

AI & ML - PROJECT

# SMART EDGE LENS AND CLOUD OPTIMIZATION

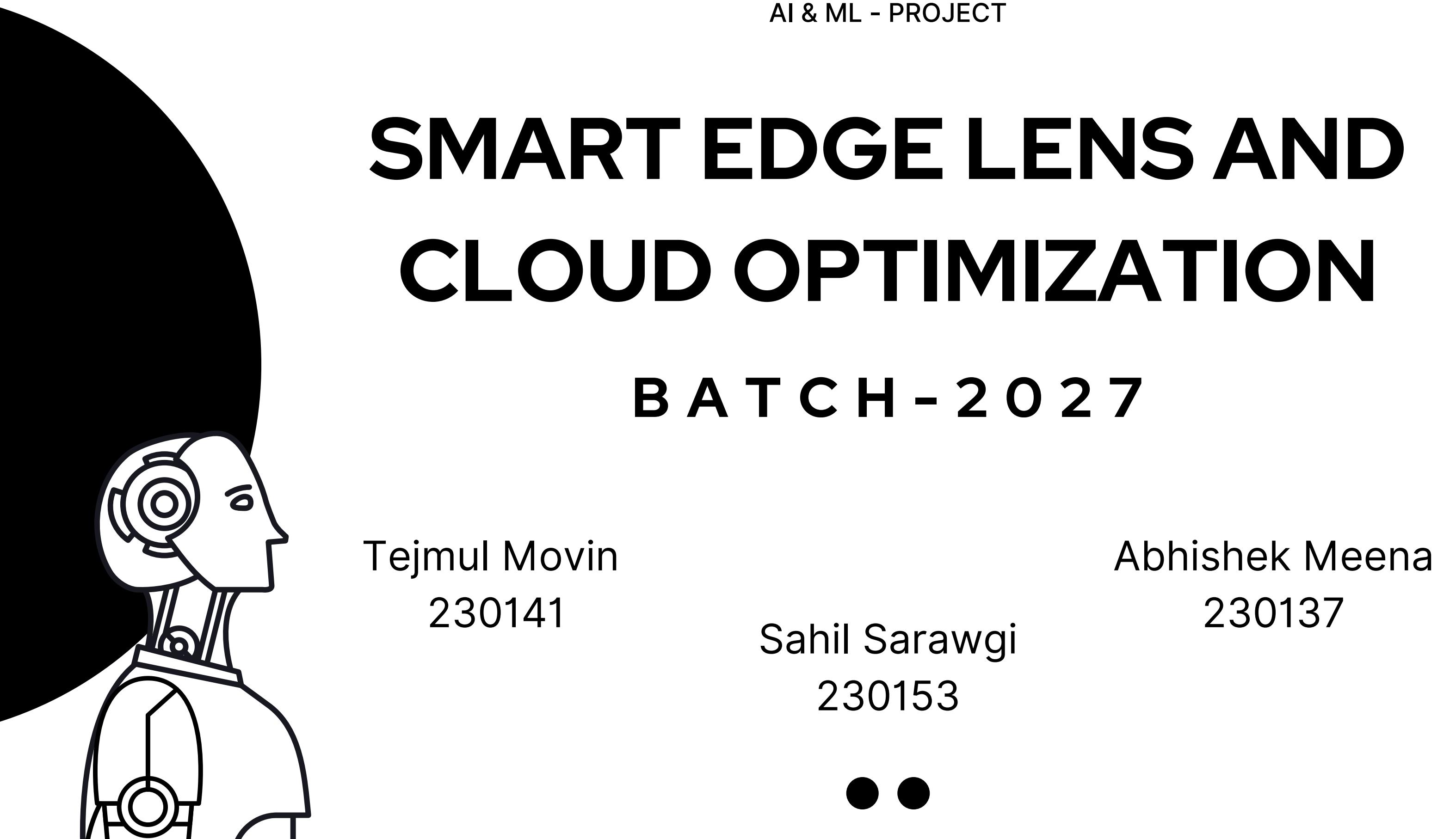
BATCH - 2027



Tejmul Movin  
230141



Sahil Sarawgi  
230153



Abhishek Meena  
230137

# PROBLEM STATEMENT & MOTIVATION

## Problem Statement:

Traditional CCTV systems send raw video to cloud → high bandwidth cost, delay, human monitoring for congestion & inefficient anomaly detection.

## Why this project?

- Reduce Human Effort for Monitoring
- Reduce latency for real-time action
- Immediate detection of anomalies and Accidents
- Raw Videos to Quantized Json Report for Cloud Optimization

## Goal:

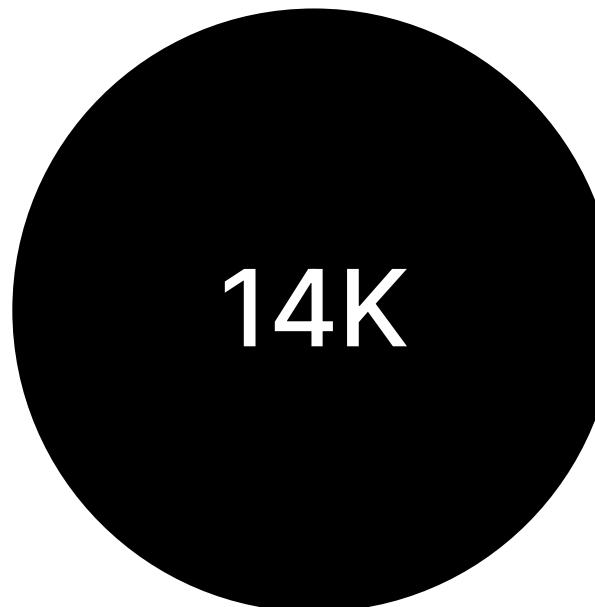
Process video frames on edge using Vision Transformers + Quantization to detect anomalies efficiently.



# QUANTIZATION

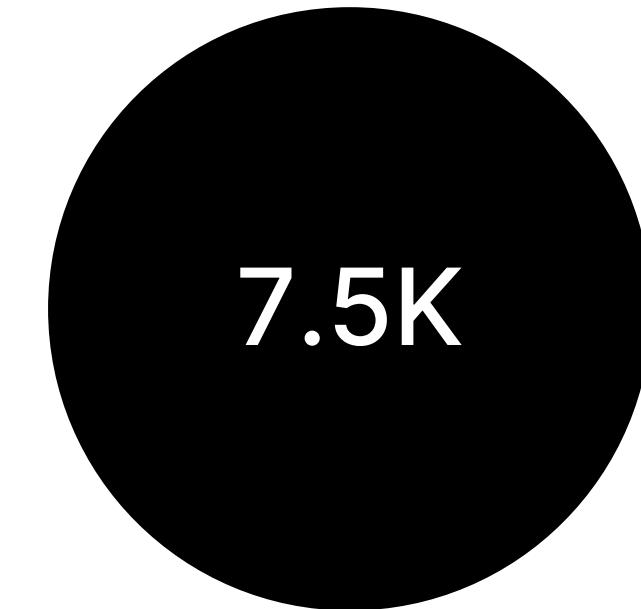
03

```
Processing video: cctv052x2004080619x00104
Processing video: cctv052x2004080620x00105
Processing video: cctv052x2004080620x00106
Processing video: cctv052x2004080620x00107
    Processed 4500 frames...
Processing video: cctv052x2004080620x00108
=====
FEATURE EXTRACTION COMPLETE
Total frames processed: 4535
Feature vector shape: (151296,)
```



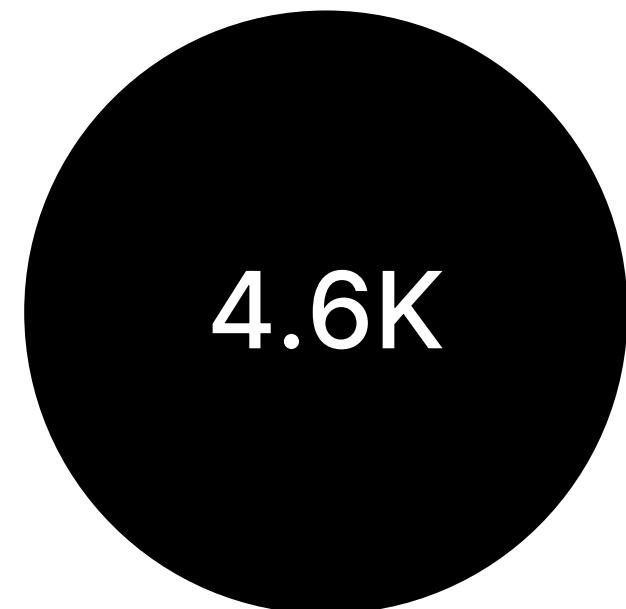
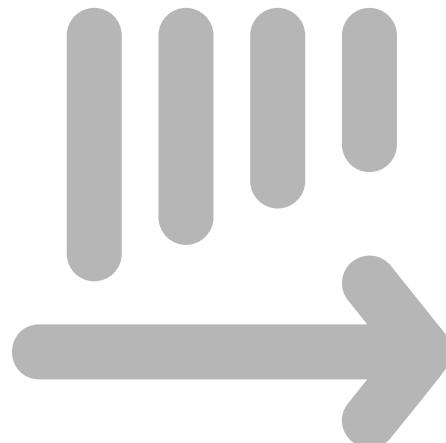
RAW FRAMES

Extracted Frames From 300  
Videos



DIVIDED QUANTIZATION

Reduced the frame rate by Half



FINAL FRAMES

ODD Frames after the divided  
Quantization

This Quantization Mainly focused on Extracted Frames from the Videos and Reduce Computation  
and Optimized Frame Rate for Json

# METHODOLOGY-1

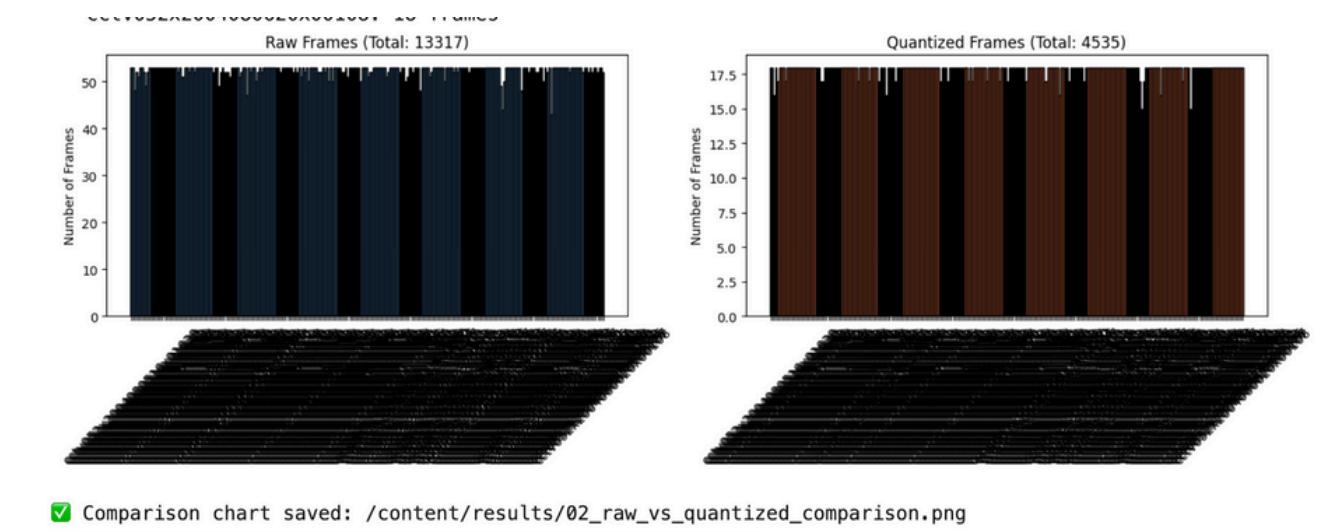
## K-Means + CNN-VGG16 Based Traffic Classification

**Goal: Classify traffic as Low / Medium / High based on video frames**

- Extract 15,000 frames → 4,500 frames via third-stage quantization
- Pre-processing (224x224, normalization, noise reduction)
- K-Means Clustering → auto-generate congestion labels instead of Manual Labelling
- VGG16 CNN used for spatial feature extraction

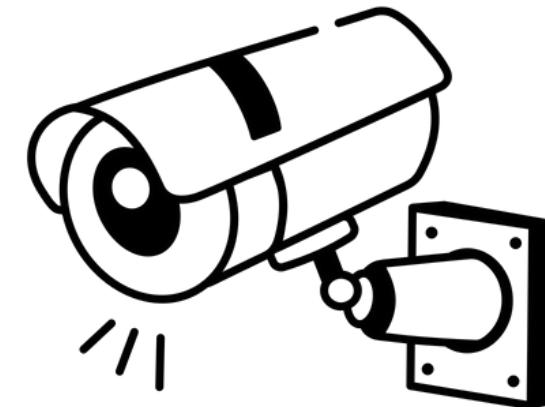
**Output:**

- 60% accuracy achieved Via Clustering and 98% via CNN
- Good for predefined congestion monitoring



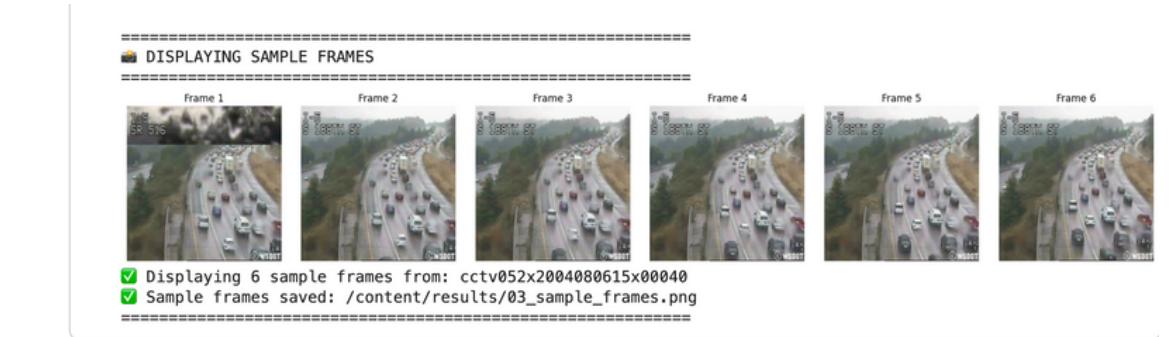
# METHODOLOGY-2

Vision Transformer + Isolation Forest



**Goal: Accident & anomaly detection without labels**

- SAME PRE-PROCESSING & QUANTIZATION (15K→4.5K FRAMES)
- VIT EXTRACTS HIGH-DIMENSION GLOBAL ATTENTION FEATURES
- ISOLATION FOREST DETECTS ANOMALIES UNSUPERVISED



**Output:**

- Anomaly score
- Severity level
- Frame ID & timestamp

**ADVANTAGES:**

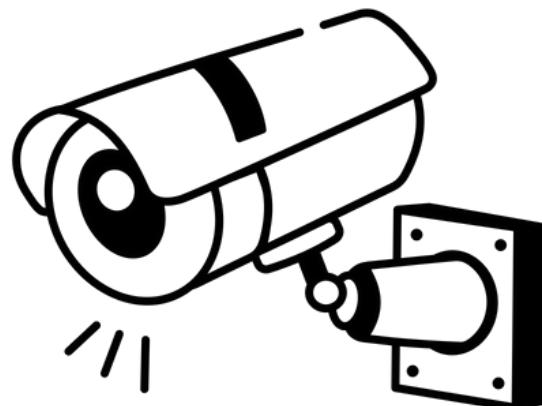
- NO MANUAL LABELING REQUIRED
- GLOBAL UNDERSTANDING > CNN
- DETECTS UNSEEN ANOMALIES

# METHODOLOGY-3

## Vision Transformer + Isolation Forest

### Goal: Fine Tuning the Vision Transformer

- VIDEO - TEXT IS A COMPLEX PROBLEM
- GPU - RAM - 15GB EXCEEDED AND SYSTEM CRASHED
- BECAUSE LACK OF RESOURCES IT



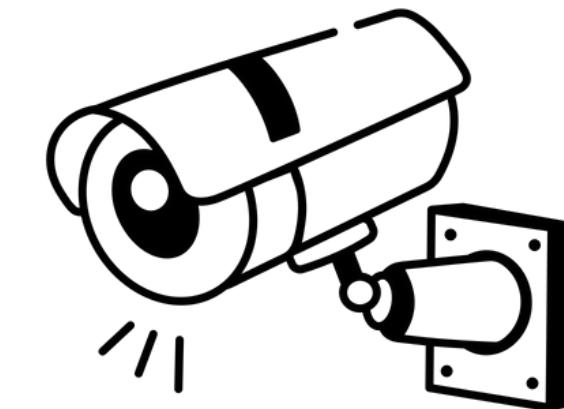
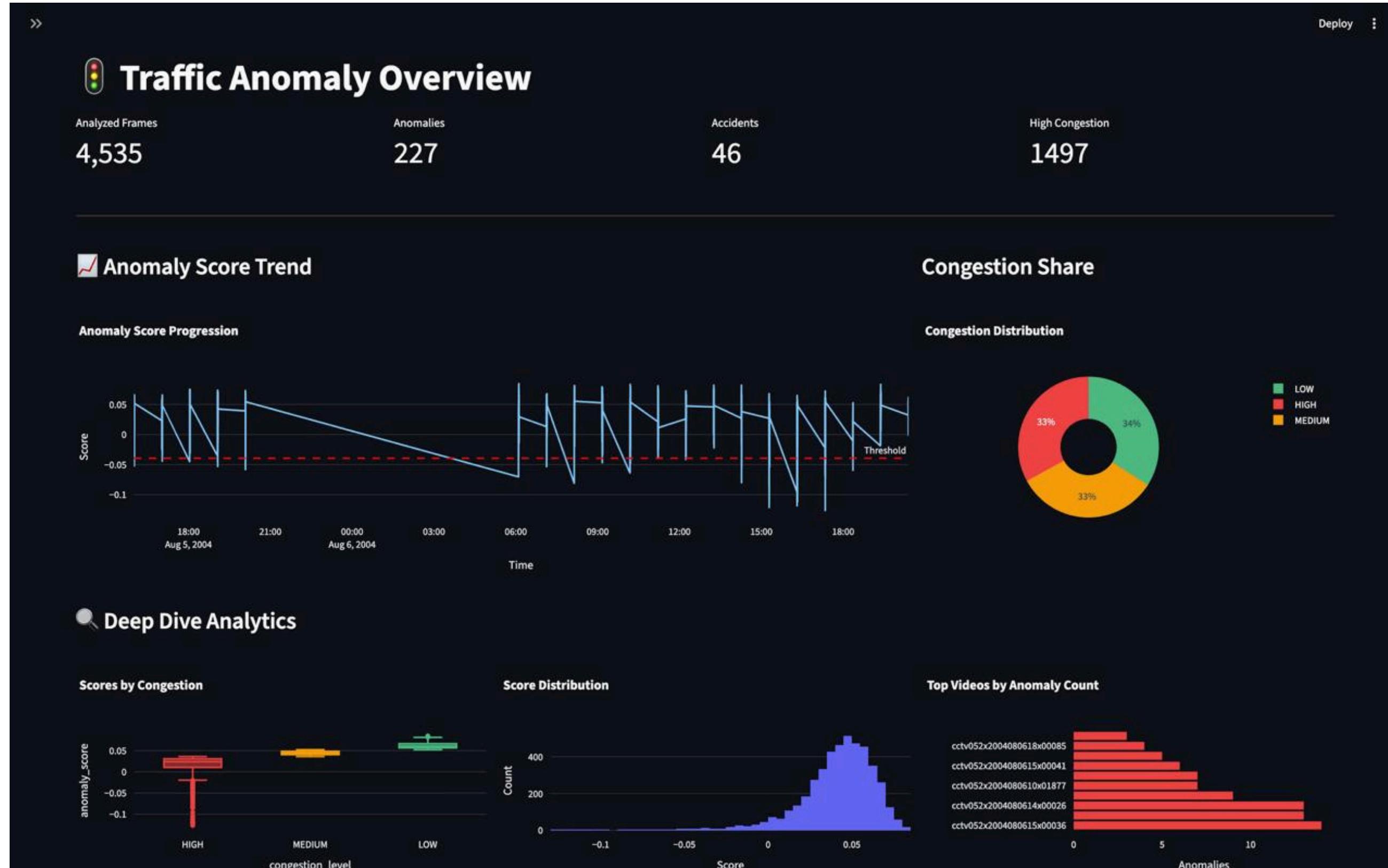
```
Model moved to device: cpu
=====
STARTING TRAINING
=====

Training Configuration:
Learning rate: 0.0001
Epochs: 20
Optimizer: AdamW
Criterion: CrossEntropyLoss
Scheduler: ReduceLROnPlateau
=====

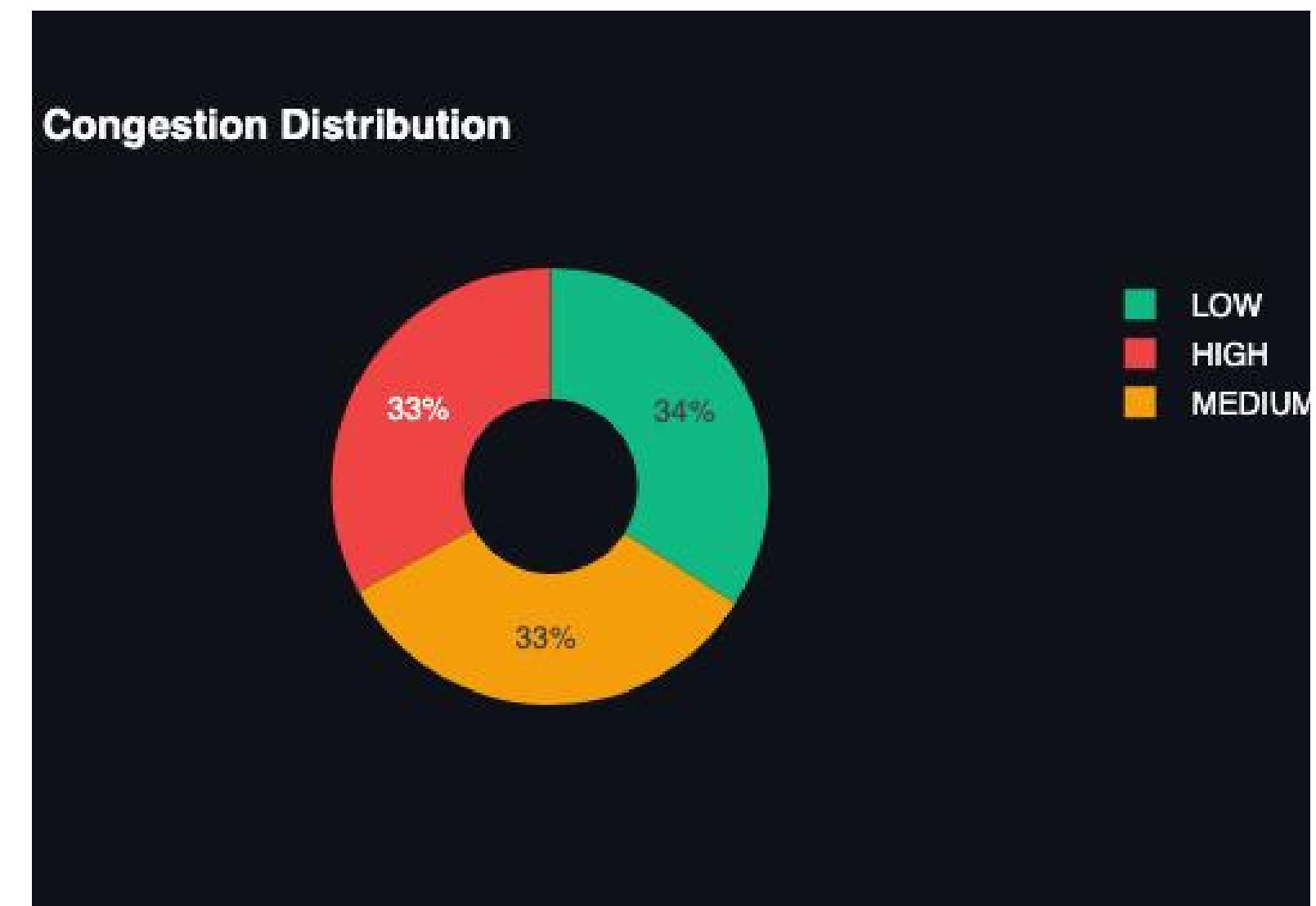
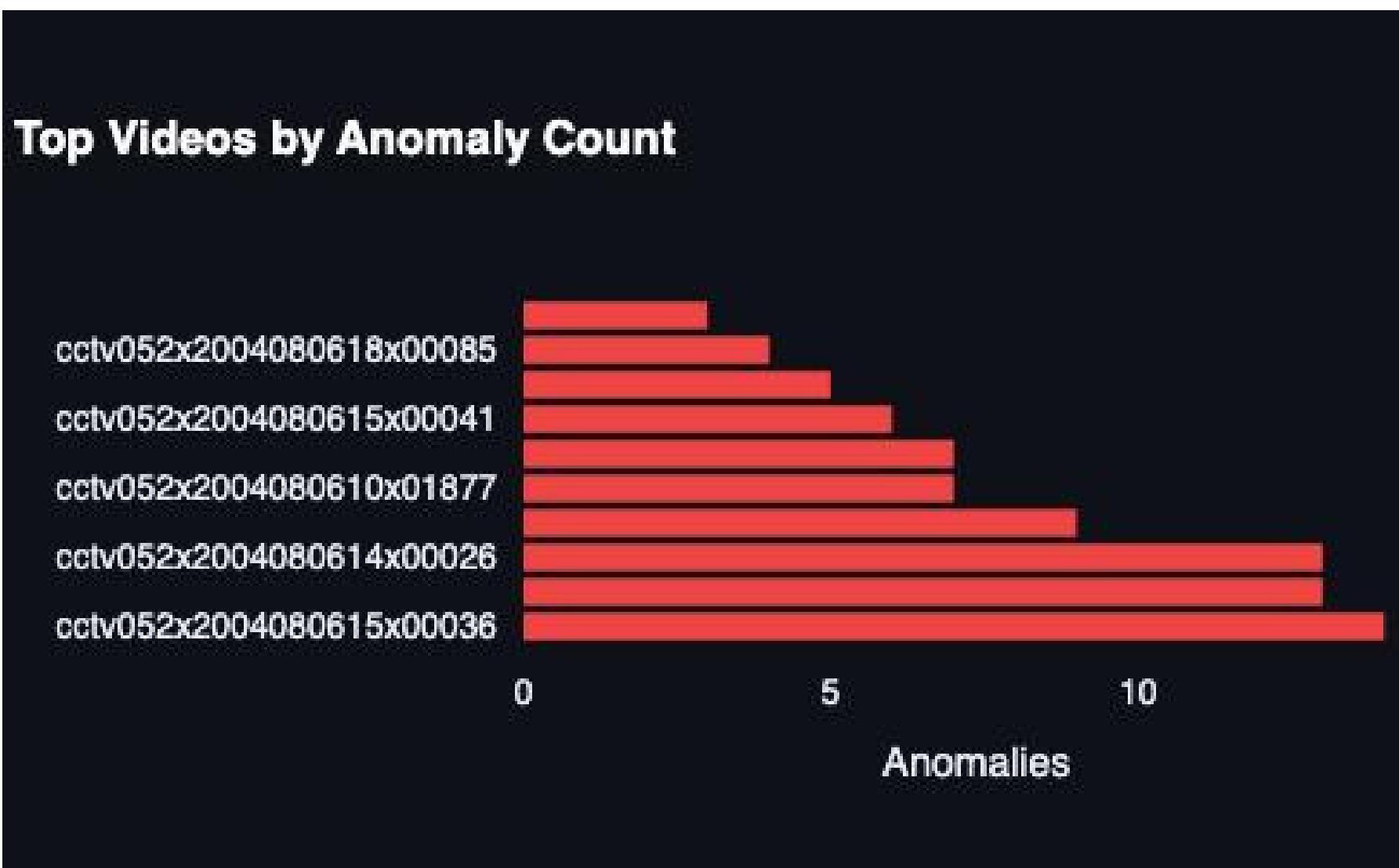
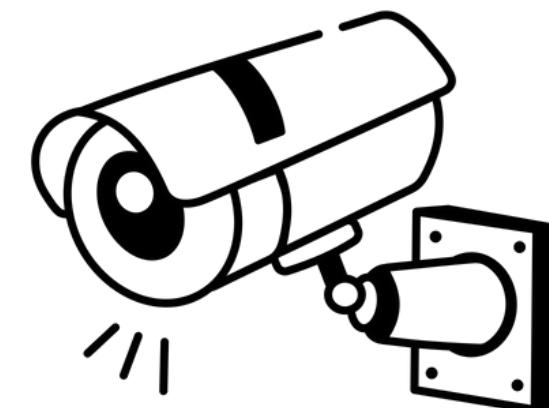
Epoch 1/20
```

 Executing (42m 49s)

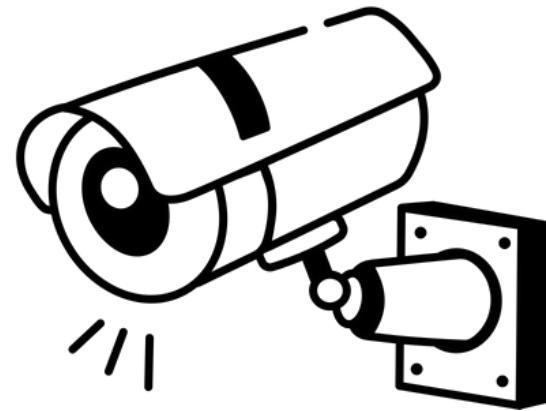
# STREAMLIT-ANALYSIS



# CONGESTION - ANAMOLY



# GITHUB - STRUCTURE



```
Docs
Final Report
  AI_ML_Final_Report.pdf
Literature Review
  Literature Review.pdf
Methodology
  AI_ML_methodology_1.pdf
  AI_ML_methodology_2.pdf
Streamlit-app
  streamlit_app.py
notebook
  aiml_project.ipynb
  README.md
  requirements.txt
  traffic_anomaly_detection_report....
```

Addded Everything from Literature Review to Methodologies and even StreamLit by maintaining proper Folder structure and Code

# CONCLUSION

- Our project focuses on using Vision Transformer and Isolation Forest for traffic monitoring.
- We process video frames and detect anomalies without sending full data to cloud.
- Preprocessing, frame extraction and model pipeline have been successfully implemented in Google Colab.
- ViT gives a better global understanding of traffic scenes compared to CNN.
- Quantization will help in faster and lightweight edge deployment.

## Future Work

- Fine-tune model for higher accurate anomalies
- Better vision transformer optimization
- Edge deployment (Jetson)
- Production-ready pipeline

Thank You --