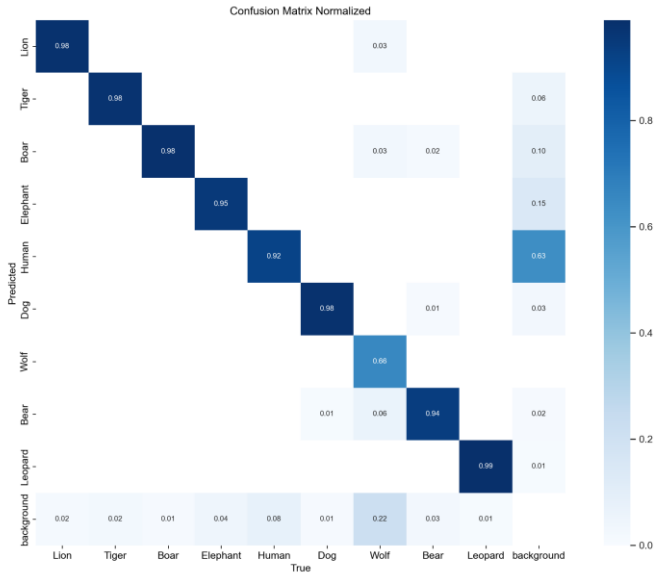

Fig 8.3 -Normalised confusion matrix

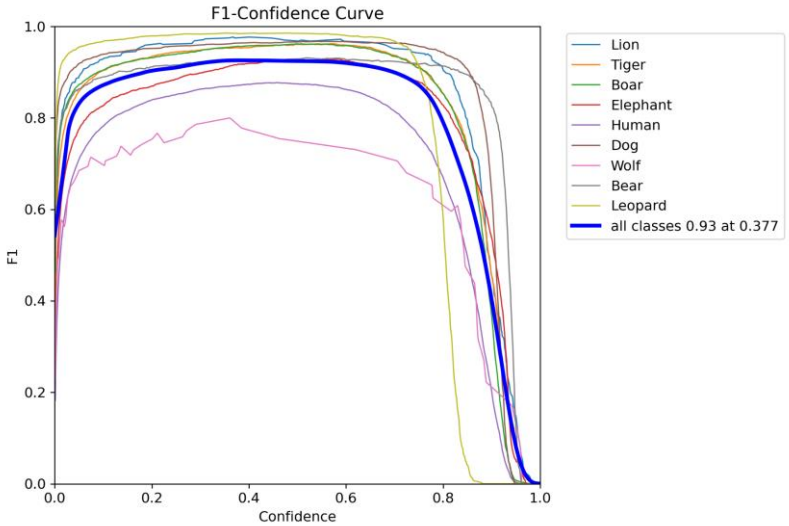

Fig 8.4 F1 - Confidence curve

An F1-confidence curve illustrates the relationship between the F1 score and confidence thresholds in a binary classification model. The F1 score is a single metric that balances both precision and recall, making it useful for evaluating model performance, especially in scenarios where there's an imbalance between positive and negative classes.

Confidence, in this context, refers to the model's certainty in its predictions, often represented by the probability assigned to instances belonging to the positive class.

The F1-confidence curve helps visualize how changes in confidence thresholds affect the F1 score. As the confidence threshold varies, the F1 score adjusts accordingly. Generally, increasing the confidence threshold tends to result in higher precision but lower recall, while decreasing the threshold typically leads to higher recall but lower precision. The F1-confidence curve allows for examining the trade-offs between precision and recall across different confidence levels, aiding in the selection of an optimal threshold for a specific application. This curve provides valuable insights into how to balance precision and recall effectively, thus improving the overall performance of the classification model.

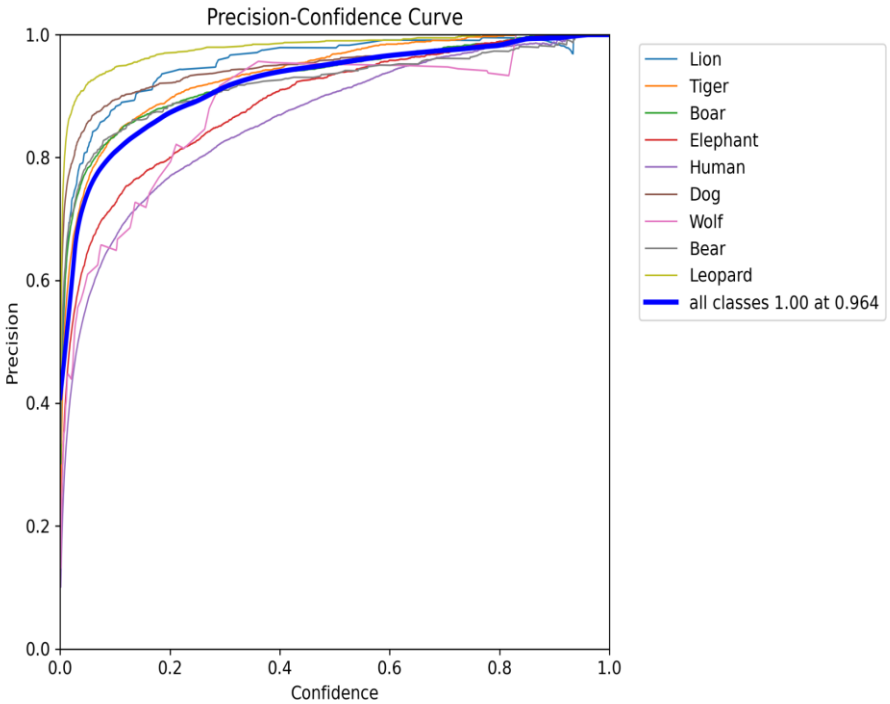

Fig 8.5 Precision - Confidence curve

The Precision-Confidence curve illustrates how the precision of a binary classification model changes with different confidence thresholds. Precision measures the accuracy of positive predictions, indicating the proportion of true positive instances among all predicted positive instances. Confidence, on the other hand, represents the model's certainty in its predictions, often denoted by the probability assigned to instances belonging to the positive class.

By adjusting the confidence threshold for classifying instances as positive or negative, the curve reveals shifts in precision. Higher confidence thresholds typically result in higher precision but lower recall, as the model becomes more conservative in making positive predictions. Conversely, lowering the confidence threshold tends to increase recall but may introduce more false positives, thereby decreasing precision. The Precision-Confidence curve provides valuable insights into the trade-offs between precision and confidence thresholds, aiding in the selection of an optimal threshold for achieving the desired balance between precision and recall in a specific application.

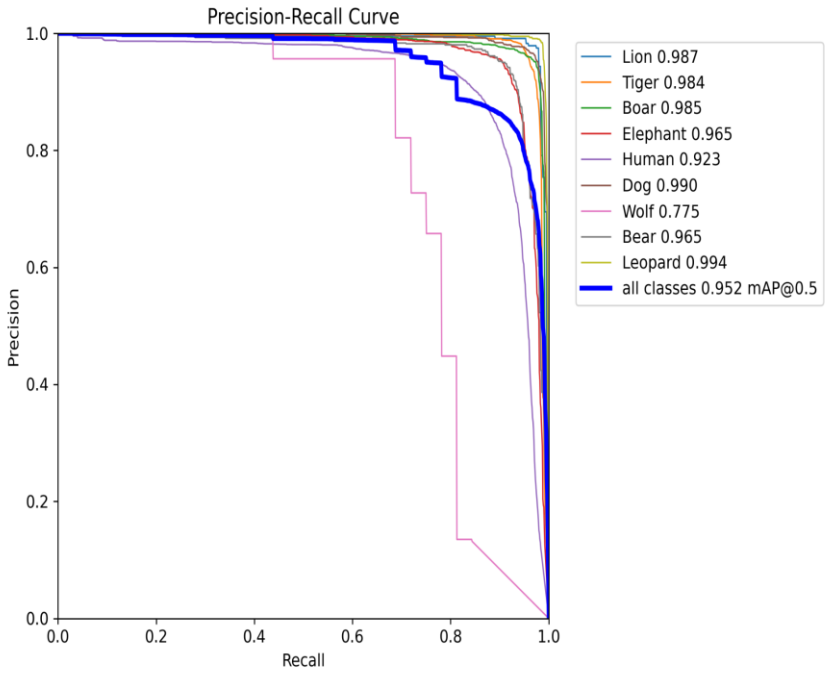

Fig 8.6 - Precision Recall Curve

A precision-recall curve offers a comprehensive visualization of a binary classification model's performance, illustrating the trade-off between precision and recall across various classification thresholds. Precision measures the accuracy of positive predictions, reflecting the ratio of true positives to the sum of true positives and false positives. On the other hand, recall assesses the model's ability to identify all positive instances, representing the ratio of true positives to the sum of true positives and false negatives. By plotting precision against recall for different classification thresholds, the curve provides insights into the model's behavior and performance characteristics. A well-balanced model ideally achieves high precision and recall simultaneously, but in practice, adjustments in the threshold typically lead to a trade-off between the two metrics. The area under the precision-recall curve serves as a summary metric, quantifying the overall performance of the model across all possible thresholds.

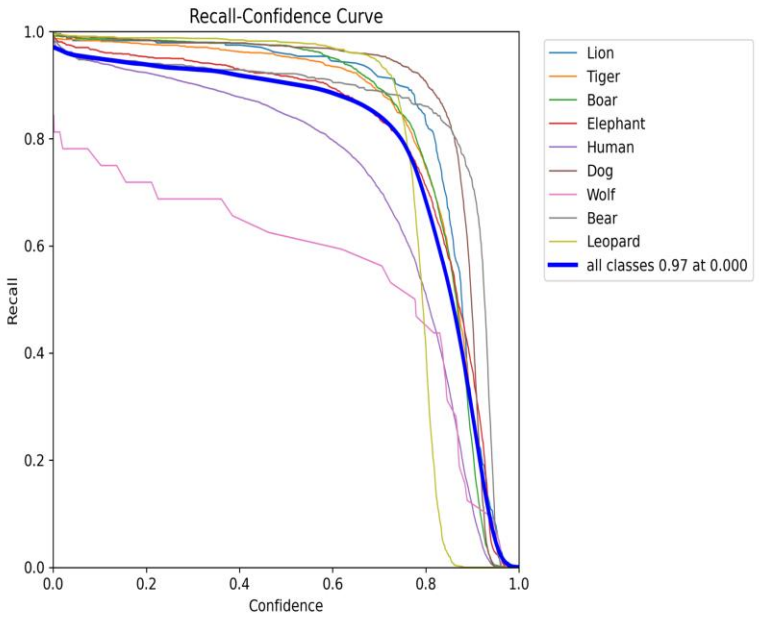

Fig 8.7 Recall - Confidence curve

A recall-confidence curve shows how the recall of a binary classification model changes with different confidence thresholds. Recall measures the model's ability to identify positive

instances, while confidence reflects the certainty of its predictions. By adjusting the confidence threshold, we observe shifts in recall: higher thresholds lead to lower recall as the model becomes more conservative, while lower thresholds increase recall but may introduce more false positives. This curve helps understand the trade-offs between recall and confidence thresholds, crucial for optimizing model performance to meet specific application needs.

8.4 OUTPUTS OBTAINED

Currently, we have successfully completed the training of a basic model to detect and identify 8 wild animals such as Tiger, Lion, Leopard, Elephants from various footage with minimal errors.

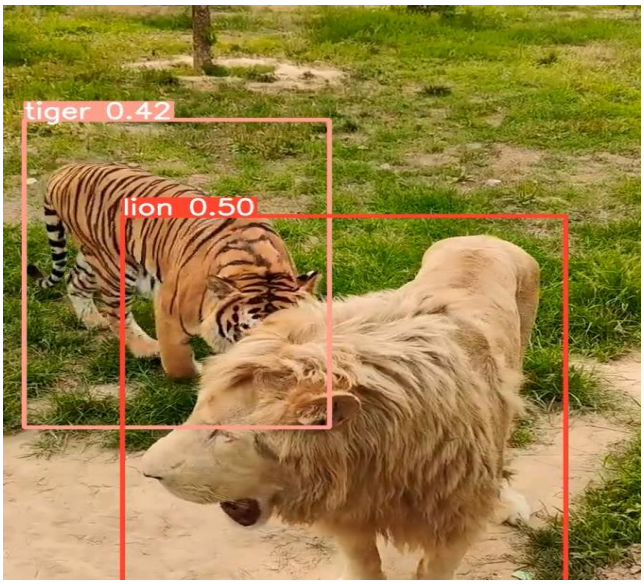

Fig. 8.8- Output 1 from our model 1

Fig. 8.9- Output 2 from our model 1

8.5 OUTPUTS FROM FINAL

Detected Lion with a confidence score of 0.82.

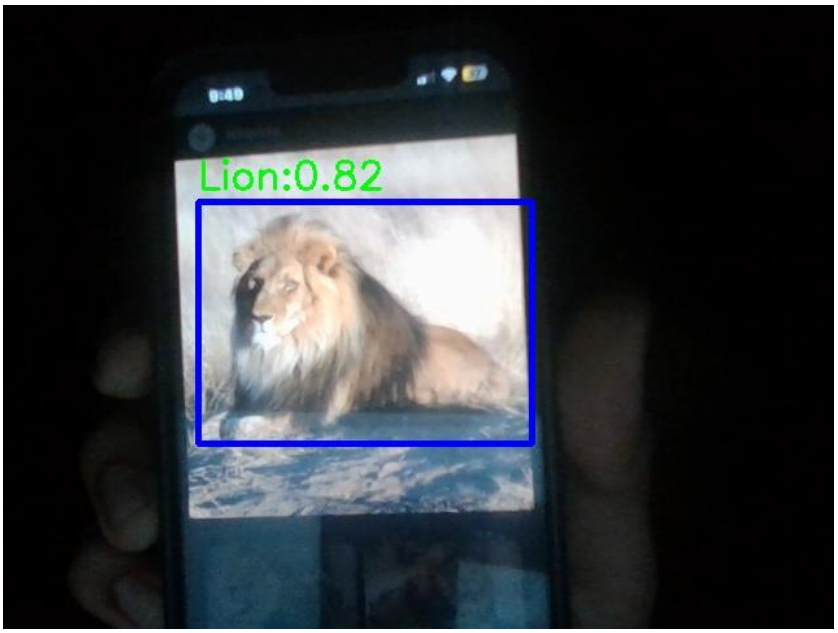

Fig 8.10- Detection of lion

Detected Bear with a confidence score of 0.92.

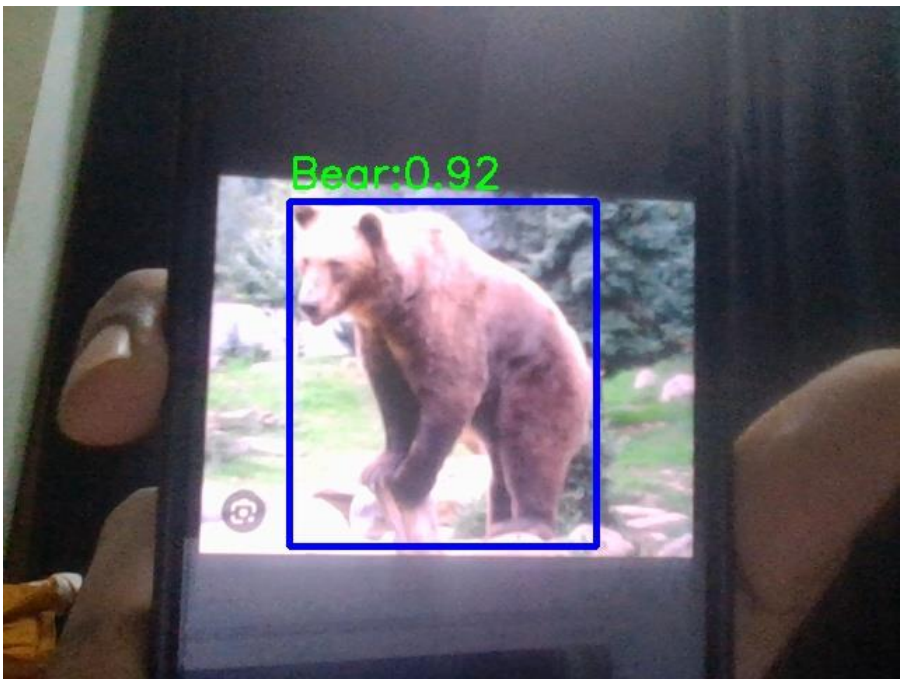

Fig 8.11 - Detection of bear

Detected Wolf with a confidence score of 0.91.

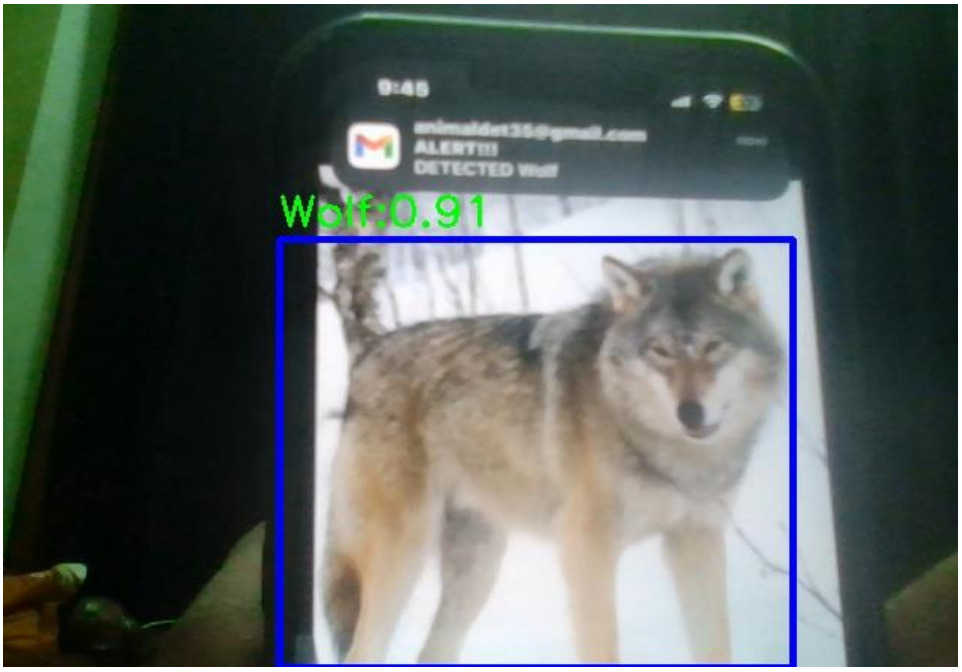

Fig 8.12 - Detection of wolf

Detected Tiger with a confidence score of 0.85.

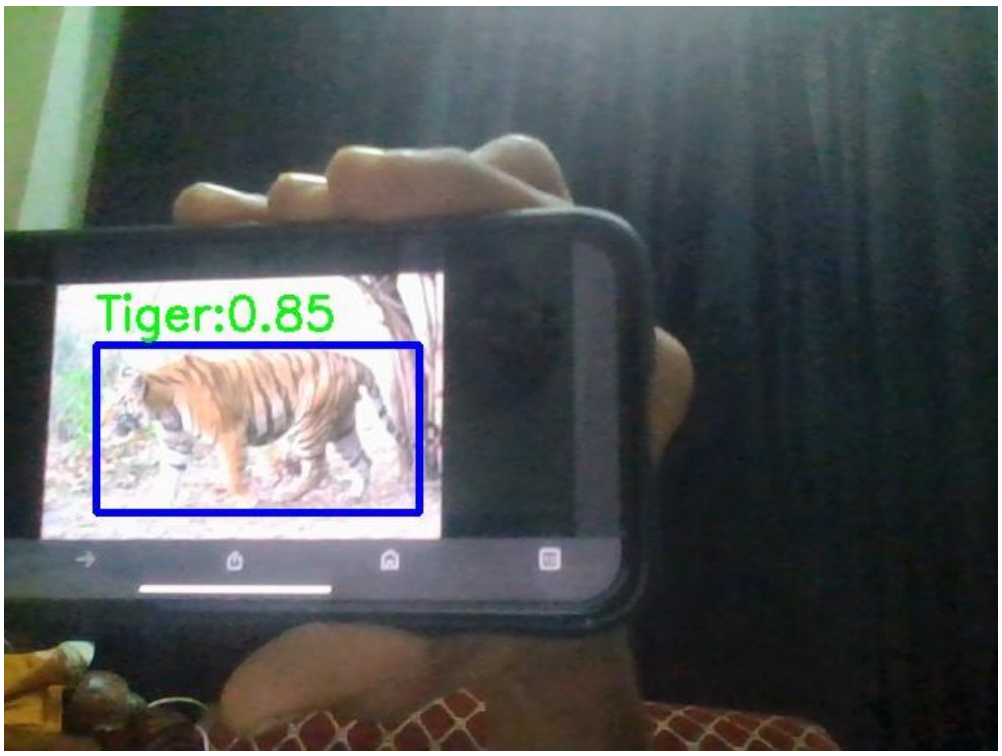

Fig 8.13 - Detection of tiger

Detected Boar with a confidence score of 0.83.

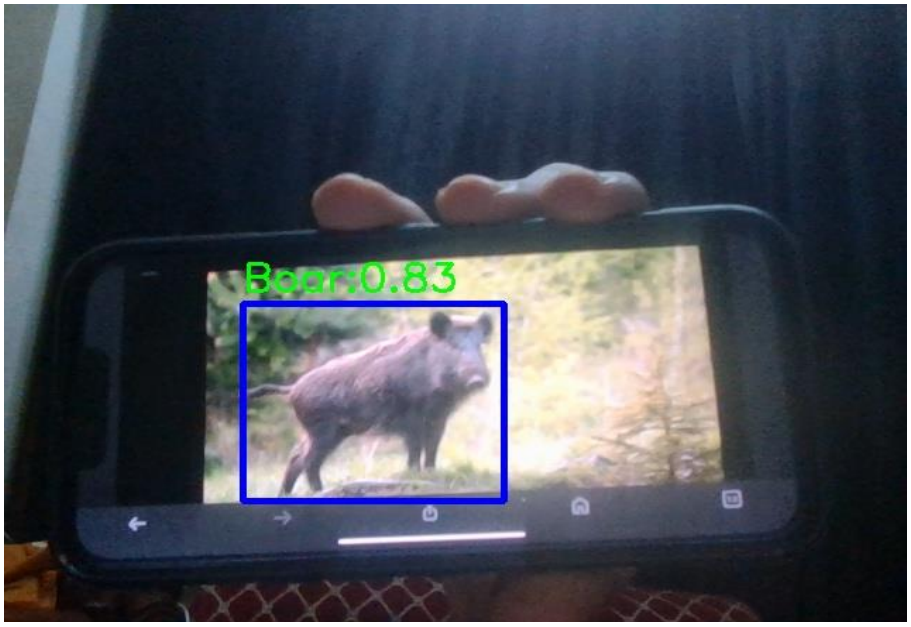

Fig 8.14 - Detection of boar

8.6 DETECTION FROM NIGHT FOOTAGE

Detected Night vision footage of Leopard with a confidence score of 0.81.

Fig 8.15 - Detection of leopard

Detected Night vision footage of Elephant with a confidence score of 0.81.

Fig 8.16 - Detection of elephant

8.7 NOTIFICATION ALERT

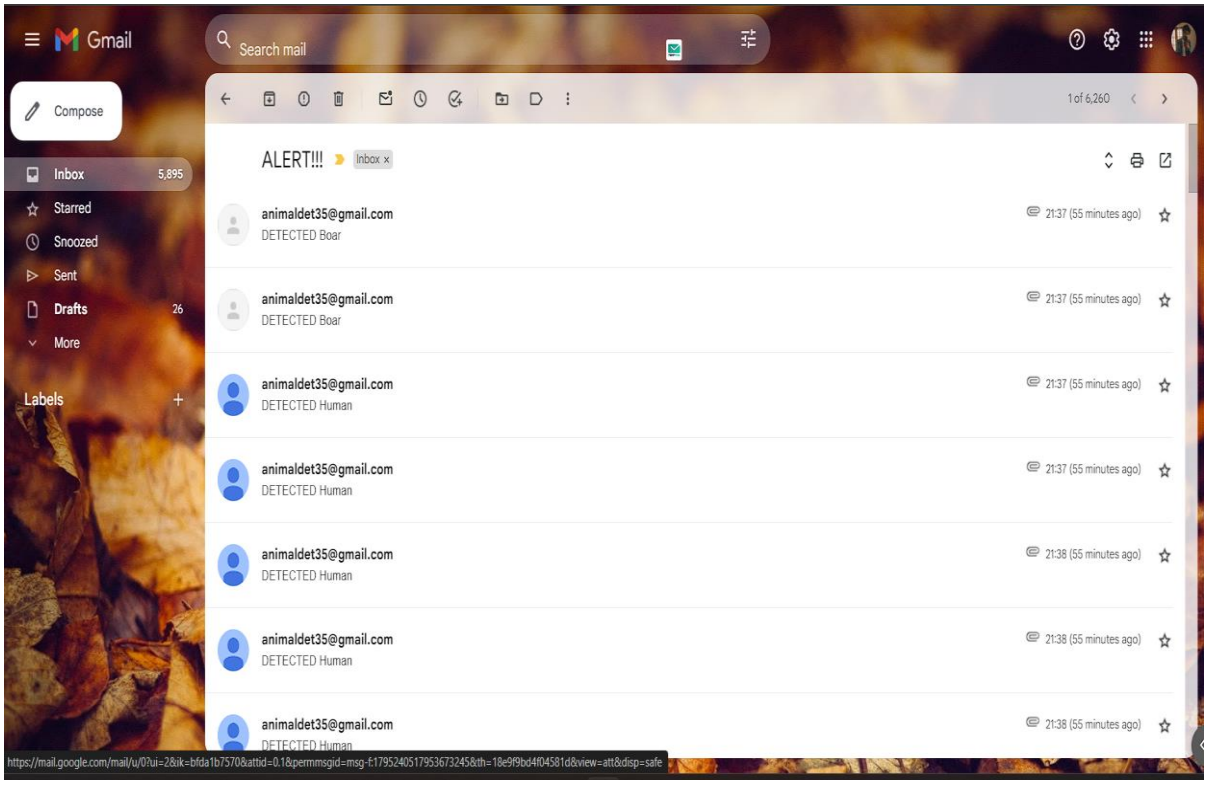

Fig 8.17 - Detected animals snapshots sent via email

Live footage from the CCTV camera is analysed and the detected animals snapshots are taken and sent to an authorized email id in real time.

8.8 NOTIFICATION VIA SMS

Real time alert being sent to a phone via sms using Twilio.

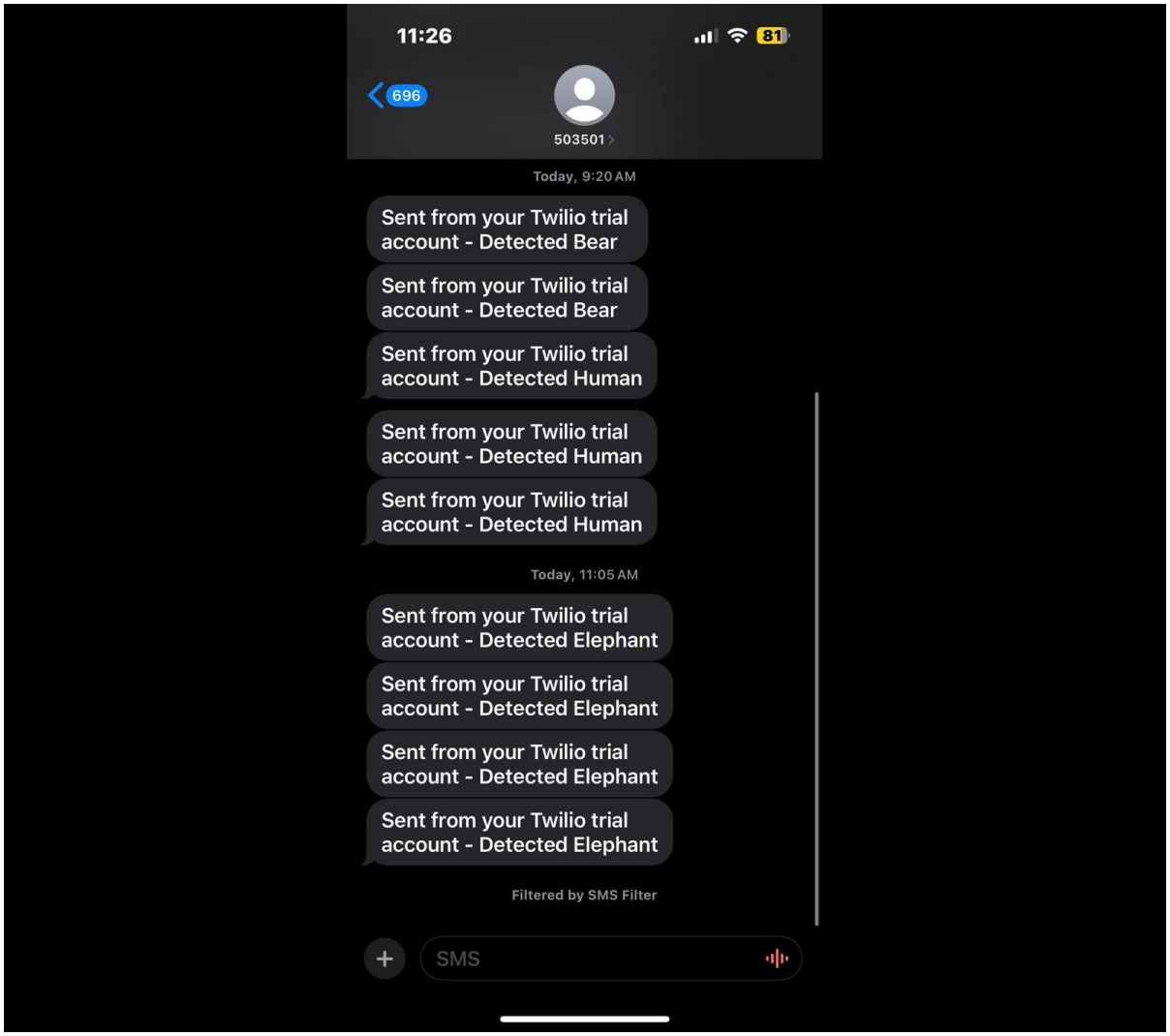

Fig 8.18 - Notification via sms

CHAPTER 9

CONCLUSION AND FUTURE SCOPE

The animal detection machine learning model powered by YOLOv8 has been remarkable in identifying eight distinct animal species. Through its advanced object detection capabilities, it has provided a reliable and efficient solution for wildlife monitoring and conservation efforts. By accurately identifying animals in diverse environments and conditions, this model offers valuable insights into population dynamics, habitat utilization, and ecosystem health. Its successful deployment underscores the potential of machine learning in aiding biodiversity research and management initiatives. As we continue to refine and expand upon this technology, we move closer to achieving more effective and sustainable practices in wildlife conservation and environmental stewardship.

9.1 FUTURE SCOPE

There is a lot of room for growth in this project like improving the detecting techniques that the AI model uses is one way to advance. In order to increase the precision and speed of animal identification, this may include investigating cutting-edge computer vision techniques like deep learning architectures like YOLO or Faster R-CNN. Furthermore, adding infrared or thermal imaging sensors could improve detection performance, particularly in dimly lit areas. Another possible avenue for the project is to expand into real-time tracking and monitoring of animals that are spotted. Creating algorithms to monitor animal movement patterns over time could be one way to do this supporting threat analysis and behavioral analysis. It's also worthwhile to think about working with groups dedicated to wildlife protection to incorporate the system into larger conservation initiatives. The project has the potential to make a substantial contribution to wildlife conservation efforts by creating tools for gathering data on animal numbers and habitat utilization. Moreover, improving accessibility and community involvement would involve developing a smartphone application that lets users report animal sightings, view live feeds, and receive notifications. Future research options include

investigating predictive analytics to evaluate the likelihood of encounters with wildlife and integrating with IoT devices to improve monitoring and reaction capabilities. All things considered, the initiative has the potential to develop into an effective wildlife management system that supports both human safety and wildlife conservation with continuing innovation and cooperation.

REFERENCES

[1]Mai Ibraheam; Kin Fun Li; Fayez Gebali, *" An Accurate and Fast Animal Species Detection System for Embedded Devices ",* IEEE Access, 13 March 2023.

[2]J. K. Vishwas1; S. M. Prajwal Raj ; Bhagya, M. Anand ,*"CNN Based Animals Recognition using Advanced YOLO V5 and Darknet",International Journal of Research in Engineering, Science and Management*, 6 June 2022 .

[3]Ms.V.Megala ; Dr.S.K.Jayanthi,*"An object detection method using the modified YOLO V4 algorithm- An abnormal activity detection in roads and railways".*

APPENDIX

```
from ultralytics import YOLO
import cv2
import math
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.image import MIMEImage
import torch
import threading
from twilio.rest import Client

print(f'PyTorch version: {torch.__version__}')
print('*'*10)
print(f'_CUDA version: ')
print('*'*10)
print(f'CUDNN version: {torch.backends.cudnn.version()}')
print(f'Available GPU devices: {torch.cuda.device_count()}')
print(f'Device Name: {torch.cuda.get_device_name()}')

model = YOLO("best.pt")
model.to('cuda')
classNames = {0:'Lion',1:'Tiger',2:'Boar',3:'Elephant',4:'Human',5:'Dog',6:'Wolf',7:'Bear',8:'Leopard'}

cap = cv2.VideoCapture(0)
cap.set(3, 640)
cap.set(4, 480)

counter = 0  # Counter for naming the saved images

def send_emails(email_list, image_filenames, cls):
    smtp_port = 587                 # Standard secure SMTP port
    smtp_server = "smtp.gmail.com"  # Google SMTP Server
    email_from = "animaldet35@gmail.com"
    pswd = "bwnx wkjw cpyr rkdd"
    subject = "ALERT!!!"

    for person in email_list:
        try:
            #Body of the email
            body = f"DETECTED {classNames[cls]}"

            # make a MIME object to define parts of the email
            msg = MIMEMultipart()
            msg['From'] = email_from
            msg['To'] = person
            msg['Subject'] = subject

            # Attach the body of the message
            msg.attach(MIMEText(body, 'plain'))
```

```
            # Attach the detected images
            for filename in image_filenames:
                with open(filename, 'rb') as f:
                    img_data = f.read()
                    image = MIMEImage(img_data, name=filename)
                    msg.attach(image)

            # Cast as string
            text = msg.as_string()

            # Connect with the server and send email
            with smtplib.SMTP(smtp_server, smtp_port) as TIE_server:
                TIE_server.starttls()
                TIE_server.login(email_from, pswd)
                TIE_server.sendmail(email_from, person, text)

            print(f"Email sent to: {person}")
        except Exception as e:
            print(f"Error occurred while sending email: {e}")

def send_sms(message_body, twilio_phone_number, recipient_phone_number, account_sid,
auth_token):
    try:
        client = Client(account_sid, auth_token)
        message = client.messages.create(
            body=message_body,
            from_=twilio_phone_number,
            to=recipient_phone_number
        )
        print("SMS sent successfully. SID:", message.sid)
    except Exception as e:
        print(f"Error occurred while sending SMS: {e}")

def detect_objects_and_notify():
    global counter
    while True:
        success, img = cap.read()

        results = model(img, stream=True, verbose=False)
        detected_image_filenames = []

        for r in results:
            boxes = r.boxes
            for box in boxes:
                confidence = math.ceil((box.conf[0] * 100)) / 100
                if confidence > 0.8:
                    cls = int(box.cls[0])
                    if cls in classNames:
                        x1, y1, x2, y2 = box.xyxy[0]
                        x1, y1, x2, y2 = int(x1), int(y1), int(x2), int(y2)  # convert to int values
                        cv2.rectangle(img, (x1, y1), (x2, y2), (255, 0, 0), 3)
                        org = (x1, y1 - 10)
```

```
                font = cv2.FONT_HERSHEY_SIMPLEX
                fontScale = 1
                color = (0, 255, 0)
                thickness = 2
                text = f"{classNames[cls]}:{confidence}"
                cv2.putText(img, text, org, font, fontScale, color, thickness)

                # Extract and save the detected object
                cropped_img = img#[y1:y2, x1:x2]
                filename = f'detected_{counter}.jpg'
                cv2.imwrite(filename, cropped_img)
                detected_image_filenames.append(filename)

    # Email
    if counter % 10 == 0 and detected_image_filenames:
        threading.Thread(target=send_emails, args=(["adarshv8045@gmail.com"],
detected_image_filenames, cls)).start()

    # SMS
    #if counter % 20 == 0 and detected_image_filenames:
        #threading.Thread(target=send_sms, args=(f"Detected {classNames[cls]}", "+14086693482",
"+917306058764", "ACf3a9b3b8735c034418ae35f5e4ee7683",
                                #"3f96072d1db80fcfb9c34164767533d5")).start()

    cv2.imshow('Result', img)
    if cv2.waitKey(1) == ord('q'):
        break

    counter += 1

  cap.release()
  cv2.destroyAllWindows()

# Call the function to start object detection and notification
detect_objects_and_notify()
```

Final report

ORIGINALITY REPORT

23%	18%	11%	12%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.researchgate.net Internet Source	3%
2	eurchembull.com Internet Source	2%
3	www.coursehero.com Internet Source	2%
4	rgu-repository.worktribe.com Internet Source	1%
5	Saad Mboutayeb, Aicha Majda, Khalid Zenkouar, Nikola S. Nikolov. "FCOSH: A novel single-head FCOS for faster object detection in autonomous-driving systems", Intelligent Systems with Applications, 2024 Publication	1%
6	nl.overleaf.com Internet Source	1%
7	www.mdpi.com Internet Source	1%
8	Submitted to Coventry University Student Paper	

		1 %
9	Ming Li, Li Zhang. "Deep Learning-based License Plate Recognition in IoT Smart Parking Systems using YOLOv6 Algorithm", International Journal of Advanced Computer Science and Applications, 2023 Publication	1 %
10	Submitted to Liverpool John Moores University Student Paper	1 %
11	Submitted to Mar Baselios Engineering College Student Paper	1 %
12	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	1 %
13	acikerisim.aydin.edu.tr Internet Source	1 %
14	pt.slideshare.net Internet Source	<1 %
15	Deming Yang, Ling Yang. "A Deep Learning-based Framework for Vehicle License Plate Detection", International Journal of Advanced Computer Science and Applications, 2024 Publication	<1 %

16	dipankarmedh1.medium.com Internet Source	<1%
17	Submitted to Sheffield Hallam University Student Paper	<1%
18	scientifictrends.org Internet Source	<1%
19	Submitted to The University of Wolverhampton Student Paper	<1%
20	Submitted to Federal University of Technology Student Paper	<1%
21	Submitted to Lebanese International University Student Paper	<1%
22	Submitted to Hong Kong University of Science and Technology Student Paper	<1%
23	Submitted to Instituto Tecnológico y de Estudios Superiores de Occidente Student Paper	<1%
24	Submitted to Riga Technical University Student Paper	<1%
25	medium.com Internet Source	<1%

26	Submitted to The Scientific & Technological Research Council of Turkey (TUBITAK) Student Paper	<1 %
27	discuss.python.org Internet Source	<1 %
28	Swethaa Prabhu, MV Sreenath, V Malavika, Himani Om, S Swetha. "Detection and Recognition of Animals Using Yolo Algorithm", 2023 International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE), 2023 Publication	<1 %
29	Submitted to University of Glasgow Student Paper	<1 %
30	arxiv.org Internet Source	<1 %
31	unsworks.unsw.edu.au Internet Source	<1 %
32	Submitted to Oxford Brookes University Student Paper	<1 %
33	Submitted to Middle East Technical University Student Paper	<1 %
34	fastercapital.com Internet Source	<1 %
35	Submitted to University of Hertfordshire Student Paper	

		<1%
36	dspace.library.uvic.ca Internet Source	<1%
37	journal.ijresm.com Internet Source	<1%
38	kylo.tv Internet Source	<1%
39	www2.mdpi.com Internet Source	<1%
40	Submitted to Brunel University Student Paper	<1%
41	"Applications of Computer Vision and Drone Technology in Agriculture 4.0", Springer Science and Business Media LLC, 2024 Publication	<1%
42	Submitted to British University in Egypt Student Paper	<1%
43	Submitted to University of Newcastle upon Tyne Student Paper	<1%
44	Submitted to Arts, Sciences & Technology University In Lebanon Student Paper	<1%

45	Submitted to Higher Education Commission Pakistan Student Paper	<1%
46	Submitted to International Islamic University Malaysia Student Paper	<1%
47	Submitted to Malayan Colleges Mindanao, A Mapua School Student Paper	<1%
48	Submitted to Southampton Solent University Student Paper	<1%
49	Submitted to Southern New Hampshire University - Continuing Education Student Paper	<1%
50	eescholars.iitm.ac.in Internet Source	<1%
51	Submitted to Swinburne University of Technology Student Paper	<1%
52	Submitted to University of Hong Kong Student Paper	<1%
53	Submitted to Yonsei University Student Paper	<1%
54	rosap.ntl.bts.gov Internet Source	<1%

55	Bharani Nammi, Sita Sirisha Madugula, Pranav Pujar, Vindi Mahesha Jayasinghe Arachchige, Jin Liu, Shouyi Wang. "Transformer-Based Deep Learning Model with Latent Space Regularization for CRISPR-Cas Protein Sequence Classification", Cold Spring Harbor Laboratory, 2024 Publication	<1 %
56	Submitted to National School of Business Management NSBM, Sri Lanka Student Paper	<1 %
57	Submitted to Vardhaman College of Engineering, Hyderabad Student Paper	<1 %
58	pdfcoffee.com Internet Source	<1 %
59	www.cscjournals.org Internet Source	<1 %
60	www.science.org Internet Source	<1 %
61	Submitted to Dhirubhai Ambani International School Student Paper	<1 %
62	Submitted to Sri Sairam Engineering College Student Paper	<1 %
63	Submitted to University of Glamorgan	

	Student Paper	<1%
64	bmcbioinformatics.biomedcentral.com Internet Source	<1%
65	d197for5662m48.cloudfront.net Internet Source	<1%
66	ds.libol.fpt.edu.vn Internet Source	<1%
67	www.geeksforgeeks.org Internet Source	<1%
68	www.ijraset.com Internet Source	<1%
69	www.packtpub.com Internet Source	<1%
70	Madallah Alruwaili, Muhammad Hameed Siddiqi, Muhammad Nouman Atta, Mohammad Arif. "Deep learning and ubiquitous systems for disabled people detection using YOLO models", Computers in Human Behavior, 2024 Publication	<1%
71	Submitted to University of Northampton Student Paper	<1%
72	Vedanshu Dewangan, Aditya Saxena, Rahul Thakur, Shrivishal Tripathi. "Application of	<1%

Image Processing Techniques for UAV Detection Using Deep Learning and Distance-Wise Analysis", Drones, 2023

Publication

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		