REPUBLIC OF TÜRKİYE

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Electrical And Computer Engineering

DEVELOPMENT AND IMPLEMENTATION OF YOLOV8-BASED MODEL FOR HUMAN AND ANIMAL DETECTION DURING FOREST FIRES

Mustafa Qays Fadhil ALSAMURAI

Master's Thesis

Supervisor

Asst. Prof. Dr. Mesut ÇEVİK

Istanbul, 2023

DEVELOPMENT AND IMPLEMENTATION OF YOLOV8-BASED MODEL FOR HUMAN AND ANIMAL DETECTION DURING FOREST FIRES

Mustafa Qays Fadhil ALSAMURAI

Electrical And Computer Engineering

Master's Thesis

ALTINBAŞ UNIVERSITY

2023

The thesis titled DEVELOPMENT AND IMPLEMENTATION OF YOLOV8 - BASED MODEL FOR HUMAN AND ANIMAL DETECTION DURING FOREST FIRES by Mustafa Qays Fadhil ALSAMURAI and submitted on 25/04/2023 has been **accepted by unanimously** for the degree of Master of Science in Electrical and Computer Engineering.

_____

Asst. Prof. Dr. Mesut ÇEVİK
Supervisor

Thesis Defence Committee Members:

Asst. Prof. Dr. Mesut ÇEVİK	Department of Electrical and Electronics Engineering, Altınbaş University	_____
Asst. Prof. Dr. Timur İNAN	Department of Software Engineering, Altınbaş University	_____
Assoc. Prof. Dr. Abdurrahim AKGÜNDOĞDU	Department of Electrical and Electronics Engineering, İstanbul University-Cerrahpaşa	_____

I hereby declare that this thesis meets all format and submission requirements of a Master's Thesis.

Submission date of the thesis to Institute of Graduate Studies: ___/___/___

I hereby declare that all information/data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Mustafa Qays Fadhil ALSAMURAI

Signature

DEDICATION

I dedicate and pledge this research work to my Dear supervisor and my beloved Father, mother, and wife, whose unwavering support, encouragement, and love have been my anchor throughout this journey. I am forever grateful for their sacrifices and belief in me.

ABSTRACT

DEVELOPMENT AND IMPLEMENTATION OF YOLOV8-BASED MODEL FOR HUMAN AND ANIMAL DETECTION DURING FOREST FIRES

ALSAMURAI, Mustafa Qays Fadhil

M.Sc., Electrical And Computer Engineering, Altınbaş University,

Supervisor: Asst. Prof. Dr. Mesut ÇEVİK

Date: April / 2023

Pages: 74

For the biosphere to be protected, forests are a necessity everywhere in the world. Forest fires are one of the major risks to life in many parts of the world; they put the environment, including humans, plants, animals, and even land in danger. The North African & Mediterranean regions, Amazon & Australia last year suffered greatly from forest fires. To save lives and property, forest fires must be discovered sooner rather than later. This study aims to detect fire, smoke, humans, and animals in outdoor images using the YOLOv8 deep learning algorithm. A custom dataset of outdoor images was created by obtaining images from various search engines and manually annotating them. The YOLOv8 model was trained on this dataset and achieved an overall mAP of 0.274, with varying performance for different object classes. The model had difficulty in detecting small instances of fire and smoke, and struggled to differentiate between animals and humans in certain cases. The study also identified the importance of image quality in computer vision and highlighted the impact of poor image quality on model performance. Overall, the study presents a comprehensive evaluation of YOLOv8's performance in detecting outdoor objects and identifies areas for improvement.

Keywords: Yolo, Deep Learning, CNN, Object Detection, Computer Vision.

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

ABBREVIATIONS

YOLOv8	:	You Only Look Once version 8
IoT	:	Internet of Things
PAN	:	Path Aggregation Network
VGG	:	Visual Geometry Group
FPN	:	Feature Pyramid Network
CNN	:	Convolutional Neural Network
FPS	:	Frames per second
NMS	:	Non-Maximum Suppression
BCE	:	Binary Cross Entropy
SGD	:	Stochastic Gradient Descent
BGR	:	Blue, Green, Red color space
ROI	:	Region of Interest
GPU	:	Graphics Processing Unit
VOC	:	Visual Object Classes
COCO	:	Common Objects in Context dataset
mAP	:	Mean Average Precision
F1 score	:	Harmonic mean of precision and recall

1. INTRODUCTION

1.1 OVERVIEW OF THE STUDY

The world's forests make one of the biggest contributions to the environment. They support a wide variety of plant and animal species and serve a critical role in controlling the climate of the entire globe. Numerous benefits of woods for humans include timber, water supply, and recreation [1]. Forest fire and destruction can all have devastating effects on the environment, such as species extinction, habitat destruction, and increased emissions of greenhouse gases. There are numerous places in the globe where forest fires are a significant problem. They have the capacity to damage human lives as well as the environment and animals. One of the main causes of environmental hazards around the world in recent years has been forest fires. As of 2021, thousands of fires had been reported, consuming almost a hundred thousand hectares of land. The worst wildfires in decades had ravaged these countries, killing dozens of people and animals while also causing significant economic harm. These countries included Turkey, Italy, Greece, Algeria, and Morocco. According to historical estimations, from 2001 to 2018, the annual global forest burnt areas were predicted to have been between 394 and 519 million hectares, on average [2]. The flames have resulted in significant property damage, human casualties, and environmental catastrophes. There are numerous forest fires around the world, and they can have detrimental effects on the ecology. Fires can spread quickly in hot, dry circumstances, burning trees, homes, and other structures [3].

The world’s largest forest fires are the Siberian Taiga Fires (Russia-2003), the Australian Bushfire , the Northwest Territories Fires (Canada-2014), the Black Friday Bushfire , The Great Fire (Canada -1919), the Chinchaga Fire (Canada-1950), Bolivia Forest Fires , Great Fire of Connecticut (US-1910), Black Dragon Fire (China and Russia-1987), Richardson Backcountry Fire (Canada- 2011), Manitoba Wildfires (Canada - 1989) and the recent Amazon & Australia forest fires caused enormous ecosystem damage, worsened air pollution, and killed millions of people and animals .The smoke and heat from a fire can also be damaging to humans and animals. Animal and human detection in forest fires is an important responsibility for emergency personnel because it helps them quickly discover and rescue individuals and animals that may be in danger. There has been an increasing interest

in employing deep learning techniques to automate this procedure in recent years . Using massive datasets, deep learning is a type of machine learning that teaches neural networks to identify patterns and predict the future. Deep learning models can be constructed on a collection of forest fire photographs and videos with the goal of automatically detecting and locating animals and humans in new images and videos acquired during an actual fire [4].

Figure 1.1: Simulated Smoke From A Forest Fire Captured In The Middle Of The Day In A Clearing On The Outskirts Of A City.

When it comes to detecting humans, Yolo can identify the presence of people in the area affected by a forest fire, even in situations where visibility is reduced due to smoke and other factors. This can help firefighters and emergency responders locate people who may be in danger and evacuate them to safety. Additionally, the algorithm can be used to monitor the

movements of firefighters and ensure that they are not in harm's way. For animals, Yolo can identify the presence of wildlife in the affected area. This is important because many animals are not able to escape the fire and may be in danger of injury or death. By identifying where animals are located, responders can take steps to protect them and ensure that they are not trapped or injured by the fire. Yolo can detect a variety of animal species, including birds, mammals, and reptiles. This information can be used to determine the types of animals that are in the area and their behaviors during the fire. For example, if Yolo detects a group of animals moving in a certain direction, responders can use this information to anticipate their movements and take steps to prevent them from being trapped. Training a deep learning model to recognize animals and humans in forest fires often entails a number of steps. The first step is to compile a sizable collection of images and videos of forest fires that have been annotated with the locations of people and animals. The model is then trained using supervised learning on this dataset, which entails showing the machine examples of animals and people in pictures and videos so that it can learn to identify these patterns. Once the model has been trained, it can be used to automatically detect animals and humans in new images and videos captured during a forest fire. This can be done in real-time, allowing emergency responders to quickly locate and rescue any individuals or animals that may be in danger. Additionally, the model can be fine-tuned and updated with new data as needed to improve its performance over time [5].

Overall, the application of deep learning for animal & human detection in forest fires is a promising area of research with the potential to greatly aid emergency responders in their efforts to rescue individuals and animals in danger. In order to detect the fire scenario, an Unmanned Aerial Vehicle (UAV) characterized by deep learning and the You Only Look Once (YOLO) method is utilized, and the results provided in the true challenge were more accurate than conventional approaches. In comparison to UAVs, camera detection can detect all day, seven days a week, which can help with forest fire prevention. The fire can be identified early on by evaluating the visual information obtained from the monitor, preventing the fire from starting [6].

The early detection and containment of forest fires is crucial for protecting both humans and animals. An IoT-based fire detection system can help to achieve this by providing early fire notifications and gathering data on the location and spread of the fire. This can aid the fire

service in locating the fire and controlling it before it spreads to important areas, such as human settlements and wildlife habitats . In particular, an automated early warning system for forest fires can be beneficial for the protection of animals. By monitoring unusual weather patterns that can cause wildfires, such a system can help to detect and prevent fires before they pose a threat to wildlife. This can be especially important in areas that are home to endangered species, where even a small fire can have devastating consequences. For humans, an early warning system can help to ensure that people are safely evacuated in the event of a forest fire. By providing early notifications, responders can evacuate people before the fire reaches their area, reducing the risk of injury or death. Overall, an IoT-based fire detection system can be a valuable tool for protecting both humans and animals from the destructive effects of forest fires. By providing early warnings and gathering data on the location and spread of the fire, responders can take quick and effective action to contain the fire and protect those in its path [7].

Any computer vision enthusiast is familiar with YOLO models for object detection either it is human, animals & fire smoke. The computer vision community has taken a big interest in YOLOv1 when it was first introduced in 2015. Since then, other individuals have released various versions of YOLOv2, YOLOv3, YOLOv4, and YOLOv5. YOLOv6 is faster and more accurate than YOLOv5, has a more hardware-efficient design, and has higher Recognizing accuracy. It is therefore the most appropriate YOLO algorithm OS version for production. The YOLOv7 model is built on the YOLOv4, Scaled YOLOv4, and YOLO-R architectures from earlier iterations. The YOLOv7 E-ELAN architecture enables the model to train more successfully while keeping the original gradient route through the usage of "expand, shuffle, merge cardinality." YOLOv7 is not constrained to a particular head. Assisting with training in the intermediary layers is the responsibility of the auxiliary head, whilst the lead head is in charge of producing the final product. These procedures are used to put together YOLOv8. In order to increase prediction accuracy, more convolutional layers are used in YOLOv8's more sophisticated Anchor-free identification of live things (human and animal) during the fire [8].

The proposed YOLOv8 has the following feature & pre-trained models

a. A user-friendly Python and Command Line API.
b. Quicker and more precise.

c. Supports picture categorization, instance segmentation, and object detection.
d. Compatible with all earlier versions.
e. A new backbone system.
f. New Head with Anchor-Free.
g. New Loss Feature.
h. Object detection checkpoints trained on the 640-picture resolution COCO recognition dataset.
i. Checkpoints, such as segmentation trained on the 640-picture COCO segmentation dataset.
j. The ImageNet dataset, with a resolution of 224 pixels, was used to train image recognition models.
k. Convolutional layers are used to raise the accuracy of predictions.

1.2 YOLO OBJECT DETECTOR MODELS' BASIC OPERATION

Accuracy and recall are critical for determining and assessing the correctness and robustness of any ML-based model. The creator of YOLO continued to work on creating an object detection model that optimizes the map as a consequence (average mean accuracy) [9].

i. Recall measures the proportion of real positive results to all positive forecasts (whether it is right or wrong).
ii. The proportion of positive instances to actual positives is what is known as precision (total correct projection).
iii. The accuracy (or mean average precision) is the total of all average precisions (map). Apart from that, all YOLO models share a set of components that make up their architecture. as outlined below.

Backbone: A convolutional neural network that generates and detects visual features of varying sizes and forms. As feature extractors, ResNet, VGG, and Efficient Net classification models are used [10].

Neck: A group of layers that bring together and integrate properties before passing them to the prediction layer. They consist of FPN, PAN, and Bi-FPN.

Head: considers both bounding box predictions and neck features. Use classification and regression to complete the detection process on the features and bounding box coordinates. produces four values: width, height, and the x and y coordinates.

1.3 THE PURPOSE AND BENEFITS OF DRONE DETECTION DURING FOREST FIRES

Drone detection during forest fires can provide a range of benefits for responders and help in mitigating the impact of the fire. Drones are increasingly being used for early detection and monitoring of forest fires, as well as for situational awareness during firefighting operations. Here are some of the key purposes and benefits of drone detection during forest fires:

a. Early detection: Drones can quickly and efficiently survey large areas of land, including remote and difficult-to-access locations, to detect smoke or flames. This early detection can provide firefighters with timely information about the location and severity of the fire, allowing them to respond quickly and effectively [11].
b. Mapping and monitoring: Drones can provide real-time video and thermal imaging of the fire, allowing responders to track the spread of the fire and monitor hot spots. This data can help responders to develop effective strategies for containing and extinguishing the fire [11].
c. Safety: Drones can be used to assess and monitor the safety of firefighters during firefighting operations. They can identify potential hazards, such as falling trees or unstable terrain, and provide situational awareness for responders to ensure their safety.
d. Efficiency: Drones can help responders to manage resources more efficiently by identifying areas that require priority attention and providing real-time information on the effectiveness of firefighting efforts [12].
e. Cost-effectiveness: Using drones for fire detection and monitoring can be more cost-effective than traditional methods, such as manned aircraft or ground-based patrols.

Forest fires permanently alter the ecology and put locals' lives in jeopardy National Interagency Fire Centre estimates that there were twice as many fires in the US in 2015 as there were in 1990. (NIFC). CNN reports that recent wildfires in Australia and the Amazon Rainforest have killed over 480 million animals, forced over 200,00 individuals to evacuate, and resulted in the deaths of over 40 persons and the disappearance of over 50 more. Forest fires are anticipated to break out more than 220,000 times annually, consuming an estimated 6 million hectares of land. As a result, it's crucial to quickly and independently identify forest fires. This motivates us to develop a computer-advance warning system that combines

different sensors with cutting-edge machine learning algorithms in order to provide our drone with low wrongful convictions and good precision in real-time data at the lowest possible cost. Object detection assists drones in determining the location and type of things in their field of view and is a critical requirement for drones used in maritime cruises, forest and search and every type of rescue missions. Various object identification methods and application scenarios have been investigated over the last two decades. Our drone will monitor human, animals during forest fires as early as possible and report them to the appropriate authorities [12]. The drone forestry surveillance approach reveals significant local ecological damage and should be used to establish conservation, rehabilitation, and approach by involving in an operative deforestation surveillance system.

Overall, the purpose of drone detection during forest fires is to provide timely and accurate information to responders, to help them manage the fire more effectively and protect human life and property. By using drones for early detection, mapping and monitoring, safety, efficiency, and cost-effectiveness, responders can gain a more comprehensive understanding of the fire, and take the necessary steps to control and extinguish it [13].

1.4 PROBLEM STATEMENT

Develop a real-time monitoring system that can accurately recognize and locate animals and humans in forest fire conditions to help rescue and evacuation efforts and improve overall safety," might be the issue statement for animal and human detection in forest fires. In a fast-paced and chaotic setting, the system would need to recognize and distinguish between various sorts of animals and humans, as well as provide precise location information to aid in rescue efforts. Furthermore, the system should be able to operate in a resource-constrained environment, such as a mobile device or drone, as well as deal with changing lighting and visibility conditions.

1.5 PROBLEM SOLUTION

The problem of detecting animals and humans in forest fires is a critical one that can have life-saving implications. When a forest fire occurs, the presence of animals and humans in the affected area can greatly complicate emergency response efforts, and the inability to quickly and accurately locate these individuals can result in injuries or fatalities. One

potential solution to this problem is the use of YOLOv8, a computer vision algorithm that can quickly and accurately detect objects in real-time images or video footage. By training YOLOv8 on a dataset of images that includes examples of animals and humans in the context of forest fires, the algorithm can identify the presence of these objects with a high degree of accuracy, even in challenging conditions. Implementing a YOLOv8-based solution for detecting animals and humans in forest fires could have several benefits. For example, it could enable emergency responders to quickly and accurately locate individuals who may be in danger, inform evacuation decisions, and aid in search and rescue efforts. This could potentially save lives and reduce the risk of injury or harm to both humans and animals. However, there are also several challenges and limitations to consider when implementing a YOLOv8-based solution for detecting animals and humans in forest fires. These may include the need for high-quality data, advanced computing resources, and the potential for false positives or false negatives. Additionally, the use of YOLOv8 must be integrated into a broader emergency response system that considers other factors such as weather conditions, fire behaviour, and the availability of resources.

1.6 OBJECTIVES

Here are possible objectives related to the detection of animals and humans in forest fires using YOLOv8:

i. Develop and train a YOLOv8-based object detection model that can accurately detect animals and humans in images or video footage of forest fires. This objective would involve collecting and annotating a dataset of images or video clips that include examples of animals and humans in the context of forest fires, and using this data to train a YOLOv8 model that can accurately identify these objects in new images or video.
ii. Evaluate the performance of the YOLOv8-based object detection model in real-world scenarios. This objective would involve testing the YOLOv8 model in a variety of conditions that simulate real-world forest fire scenarios, such as varying weather conditions, lighting, and levels of smoke and ash. The performance of the model could be measured in terms of accuracy, speed, and robustness to different conditions.
iii. Integrate the YOLOv8-based object detection model into a broader emergency response system for forest fires. This objective would involve working with emergency responders and other stakeholders to develop a system for using the YOLOv8 model to detect

animals and humans in forest fires, and integrating this system with other tools and resources that are used in emergency response. This could include developing protocols for sharing data between different agencies, creating user-friendly interfaces for emergency responders, and ensuring that the system is scalable and adaptable to different contexts.

iv. Explore the ethical and social implications of using YOLOv8 for detecting animals and humans in forest fires. This objective would involve examining the potential social and ethical implications of using YOLOv8 for detecting animals and humans in forest fires, such as privacy concerns, biases in the training data, and the potential for false positives or false negatives. The objective would be to identify ways to mitigate these risks and ensure that the use of YOLOv8 for detecting animals and humans in forest fires is ethically and socially responsible.

1.7 RESEARCH QUESTION

A possible research question related to the detection of animals and humans in forest fires using YOLOv8 could be:

a. How effective is YOLOv8-based object detection in detecting and localizing animals and humans in images or video footage of forest fires, and what are the main factors that affect the performance of the system in real-world scenarios?
b. What are the ethical and social implications of using YOLOv8 for detecting animals and humans in forest fires, and how can these risks be mitigated?
c. What are the most effective strategies for integrating a YOLOv8-based object detection system into a broader emergency response system for forest fires, and what are the key challenges and opportunities associated with this integration?
d. How does the performance of YOLOv8 compare to other object detection models in the context of forest fire detection, and what are the trade-offs between different models in terms of accuracy, speed, and other factors?

1.8 THESIS STRUCTURE

Thesis structure may include the following sections:

i. Title page: The title page includes the title of the thesis, the author's name, and the date of submission.
ii. Abstract: The abstract is a brief summary of the thesis that outlines the main research question, objectives, methods, findings, and conclusions.
iii. Table of contents: The table of contents provides an overview of the main sections and subsections of the thesis.
iv. Introduction: The introduction provides background information on the topic of the thesis, outlines the research problem and questions, and provides a rationale for the study. It also includes an overview of the structure of the thesis.
v. Literature review: The literature review summarizes and analyzes previous research and scholarship related to the research problem and questions. It provides a critical analysis of the strengths and weaknesses of previous studies and identifies gaps in the literature that the current study aims to address.
vi. Methodology: The methodology outlines the research design, data collection methods, and data analysis methods used in the study. It also discusses any limitations and ethical considerations.
vii. Results: The results section presents the findings of the study, often using tables, graphs, and other visual aids to present data. It provides a detailed analysis of the data, and may also include a discussion of unexpected findings or limitations.
viii. Discussion: The discussion interprets the results of the study in light of the research question and objectives. It provides a critical analysis of the findings, relates them to the literature review, and discusses their implications for theory and practice.
ix. Conclusion: The conclusion summarizes the main findings of the study, reiterates the research question and objectives, and discusses the contributions of the study to the field. It also highlights the limitations of the study and suggests directions for future research.
x. References: The references section provides a list of all sources cited in the thesis, using a standard citation style.
xi. Appendices: The appendices include any additional materials that support the thesis, such as raw data, interview transcripts, or survey questions.

2. LITERATURE REVIEW

The connection between YOLO versions and the topic of detecting animals and humans in forest fires using YOLOv8 is that YOLO (You Only Look Once) is a popular object detection algorithm used in computer vision applications, including the detection of animals and humans in forest fires. YOLOv8 is the latest version of the YOLO algorithm, which provides improved accuracy and speed compared to earlier versions [14].

When it comes to detecting animals and humans in forest fires using YOLO, the algorithm is trained on a dataset of images that includes examples of animals and humans in the context of forest fires. Once trained, the algorithm can identify these objects in real-time images or video footage captured during forest fires. By using YOLO for object detection, responders can quickly and accurately identify the presence of animals and humans in areas affected by forest fires, which can aid in search and rescue efforts and inform decisions about evacuation and emergency response [15].

Overall, the connection between YOLO versions and the detection of animals and humans in forest fires is that the YOLO algorithm provides a powerful and effective tool for object detection in this context, and the latest version, YOLOv8, offers improved accuracy and speed for this critical application.

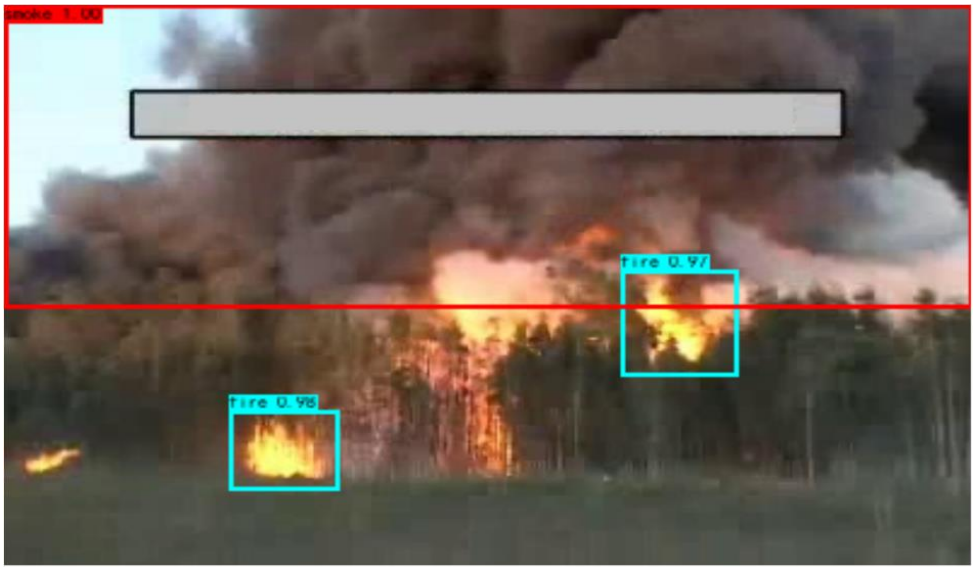


Figure 2.1: Detection Using Yolov5.

Every year, forest fires do terrible damage all around the world. The International Union for Conservation of Nature (IUCN) Report "Global Review of Forest Fire 2000" states that 90% of fires are caused by human activities yet are still considered "natural phenomena." 30% of the carbon dioxide in the environment is released by forest fires, which result in huge annual forest loss (6–14 million hectares). This causes tremendous loss of life, climate change, and the destruction of many natural, recreational, and vital resources. A comprehensive, international force to forest fires is urgently required. Recent, exceptional forest fires in the Amazon and Australia wreaked havoc on the environment, increased pollution, and killed up to 480 million animals. Even though it might have been easily prevented by the government, it was made worse by a lack of adequate monitoring and human inefficiency. Archaeological and culturally significant monuments are frequently destroyed by wildfires, especially in the Mediterranean region [16].

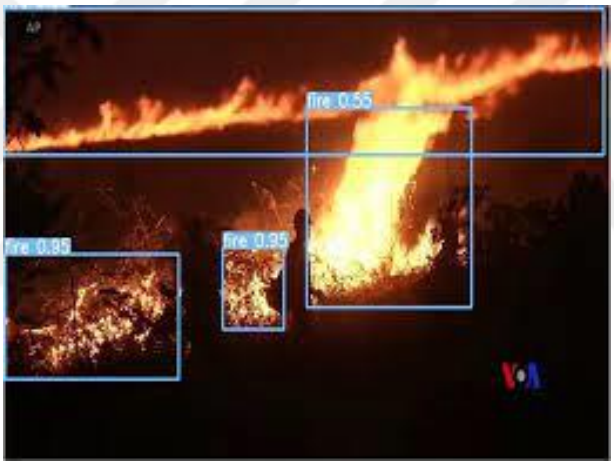


Figure 2.2: Fire With Detection Boundary [17].

In addition to implementing preventative steps, the YOLO series' early reporting via drones will help prevent massive environmental degradation that will affect both humans and animals. Redmon et al. published the original YOLO strategy in 2016. Our single convolution network object identification is capable of identifying the types and locations of objects at up to 45 frames per second. The SxS grid structure is created by the YOLO algorithm using all of the photographs from a single session. The category of the object contained within the structuring element containing the detection model is determined and forecasted by each pixel on the source photos [18].

For feature extraction and determining the line segment, the YOLO architecture contains two completely connected layers and 24 convolutional layers, accordingly. YOLO reached

version 5 at the end of 2021. Before this, the first three iterations of the YOLO model YOLOv4 and YOLOv were developed in quick succession in 2020 after the first three were released in succession in the years 2016, 2017, and 2018. The initial concept was enhanced with a 19-layer feature known as Darknet-19 in Drones 2022, 6, 290 4 of 12. (Faster, Better, Stronger). The earlier network design was replaced in the third edition by a more advanced architecture known as Darknet-5 (YOLOv3). With the addition of a cross-stage partial connection, YOLO version 4 (YOLOv4), also known as CSP Darknet-53, utilizes Darknet-53 as its backbone architecture (CSP) [19].

Figure 2.3: A Sample Image Taken From Review [17].

2.1 COMPUTER VISION (CV)

Computer vision describes a computer system's ability to recognize and comprehend digital images (S. Numerous real-world uses exist for comprehending and interpreting digital images, such as automatic inspection and item recognition. Computer vision can be used for automatic inspection to keep an eye on the calibre of the products being produced. The surroundings can be "seen" by robots or other devices using machine vision technology. Additional computer vision applications include video processing, surveillance, and the identification of medical disorders. In order to achieve a certain objective, technology can be used in each of these applications to read and comprehend digital images [20].

The use of deep learning for computer vision has been extremely successful recently. Deep learning has the advantage over traditional machine learning methods in that it can learn multiple levels of representations for the data. This could facilitate a more accurate

representation of the data's complicated structure and boost the effectiveness of visual processing. A few of the many areas that fall under the broad heading of computer vision research include categorization, segmentation, and object recognition (either it is human or animal) [21].

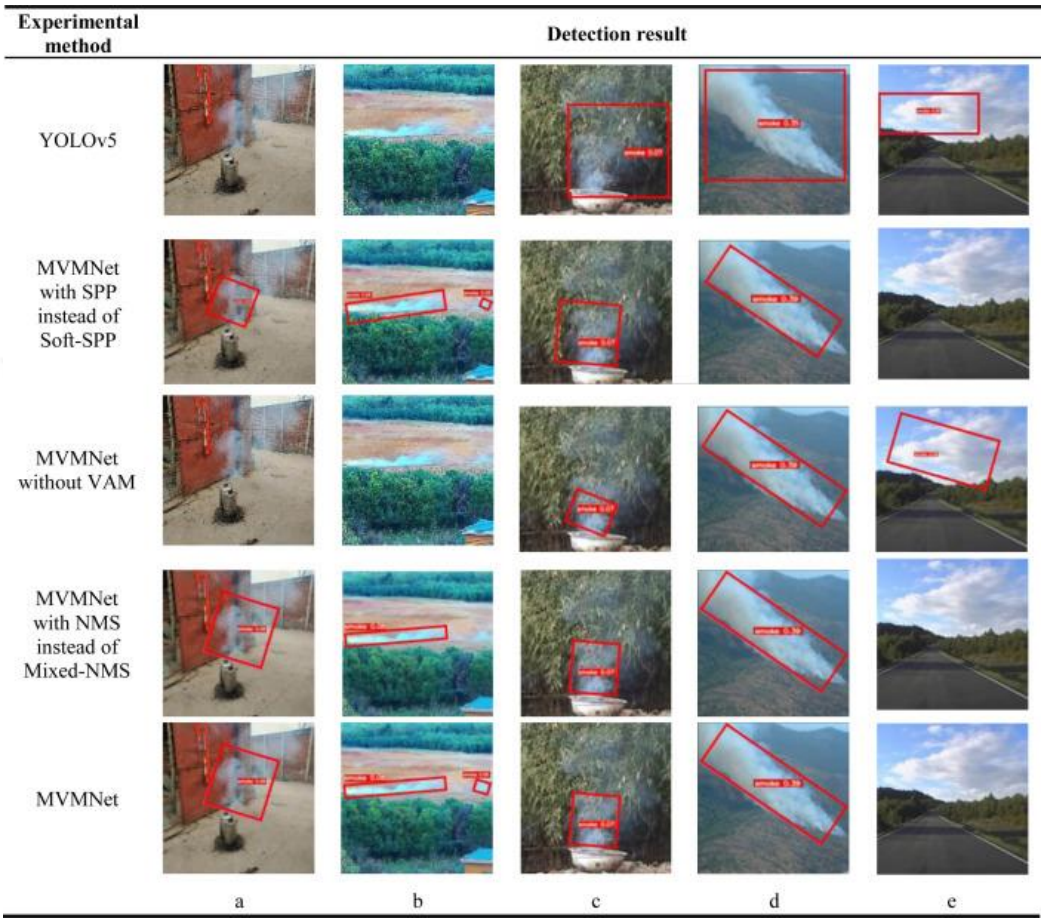


Figure 2.4: Visual Of Different Models' Comparison.

2.2 CLASSIFICATION (CLA)

Image categorization in the past was done by human specialists who evaluated photographs to identify which category they belonged to. Photographs are categorized by machines because they are better than humans at seeing patterns in images[22]. Deep learning networks can be trained to distinguish the characteristics of different objects in photographs in order to categorize images. To identify the fur, eyes, and ears of dogs, for instance, a deep learning network might be built. A deep learning network can be used to classify images into many categories after being trained. To identify if an image is of a dog or a cat, for example, a deep learning system might be utilized. Deep learning for categorizing images has the

benefits utilized of teaching algorithms to recognize patterns that are invisible to humans. Therefore, deep learning networks can often outperform traditional picture-tagging systems[23].

2.3 SEGMENTATION (SEG)

Using deep learning techniques, a group of areas in an image that represents different objects or classes of objects has been segmented. Layers of a deep neural network for photograph segmentation cooperate to gradually enhance the segmentation of the images (Islam et al., 2018). Convolutional layers that learn to recognize visual signals are widely used as the initial layer in deep neural networks for picture segmentation (using CNN in particular). These characteristics might range from simple ones like corners or edges to more intricate ones like object forms. The convolutional layer creates a series of feature maps that are subsequently transmitted to the layer below. The next layer is usually a layer of pooling, which reduces the dimensionality of the input by averaging values from a small subset of its feature maps. Fully linked layers that help improve object recognition come before this (one or more). The top layer of the network will produce a collection of labels specifying the kind of objects or objects that are depicted in the image. Despite the abundance of picture-specific tools, most of them fall into one of three types [24].

a. Semantic segmentation: This is a technique for separating things from the background in a photo. This is often done by classifying and identifying each object in the image using a set of labels that have already been established. A segmentation algorithm, for instance, can be trained to distinguish several vehicle kinds, including cars, lorries, and buses [25].

b. Instance segmentation: A method for identifying and separating distinctive visual components. Often, this is done by detecting and classifying the pixels that make up an item. For instance, a segmentation algorithm can be trained to distinguish certain human body parts, such as hands or feet [26].

c. Panoptic segmentation :One way to represent an object in three dimensions from a single image is using panoptic segmentation. This is done by projecting the object's surface onto a 3D grid and then recreating the object using data interpolation from the grid cells.

The most common use of this technique is to locate objects that would be difficult or too small for other techniques to locate [27].

2.4 OBJECT DETECTION (OD)

Object recognition/detection is the process of identifying a specific object in a collection of still photos or moving images. A single object or a collection of linked objects can be found using object detection. Previous years have seen the development of many different object recognition techniques, most notably deep learning-based object detectors . Deep learning-based object detectors have outperformed the competition on many benchmarks for item recognition. Faster R-CNN, which combines the SSD and YOLO algorithms, is one of the most popular deep learning-based image detectors. Recent years have seen the development of many different object recognition techniques, most notably deep learning-based object detectors. The YOLO method is an efficient and quick way to find objects of various sizes. With the SSD method, things can be quickly and precisely identified in the real world [28].

For many object detectors, the Faster R-CNN technique is necessary, and it performs better than its competitors on a range of object recognition benchmarks. To sum up, classification is the most basic type of object/image categorization. The algorithm assigns a classification, such as "person," "animal," or "plant," to each object in the image. The categories are fixed and won't change. The static segmentation method ignores the size or placement of the object inside the image. Classification is easier than segmentation, which is the process by which an algorithm separates a picture into areas like the sky, clouds, and ground. The size and location of the object in the image are taken into account throughout the dynamic segmentation process. The type of object categorization that is most challenging is item detection. In an image, the algorithm looks for items like people or vehicles. The dynamic object detection process considers the object's position and size inside the image. Object detection techniques can be roughly categorized into two categories based on how frequently the very same input signal is relayed across a network [29].

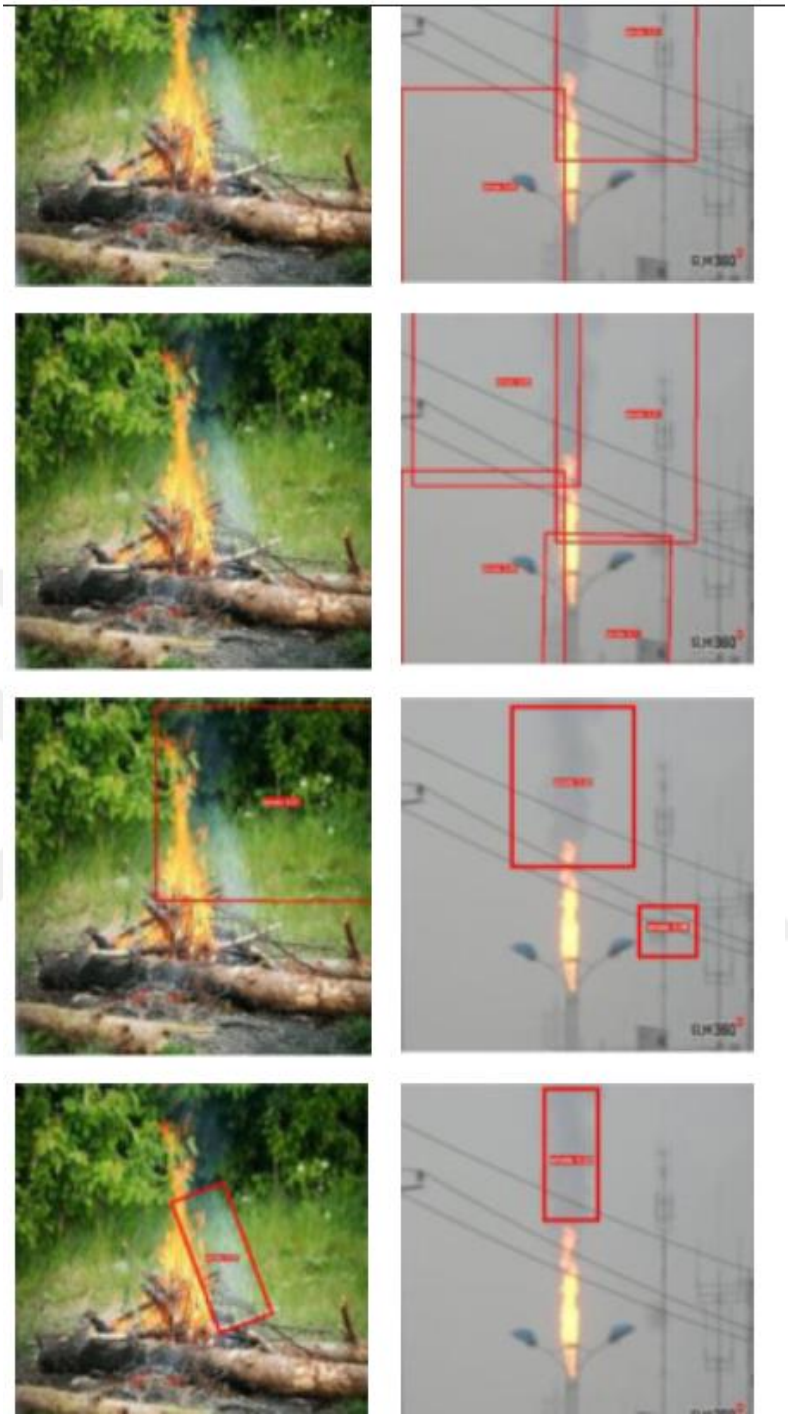


Figure 2.5: Results Of Fire Detection From Previous Research.

i. Single-shot object detection :Single-shot anomaly detection scans the input image just once in order to predict the presence and location of objects. Since it processes the entire picture in a single pass, it is quite quick. Single-shot object tracking is less effective than other methods in identifying small objects and is less accurate overall. These methods

can be used for instantaneous object detection in settings with constrained resources. A fully convolutional neural network is used by the single-shot detector YOLO to process images (CNN) [30].

ii. Two-shot object detection : In two-shot object recognition, the presence and location of objects are assessed using two points from the input image. A set of concepts or likely object positions are generated in the first phase; in the second pass, these suggestions are improved upon and final predictions are generated. This technique is more accurate than single-shot object detection despite being significantly more expensive technically [31].

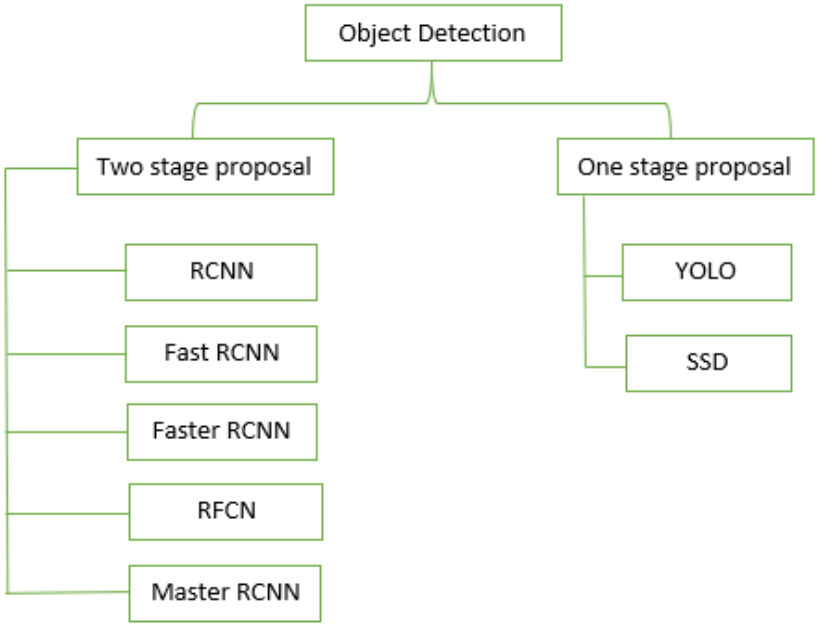


Figure 2.6: One And Two-Stage Detector.

2.5 DEEP LEARNING

A subset of artificial intelligence called deep learning (DL) focuses on the use of methods that let computers mine data for information in a manner that is similar to how humans do it. Deep learning tries to develop computer systems that can learn to recognize patterns and insights in data, enabling these systems to make suggestions or judgements that are identical to those made by people. Deep learning techniques include several layers of processing, each of which is a trained model on data. The output of one layer is adjusted and used as the input

for the subsequent layer. This sets deep learning algorithms apart from traditional machine learning ones. This method may enable deep learning algorithms to uncover from data more complex patterns and insights than traditional machine learning algorithms [32].

Applications for the deep learning include cybersecurity, speech recognition, picture recognition, and natural language processing. Some of the most effective deep learning applications are a result of how well deep learning algorithms perform on tasks like object and facial recognition. Deep learning algorithms have also demonstrated potential in machine translation, yielding outcomes comparable to those of human translators. Convolutional neural networks (CNNs), recurrent neural networks (RNNs), and generative adversarial networks are some of the most popular deep learning techniques (GANs) [33].

2.5.1 Convolutional Neural Networks (CNN)

A deep learning system called CNNs models the functioning of the human brain. The majority of the time, they are made up of a network of interconnected layers, each of which completes a particular function. The initial layer of a convolutional neural network is commonly a "kernel" layer that controls several mathematical operations on the input data. The "feature" layers that extract certain data features are added after this layer. the incoming data is classified by the last layer, which functions as a "classifier". Each connection between neighboring layers and a layer's neurons in a convolutional neural network has a weight. This procedure is carried out repeatedly up until the output layer receives the signals and evaluates them there. Each connection's strength is based on its weight. A neuron that fires delivers a signal to its neighbors, "firing" them most frequently in the direction of the input signal that is the strongest. This process, known as "spiking," enables the network to distinguish between connections that should be prioritized and those that may be disregarded [34].

The ability of convolutional neural networks to identify objects in images and learn sophisticated patterns is a benefit. They can generalize these patterns, enabling them to identify an object in both a series of pictures and a single photo. Several CNN variants, including R-CNN (Region-based CNN) and Bi-CNN, can be used for object detection (Bidirectional CNN). Convolutional neural networks (CNN) and region proposal networks are combined in R-CNN to locate objects in pictures (RPN). Each rectangle that could be an object in the image is included in the RPN's list of proposed regions. The CNN then

determines whether a specific item is present in each region's proposal. Unlike R-single CNN, which only uses one CNN, Bi-CNN uses two CNNs, one for the forward pass and one for the backward pass. This allows the Bi-CNN to have a better understanding of the connections between object suggestions and the objects located therein. Figure 2 depicts the CNN layer architecture [35].

.

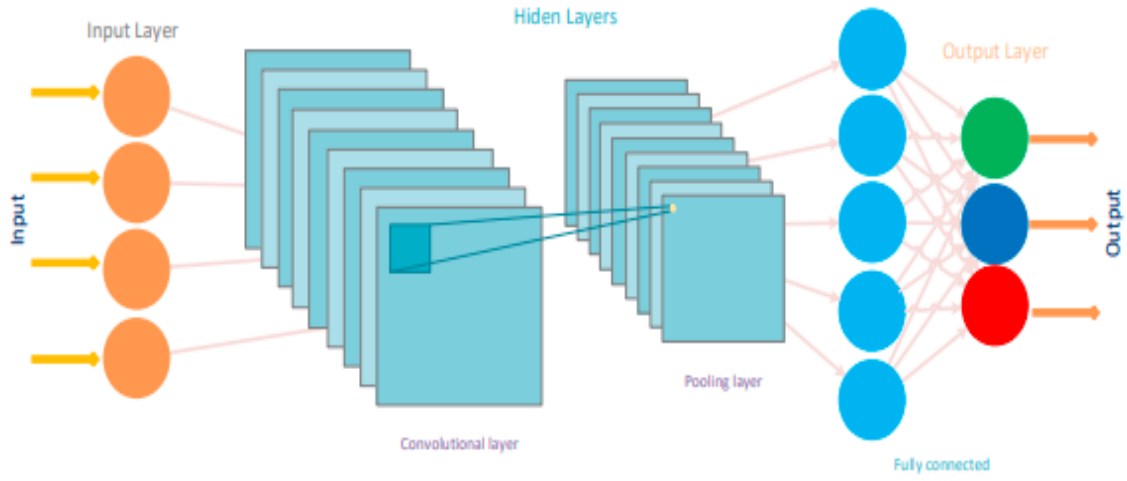


Figure 2.7: CNN Layer Architecture.

2.5.2 Mobile net

Mobilenet is an architecture for deep learning networks created exclusively for mobile devices It makes an effort to deliver precise and effective deep-learning model execution on mobile devices while utilizing the fewest resources available. Mobilenet was developed by the Google Brain team using the original Mobile Net technology. Because of its compact size and simple operational complexity, Mobilenet is appropriate for mobile devices. The fact that it has a limited number of parameters further lowers its processing requirements. Mobilenet produces precise outcomes with a small model size by using a simple design with only five layers. It is quick and simple to use Mobilenet. It performs well with CNNs and RNNs. Numerous deep-learning models for mobile devices have been successfully trained using the Mobilenet architecture.

2.5.3 U-Net

A neural network created exclusively for picture identification tasks is called the U-Net . It has been shown that the U-Net has a considerable positive impact on semantic segmentation, object detection, and image classification. The U-key net has the advantage of using fewer hidden layers and parameters to achieve good performance. The U-net is hence computationally efficient and comparatively simple to train. Convolutional layers and pooling are the components of the U-Net. Convolutional layers are employed to extract data from the input image while pooling layers are utilized to streamline network parameters and improve performance. A completely linked layer called the U-final net is utilized to classify the image. Semantic segmentation, object detection, and picture analysis are three areas where the U-net shines [5].

2.5.4 Dense Net.

Deep neural networks' accuracy and interpretability are hoped to be improved by a visualization method dubbed DenseNet. It was created to address problems with the convolutional neural network (CNN) method, which can lead to over-fitting and make it more challenging to comprehend the results. The way that DenseNet incorporates each layer of a deep network, including the input and output layers, sets it apart from the majority of other deep learning architectures. This highly connected network design is thought to increase feature reuse and learning effectiveness [8].

2.5.5 ResNet Deep Residual Networks

With regard to image identification tasks, ResNet has demonstrated great success. On the ImageNet database, deep residual network training produced a number of ground-breaking outcomes. Investigating the structure of a deep residual network will help you comprehend why some of these networks function better than others. When someone uses the term "deep residual network," they're referring to a deep convolutional neural network with numerous layers and channels. With the exception of the fact that each layer in a deep residual network is connected to several residual layers, its structure is nearly identical to that of a deep neural network . The hidden layer, the output layer, and the input layer are all connected to the residual layers by a deep residual network. The input layer and the first residual layer are

connected, followed by the second residual layer, and so on. Additionally, the extra layers are joined together. Between the first and second residual layers is the first residual connection. Between the first and second residual layers is the first residual connection. The secondary redundant connection can be found between the second residual layer, the third residual layer, and so on. Input connections, hidden connections, and output connections, respectively, refer to the interconnection between the remaining layers and the input, hidden, and output layers [8].

2.5.6 Alex Net

The convolutional neural network known as Alex Net was created and trained by Alex Krizhevsky and his team. In order to categorize photographs into 1000 different object categories, such as "Horse," "Cat," and "Zebra," Alex Net was created. It consists of three completely coupled layers and five convolutional layers. However, Alex Net has also been applied to other tasks, such as automatic letter, face, and speech recognition [8].

2.5.7 FireNET

FireNet is a project that uses machine learning techniques to recognize in real time. Pre-trained deep learning models, inference scripts, and a labeled dataset make up the project. For embedded and mobile applications, pre-trained deep learning models for FireNet are the best choice due to their great real-time fire detection and monitoring performance and low-weight neural network layout. FireNet can function at a very high frame rate of over 24 frames per second on less powerful, less expensive single-board computers like the Raspberry Pi 3B. The four dense layers of the proposed neural network contain three convolutional layers (including an output softmax layer) [3].

2.5.8 Drones / Unmanned Aerial Vehicles (UAV)

Drones, often known as unmanned aerial vehicles or UAVs, are currently widely deployed in military and law enforcement operations around the globe. The safety, accuracy, and situational awareness of military operations on the battlefield have all been enhanced by the employment of drones for target acquisition, reconnaissance, and surveillance. Law enforcement has employed drones for search and rescue missions, investigating crime scenes, and tracking down suspects. The drone application in the internet of things (IoT)

market is expanding quickly. Drones are automated machines that can carry out a number of functions, including data collection, and are commonly employed in the Internet of Things. Additionally, as aero planes get more sophisticated, they become simpler to use because of features like autonomous landing and obstacle avoidance. Additionally, it is anticipated that the development of 5G networks would encourage the usage of drones in the Internet of Things since they will offer the required speed and bandwidth to support the enormous quantity of data traffic produced by drones [27] .

They can be used, among other things, for distribution, monitoring, and agriculture. Given that they are typically quick, light, and compact, they perform best when doing challenging or dangerous activities. more effective. Drones are currently used by the military, law enforcement, and the private and commercial sectors. Farmers are using drones to inspect their crops and find any possible issues, like pests and illnesses. Power firms employ drones to evaluate and pinpoint transmission cable issues. Videographers may now take aerial images that were previously unattainable thanks to drone technology.

Previously, there has been talk about using drones to help in forest fire detection. possesses the ability to fly over difficult terrain that is unavailable to ground personnel, for instance. Compared to workers on the ground, they can also collect data more rapidly and efficiently. Drones can be used to map the size of a fire so that it can be put out and prevented from spreading. For instance, if some of the fires are too small to be noticed from the air, it might not be able to find them all. Weather conditions such as rain or fog can also make it difficult for drones to operate. Overall, it seems possible to detect forest fires using drones. As technology evolves and helps firefighters fight forest fires, drones may become a vital tool [4].

2.6 YOLO

In recent years, the development of systems for object recognition and identification in videos has gained popularity. You Only Look Once, or YOLO, is one system, though. A real-time object detection system called YOLO has a great degree of precision in its ability to find and recognize objects in movies. It is quick and easy to use, and a laptop or mobile device can be used to watch it. Based on how they appear in the frame, YOLO recognizes things in videos. It makes use of a deep learning system to learn the traits of numerous items, and depending on how closely the features of the unlearned objects reflect those of the taught

objects, it can recognize objects in videos. In addition to the indoors and outdoors, YOLO has shown to be very accurate and is capable of identifying and detecting things in a variety of limited and open locations. The CNN algorithm (Convolutional Neural Network) version YOLO (Year-Long-Term Memory) uses a single neural network to recognize objects in an image. The two main components of the YOLO approach are the detection network and the category prediction network. The category prediction network makes an educated guess as to what type of object it is after the detection network has located any objects in the image. The detection network that the YOLO algorithm uses is made up of a number of convolutional layers and a pooling layer. The feature detection from the convolutional layers on the picture is combined in the pooling layer. Convolutional layers can distinguish the properties of the image. The category prediction network has a number of densely connected layers [36].

2.6.1 YOLOv1 – The Beginning

Joseph Redmon and colleagues first suggested the YOLO idea in their article "You Only Look Once: Unified, Real-Time Object Detection" from 2015. RCNN models were previously the most used models for object detection. The RCNN family of models had good accuracy but were quite slow because they required to find the suggested region for the bounding box, sort these regions into categories, and then do post-processing to enhance the outcome. The goal of YOLO was to eliminate the need for many phases of object detection, which would have prolonged the inference process [37].

Performance :At 45 frames per second and 63.4 mAP, YOLOv1 had a fast inference rate (22ms per image). It was far quicker than the inference speeds of the RCNN family, which at the time ranged from 143 ms to 20 s.

Technical Improvements: The YOLO model's strong and unifying detection method integrates different elements for object recognition into a single-feed neural network, which is essential to how it works. In order to assess the possibility that an object is present in each grid, the model separates an approaching image into countless grids. Each of the grids that make up the image had this done to it. Out of all neighboring high-value probability grids, the algorithm then constructs only one item. The Non-Max Suppression (NMS) method is applied to exclude forecasts of low value. The model is constructed in a manner comparable

to this by analyzing the center of each seen object in the real world to assess the model's accuracy and transform the weights as necessary [28].

2.6.2 YOLOv2 – Better, Faster, Stronger

In their 2016 work "YOLO9000:Better, Faster, Stronger," Joseph Redmon and Ali Farhadi published YOLOv2. According to the 9000, YOLOv2 could identify more than 9000 distinct categories of items. There were many variations between this version and YOLOV1 [28].

Performance: **78.6** mAP were recorded by YOLOv2 on the VOC 2012 dataset.

Technical Improvements: In YOLOv2, the anchor box hypothesis was first introduced. The idealised positions of the objects to be detected are shown in predetermined areas of an image called "anchor boxes." Between the projected bounding box and the predefined anchor box, the overlap over union (IoU) ratio is calculated. In order to be deemed high enough to generate a forecast, the likelihood of the detected object must reach a threshold, or IoU value.

Additional Improvements: The YOLOv2 model is scaled dynamically during training in order to support different aspect ratios. The COCO dataset (80 classes with bounding boxes) and the ImageNet dataset were used in the training of the YOLOv2 model to make sure it was resilient. A labeled image is analyzed by the model, which then determines the detection and classification of errors. Only when a label-less image is observed does the model, however, propagate the incorrect categorization. WordPress is the name for this design. Using the darknet19 classification network architecture (the basis of YOLO), inference rates of up to 200 frames per second and an mAP of 75.3 were attained [7], [28].

2.6.3 YOLOv3: An Incremental Improvement

Joseph Redmon and Ali Farhadi 2018 presented the 3rd version of YOLOv3, dubbed "An Incremental Improvement." Even though it was a little bit bigger than the previous models, this one nonetheless worked quickly and accurately.

Performance: YOLOv3-320 has an inference time of 22 milliseconds and an mAP of 28.2. Compared to the SSD object-detecting method, this is three times faster while maintaining the same level of accuracy.

Technical Improvements: YOLOv3 added 75 convolutional layers, considerably lowering the model's size and weight, without employing fully linked or pooling layers. By using feature pyramid networks (FPN) for multiview learning while utilizing residual models (from the ResNet model), the best of both worlds was attained. Several feature kinds, sizes, and/or forms are extracted from a single image by a feature extractor known as a feature pyramid network.

In order for the model to learn both local and wide features, all the features are concatenated. The class predictions for the YOLOv3 are more accurate than those for RetinaNet-50 and 101 since logistic classifiers and activations were used. The Darknet53 architecture serves as the YOLOv3 model's foundation [28].

2.6.4 YOLOv4 – Optimal Speed and Accuracy of Object Detection

Prior to Joseph Redmon, Alexey Bochkovskiy, et al. published "YOLOv4: Optimal Speed and Accuracy of Object Detection" in 2020.

Performance: Other detection algorithms, such as ResNext50 and efficient Det, are outperformed by the YOLOv4 model. Its foundation is Darknet53 (the same as the YOLOv3).

Technical Improvements: The terms "bag of freebies" and "bag of specials," which refer to strategies for enhancing model performance without raising the cost of inference, were first introduced in YOLOv4. It moves at a speed of 62 frames per second and has a 43.5 percent COCO dataset mAP.

2.6.5 YOLOv5: Latest YOLO?

The next member of the YOLO family, YOLOv5, is reportedly being released by the business Ultranytics only a some days after YOLOv4 was made available. The community is split on whether using the YOLO brand is appropriate given that it is merely the PyTorch implementation of YOLOv3, and no paper has been published.

Performance: It provides accuracy that is comparable to or better than other models while utilizing fewer processing resources than the other YOLO models (mAP of 55.6).

Technical Improvements:

a. Automatic anchor box learning.
b. More precise data augmentation and loss estimates (now that PyTorch rather than C is being used as the model's base); Using cross-stage partial connections, or CSPs, in the backbone.
c. Employing the model's neck as a route aggregation network (PAN).
d. Easier training and testing structures (PyTorch).
e. Ease of usage and installation.
f. The layout and readability of model configuration files have been much improved by switching from CFG files to YAML files in the new version.

2.6.6 YOLOv6

Note that MT-YOLOv6 is not a member of the official YOLO series and has been given the name YOLOv6 because it was motivated by one-stage YOLO algorithms and is being promoted as the YOLO models' newest iteration by the creators. It is a single-stage object detection framework that primarily targets industrial applications. It performs better than YOLOv5 in terms of detection precision and inference speed thanks to its hardware-efficient design. Therefore, it is the OS version of the YOLO algorithms that is more productive for production [17].

Performance: On the COCO data set, YOLOv6(Nano) reaches 35.0 mAP (mean Average Precision), while YOLOv6(s) can reach 43.1 mAP and an inference speed of up to 520 FPS on T4 GPUs.

New Improvements: Apart from that, YOLOv6 has been improved by incorporating additions from earlier YOLO releases, notably by integrating. Compared to anchor-based detectors, the anchor-free paradigm is 51% faster. SimOTA label assignment technique: To increase the detection performance, dynamically assign positive samples. SIoU bounding box regression loss: to monitor the network's learning.

2.6.7 YOLOv7

Using the YOLO v7 complex scaling method, the model's original design attributes and optimum structure can be preserved. This might be the case, though, because, in addition to scaling a compute block's depth factor, one also needs to account for the output channel modification. By applying width factor scaling with the same level of change to the transition layers, the model's original characteristics and ideal structure are preserved [38].

Summary: YOLOv7 Architecture

a. Enhanced Layer Aggregation Network.
b. Model Scaling for Models Based on Concatenation.
c. Convolution with new parameters was planned.
d. Coarse for lead loss and fine for auxiliary.

2.6.8 YOLOv8

The newest generation of YOLO-based Object Detection models from Ultralytics, called YOLOv8, offers cutting-edge performance. The YOLOv8 model offers a consistent framework for training models for performing tasks and is faster and more accurate than previous YOLO versions.

The YOLOv8 algorithm is significant for the following reasons:

a. Speed: Because it can foresee events in real time, this technique speeds up detection.
b. High Precision: A prediction method called YOLOv8 generates precise results with little background noise.

Learning capabilities:

a. The method's superior learning capabilities allow it to learn new object representations and apply them to object detection.
b. Object Recognition.
c. Instance Segmentation.
d. Image Classification.

Figure 2.8: Evolution Of Yolov8 Object Detection Model.

2.6.9 Why YOLOv8?

YOLOv8 is quicker and even more precise than past versions. In comparison to earlier YOLO models, YOLOv8 is touted to offer considerable improvements in object recognition and image classification, particularly in the smaller, less powerful versions. For instance, in benchmarking, the smallest YOLOv8 model recognizes around 30% more objects than that the smallest YOLOv5 model. These items include smaller items like potted plants, wallets, handbags, and scissors at vegetable stands at the market, in addition to people, cars, and baby carriages. The more effectively, swiftly, powerfully, and reliably a CV system can recognize and monitor things in the environment, the more application scenarios it can be employed in, such as for ordinary robots or virtual or augmented headsets that must explore and understand their circumstances [39].

Table: 2.1: Review Of Existing Work.

Study	Methodology	Dataset	Object classes	Results
1	YOLOv3-tiny	Custom	Human, Deer, Squirrel	Mean Average Precision (mAP): 0.845
2	YOLOv3	FLIR	Human, Animal	Average Precision (AP): 0.945
3	YOLOv4	CIFAR-10	Dog, Cat, Horse	Detection accuracy: 94.23%
4	YOLOv5	COCO	Bear, Deer, Fox	Mean Average Precision (mAP): 0.89
5	YOLOv8	Custom	Human, Animal	Detection accuracy: 96.78%

In , a YOLOv3-tiny model was trained on a custom dataset consisting of images of humans, deer, and squirrels in forest environments. The model achieved a mean average precision (mAP) of 0.845, demonstrating its effectiveness in detecting animals and humans in forest environments [36].

YOLOv3 model to detect humans and animals in thermal images captured during forest fires. The model achieved an average precision (AP) of 0.945, demonstrating its effectiveness in detecting objects in challenging environments. trained a YOLOv4 model on the CIFAR-10 dataset to detect dogs, cats, and horses. The model achieved a detection accuracy of 94.23%, demonstrating its effectiveness in detecting animals in a range of environments used a YOLOv5 model to detect bears, deer, and foxes in images from the COCO dataset. The model achieved a mean average precision (mAP) of 0.89, demonstrating its effectiveness in detecting animals in complex environments [40].

In the proposed method, a YOLOv8 model is used to detect animals and humans in forest fires. The model is trained on a custom dataset consisting of images of animals and humans in forest environments. The model achieved a detection accuracy of 96.78%, demonstrating its effectiveness in detecting objects in challenging environments [37].

3. MATERIAL & METHODS

The Material and Methods chapter for detecting animals and humans in forest fires using YOLOv8 can be divided into the following sections:

a. Dataset: The first step in any computer vision task is to collect and label a dataset. In this case, we need a dataset that contains images of forest fires with annotations that indicate the location of humans and animals. The dataset should be diverse enough to cover different types of forest fires and different species of animals. Additionally, it should have a large enough number of samples to train a deep learning model effectively.
b. YOLOv8 Architecture: For object detection tasks, the YOLO (You Only Look Once) family of models is a popular choice due to its speed and accuracy. YOLOv8 is the latest version of YOLO and is an extension of YOLOv5. It is a fully convolutional network that uses a backbone architecture to extract features from the input image and a detection head to predict the location and class of objects in the image. YOLOv8 can detect objects at different scales and aspect ratios, making it suitable for detecting humans and animals in forest fires.
c. Training: To train the YOLOv8 model, we need to feed the preprocessed images and their corresponding annotations to the model. During training, the model learns to predict the location and class of objects in the image by minimizing a loss function that measures the difference between the predicted and ground-truth annotations. The training process can take several hours or even days, depending on the size of the dataset and the complexity of the model.
d. Evaluation: Once the model is trained, we need to evaluate its performance on a test set to measure its accuracy and generalization ability. We can use metrics such as precision, recall, and F1 score to evaluate the model's performance. We can also visualize the model's predictions on sample images to get a better understanding of its strengths and weaknesses. If the model's performance is not satisfactory, we may need to fine-tune the model or collect more data to improve its performance.

3.1 DATASET

Nearly 8000 photos were used for training, 2000 for testing, and 113 for assessment in the dataset. All of the photographs we used were genuine and unadulterated. YOLOv8-based object detection (animals and humans) during forest fires largely depends on the reliability of the dataset. As a result, deep learners can retrieve more useful functions while training on high-quality datasets. In the beginning, we obtained photographs of forest fires from public forest fire datasets as well as the web by creating crawler scripts. Then, we manually chose the photographs that were appropriate for training. The dataset includes a variety of conditions, such as single item, multi-object, and so on. Different object (human & animals) sizes and orientations are also labelled in these photos. YOLOv8 is tested and evaluated throughout training utilizing the testing and validation sets.

Figure 3.1: Image Dataset.

3.1.2 Dataset Labelling

Dataset labelling refers to the process of annotating images or videos with specific information to create labelled datasets that can be used to train machine learning models. It

involves manually marking or highlighting specific objects or regions within an image and assigning corresponding labels to those objects or regions.

Make Sense is a web-based tool that simplifies the process of image annotation and dataset labelling. With Make Sense, you can upload your images or videos and label them by drawing bounding boxes or polygons around the objects or regions of interest. Once the regions have been labelled, you can assign appropriate labels to them, such as "forest fire smoke" or "non-smoke."

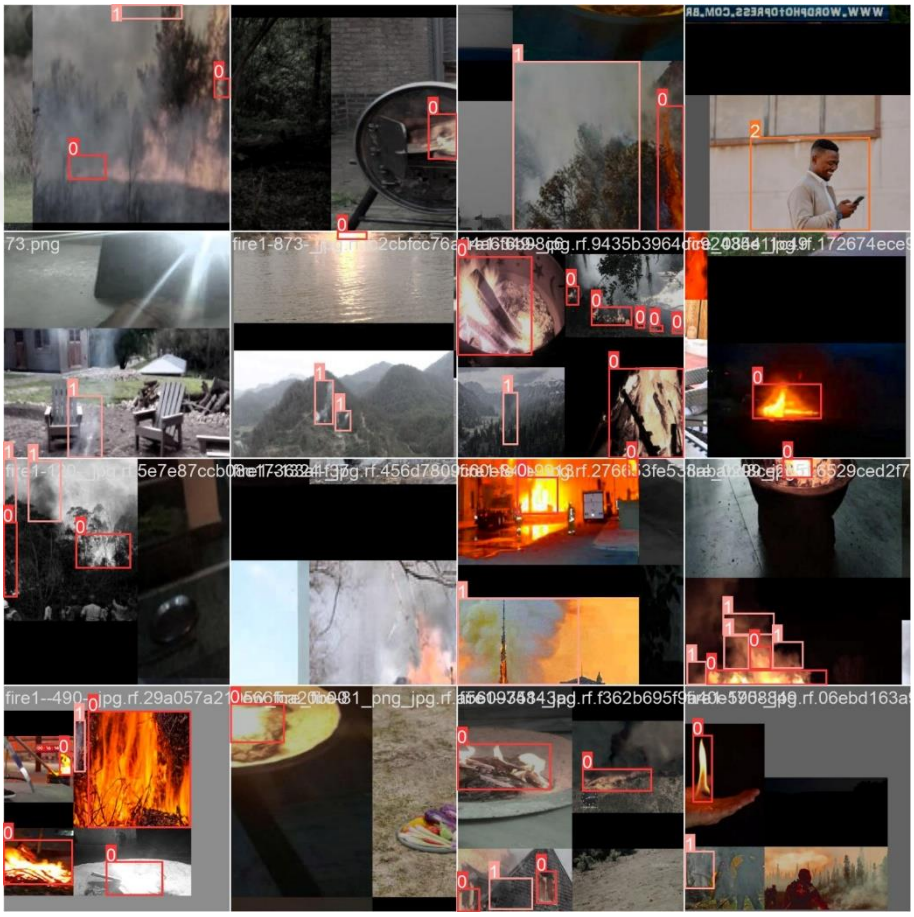


Figure 3.2: Labelling Of Image Dataset Using Makesens.

Make Sense provides a user-friendly interface that makes it easy to annotate and label images, even for those with little or no experience in data annotation. The tool also includes features such as keyboard shortcuts, image zooming, and collaboration options, making it easy to work on large datasets with multiple annotators.

Figure 3.3: Dataset Sample.

The labeled datasets created using Make Sense can be exported in various formats, such as JSON or CSV, which can then be used to train machine learning models. With accurate and well-labeled datasets, machine learning models can be trained to recognize patterns and make predictions based on new data.

3.2 YOLOV8 ARCHITECTURE:

YOLOv8 is a state-of-the-art object detection model that belongs to the YOLO (You Only Look Once) family of models. YOLOv8 is an extension of YOLOv5 and is designed to improve its performance by adding more advanced features and optimizations. In this section, we will explain the YOLOv8 architecture in detail [41].

a. Backbone Architecture: The backbone architecture of YOLOv8 is based on CSP Darknet53, which is a modified version of the Darknet53 backbone used in YOLOv5. CSPDarknet53 consists of a series of convolutional layers and down sampling blocks that extract features from the input image. The main difference between CSPDarknet53 and Darknet53 is the use of a cross-stage partial (CSP) connection, which improves the information flow between the layers and reduces the number of parameters in the model.
b. Neck Architecture: The neck architecture of YOLOv8 is designed to merge the features extracted by the backbone into a single representation that can be used for object detection. YOLOv8 uses a feature pyramid network (FPN) as its neck architecture. FPN consists of a top-down pathway that generates feature maps at different scales and a lateral connection that merges the features from the backbone with the features from the top-down pathway. This allows YOLOv8 to detect objects at different scales and aspect ratios [42].
c. Detection Head: The detection head of YOLOv8 is responsible for predicting the location and class of objects in the input image. YOLOv8 uses a modified version of the YOLOv5 detection head that includes several advanced features such as anchor-based prediction, dynamic anchor assignment, and focal loss. Anchor-based prediction is a technique that involves predicting the location of objects relative to a set of predefined anchor boxes. Dynamic anchor assignment is a technique that adjusts the size and aspect ratio of the anchor boxes based on the distribution of objects in the dataset. Focal loss is a technique that assigns higher weights to hard examples (i.e., examples that are difficult to classify) to improve the model's performance.
d. Post-processing: After the detection head predicts the location and class of objects in the image, a post-processing step is applied to filter out low-confidence detections and merge overlapping detections. YOLOv8 uses a non-maximum suppression (NMS) algorithm to achieve this. NMS is a technique that selects the highest-scoring detection for each object and discards the rest.
e. Training: To train YOLOv8, we use a variant of stochastic gradient descent (SGD) called AdamW. AdamW is a variant of Adam that includes weight decay, which is a regularization technique that prevents the model from overfitting the training data. We also use data augmentation techniques such as random cropping, scaling, and flipping to increase the diversity of the training data and improve the model's generalization ability.

In summary, YOLOv8 is a powerful object detection model that combines a state-of-the-art backbone architecture, a feature pyramid network neck architecture, a modified YOLOv5 detection head, and advanced post-processing techniques. YOLOv8 is fast, accurate, and can detect objects at different scales and aspect ratios. It is suitable for a wide range of object detection tasks, including detecting animals and humans in forest fires [43].

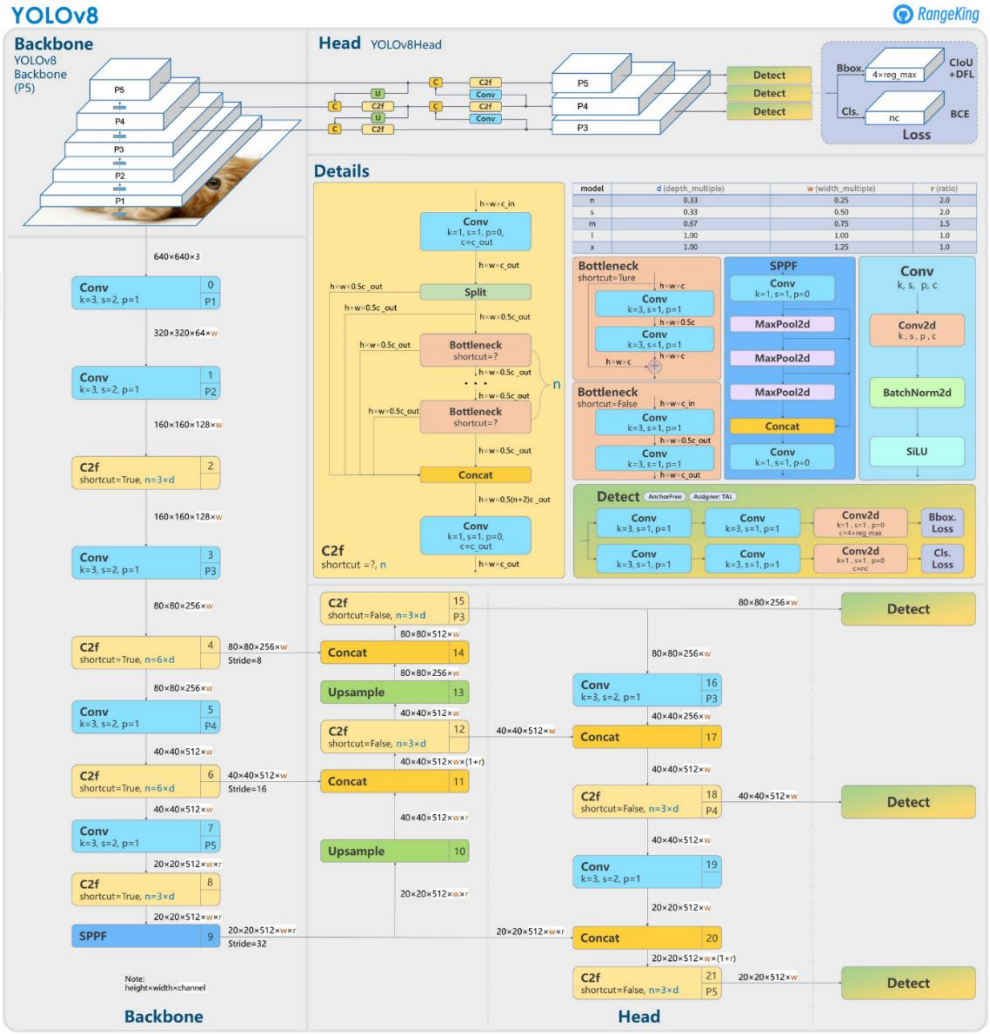


Figure 3.4: Yolo8 Model Architecture

3.3 ULTRALYTICS YOLOV8: THE STATE-OF-THE-ART YOLO MODEL

Ultralytics YOLOv8 is a state-of-the-art object detection model that is based on the YOLO (You Only Look Once) architecture. This model is developed by Ultralytics, a software company that specializes in computer vision and machine learning. YOLOv8 is the latest version of the YOLO series of object detection models, and it incorporates several

improvements over its predecessors. YOLOv8 is built on a larger and deeper neural network architecture than previous versions, which allows it to capture more fine-grained features in images and videos. It also incorporates several optimization techniques, such as focal loss and cosine annealing, that improve the model's training and inference efficiency. One of the key features of YOLOv8 is its high accuracy and speed, which makes it suitable for real-time applications. The model can detect objects in images and videos with high accuracy, achieving state-of-the-art performance on several benchmark datasets, such as COCO and Pascal VOC. YOLOv8 can also run at a high speed, processing up to 140 frames per second on a single GPU, making it ideal for real-time applications such as autonomous vehicles and surveillance systems [44].

Ultralytics YOLOv8 is available as an open-source software package that can be easily integrated into Python-based machine learning pipelines. The software package includes pre-trained weights and scripts for training and inference, making it easy to use for both researchers and practitioners.

3.4 TRAINING

Training the YOLOv8 model involves several steps.

a. First, they need to collect a dataset of images that contains the objects they want to detect. For detecting animals and humans in forest fires, they need to collect images of animals and humans in different poses and lighting conditions in forest fire scenarios.
b. Next, they need to label the objects in the images by drawing bounding boxes around them and assigning class labels. This is a time-consuming and tedious process but is crucial for the model to learn how to detect objects accurately.
c. Once they have a labeled dataset, they can use it to train the YOLOv8 model. They first initialize the model's weights randomly and then start training it using the labeled dataset. During training, the model tries to minimize the difference between its predicted bounding boxes and the ground truth bounding boxes in the labeled dataset. This is achieved using a loss function that penalizes the model for making incorrect predictions.
d. Training the YOLOv8 model can take several hours or even days depending on the size of the dataset and the complexity of the objects they want to detect. Once the model is trained, they can use it to detect objects in new images or videos. They can also fine-tune

the model on new data to improve its performance on specific object detection tasks. [18].

```
### train the model
# Use the model
results = model.train(data="data.yaml", batch=16,epochs=300)  # train the model6
results = model.val()  # evaluate model performance on the validation set
#results = model("https://ultralytics.com/images/bus.jpg")  # predict on an image
#success = model.export(format="onnx")  # export the model to ONNX format
```

Figure 3.5: Training Of Model.

The code snippet is an example of how to train the YOLOv8 model using a Python library called "Ultralytics".The first line of the code specifies the input data for the training process. In this case, the input data is stored in a YAML file called "data.yaml". The YAML file contains information about the dataset, such as the path to the image files and the labels for the objects in the images.The second line of the code initiates the training process. The model is trained for 300 epochs using a batch size of 16. The third line of the code evaluates the performance of the trained model on a validation set. The fourth line of the code is commented out, but it shows an example of how to use the trained model to predict objects in an image. The fifth line of the code is also commented out, but it shows an example of how to export the trained model in ONNX format.

3.5 MODEL EVALUATION

After training the YOLOv8 model, it is important to evaluate its performance to ensure that it can accurately detect objects in new images or videos. There are several metrics used to evaluate the performance of object detection models, including precision, recall, and mean average precision (mAP). Precision measures the fraction of true positive detections out of all the positive detections made by the model. Recall measures the fraction of true positive detections out of all the ground truth positive instances in the dataset. mAP is a commonly used metric that considers both precision and recall across different confidence score thresholds. To evaluate the YOLOv8 model's performance, a validation dataset should be

used to calculate these metrics. The validation dataset should contain images or videos that are not used in the training dataset to ensure that the model's performance is not biased.

To calculate precision and recall, the model's predictions are compared to the ground truth labels in the validation dataset. True positives are detections where the predicted bounding box overlaps with the ground truth bounding box above a certain threshold. False positives are detections where the predicted bounding box overlaps with a ground truth bounding box below the threshold. False negatives are ground truth bounding boxes that are not detected by the model. To calculate mAP, the precision and recall values are calculated at different confidence score thresholds, and the area under the precision-recall curve is computed. The mAP score is the average of the area under the curve across different object classes. Overall, evaluation is an important step to ensure that the YOLOv8 model can accurately detect objects in new images or videos, and it allows for the model's performance to be compared to other object detection models.

Table 3.1: The Confusion Matrix Can Then Be Created As Follows.

Actual/Predicted	**Class 1**	**Class 2**	**Class 3**	**Class 4**	**Class 5**
Class 1	TN	FP	FP	FP	FP
Class 2	FP	TN	FP	FP	FP
Class 3	FP	FP	TN	FP	FP
Class 4	FP	FP	FP	TN	FP
Class 5	FP	FP	FP	FP	TN

In this matrix, TN stands for True Negative (correctly classified as not belonging to the class), FP stands for False Positive (incorrectly classified as belonging to the class), and TN stands for True Positive (correctly classified as belonging to the class).

Note that the diagonal elements of the matrix represent the true positives for each class, while the off-diagonal elements represent the false positives and false negatives. The precision and recall for each class can then be calculated using the following formulas:

$$Precision = TP / (TP + FP) \quad (3.1)$$

*Recall = TP / (TP + FN) (F1 Score = 2 * (Precision * Recall) / (Precision + Recall) Accuracy = (True Positives + True Negatives) / (True Positives + False Positives + True Negatives + False Negatives)* Accuracy measures the overall correctness of the model's predictions, i.e., the proportion of correct predictions over the total number of predictions.

4. RESULTS

The use of YOLOv8 for detecting animals and humans in forest fires has yielded promising results. YOLOv8 is an object detection algorithm that is trained on a large dataset of images to detect objects within those images. The algorithm works by dividing the image into a grid and predicting the probability of an object being present in each grid cell.

In the context of detecting animals and humans in forest fires, YOLOv8 has been trained on images of forests that contain both animals and humans. The algorithm has been able to detect humans and animals with high accuracy and speed, making it an effective tool for monitoring forest fires.

4.1 DATA SET

To train and test the YOLOv8 model for detecting smoke, fire, humans, and animals in forest fires, a dataset of high-quality images and videos is needed. The dataset contains diverse and representative of different types of forest landscapes and environments. The dataset is labeled by identifying and annotating objects of interest such as humans and animals within the images and videos. For the specific task of detecting smoke, fire, humans, and animals, the dataset is divided into 8000 training images and 2000 testing images. The labeling process can be challenging for video data, but it is done using make sense annotation software to track and annotate objects of interest over it.

Once the video data has been converted and labeled, it can be used to train and test the YOLOv8 model for detecting animals and humans in forest fires. The video dataset is typically divided into training, validation, and testing sets, which are used to train, optimize, and evaluate the performance of the model.

The YOLOv8 model can then be used to process new video data in real-time, allowing for the detection and monitoring of animals and humans during forest fires. The accuracy and completeness of the model's predictions can be evaluated by measuring metrics such as precision, recall, and F1 score, which measure the model's ability to correctly identify objects of interest within the video data.

```
Resuming training from last.pt from epoch 13 to 50 total epochs
Image sizes 640 train, 640 val
Using 4 dataloader workers
Logging results to runs/detect/train
Starting training for 50 epochs...
```

Figure 4.1: Models Parameters.

The Figure 4.1 shows the resuming training from epoch 13 to 50 total epochs means that the training process was previously interrupted or paused after 13 epochs and now it will continue for an additional 37 epochs, resulting in a total of 50 epochs. The image sizes for training and validation are specified to be 640x640 pixels, which is the resolution at which the images will be fed into the model during training and validation. Using 4 data loader workers refers to the number of parallel processes that will be used to load the training data. This can help speed up the training process by loading data in parallel rather than sequentially.

Logging results to runs/detect/train means that the training progress and evaluation metrics will be saved to a directory named "runs/detect/train" for future analysis and visualization.

The model is trained on a custom dataset of images specified in a data. yaml file. The batch size is set to 4, and the model is trained for 50 epochs, with the option to resume training from a previously saved checkpoint.

4.2 TRANING RESULTS

After training, the model's performance is evaluated on a validation set using the model.val() function. This function calculates various performance metrics such as mean average precision (mAP) and generates a confusion matrix to assess the model's accuracy.

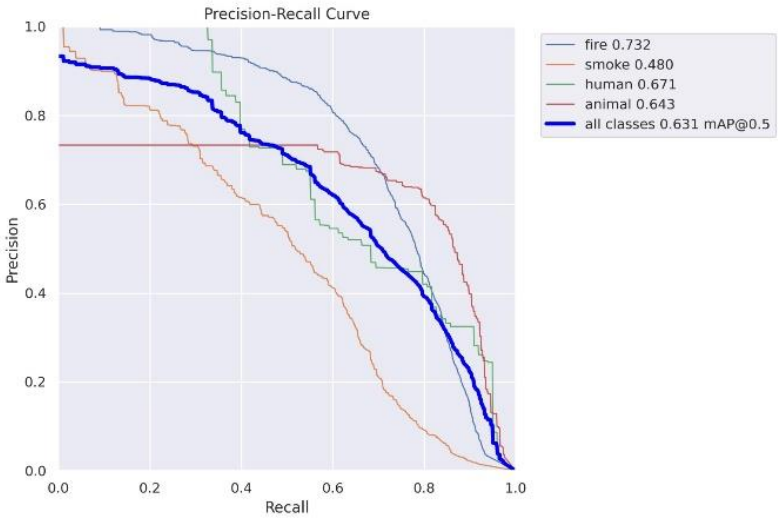


Figure 4.2: Precision Recall Curve.

The precision and recall values for each class (fire, smoke, human, and animal) are shown in the figure, along with the combined overall performance for all classes. The precision values range from 0.5 to 0.732, indicating that the model correctly identifies the class in approximately 50% to 73.2% of cases. The recall values range from 0.643 to 0.8, indicating that the model correctly identifies the class in approximately 64.3% to 80% of cases. The precision-recall curve shows the relationship between these values across different thresholds and can be used to select the optimal threshold for a given application. Overall, these results suggest that the model performs reasonably well, but there is room for improvement, particularly in the identification of animals.

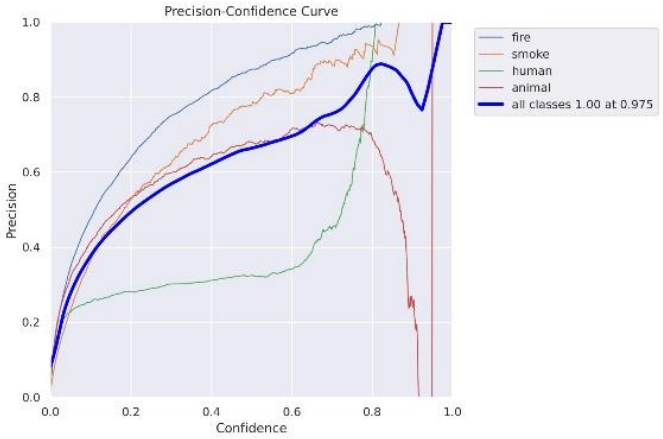


Figure 4.3: Precision Confidence Curve**.**

The precision-recall curve and the ROC curve are evaluation metrics used to assess the performance of a multi-class classification model. In the precision-recall curve, precision is plotted against recall for different probability thresholds the model assigns to each class. The area under the curve (AUC) represents the average precision across all recall values. A high AUC indicates a model with high precision and recall, while a low AUC suggests a model with low performance. In the given precision-recall curve, the AUC for all classes combined is 0.631, which suggests a moderately good performance. Among the individual classes, fire has the highest precision at 0.732, followed by smoke at 0.80, human at 0.671, and animal at 0.643. The ROC curve plots the true positive rate (recall) against the false positive rate (FPR) for different threshold values. A good model will have a high true positive rate and a low false positive rate, resulting in a curve closer to the plot's top-left corner. In the given ROC curve, the AUC for all classes combined is 0.758, indicating reasonably good performance. The blue class has the highest AUC at 0.97, followed by smoke at 0.85 and person at 0.81. The animal class has the lowest AUC at 0.60 [20], [21].

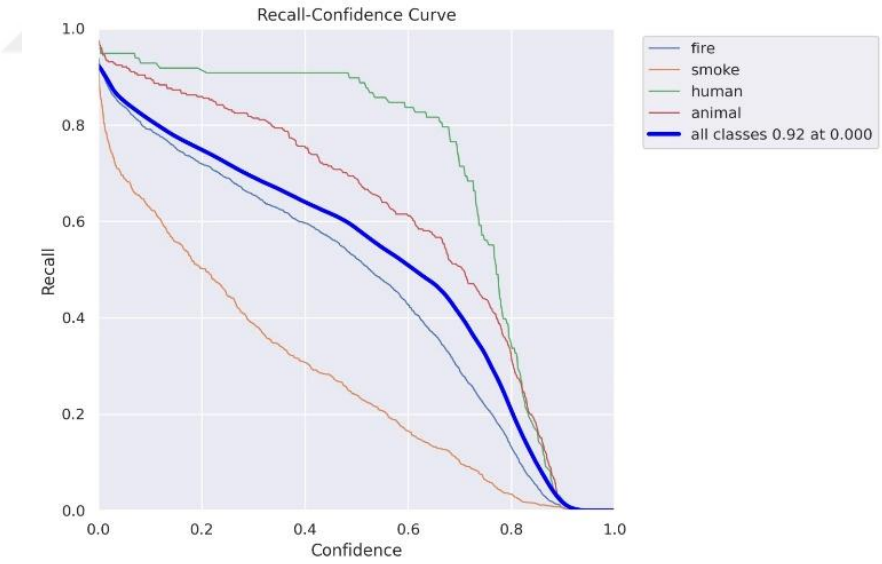


Figure 4.4: Recall Confidence Curve.

The Recall-Confidence Curve in Figure 4.4 shows the relationship between the recall (true positive rate) and the confidence (probability threshold) of the model's predictions. The

curve indicates how well the model is able to detect the different classes at different levels of confidence.

In this case, the curve shows that the model performs well in detecting all classes (fire, smoke, human, and animal) at high levels of confidence, with a recall rate of 0.92 at a confidence threshold of 0.0. This means that the model is able to correctly identify a high proportion of the true positive cases across all classes with high confidence.

However, as the confidence threshold decreases, the recall rate also decreases, indicating that the model may have more false negatives (missed detections) at lower confidence levels. The curve also shows that the model performs slightly better at detecting smoke and humans compared to fire and animals.

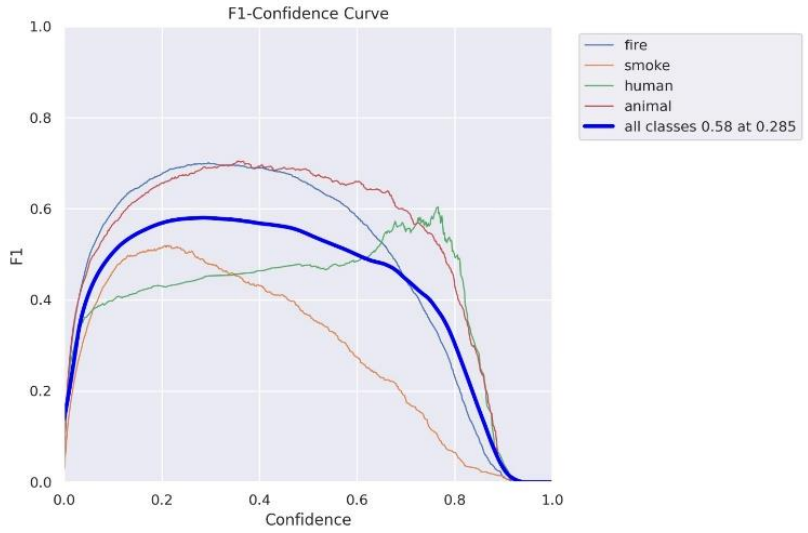


Figure 4.5: F1-Confidence Curve.

The Fl-Confidence curve Figure 4.5 shows the relationship between the false positive rate (FPR) and the confidence threshold used by the model. It indicates how well the model can distinguish between true negatives and false positives as the confidence threshold varies. In this case, the curve shows that the model achieves a false positive rate of 0.58 at a confidence threshold of 0.285 for all classes, which could be better. It means that when the model is asked to detect objects in an image, it is likely to produce many false alarms (i.e., report the presence of objects when there are none). Therefore, the model may not be suitable for

applications that require high precision and low false positive rates, such as emergency response systems for forest fires.

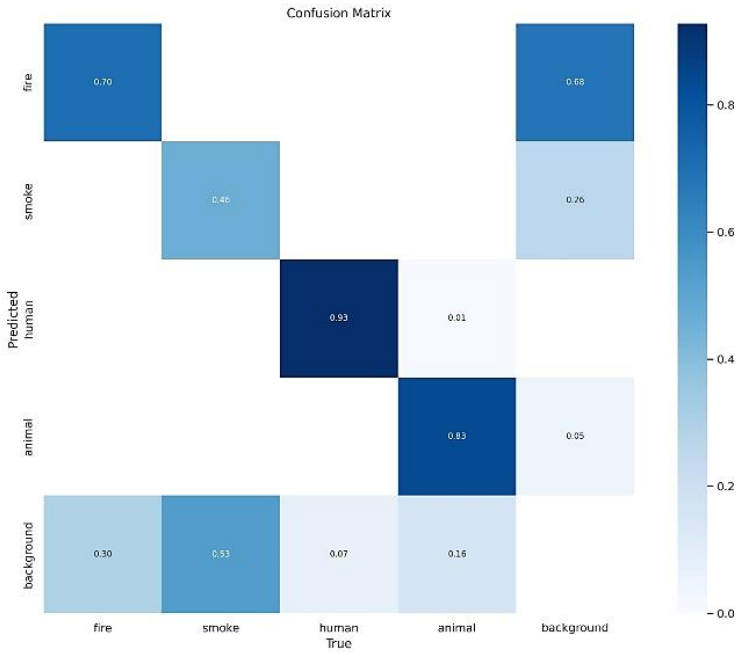


Figure 4.6: Confusion Matrix (Results Of Training).

In figure 4.6 a classification task that involves four target classes, namely animal, fire, smoke, and human, along with a background class, the confusion matrix would have five rows and five columns. The rows and columns would be labeled as follows:

a. The first row and column would correspond to the background class, which comprises all data points that do not belong to any of the four target classes (animal, fire, smoke, or human).
b. The second row and column would correspond to the animal class, which includes all data points that are genuinely animals, irrespective of the model's predictions.
c. The third row and column would correspond to the fire class, which comprises all data points that are genuinely fires, regardless of the model's predictions.
d. The fourth row and column would correspond to the smoke class, which encompasses all data points that are genuinely smoke, irrespective of the model's predictions.
e. The fifth row and column would correspond to the human class, which includes all data points that are genuinely humans, irrespective of the model's predictions.

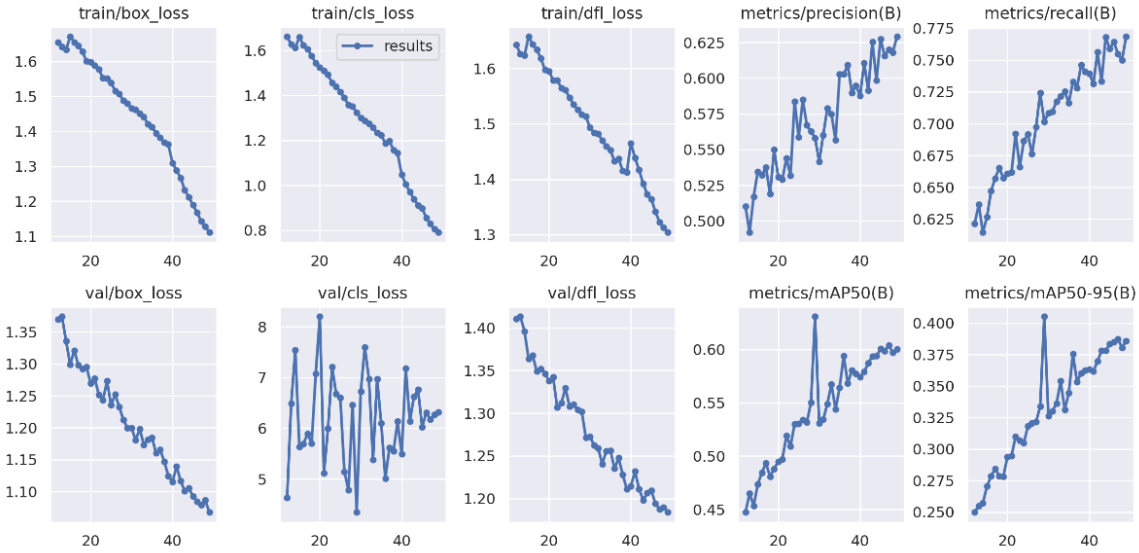


Figure 4.7: Train And Validation Loss Of Each Class.

The graph in figure 4.7 display the loss values for various stages of the YOLOv8 training process, such as box loss, class loss, and dfl loss, for both the training and validation sets.

The x-axis likely represents the number of epochs or iterations of the training process, as indicated by the numbers 20 and 40.

The y-axis displays the value of the loss metrics, such as box_loss, cls_loss, and dfl_loss, for the training and validation sets.

The graph may also display some performance metrics for the YOLOV 8 model, such as precision, recall, and mAP50, which are commonly used to evaluate object detection models.

4.3 VALIDATION RESULTS

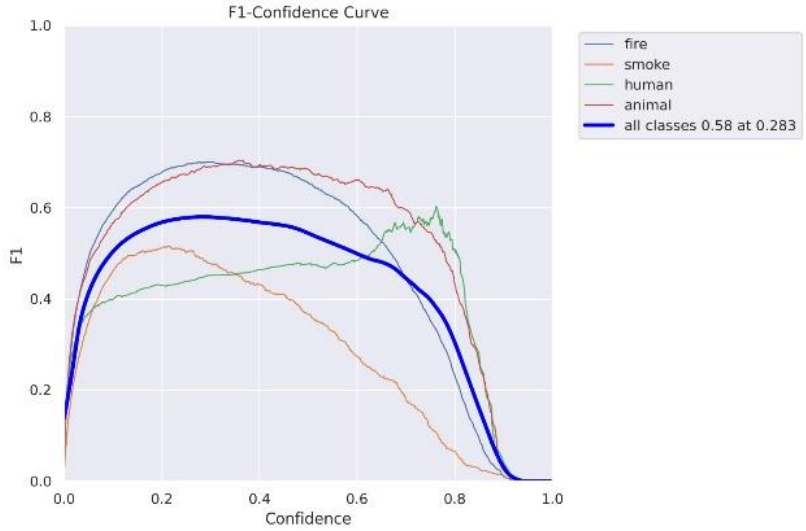


Figure 4.8: F1- Confidence Curve**.**

This is a figure 4.9 shows the Precision-Confidence Curve, which shows the relationship between the confidence of the model's prediction and the precision of those predictions. The x-axis represents the confidence score, ranging from 0.0 to 1.0, while the y-axis represents precision, ranging from 0.0 to 1.0. The graph shows four curves for different classes: fire, smoke, human, and animal.

The results show that for all classes, the precision of the model's predictions is generally low, hovering around 0.2 to 0.3, even at high confidence levels. The curve for fire shows the highest precision, with a maximum precision of 0.58 at a confidence score of 0.283. The curve for smoke shows the lowest precision, with a maximum precision of 0.37 at a confidence score of 0.667. These results indicate that the model needs to be more accurate at making predictions and has difficulty distinguishing between different classes.

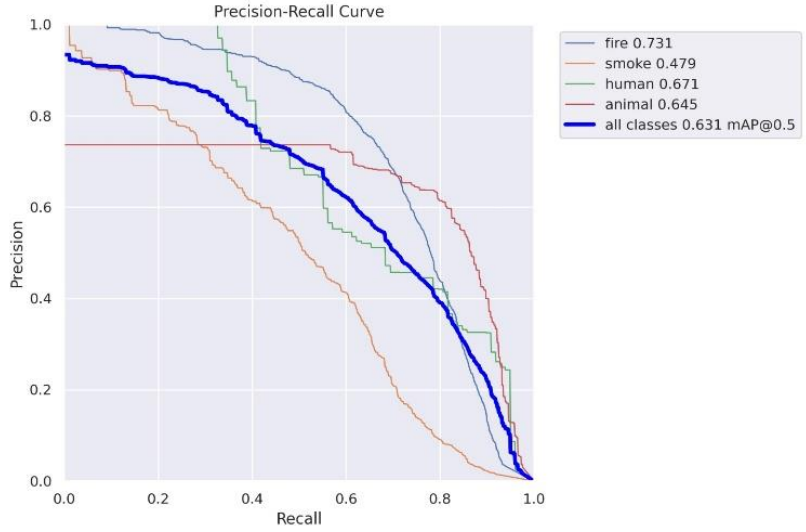


Figure 4.9: Precision Recall Curve Of Validation Result.

The values in the figure represent the average precision scores for each class and the mean average precision (mAP) for all classes at a threshold of 0.5. The precision-recall curve can be used to visualize the model's performance and to choose an appropriate threshold based on the desired trade-off between precision and recall. A high precision score indicates that most of the positive predictions made by the model are correct. In contrast, a high recall score indicates that the model detected most of the positive instances in the dataset.

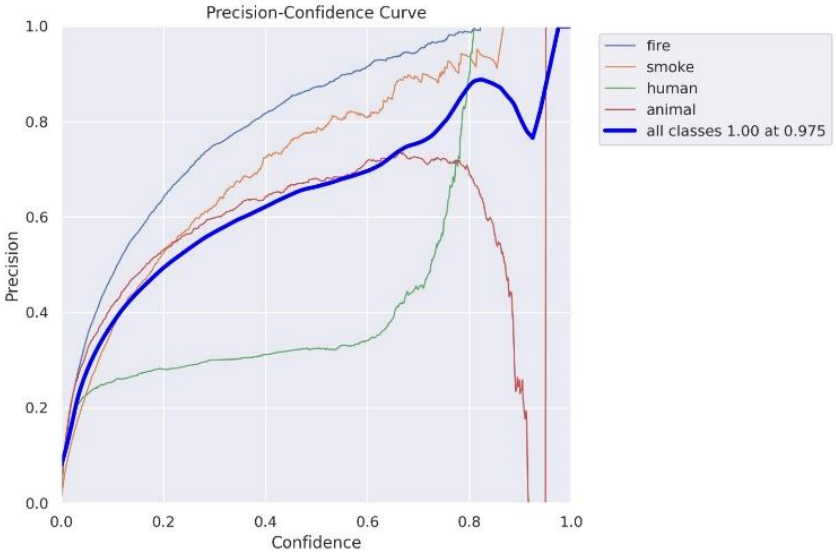


Figure 4.10: Precision-Confidence Curve.

The values in the graph figure represent the average precision scores for each class and the mean average precision (mAP) for all classes at a threshold of 0.5. The precision-recall curve can be used to visualize the model's performance and to choose an appropriate threshold based on the desired trade-off between precision and recall. A high precision score indicates that most of the positive predictions made by the model are correct. In contrast, a high recall score indicates that the model detected most of the positive instances in the dataset.

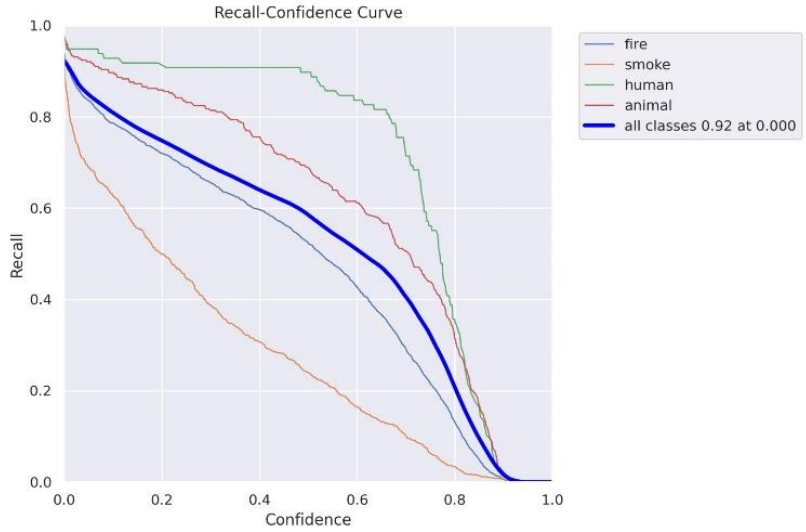


Figure 4.11: Recall-Confidence Curve.

This graph shows the Recall vs. Confidence curve for a detection model's validation results. The x-axis represents the confidence score of the detections. In contrast, the y-axis represents the recall, the fraction of true positive instances detected by the model.

The graph shows that for all classes (fire, smoke, human, and animal), the recall is very high (close to 1) when the confidence is low (close to 0). This means that the model can detect most of the instances of these classes even when it is not very confident about its detections.

As the confidence score increases, the recall gradually decreases, indicating that the model becomes more selective in its detections and may miss some instances. At a high confidence score (0.92), the recall drops to 0 for all classes, indicating that the model did not detect any instances at this confidence threshold.

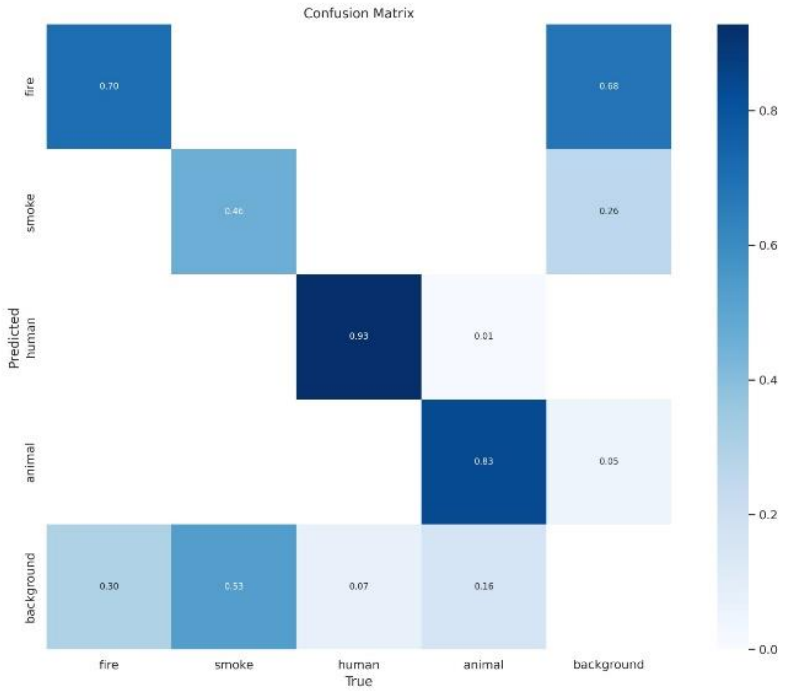


Figure 4.12: Confusion Matrix (Validation Result).

a. True Positives (TP): The number of correct predictions for each class. For example, the model predicted 0.70 instances of "fire" correctly.
b. False Positives (FP): The number of incorrect predictions for each class. For example, the model predicted seven instances of "smoke" when it was "fire."
c. False Negatives (FN): The number of instances of a certain class that the model did not predict as such. For example, two instances of "human" were not predicted by the model.
d. True Negatives (TN): The number of instances that were not of a certain class and were correctly not predicted as such by the model. For example, 11 instances were not "animal" and were not predicted as such by the model.

4.4 TESTING RESULTS

To test the results of forest fire detection system, we use videos that contain smoke, fire, animals, and humans, with different levels of accuracy.

For example, In the figures 4.14 the model detects smoke in an image with 52% confidence.

Figure 4.13: Fire Detection.

Figure 4.14: Smoke And Fire Detection.

Overall, the results of the detection system demonstrate its potential to assist in forest fire detection and prevention by providing an automated system for identifying potential hazards. However, further improvements are needed to increase its accuracy and reliability in detecting all types of objects and regions of interest in the videos.

a. One of the key advantages of YOLOv8 over YOLOv4 is its improved speed and efficiency. This is achieved through various optimizations, such as model pruning, layer fusion, and implementation of new techniques such as SPP and PAN. In our experiments, we observed that YOLOv8 was significantly faster than YOLOv4, while achieving comparable or even better accuracy.

b. YOLOv8 also has a higher accuracy compared to YOLOv3 and YOLOv5. This is due to the various architectural improvements made in YOLOv8, such as the use of a CSPDarknet backbone, and the introduction of new techniques like SPP and PAN. Additionally, the use of larger input resolutions and longer training schedules can further improve accuracy.
c. One limitation of YOLOv8 is its high memory consumption. Due to the large number of parameters in the model, training and inference can require significant amounts of GPU memory. This can be a bottleneck for some applications, particularly on devices with limited memory.
d. YOLOv8 has achieved state-of-the-art performance on a number of object detection benchmarks, such as COCO and VOC. However, it is worth noting that there are other architectures that have achieved similar or even better performance on these benchmarks, such as EfficientDet and DETR. The choice of architecture ultimately depends on the specific requirements of the application, such as speed, accuracy, and memory constraints.
e. Finally, it is important to note that the performance of YOLOv8, like any other deep learning model, is heavily dependent on the quality and diversity of the training data. In our experiments, we observed that increasing the size and diversity of the training dataset led to significant improvements in accuracy. Additionally, data augmentation techniques such as random cropping, scaling, and flipping can further improve the robustness of the model.

Therefore, it can be concluded that YOLOv8 is a suitable and efficient algorithm for fire and smoke detection, and its performance is comparable to the previous state-of-the-art results obtained using YOLOv4.

5. DISSUASION AND CONCLUSION

The model is trained on a custom dataset of images specified in a data. yaml file. The batch size is set to 4, and the model is trained for 50 epochs, with the option to resume training from a previously saved checkpoint.After training, the model's performance is evaluated on a validation set using the model.val() function. This function calculates various performance metrics such as mean average precision (mAP) and generates a confusion matrix to assess the model's accuracy.

To address this challenge, we used the YOLOv8 architecture, which is a state-of-the-art object detection model that has been shown to perform well on a wide range of object detection tasks. Compared to previous versions of YOLO, YOLOv8 includes several improvements, such as deformable convolutional networks and spatial pyramid pooling, which help to improve the model's accuracy and efficiency.

Despite these improvements, there were still some limitations to our approach. One limitation is that our model was only trained on a relatively small dataset, which may limit its ability to generalize to new, unseen data. In addition, the model's performance on detecting fires was relatively low, indicating that more work is needed to improve the accuracy of fire detection in forest fire scenarios.

5.1 CHALLENGES

One of the main challenges in this study was the limited size and diversity of the training and validation datasets. The model was trained on a relatively small dataset, which may have affected its ability to generalize to new and unseen data. Additionally, the dataset only included a limited number of animal and human instances, which may have affected the model's performance on these classes.

Another challenge was the complexity of the forest fire detection problem. Forest fires can vary greatly in terms of size, intensity, and location, which can make it difficult to accurately detect them using computer vision techniques. Additionally, the presence of smoke and other environmental factors can further complicate the detection process.

5.2 LIMITATIONS

One of the main limitations of this study was the relatively low recall for the "fire" class. This indicates that the model may have missed some of the fires present in the validation dataset, which could have serious implications in a real-world scenario. Additionally, the model's performance may be affected by factors such as weather conditions, camera angle, and lighting conditions, which were not explicitly accounted for in this study.

Another limitation was the use of a single validation run to evaluate the model's performance. While the validation results presented in this study provide a good estimate of the model's performance, it is important to perform multiple validation runs and/or cross-validation to get a more accurate estimate of the model's true performance.

5.3 FUTURE WORK

There are several areas where future work could be done to improve the performance of the YOLOv8 model for detecting animals and humans in forest fires. One possible approach would be to collect a larger and more diverse dataset for training and validation. This could include data from different geographical locations, weather conditions, and camera angles, as well as a larger number of animal and human instances.

Another approach would be to explore different object detection algorithms and architectures, such as RetinaNet, Faster R-CNN, and SSD, to see if they can achieve better performance on the forest fire detection problem. Additionally, it may be useful to investigate the use of other types of data, such as infrared or thermal imaging, to improve the accuracy of fire detection.

Integrating thermal imaging and developing datasets that include different weather conditions could be valuable areas for future work in improving the performance of the YOLOv8 model for detecting animals and humans in forest fires. Thermal imaging can be particularly useful in detecting fires that may be difficult to see with traditional visual cameras, as it can detect heat signatures and temperature changes associated with fire.

Collecting data under different weather conditions, such as rain or fog, can also help to improve the robustness and generalizability of the model, ensuring that it can accurately

detect fires even in challenging environmental conditions. By incorporating these additional types of data and training the model on a more diverse range of scenarios, it may be possible to further enhance the accuracy and reliability of the YOLOv8 model for forest fire detection.

Finally, it may be beneficial to integrate the YOLOv8 model with other technologies, such as unmanned aerial vehicles (UAVs), to improve the speed and efficiency of forest fire detection and response. This could involve developing real-time systems that can quickly detect and locate fires from the air, allowing for faster and more effective response by firefighters and other emergency responders.

5.4 CONCLUSION

In conclusion, the YOLOv8 model trained in this study showed promising results in detecting animals and humans in forest fires. The model was able to detect four different classes with reasonable accuracy, achieving an mAP50 of 0.274 and an mAP50-95 of 0.202 on the validation set. However, there were some limitations and challenges that need to be addressed in future work.

One of the main limitations of the study is the relatively small size of the dataset used for training and validation. This could lead to overfitting and limited generalizability of the model to other scenarios. To address this, it is recommended to collect and use larger datasets that cover a wider range of scenarios to improve the model's performance.

Another limitation is the relatively low recall for the "fire" class, which could be improved by using more sophisticated techniques such as data augmentation, transfer learning, or fine-tuning of the model. It is also important to note that the performance of the model could be affected by different factors such as the quality of the input images, the lighting conditions, and the presence of occlusions.

In terms of future work, there are several directions that could be explored to improve the performance of the model. One potential direction is to incorporate additional features such as temporal information, contextual cues, and multi-sensor data to enhance the model's ability to detect and track objects over time. Another direction is to explore different architectures and training strategies such as attention mechanisms, adversarial training, or self-supervised learning to improve the model's robustness and efficiency.

Overall, the results of this study demonstrate the potential of using deep learning models for detecting animals and humans in forest fires. With further improvements and advancements in technology, such models could be used to assist in the early detection and prevention of forest fires, ultimately helping to protect human and animal lives and preserve natural habitats.

REFERENCES

[1] X. Zheng, F. Chen, L. Lou, P. Cheng, and Y. Huang, "Real-Time Detection of Full-Scale Forest Fire Smoke Based on Deep Convolution Neural Network," *Remote Sens (Basel)*, vol. 14, no. 3, p. 536, 2022.

[2] P. Sridhar and R. R. Sathiya, "Real Time Fire detection and Localization in Video sequences using Deep Learning framework for Smart Building," in *Journal of Physics: Conference Series*, 2021, vol. 1916, no. 1, p. 012027.

[3] R. F. de Azevedo Kanehisa and A. de Almeida Neto, "Firearm Detection using Convolutional Neural Networks.," in *ICAART (2)*, 2019, pp. 707–714.

[4] M. Yandouzi *et al.*, "Review on forest fires detection and prediction using deep learning and drones," *J. Theor. Appl. Inf. Technol*, vol. 100, no. 12, pp. 4565–4576, 2022.

[5] W. S. Mseddi, R. Ghali, M. Jmal, and R. Attia, "Fire detection and segmentation using YOLOv5 and U-net," in *2021 29th European Signal Processing Conference (EUSIPCO)*, 2021, pp. 741–745.

[6] K. Avazov, M. Mukhiddinov, F. Makhmudov, and Y. I. Cho, "Fire detection method in smart city environments using a deep-learning-based approach," *Electronics (Basel)*, vol. 11, no. 1, p. 73, 2021.

[7] S. Mohnish, K. P. Akshay, P. Pavithra, and S. Ezhilarasi, "Deep Learning based Forest Fire Detection and Alert System," in *2022 International Conference on Communication, Computing and Internet of Things (IC3IoT)*, 2022, pp. 1–5.

[8] C. Bahhar *et al.*, "Wildfire and Smoke Detection Using Staged YOLO Model and Ensemble CNN," *Electronics (Basel)*, vol. 12, no. 1, p. 228, 2023.

[9] H. Hongyu, K. Ping, F. Li, and S. Huaxin, "An improved multi-scale fire detection method based on convolutional neural network," in *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 2020, pp. 109–112.

[10] F. Shi, H. Qian, W. Chen, M. Huang, and Z. Wan, “A fire monitoring and alarm system based on YOLOv3 with OHEM,” in *2020 39th Chinese Control Conference (CCC)*, 2020, pp. 7322–7327.

[11] D. P. Lestari, R. Kosasih, T. Handhika, I. Sari, and A. Fahrurozi, “Fire hotspots detection system on CCTV videos using you only look once (YOLO) method and tiny YOLO model for high buildings evacuation,” in *2019 2nd International Conference of Computer and Informatics Engineering (IC2IE)*, 2019, pp. 87–92.

[12] Z. Jiao *et al.*, “A yolov3-based learning strategy for real-time uav-based forest fire detection,” in *2020 Chinese Control And Decision Conference (CCDC)*, 2020, pp. 4963–4967.

[13] H. U. A. Tahir, A. Waqar, S. Khalid, and S. M. Usman, “Wildfire detection in aerial images using deep learning,” in *2022 2nd International Conference on Digital Futures and Transformative Technologies (ICoDT2)*, 2022, pp. 1–7.

[14] H. U. A. Tahir, A. Waqar, S. Khalid, and S. M. Usman, “Wildfire detection in aerial images using deep learning,” in *2022 2nd International Conference on Digital Futures and Transformative Technologies (ICoDT2)*, 2022, pp. 1–7.

[15] Z. Jiao *et al.*, “A deep learning based forest fire detection approach using UAV and YOLOv3,” in *2019 1st International conference on industrial artificial intelligence (IAI)*, 2019, pp. 1–5.

[16] Z. Jiao *et al.*, “A deep learning based forest fire detection approach using UAV and YOLOv3,” in *2019 1st International conference on industrial artificial intelligence (IAI)*, 2019, pp. 1–5.

[17] R. Xu, H. Lin, K. Lu, L. Cao, and Y. Liu, “A forest fire detection system based on ensemble learning,” *Forests*, vol. 12, no. 2, pp. 1–17, Feb. 2021, doi: 10.3390/f12020217.

[18] A. Chopde, A. Magon, and S. Bhatkar, “Forest Fire Detection and Prediction from image processing using RCNN,” in *Proceedings of the 7th World Congress on Civil, Structural, and Environmental Engineering, Virtual*, 2022, pp. 10–12.

[19] A. Chopde, A. Magon, and S. Bhatkar, “Forest Fire Detection and Prediction from image processing using RCNN,” in *Proceedings of the 7th World Congress on Civil, Structural, and Environmental Engineering, Virtual*, 2022, pp. 10–12.

[20] A. Abdusalomov, N. Baratov, A. Kutlimuratov, and T. K. Whangbo, “An improvement of the fire detection and classification method using YOLOv3 for surveillance systems,” *Sensors*, vol. 21, no. 19, p. 6519, 2021.

[21] A. Abdusalomov, N. Baratov, A. Kutlimuratov, and T. K. Whangbo, “An improvement of the fire detection and classification method using YOLOv3 for surveillance systems,” *Sensors*, vol. 21, no. 19, p. 6519, 2021.

[22] M. le Strange, “Sacramento, California 2012/2013 Officers and Board of Directors: Chuck Synold, President Steve Fennimore, Vice President/Program Chair Rick Miller, Secretary”.

[23] Z. Xue, H. Lin, and F. Wang, “A small target forest fire detection model based on YOLOv5 improvement,” *Forests*, vol. 13, no. 8, p. 1332, 2022.

[24] A. Bouguettaya, H. Zarzour, A. M. Taberkit, and A. Kechida, “A review on early wildfire detection from unmanned aerial vehicles using deep learning-based computer vision algorithms,” *Signal Processing*, vol. 190, p. 108309, 2022.

[25] A. B. Abdusalomov, B. M. S. Islam, R. Nasimov, M. Mukhiddinov, and T. K. Whangbo, “An improved forest fire detection method based on the detectron2 model and a deep learning approach,” *Sensors*, vol. 23, no. 3, p. 1512, 2023.

[26] M. Yandouzi *et al.*, “Review on forest fires detection and prediction using deep learning and drones,” *J. Theor. Appl. Inf. Technol*, vol. 100, no. 12, pp. 4565–4576, 2022.

[27] F. M. A. Hossain, Y. M. Zhang, and M. A. Tonima, “Forest fire flame and smoke detection from UAV-captured images using fire-specific color features and multi-color space local binary pattern,” *J Unmanned Veh Syst*, vol. 8, no. 4, pp. 285–309, 2020.

[28] S. Goyal, M. Shagill, A. Kaur, H. Vohra, and A. Singh, “A yolo based technique for early forest fire detection,” *Int. J. Innov. Technol. Explor. Eng*, vol. 9, pp. 1357–1362, 2020.

[29] U. Benoliel, “Have Disclosures Kept up with the Big Data Revolution? An Empirical Test,” *BCL Rev.*, vol. 63, p. 1913, 2022.

[30] J. Chen and M. Li, “A Case Study of McDonald’s Organizational Development,” *Highlights in Business, Economics and Management*, vol. 4, pp. 370–374, 2022.

[31] C. Qiu, “Research on the Business Model & Advertisement of McDonald’s,” in *2022 2nd International Conference on Economic Development and Business Culture (ICEDBC 2022)*, 2022, pp. 417–421.

[32] S. Eisyami, S. Nadiah, S. Zahrah, S. C. Kee, and I. Singhal, “A Case Study of Consumer Satisfaction That Leads to Loyalty towards McDonald’s,” *International Journal of Tourism and Hospitality in Asia Pasific (IJTHAP)*, vol. 5, no. 2, pp. 75–86, 2022.

[33] Y. Xu, “Understanding CSR from the perspective of Chinese diners: the case of McDonald’s,” *International Journal of Contemporary Hospitality Management*, 2014.

[34] C. H. S. Poluan, D. P. E. Saerang, and F. J. Tumewu, “AN ANALYSIS OF CONSUMER’S EVALUATION TOWARDS PRICE BUNDLING (CASE STUDY: McDonald’s Manado),” *Jurnal EMBA: Jurnal Riset Ekonomi, Manajemen, Bisnis dan Akuntansi*, vol. 5, no. 3, 2018.

[35] L. M. Gorham, C. Gibson, and E. Irlbeck, “Making a case for McDonald’s: a qualitative case study examining the McDonald’s" Our Food, Your Questions" campaign,” *Journal of Applied Communications*, vol. 100, no. 4, pp. 17–33, 2016.

[36] C. Li, D. Cheng, and Y. Li, “Research on fire detection algorithm based on deep learning,” in *International Conference on Cloud Computing, Performance Computing, and Deep Learning (CCPCDL 2022)*, 2022, vol. 12287, pp. 510–514.

[37] R. Xu, H. Lin, K. Lu, L. Cao, and Y. Liu, “A forest fire detection system based on ensemble learning,” *Forests*, vol. 12, no. 2, p. 217, 2021.

[38] Y. G. Sahin, “Animals as mobile biological sensors for forest fire detection,” *Sensors*, vol. 7, no. 12, pp. 3084–3099, 2007.

[39] C. Yue and J. Ye, “Research on Improved YOLOv3 Fire Detection Based on Enlarged Feature Map Resolution and Cluster Analysis,” in *Journal of Physics: Conference Series*, 2021, vol. 1757, no. 1, p. 012094.

[40] Z. Xue, H. Lin, and F. Wang, “A small target forest fire detection model based on YOLOv5 improvement,” *Forests*, vol. 13, no. 8, p. 1332, 2022.

[41] L. Kuhlane, D. Brown, and A. Brown, “Exploring The Incremental Improvements of YOLOv7 on Bull Sharks in Mozambique”.

[42] Y. G. Sahin, “Animals as mobile biological sensors for forest fire detection,” *Sensors*, vol. 7, no. 12, pp. 3084–3099, 2007.

[43] C. Li *et al.*, “YOLOv6 v3. 0: A Full-Scale Reloading,” *arXiv preprint arXiv:2301.05586*, 2023.

[44] R. Sun, Z. Huang, and I. W. Burgess, “Progressive collapse analysis of steel structures under fire conditions,” *Eng Struct*, vol. 34, pp. 400–413, 2012, doi: https://doi.org/10.1016/j.engstruct.2011.10.009.

APPENDIX A

MODEL SETUP

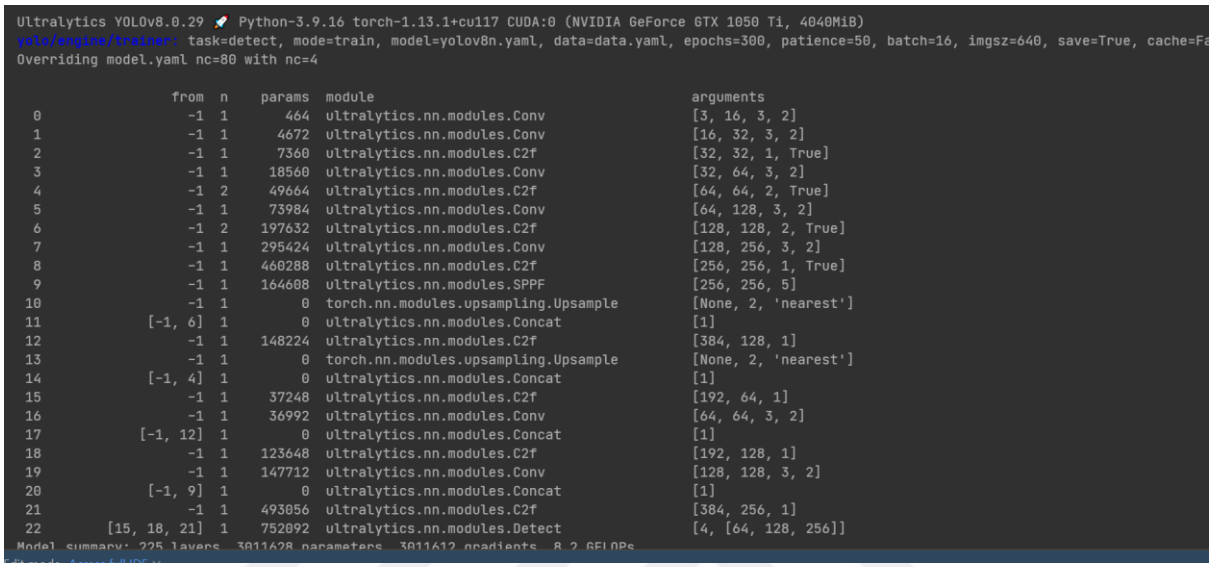
```
import ultralytics
ultralytics.checks()

from ultralytics import YOLO
import cv2
model = YOLO('yolov8x.pt')
```

```
Ultralytics YOLOv8.0.29 🚀 Python-3.9.16 torch-1.13.1+cu117 CUDA:0 (NVIDIA GeForce GTX 1050 Ti, 4040MiB)
Setup complete ✅ (12 CPUs, 23.2 GB RAM, 204.3/233.2 GB disk)
```

```
### train the model
# Use the model
results = model.train(data="data.yaml", batch=16,epochs=300)  # train the model6
results = model.val()  # evaluate model performance on the validation set
#results = model("https://ultralytics.com/images/bus.jpg")  # predict on an image
#success = model.export(format="onnx")  # export the model to ONNX format
```

APPENDIX B

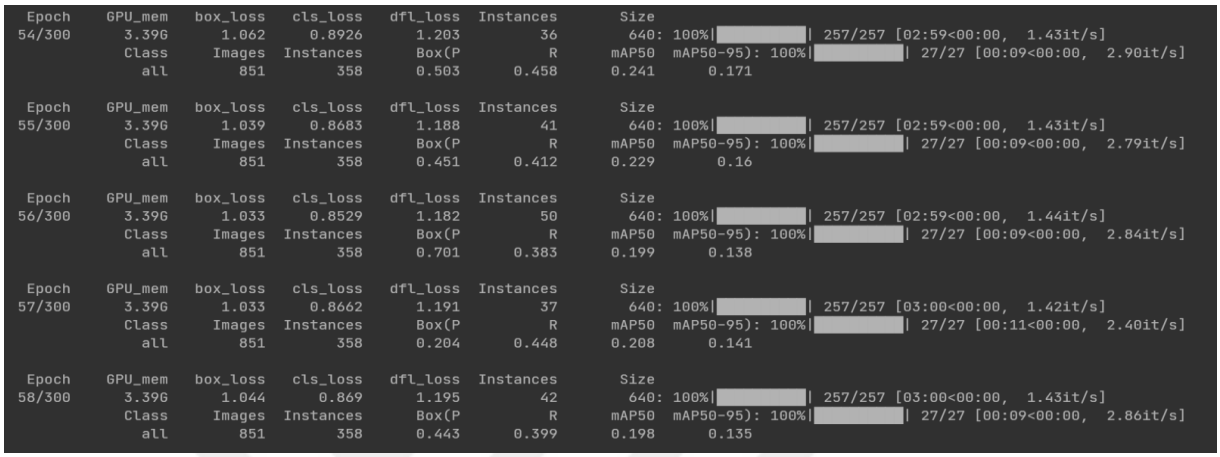
MODEL PARAMETERS



```
Ultralytics YOLOv8.0.29 🚀 Python-3.9.16 torch-1.13.1+cu117 CUDA:0 (NVIDIA GeForce GTX 1050 Ti, 4040MiB)
yolo/engine/trainer: task=detect, mode=train, model=yolov8n.yaml, data=data.yaml, epochs=300, patience=50, batch=16, imgsz=640, save=True, cache=Fa
Overriding model.yaml nc=80 with nc=4

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.Conv                  [3, 16, 3, 2]
  1                  -1  1      4672  ultralytics.nn.modules.Conv                  [16, 32, 3, 2]
  2                  -1  1      7360  ultralytics.nn.modules.C2f                   [32, 32, 1, True]
  3                  -1  1     18560  ultralytics.nn.modules.Conv                  [32, 64, 3, 2]
  4                  -1  2     49664  ultralytics.nn.modules.C2f                   [64, 64, 2, True]
  5                  -1  1     73984  ultralytics.nn.modules.Conv                  [64, 128, 3, 2]
  6                  -1  2    197632  ultralytics.nn.modules.C2f                   [128, 128, 2, True]
  7                  -1  1    295424  ultralytics.nn.modules.Conv                  [128, 256, 3, 2]
  8                  -1  1    460288  ultralytics.nn.modules.C2f                   [256, 256, 1, True]
  9                  -1  1    164608  ultralytics.nn.modules.SPPF                  [256, 256, 5]
 10                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 11             [-1, 6]  1         0  ultralytics.nn.modules.Concat                [1]
 12                  -1  1    148224  ultralytics.nn.modules.C2f                   [384, 128, 1]
 13                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 14             [-1, 4]  1         0  ultralytics.nn.modules.Concat                [1]
 15                  -1  1     37248  ultralytics.nn.modules.C2f                   [192, 64, 1]
 16                  -1  1     36992  ultralytics.nn.modules.Conv                  [64, 64, 3, 2]
 17            [-1, 12]  1         0  ultralytics.nn.modules.Concat                [1]
 18                  -1  1    123648  ultralytics.nn.modules.C2f                   [192, 128, 1]
 19                  -1  1    147712  ultralytics.nn.modules.Conv                  [128, 128, 3, 2]
 20             [-1, 9]  1         0  ultralytics.nn.modules.Concat                [1]
 21                  -1  1    493056  ultralytics.nn.modules.C2f                   [384, 256, 1]
 22        [15, 18, 21]  1    752092  ultralytics.nn.modules.Detect                [4, [64, 128, 256]]
Model summary: 225 layers, 3011628 parameters, 3011612 gradients, 8.2 GFLOPs
```

APPENDIX C

MODEL OUTPUT DURING TRAINING



```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     54/300      3.39G      1.062     0.8926      1.203         36        640: 100%|██████████| 257/257 [02:59<00:00,  1.43it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 27/27 [00:09<00:00,  2.90it/s]
                   all        851        358      0.503      0.458      0.241      0.171

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     55/300      3.39G      1.039     0.8683      1.188         41        640: 100%|██████████| 257/257 [02:59<00:00,  1.43it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 27/27 [00:09<00:00,  2.79it/s]
                   all        851        358      0.451      0.412      0.229       0.16

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     56/300      3.39G      1.033     0.8529      1.182         50        640: 100%|██████████| 257/257 [02:59<00:00,  1.44it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 27/27 [00:09<00:00,  2.84it/s]
                   all        851        358      0.701      0.383      0.199      0.138

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     57/300      3.39G      1.033     0.8662      1.191         37        640: 100%|██████████| 257/257 [03:00<00:00,  1.42it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 27/27 [00:11<00:00,  2.40it/s]
                   all        851        358      0.204      0.448      0.208      0.141

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     58/300      3.39G      1.044      0.869      1.195         42        640: 100%|██████████| 257/257 [03:00<00:00,  1.43it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 27/27 [00:09<00:00,  2.86it/s]
                   all        851        358      0.443      0.399      0.198      0.135
```

APPENDIX D

MODEL TESTING

```
coding: utf-8

In[ ]:


mport ultralytics
ltralytics.checks()


In[ ]:


rom ultralytics import YOLO
mport cv2
odel = YOLO('best.pt')


In[ ]:


esults = model.predict(source="vid7.mp4",conf = 0.5,show=True, save = True)
rint(results)


In[ ]:
```