Let's Start

Fire and Smoke Detection on edge devices

4th year multidisciplinary project

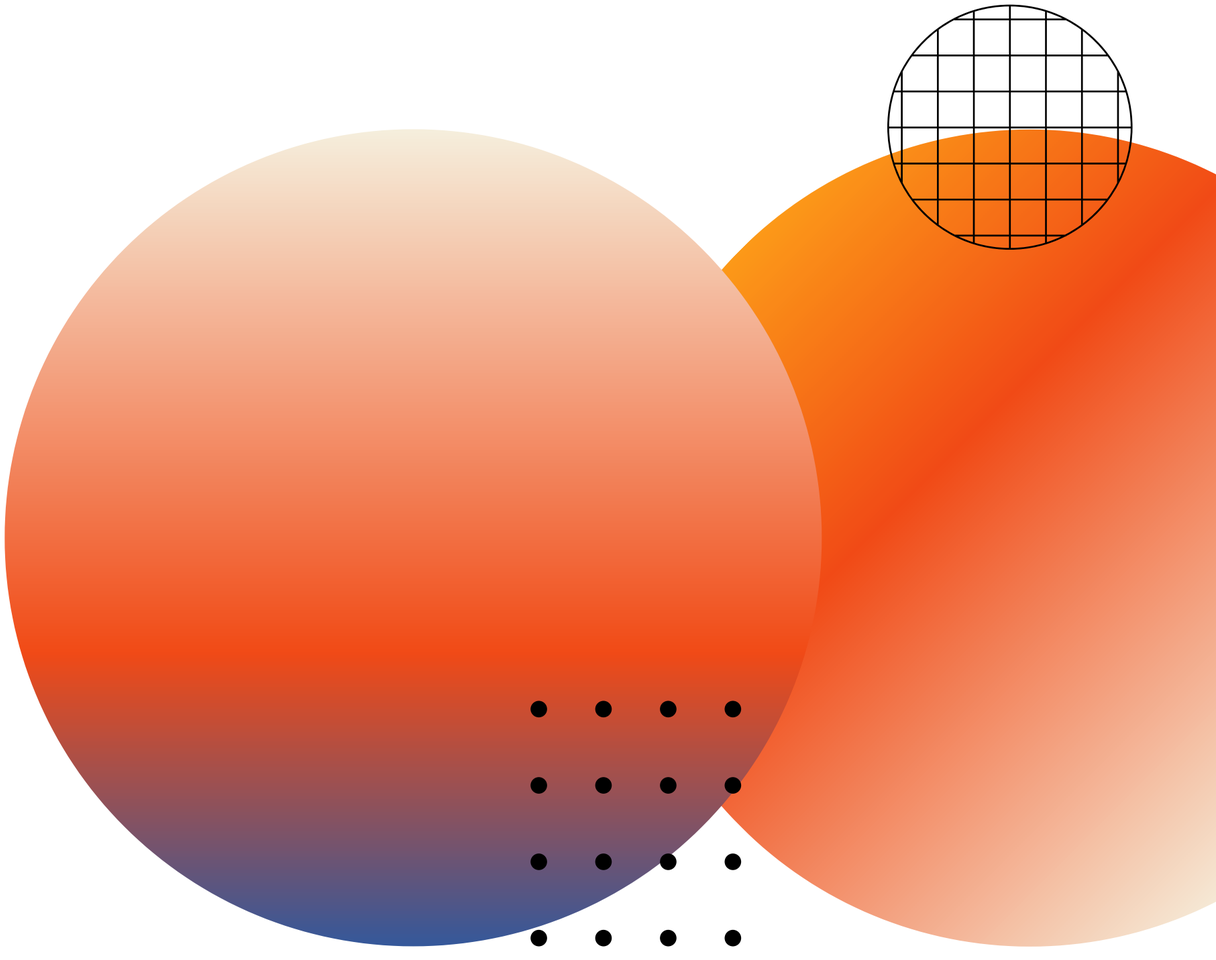


Table of contents

Introduction

In order to reduce damage and ensure people's safety, fire and smoke incident detection is essential. However, classic fire detection methods that rely on physical sensors frequently have issues like false alarms and slow response times. Advanced technologies that make use of deep learning models are required to solve these problems by offering accurate, quick, and portable solutions for real-time fire and smoke detection.

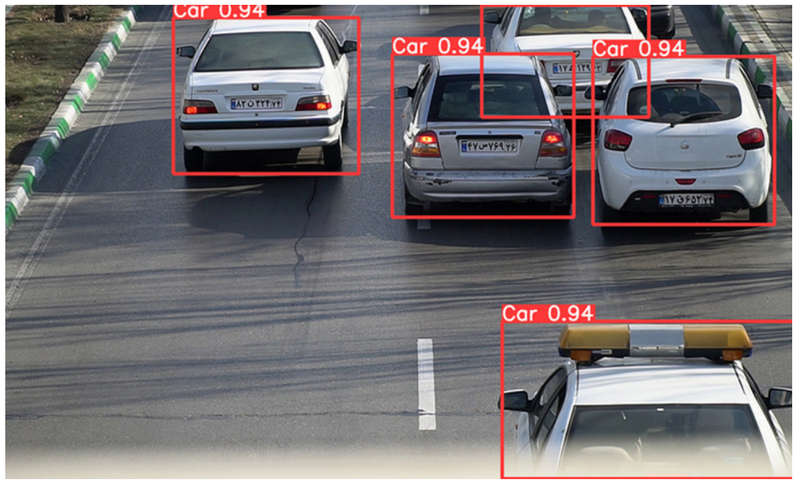
Limitations of Traditional Approaches

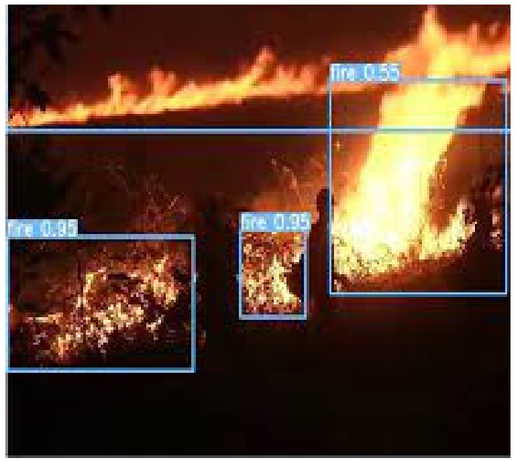
Traditional methods rely mainly on Physical Sensors which presents some limitations

- False Alarms
- Threshold Challenges
- Delayed Alarm Response

Our approch

Our fire and smoke detection is mainly about classification and object detection.





Tools

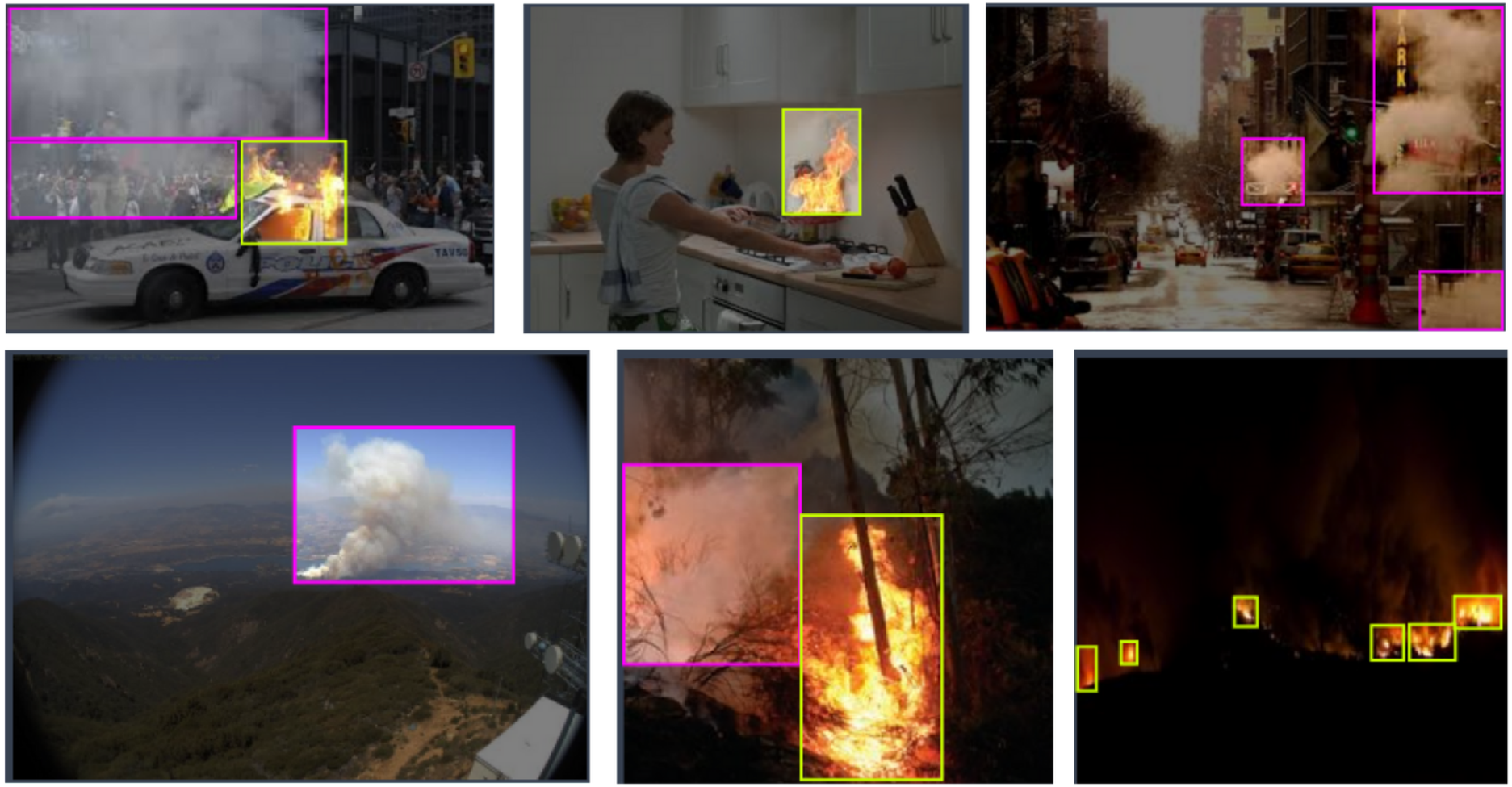
- Dataset
- Hardware components
- Technologies used

The dataset

We used The ”Wildfire and smoke” data set, sourced from Roboflow contains 3,255 images and 6,825 annotations
This dataset is diverse and realist, it contains images variations in environmental factors, such as lighting conditions, weather conditions, and camera perspectives.

3K
Images

6K
Annotations



Few images from the dataset

Hardware components

Raspberry Pi 3 +Pi Camera

- The Raspberry Pi offers real-time detection capabilities

- Contains a camera module which make it suitable for live streaming applications.

Hardware components

Jetson Nano

- The Jetson Nano is a small, powerful, and affordable single-board computer
- The Jetson Nano supports popular AI frameworks like TensorFlow, PyTorch. It is specifically built for edge computing and AI applications.

Technologies

Models

- CNN
- YOLOv4
- YOLOv5
- YOLOv7
- YOLOv8

State of the art

Object detection

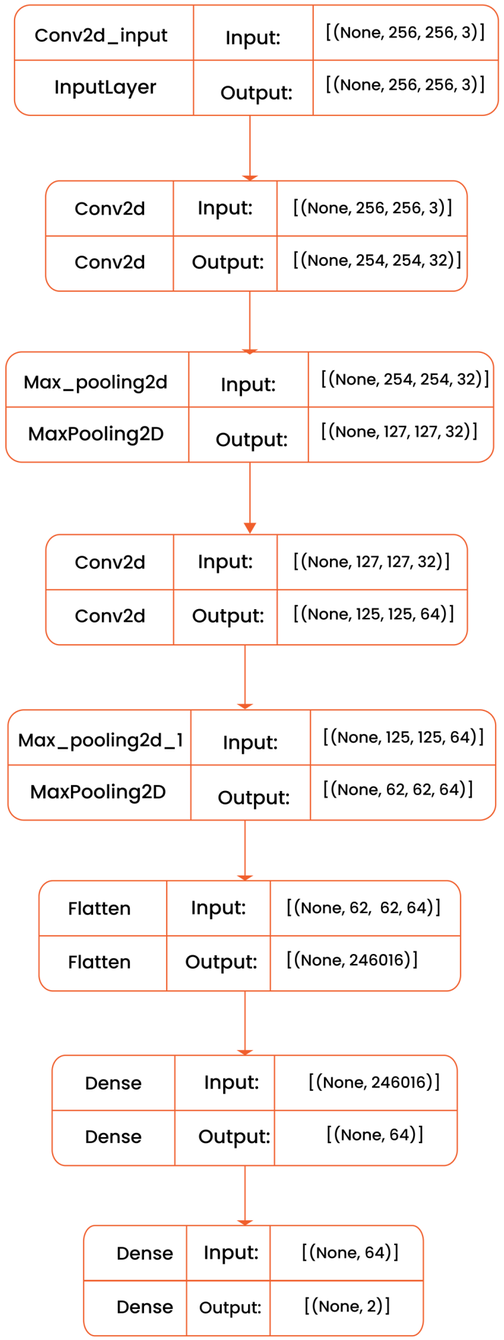
One stage models

- YOLO

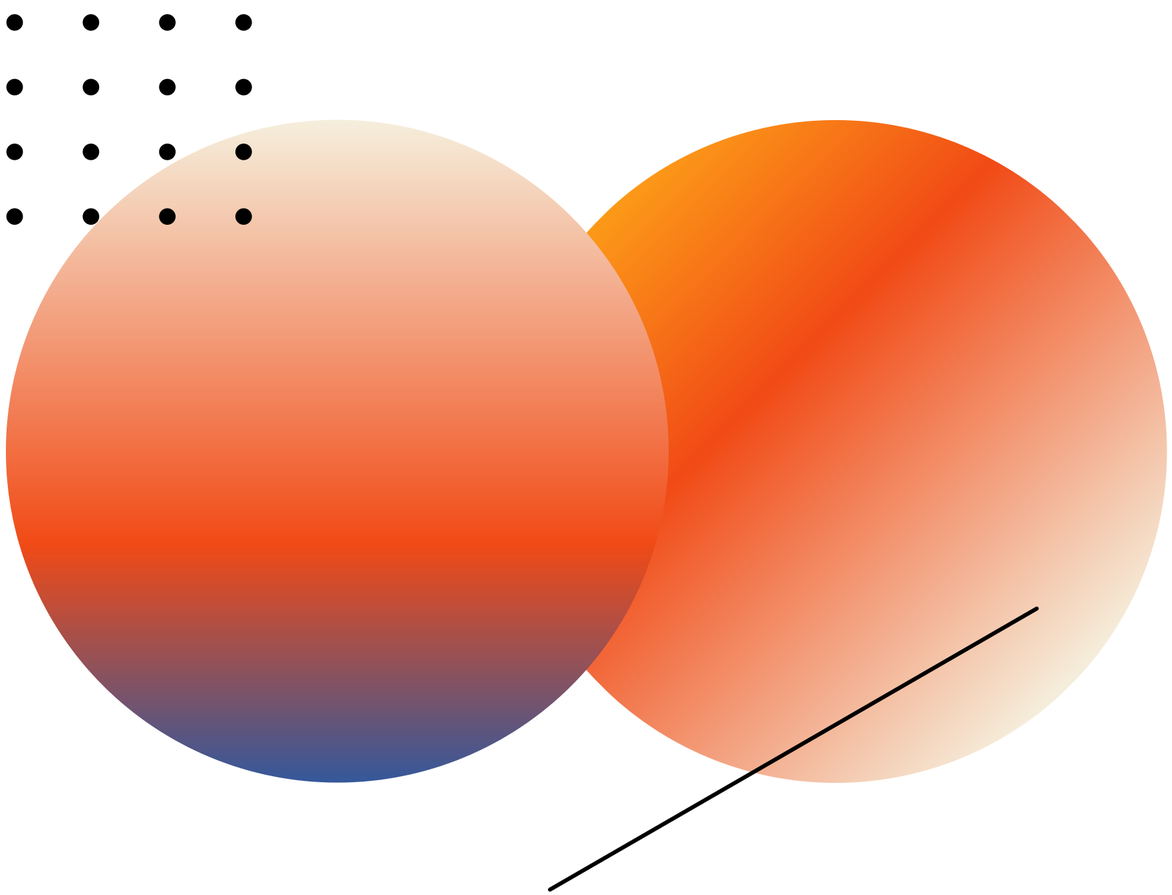
Two stage models

- Faster R-CNN

CNN



- The model architecture includes convolutional layers, pool-ing layers, and dense layerst
- The model is compiled using the Adam optimizer and cross-entropy loss
- The model was trained for 50 epochs with a batch size of 64
- The model achieved an accuracy of 96% on the test set



Why YOLO ?

1. a popular object detection model known for its speed and accuracy
2. single-shot object detection is better suited for real-time applications
3. Flexibility and Adaptability

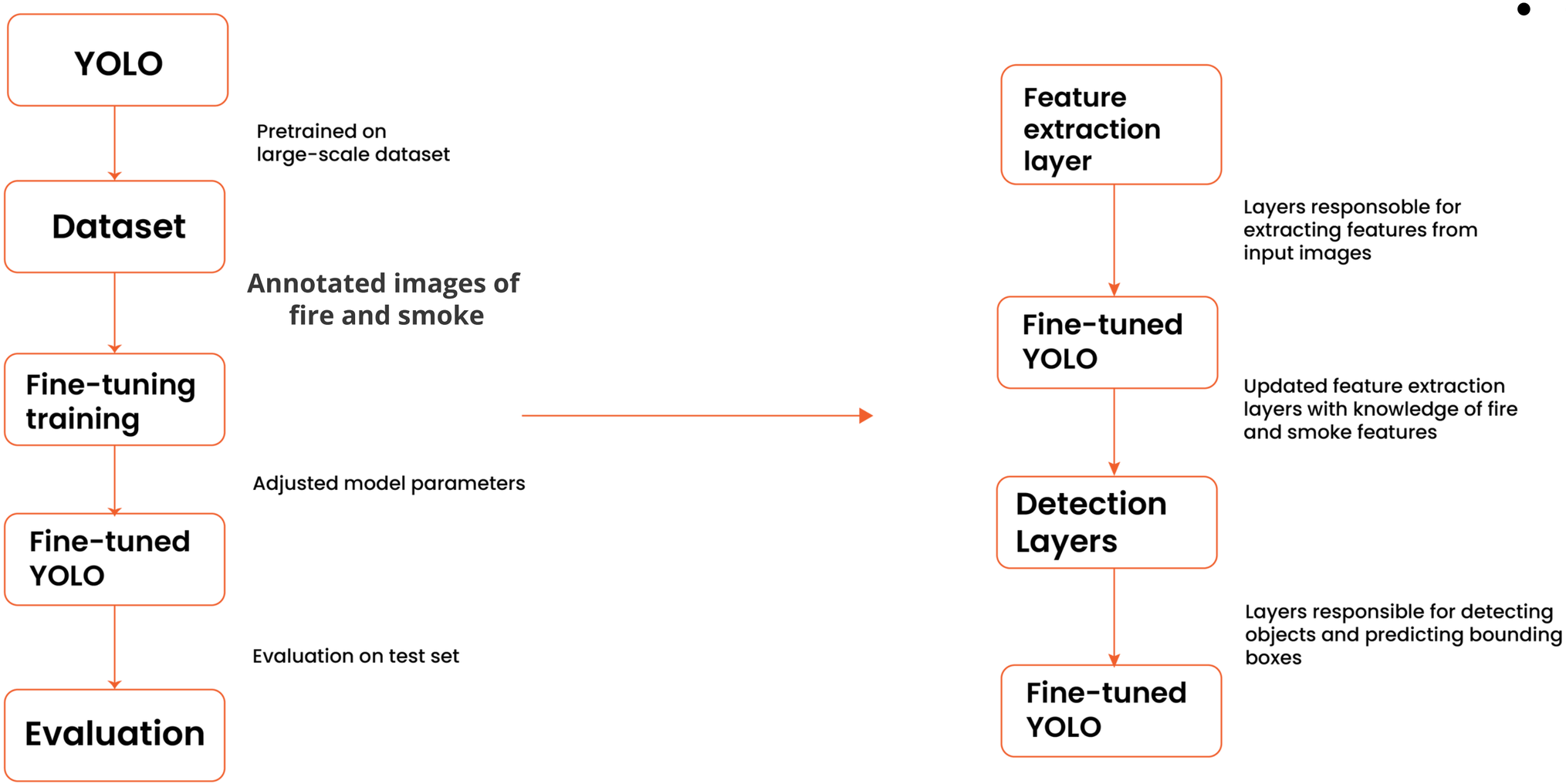
The Global Yolo Architecture

Backbone

Feature extraction Layer

Detection layer

YOLO fine-tuning



YOLOv4

- include the introduction of the CSPNet architecture
- utilizes k-means clustering for generating anchor boxes

	learning Rate	Batch Size	trainning epochs	Confidence Threshold
YOLOv4	0.001	32	50	0.5

YOLOv5

- introduces a more complex architecture(EfficientDet)
- incorporates spatial pyramid pooling (SPP) for improved detection of small objects
- introduces CIoU loss to handle imbalanced datasets

	learning Rate	Batch Size	trainning epochs	Confidence Threshold
YOLOv5	0.01	32	30	0.48

YOLOv7

- introduces the use of anchor boxes for improved object detection and reduces false positives.
- utilizes focal loss to handle small objects and focuses on hard examples

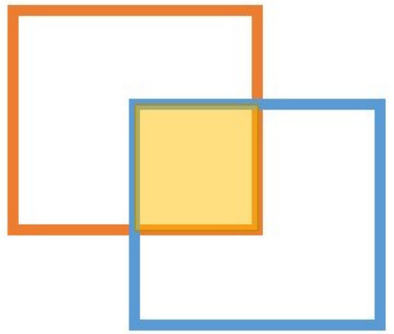
	learning Rate	Batch Size	trainning epochs	Confidence Threshold
YOLOv7	0.001	32	50	0.52

YOLOv8

- YOLO v8 boasts of a new API that will make training and inference much easier on both CPU and GPU devices .
- YOLOV8 uses the Darknet-53 backbone network.

	learning Rate	Batch Size	trainning epochs	Confidence Threshold
YOLOv7	0.001	32	25	0.6

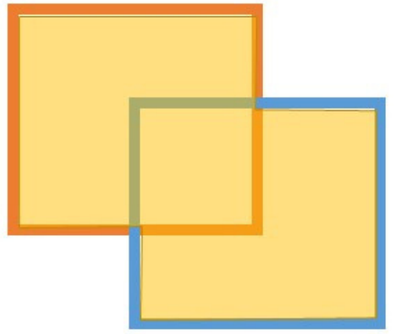
Metrics



$$Intersection\ over\ Union\ (IoU) = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$

Prediction

Ground-truth



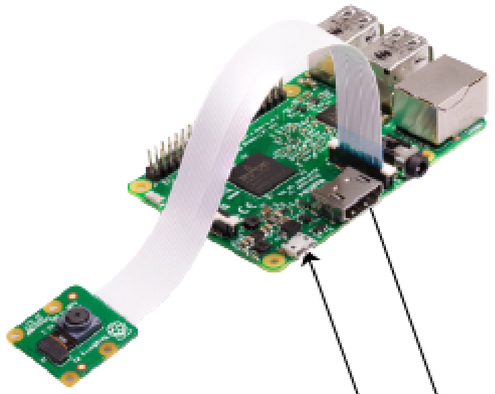
Discussion

The mAP is calculated by finding Average Precision(AP) for each class and then average over a number of classes.

	MAP
YOLOv4	0.54
YOLOv5	0.43
YOLOv7	0.59
YOLOv8	0.63

System architecture

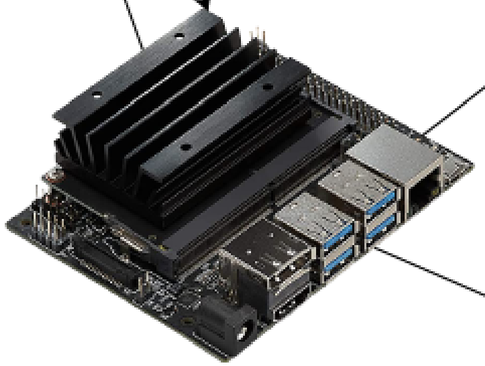
Raspberry pi



Raspberry pi camera

Camera Feed

Streaming



Jetson Nano

Camera Feed Processing and Inference

Frames

Frames

Fire and Smoke Detection Module

Model Training

Fire Detection and Segmentation

Fire and Smoke Coordinates

Demo

Perspectives and conclusion

Distance estimation

Size estimation

Enhancing the security aspects

Distance estimation

explored the possibility of training a dist-YOLO model for calculating distances

Size estimation

analyze the fire region and infer its dimensions or intensity

Enhancing the security aspects

implementing measures to protect the network communication between the devices

securing the storage and transmission of captured video data

implementing authentication and authorization mechanisms to control access to the system

Conclusion

This project has created a real-time fire and smoke detection system that can be used with embedded devices like the JetsonNano and Raspberry Pi. It has been demonstrated that the system's architecture, which combines the strength of these platforms with theYOLOv4, YOLOv7, and YOLOv8 models, is efficient in detecting fire and smoke cases

End

Thank you

Do you have any questions?