Crime (Fight & Object) Detection System

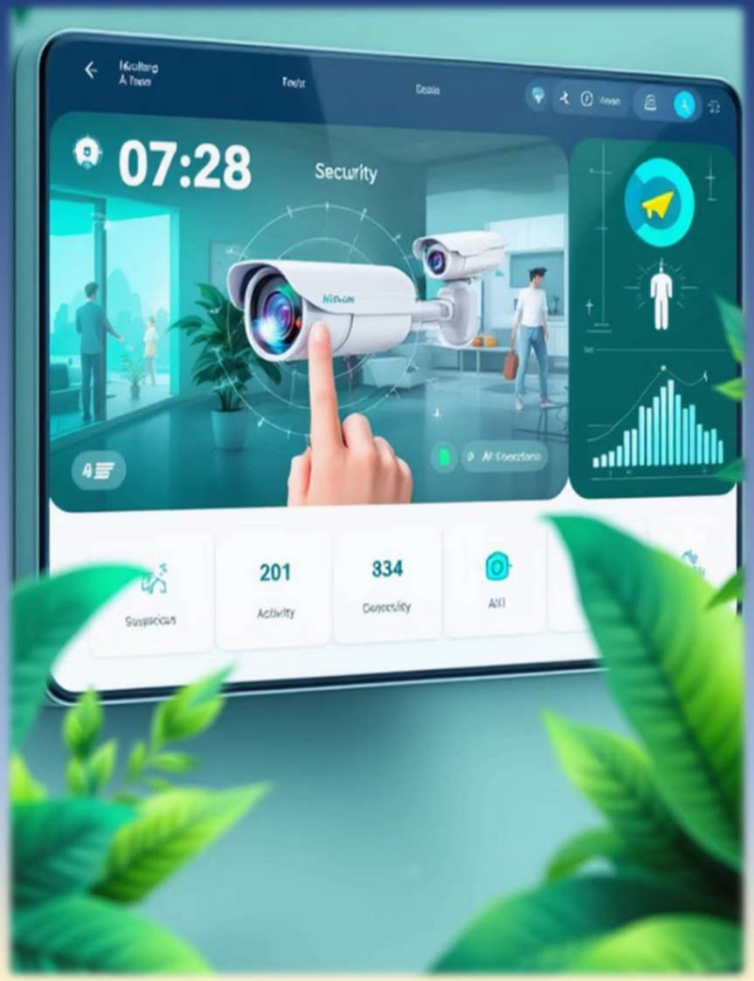
AI & Data Science Graduation Project

Our Team:

- Eng. Mohamed Osama Faid
- Eng. Mohamed Mostafa El-Sayed
- Eng. Ahmed Ezzat Moustafa
- Eng. Karim Mohamed Hafez
- Eng. Mahmoud Yossry Elmetwallli
- Eng. Rana Hassan Badrawi

Supervisors:

Eng. Mariam Elsaee & Dr. Nesma Ibrahim



Project introduction

Security Enhancement

An intelligent video analysis system that detects and recognizes criminal activities in surveillance footage using AI-powered technologies.

Dual-Stream System

Custom 3D CNN for action recognition and YOLOv8 for weapon detection & fire hazard.

Also a User-Friendly Interface

Gradio-based web application hosted on HuggingFace Spaces for individuals and developing purposes.

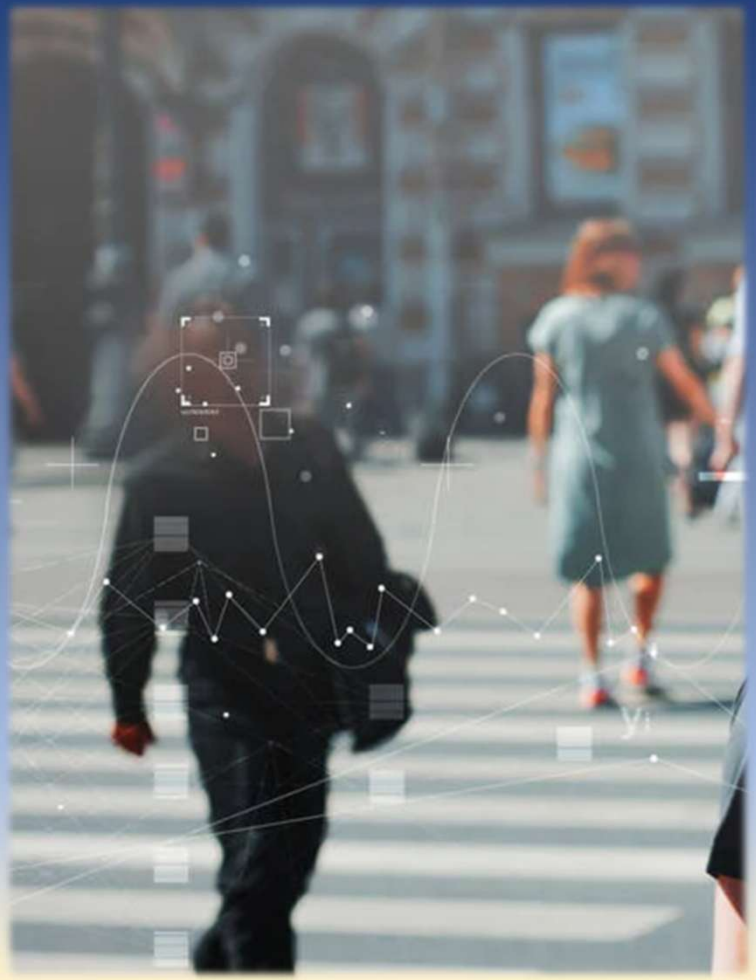
Problem Statement – What's Wrong?

- **Manual and reactive methods**: Law enforcement relied on routine patrols, manual crime analysis, and post-crime investigations, which delayed responses and allowed crime rates to remain high.
- **Limited predictive capabilities**: There was no reliable system to predict where or when crimes might occur.
- **Slow response times**: Emergency services relied on limited caller information, leading to delays.

In Los Angeles, the "Operation LASER" a similar system AI-based helped reduce violent crimes by **25%** in targeted zones through crime pattern analysis, AI tools analyze **hours of surveillance footage in minutes**, improving evidence collection.
Also real-time emergency response were 40% faster.
though, integrating a similar smart system well trained & suitable for our society is urgent need.

رواد مصر الرقمية

وزارة الاتصالات وتكنولوجيا المعلومات



🔍 Why our Project Matters?

📈 **Adapting to Growing Populations**
As population density increases, traditional security methods alone are no longer enough.

🤖 **Empowering the AI Era**
Integrating AI with surveillance systems boosts the efficiency of security operations.

🚨 **Faster Emergency Response**
Smart detection enables real-time alerts and quicker reactions to threats.

👁️ **Reduced Human Monitoring Load**
Automates threat detection, allowing security personnel to focus on critical tasks.

🎥 **Better Evidence Collection**
Advanced video analysis ensures accurate documentation of incidents.

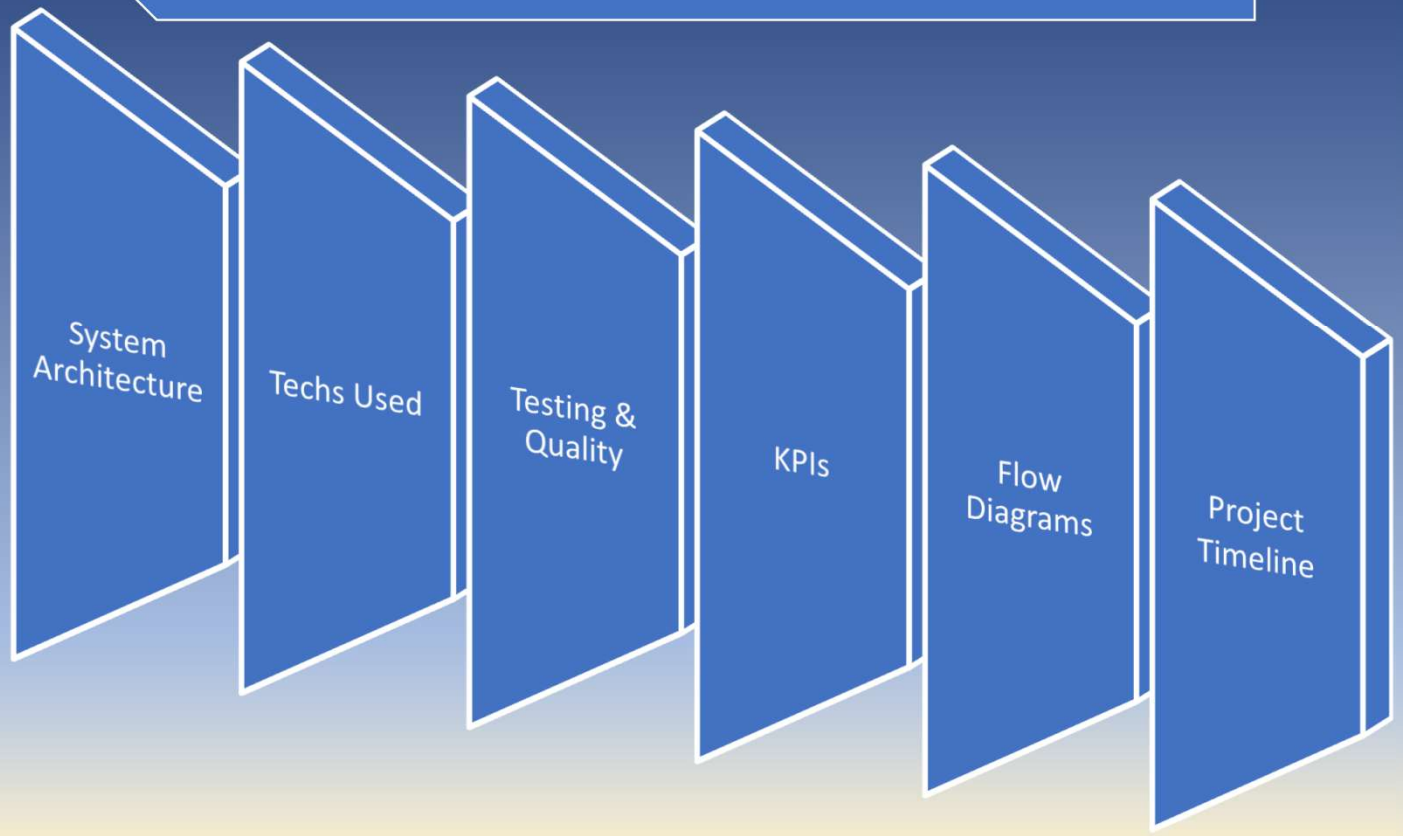
Literature Review / Related Work

In our project we reviewed some related works, after giving thanks to them we were able to focus some points, adding good progress of ComV at this field:

- **Dual Detection System**: Combines fight detection (3D CNN) and weapon detection (YOLOv8) in one pipeline.
- **Modular & Scalable**: Clean architecture with plans for real-time CCTV integration coming in Phase 2.
- **Web Interface**: Django-based upload system for user-friendly access.
- **Custom Frame Processing**: Tailored OpenCV function handles diverse video types efficiently.
- **Supports Multi-User Roles**: Designed for both general users and security staff.

Project Name	Year	Pros	Cons
Detection (Vision-based Fight Akti et al.)	2020	• LSTM + attention. • 2D CNNs. • New dataset introduced.	• No weapon detection.
JOSENet (Nardelli et al.)	2024	• RGB + Optical Flow. • Low memory cost.	• Precomputed optical flow needed. • Not real-time.
VIVIT (Singh et al.)	2022	• State-of-the-art accuracy. • Captures long-range features.	• Slow processing. • Requires large datasets.
Real-Time Weapon Detection (YOLOv5) (Senjab)	2020	• Simple PyQt5 interface. • Real-time inference.	• No action recognition. • Ignores behavioral context.
Weapon Detection in Videos (YOLOv5) (Sabari S.)	2021	• Frame-based violence scoring. • Visual weapon detection.	• No behavioral analysis. • Misclassification risks.
Violence Detection in Surveillance Videos (Moaaz)	2020	• High accuracy on standard datasets. • Uses temporal features.	• Unspecified architecture. • Not designed for real-time.

Technical details.



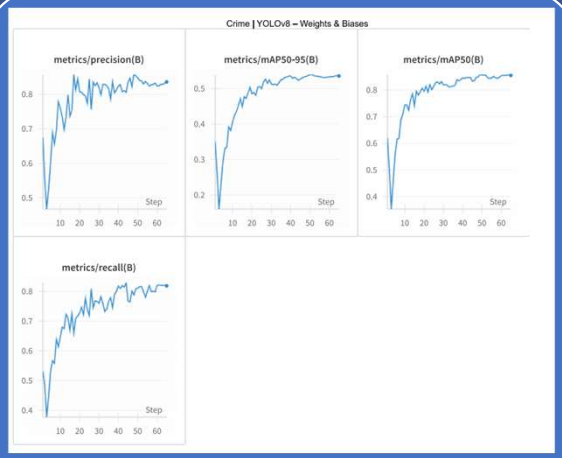
Technologies Used

- YOLOv8 - Objects Detection.
- TensorFlow-keras - 3D CNN for Action Recognition.
- Gradio - Web UI.
- OpenCV, MoviePy - Video Processing.

Robflow for collecting data & wandb for analysis & Kaggle for training.

Collection data

Roboflow for collecting yolo images data



wandb for analysis and chats

Kaggle for action videos & training

System Architecture

Input Layer

Video upload interface accepting for non-tech users & technical version for custom developing.

Preprocessing Layer

Frame slicing and resizing for model compatibility.

Detection Layers

YOLOv8 for weapons & fire, 3D CNN tensorflow-keras for violent actions.

Overlay Engine

Annotation on frames with detection results.(results only for non-tech users)

Postprocessing

Video compilation and delivery to user.

Testing & Quality Assurance

Module	Test Case	Expected Result
Video Upload	Accept .mp4, .avi, .mpeg	Success
Frame Extraction	Correct FPS extraction	Verified
Action Model	Classify 30-frame clips	≥91% accuracy
YOLO Detection	Identify Objects	≥80% precision
Integration	Combined model output	Functional

Key Performance Indicators

Fight detection Model Training Performance

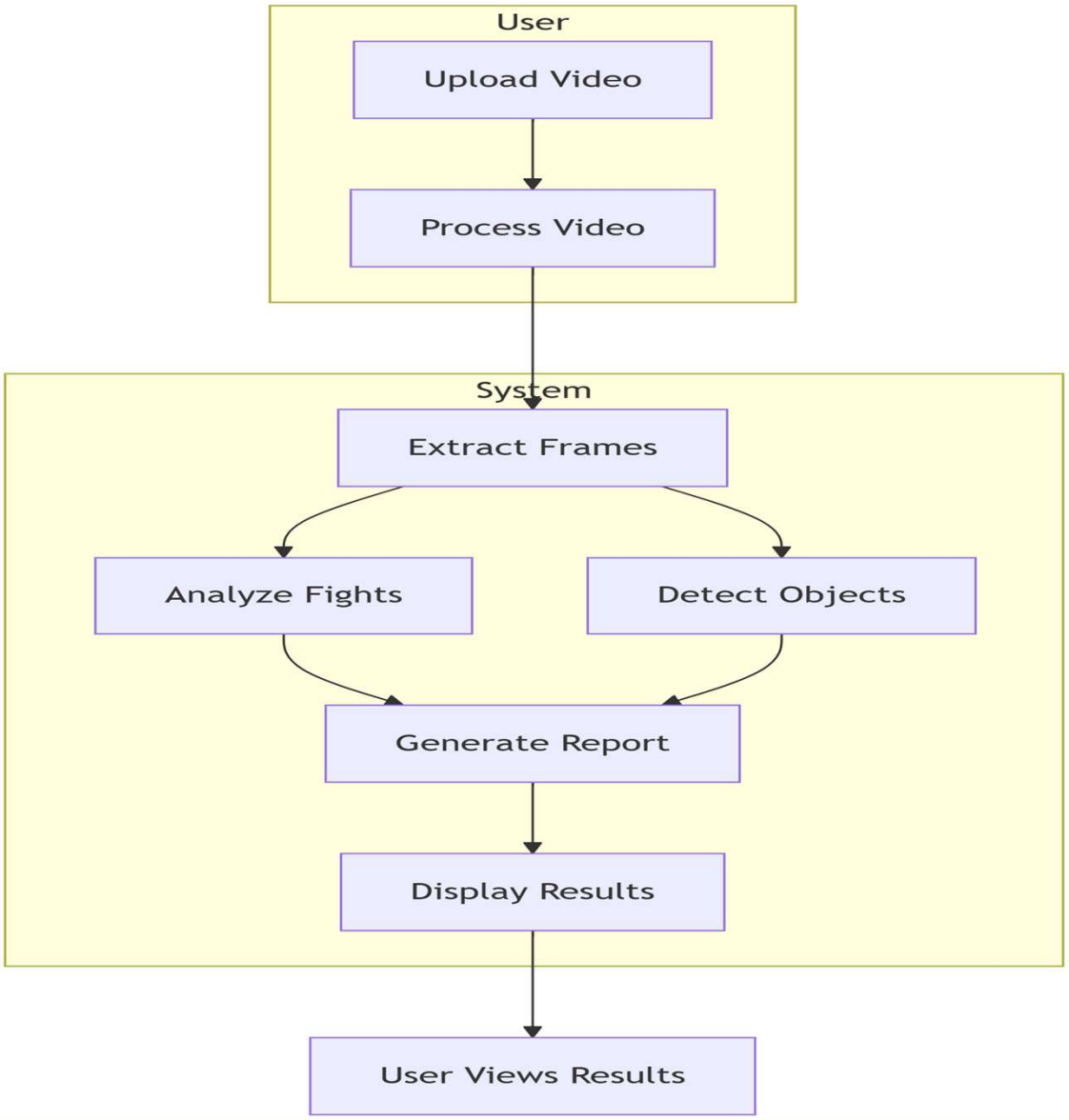
Metric	Value	Epoch
Best Training Accuracy	0.8583	7
Best Validation Accuracy	0.9167	10
Lowest Training Loss	0.3636	7
Lowest Validation Loss	0.2805	8

Key Performance Indicators

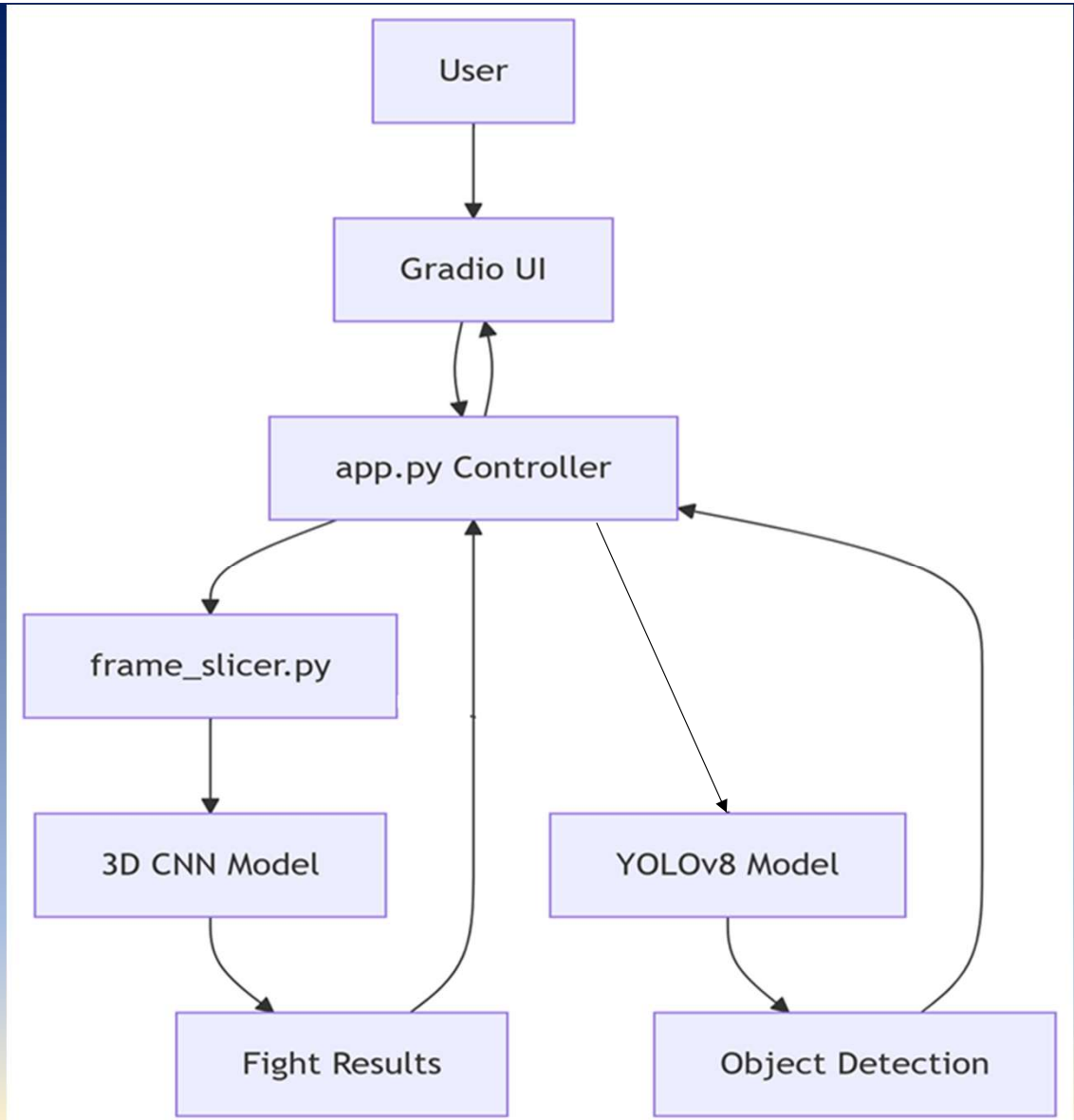
🔍 YOLOv8 Detection Performance Metrics

Metric	Value	Interpretation (What it means)
mAP@0.5	0.80	Excellent object detection at IoU ≥ 0.5. Model locates objects accurately even with slight shape/size variations.
mAP@0.5:0.95	0.50	Moderate performance. Indicates room for improvement in detecting small or complex objects.
Precision	0.80	High precision. 80% of detected objects are true positives, with few false alarms.
Recall	0.70	Medium recall. Model captures 70% of actual objects, missing 30% (false negatives).

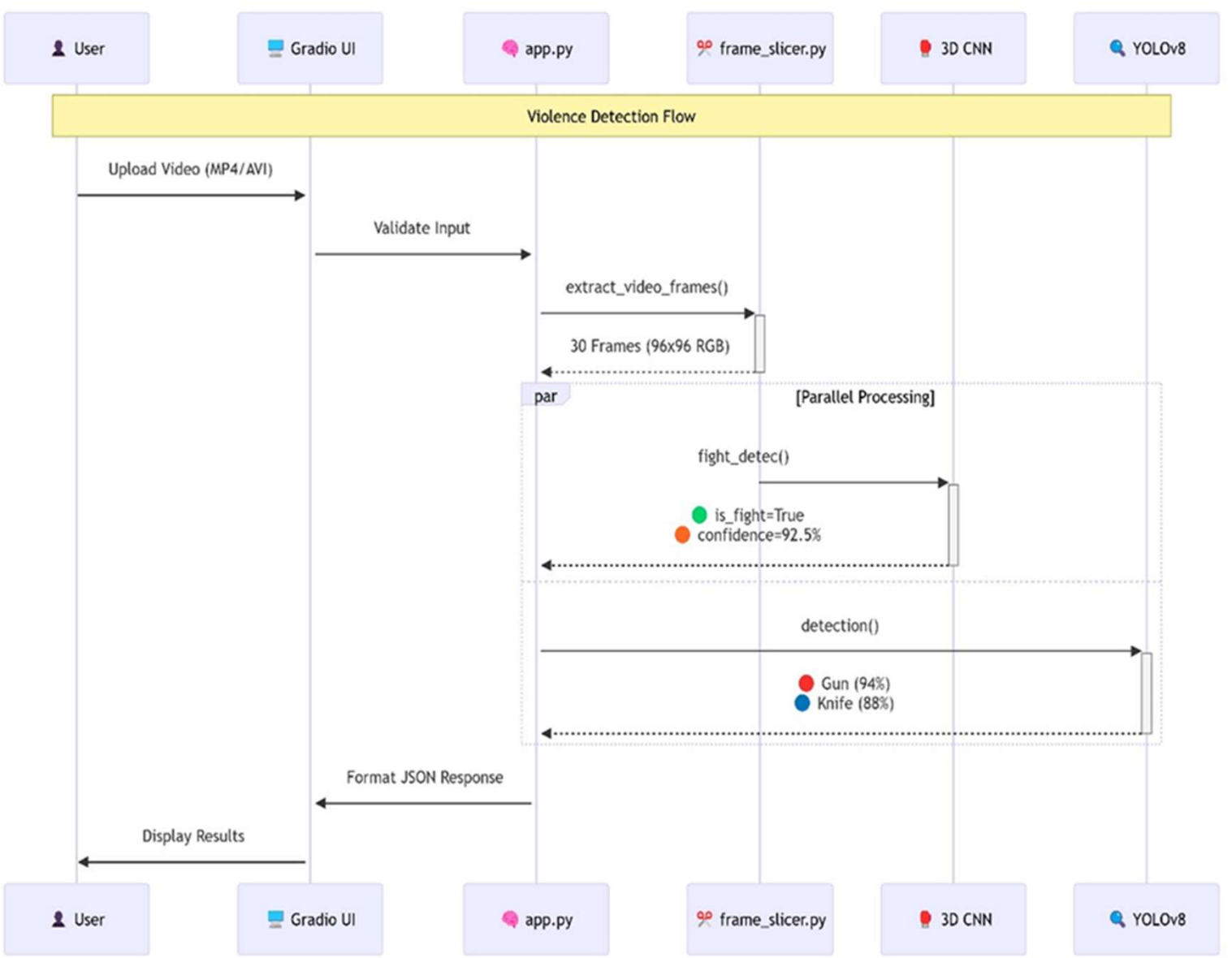
Data Flow Diagram



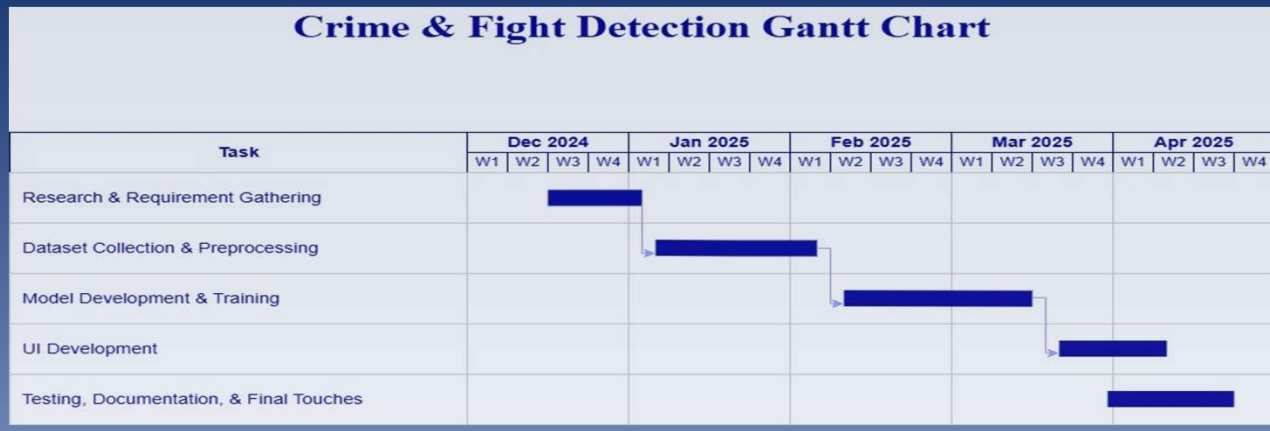
Software Architecture Diagram



Use Case


User
Gradio UI
app.py
frame_slicer.py
3D CNN
YOLOv8
Violence Detection Flow
Upload Video (MP4/AVI)
Validate Input
extract_video_frames()
30 Frames (96x96 RGB)
par
[Parallel Processing]
fight_detec()
is_fight=True
confidence=92.5%
detection()
Gun (94%)
Knife (88%)
Format JSON Response
Display Results
User
Gradio UI
app.py
frame_slicer.py
3D CNN
YOLOv8

Project Timeline



Planning Phase

Project proposal, requirements gathering, gathering, and Data Preprocessing.

Development

Model training, integration, and system architecture implementation.

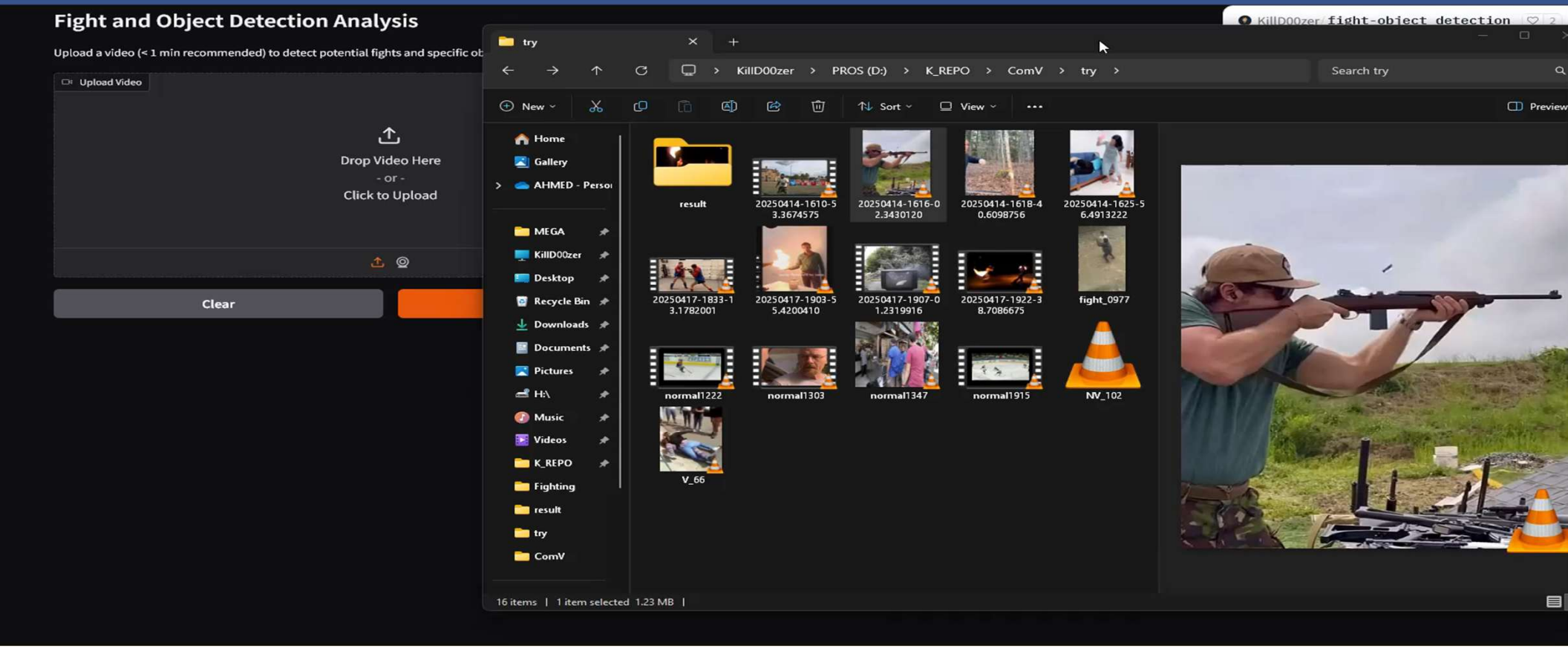
Testing

Quality assurance, bug fixes, and performance optimization.

Deployment

Hugging Face Spaces deployment with documentation.

Running Module



Running Module With Video Annotation

full_project.py

ComV > AI_made > full_project.py > ...

Run Python

```python
# --- Import Necessary Libraries ---
import cv2  # OpenCV for potential video processing (though not directly used here)
import numpy as np # NumPy for numerical operations (though not directly used here)
import os # Operating system library (though not directly used here)
from ultralytics import YOLO # YOLO object detection model (imported but seems unused directly, likely used within 'detection')
import time # Time library (though not directly used here)
import tensorflow as tf # TensorFlow for deep learning (imported but seems unused directly, likely used within 'fight_detec')
from frame_slicer import extract_video_frames # Frame extraction utility (imported but seems unused directly, likely used within 'fight_detec')
import matplotlib.pyplot as plt # Matplotlib for plotting (imported but seems unused directly, likely used within 'fight_detec')

# --- Import Custom Modules ---
# Import the fight detection function from the 'Fight_detec_func.py' file
from Fight_detec_func import fight_detec
# Import the object detection function from the 'objec_detect_yolo.py' file
from objec_detect_yolo import detection


# --- Main Execution Block ---
# This code runs when the script is executed directly.

# Prompt the user to enter the path to the video file
path0 = input("Enter the local path : ")
# Remove any surrounding quotation marks from the input path (useful for paths copied from Windows Explorer)
path = path0.strip('"')
# Print an informational message indicating which video is being processed
print(f"[INFO] Loading video: {path}")

# Call the fight detection function with the provided video path
# This will analyze the video for fight activity and print/save results (depending on the function's implementation)
fight_detec(path)

# Call the object detection function with the provided video path
# This will analyze the video to detect objects (e.g., persons, weapons) and print/save results
detection(path)

```

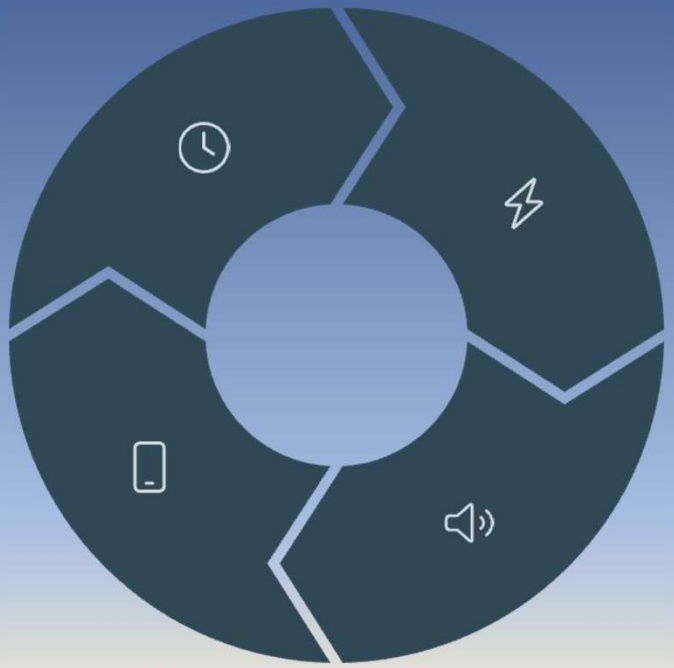
Limitations & Future Work

Real-time Support

Add webcam integration for live live video processing.

Performance Optimization Optimization

Convert models to TensorRT for faster inference.



Mobile Support

Develop TensorFlow Lite version for version for edge devices.

Audio Detection

Incorporate sound analysis for gunshots and screams.

Access & Resources

Online Demo

Available on Hugging Face Spaces

Source Code

Complete codebase with documentation

Technical Documentation

Detailed guides and API references

Visit: https://huggingface.co/spaces/KillD00zer/fight-object_detection

Visit: KillD00zer/fight-object_detection: fight & objects detector from videos

Thank You

Questions & Discussion