Smoke and fire detection using smart cameras

By Shrivas Sudharsan

01. **The Problem**

Why is fire damage a big deal?

02. **Related Works**

What people have done?

03. **Our approach?**

What our project entails?

04. **IoT**

How IoT works?

05. **Project Idea**

Methodology + Results

06. **Conclusion**

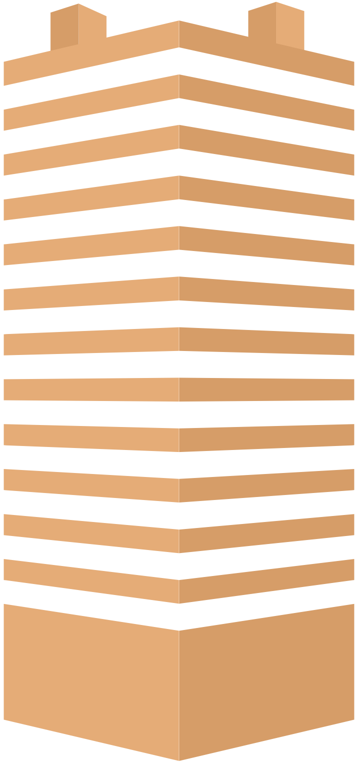
What do the results mean? Next steps?

The Problem

- Only 53% of fires have a soundable alarm
- Every 1000 homes fires, 12.3 cause deaths simply because there is no alarm system or it doesn’t work
- Whereas homes that have alarm systems only have a 5.7 death rate every 1000 homes, halving the damage caused
- Destruction and lives lost
- Inevitable

Why does it matter?

- Working smoke alarm significantly decreases chances of death
- In less than 30 SECONDS, a small fire can turn into a major one, and become deadly within two minutes
- Temperatures can rise to 600 degrees and become extremely smoky
- Most deaths are caused by the pollution, not the fire itself
  - 1.8 billion tons of CO_2 pollution caused by wildfires



What has been done

SmokeD

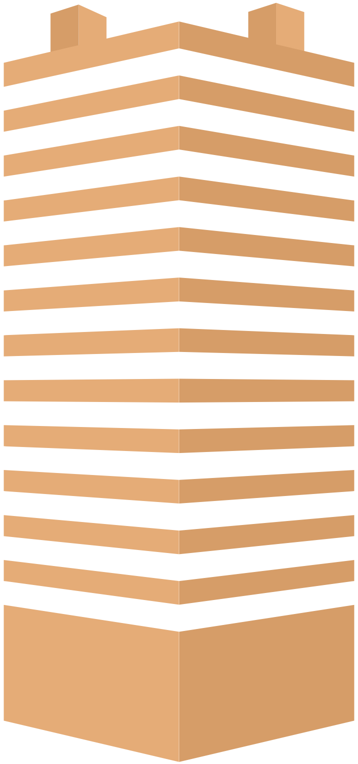
Machine learning based cameras used to detect forest fires in 10 minutes for up to 10 miles away from the location of the camera

FLIR

These are thermal imaging infrared cameras which are capable of detecting heat. MoviTherm has used these cameras to build an automated sprinkler in large scale warehouses.

AVIOTEC

Starlight 800 model uses artificial intelligence and optical analyses to detect fire in warehouses and in dark areas with little to no light



Related Works

Yolov4

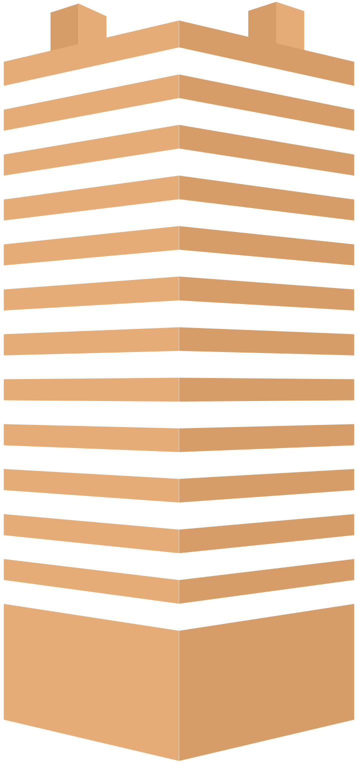
Sergio performed a study using a convolutional neural network to detect fires.

Forest Fires

A study by Krishnamoorthy uses 5 different sensors and an arduino kit to detect smoke. Considers both temperature and carbon dioxide levels

CNN

Uses denoising convolutional neural network for classifying smoke and optimizes computational cost for IoT



Our Approach

Smart Cameras

We will use a raspberry pi based implementation to capture an image, then be able to detect a fire allowing data to be stored locally and handle computation in its microprocessing

Artificial Intelligence - YOLO

The “You only look once” image detection algorithm uses machine learning models to detect a possible fire and report its accuracy.

Adjustments

Each visual detection will have a level of “confidence” in detecting a fire/smoke. We will adjust the parameters of YOLO to get the best possible camera to detect images.

An IoT based model

Local box

These have smart cameras with microprocessors capable of storing data.

Local computing

A microprocessor uses a YOLO algorithm to report detection/classification of a fire/smoke

How does data get sent

IoT gateway

Data travels from the sensor to the cpu

Raspberry Pi

Contains lager processing that runs machine learning

Wi-Fi

Due to indoor settings, we can use wifi to transfer data

IoT gateway

- IoT system connects to the servers IP address and sends data
- The microprocessor collects the data from the sensor and sends itself through a gateway
- WIFI
- The sensor layer collects data from the field
- The network layer is where this data is transmitted to the system
- The edge layer is responsible for machine learning analysis.

Our project

Sustainability

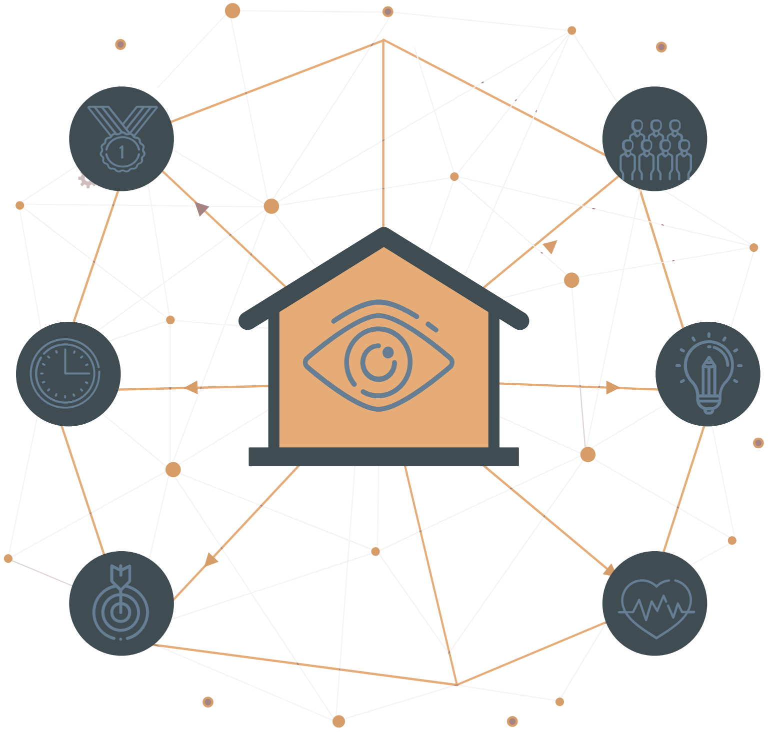
IoT allows for self-sufficient remote communication

Versatility

By testing various algorithms, we can train cameras for various images while maintaining confidence levels

Accurate

Using deep learning YOLO detection.



Efficiency

Smart cameras can pick up a fire 2 hours before a smoke can detect it (~10 miles)

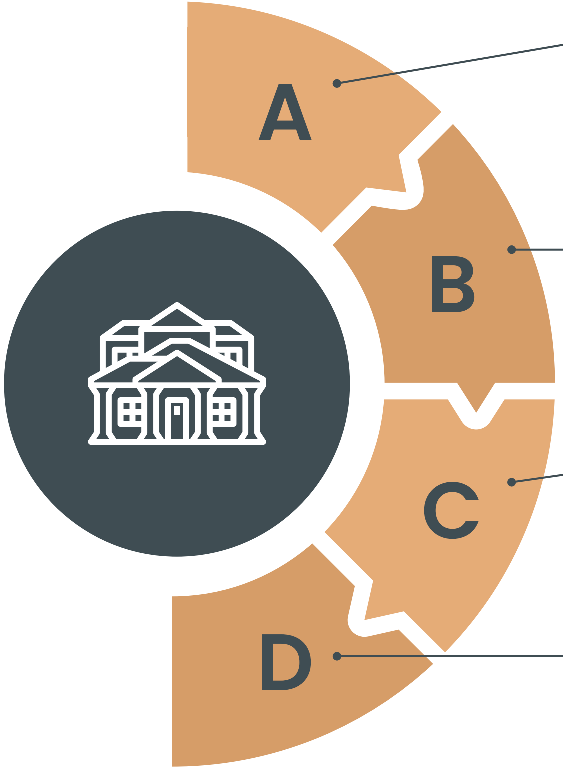
Innovative

Follows the trend of previous research, looking for automation in environmental concerns

Necessary

Using machine learning to detect cameras is capable of saving lives. The faster the response the better

Logistics



Fundamental

Learn from previous research the benefits of IoT, how it works, and what are its implications for the future.

Development

YOLOv8, learn from other papers

Fine Tuning

Try to improve confidence and accuracy values during testing

Analysis

What do our results entail? What could have been improved in our experiment?

Project Plan

- Goal: Design a learning based fire and smoke detection on smart cameras that are deployed in smart spaces
- Smart spaces: indoor + outdoor
- Smart cameras have quick vision detect and are versatile
- Machine learning: important tool which can be used in any sector
- Vision based successfully with fire and smoke
- YOLOv8: common
- Suitable for embedded devices + smart cameras

Project Plan

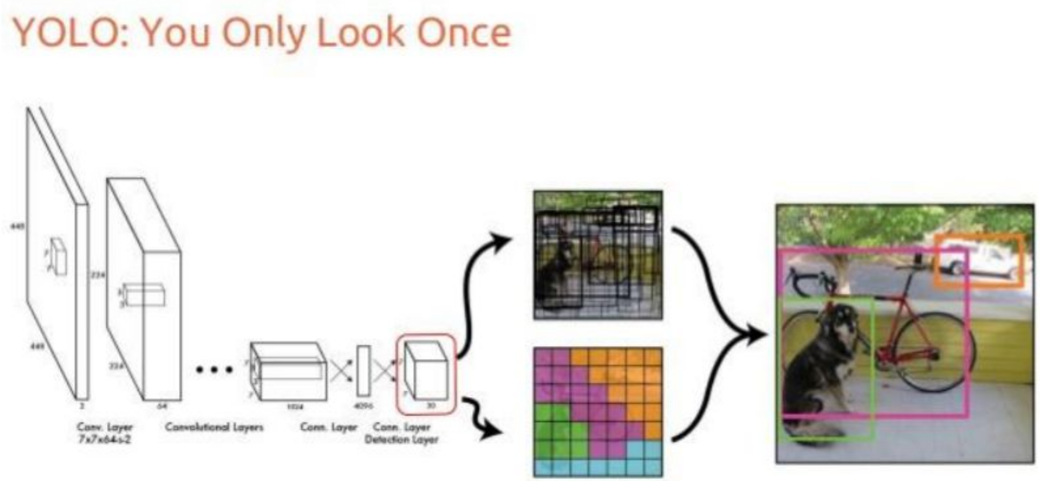
- Manual smoke sensors lack effectiveness in outdoor settings
- Camera implementation detects farther and faster
- Implementing both smoke and fire vision hurdles that obstacle
- Can be used in both indoor and outdoor settings
- Visual allows for depth + location tracking
- IoT based requires low computational power
- Smaller data, less powerful algorithms
- YOLO is typically used.

YOLOv8

- The “You Only Look Once” algorithm: machine learning
- Classification and detection
- It uses a single convolutional neural network
- Trained on large datasets using supervised machine learning
- We selected YOLO because it performs classification and detection of images with real-time detection and high accuracy
- YOLO is used outside of simple classification; it can also be used in image and instance segmentation
- Open Source

YOLO algorithm

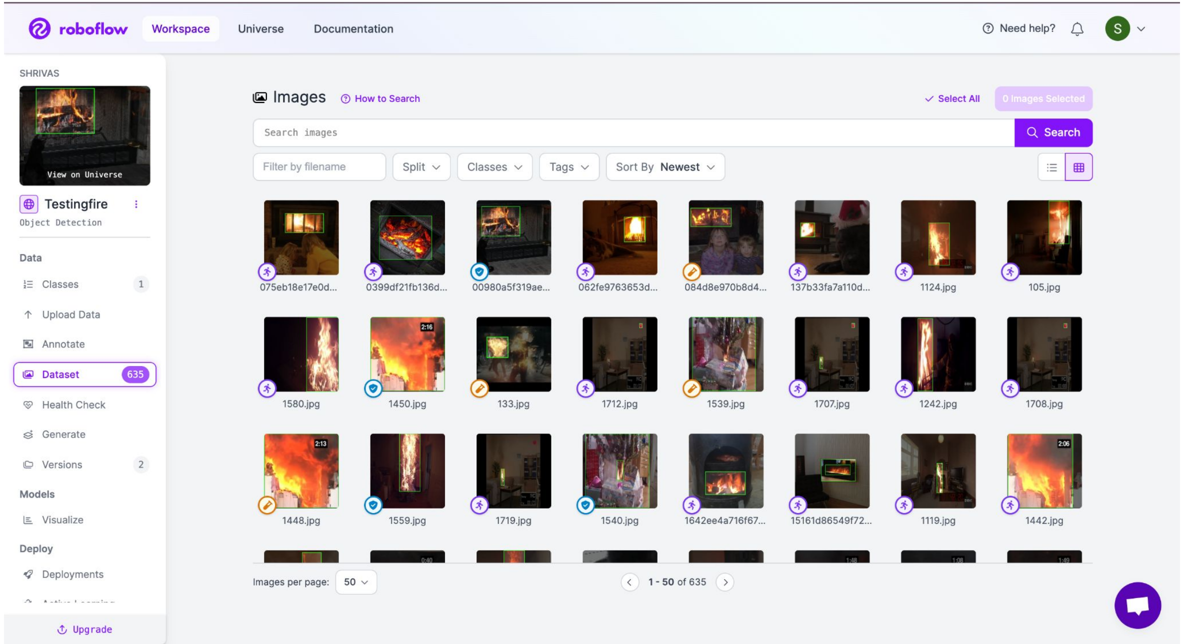
- Divides input images into grids and predicts bounding boxes and probabilities for each cell
- Learns from previous annotated bounding boxes



Fire and Smoke Datasets

- Online google images (kaggle + roboflow)
- Not pretrained
- Gather various sizes
- We want indoor, outdoor, small, and large fires
- Fires that include smoke as well for second dataset
- With individual smoke images for smoke implementation
- Upload images to roboflow and annotate with boxes.
- Translate to labels

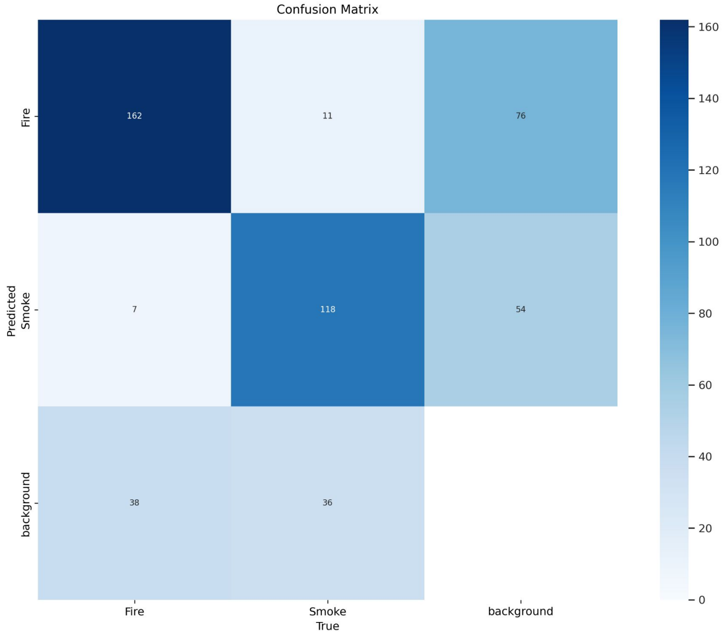
Fire and Smoke Datasets



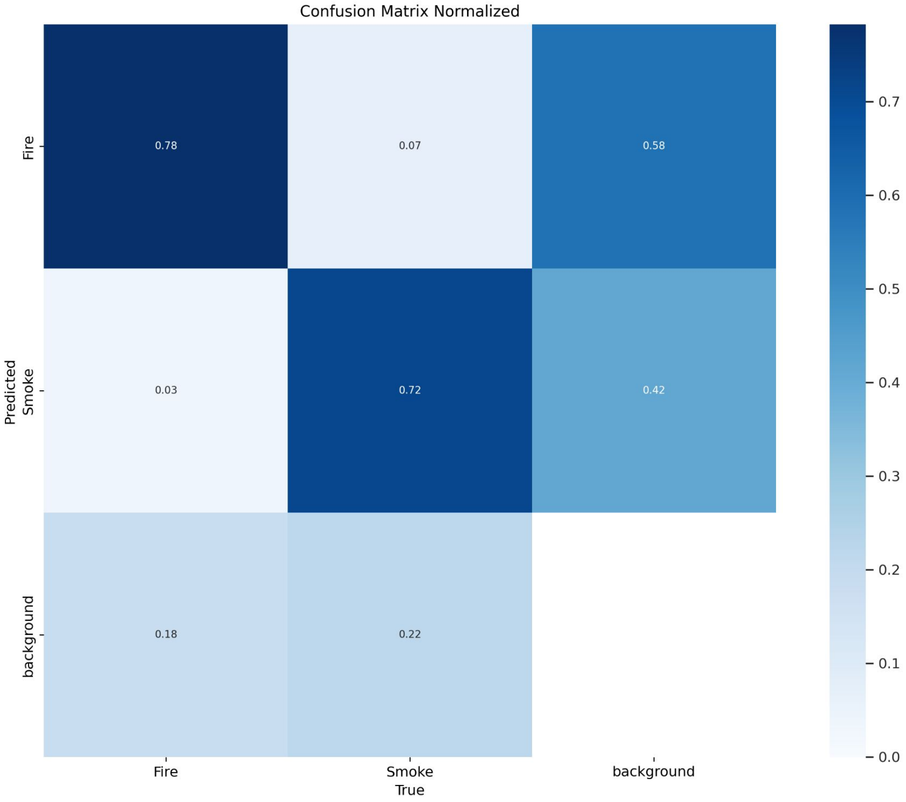
Training

- Parameterize the algorithm
- Specifically number of epochs: iterations
- Use validation phase to see results of data
- Tweak based on those results
- Metrics: precision, recall, f1, accuracy, mean average precision
- Confidence levels
- Speed of processing
- Difference between smoke and fire in terms of analysis

Training


Confusion Matrix
Fire
Smoke
background
Predicted
True
162
11
76
7
118
54
38
36
160
140
120
100
80
60
40
20
0

Training



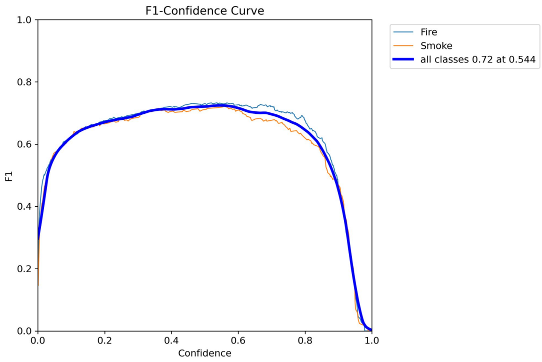
Confusion Matrix Normalized
Fire
Smoke
background
Predicted
True
0.78
0.07
0.58
0.03
0.72
0.42
0.18
0.22
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7

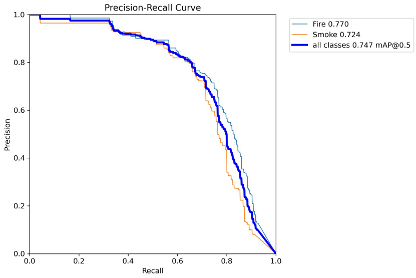
Training

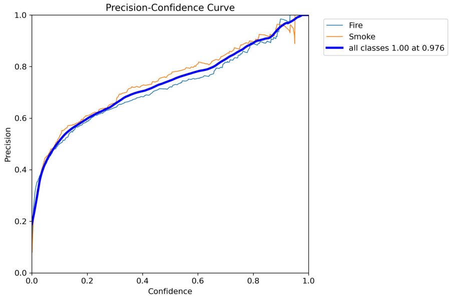

Smoke 0.9
Fire 0.9
Fire 0.6
Fire 0.4
Smoke 0.9
Fire 0.9
Fire 0.9
Fire 0.9
Smoke 0.8
Fire 0.9
SmokeSmoke 0.8
Fire 0.7
Fire 0.5


Fire
Smoke
Fire
Fire
Smoke
Fire
Smoke
Smoke
Fire
Smoke
Smoke
Fire
Smoke
Fire
Smoke
Fire

Graphs


F1-Confidence Curve
Fire
Smoke
all classes 0.72 at 0.544
F1
Confidence


Precision-Recall Curve
Fire 0.770
Smoke 0.724
all classes 0.747 mAP@0.5
Precision
Recall


Precision-Confidence Curve
Fire
Smoke
all classes 1.00 at 0.976
Precision
Confidence

Speed

```
val: Scanning /content/Fire-and-smoke-1/valid/labels.cache... 185 images, 0 backgrounds, 0 corrupt:
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 12/12
                   all        185        372      0.762      0.692      0.747       0.53
                  Fire        185        207      0.738      0.722       0.77      0.522
                 Smoke        185        165      0.785      0.661      0.724      0.539
Speed: 1.8ms preprocess, 6.5ms inference, 0.0ms loss, 6.2ms postprocess per image
Results saved to runs/detect/val
💡 Learn more at https://docs.ultralytics.com/modes/val
```

```
reated: /content/Fire-and-smoke-3/valid/labels.cache
 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%
   all        485        890      0.867       0.79      0.865      0.693
  Fire        485        507      0.856      0.826      0.893      0.714
 Smoke        485        383      0.878      0.753      0.838      0.671
process, 4.9ms inference, 0.0ms loss, 3.2ms postprocess per image
```

Results

Confidence

Fire and Smoke: Fire 90.2%, Smoke 85.02%

Fire: 85.1%

Accuracy

Fire and Smoke: Fire 92.8%, Smoke 76.1%

Fire: 85.8%

Results (Fire + Smoke)

Fire

Validation Set Size (# of images)	Precision	Recall	MAP
185	0.738	0.722	0.77
301	0.778	0.746	0.81
485	0.856	0.826	0.893

Smoke

Validation Set Size (# of images)	Precision	Recall	MAP
185	0.785	0.661	0.77
301	0.821	0.677	0.758
485	0.878	0.758	0.838

Results (Fire + Smoke)

Combined

Validation Set Size (# of images)	Precision	Recall	MAP
185	0.762	0.692	0.747
301	0.799	0.712	0.784
485	0.867	0.79	0.865

Time for computation on raspberry pi:

1.99944 s per image
(realtime)

Thank You for listening