Department of Electrical and Computer Engineering
North South University (NSU)

CSE 440: Artificial Intelligence
Section 05

Project

Total Marks: 100
Proposal (20) + Execution (60) + Presentation (20)

Instructor: Dr. Mohammad Mahmudul Alam Semester: Spring 2025

Title:	Real-Time Object Detection Using YOLOv8
Name:	Hasnat Karibul Islam (Alindo)
NSU ID:	2211275042
Name:	Md. Minhajul Islam
NSU ID:	2211022042

Final Project Report

Title: Real Time Object Detection using YOLOv8

Abstract:
For this project, we have deployed a real-time object detection using the **YOLOv8 model** on **Google Colab**. Our main goal was to detect and classify objects frame by frame within an uploaded video and then create a new output video with annotations like bounding boxes and class labels. We used the **ultralytics** implementation of **YOLOv8** and **OpenCV** for processing the video. This project showed that it is extremely possible to have rapid and precise object detection in a cloud environment, hence making it accessible to people without strong local hardware. We could monitor frame-by-frame detection output, process speed analysis, and calculate the average confidence score, giving us an equal perspective on how the model behaved with real time input video.

Introduction:
We have always been interested in real-time object detection, especially in use cases like security, traffic management, and smart automation. This interest leads us to experiment with YOLOv8 one the latest and most efficient object detection models. We were eager to experiment with a real-life example using a video file and how well and efficiently the model would detect various objects.

Problem Statement:
Training and running an object detection model generally requires a great amount of processing power and high-end setups and most of the students doesn't have this kind of setup. We really wanted to break this barrier by using a lightweight and easily accessible platforms like Google Colab and Kaggle, to show that it is still possible to train or run models like YOLOv8 without owning a high-end setup.

Another challenge to take a video from the user, detect a number of objects within it, annotate them in reakl0-time, and output a with the annotation while also monitoring the performance matrices like FPS and confidence levels.

Methodology:
For this project first we installed necessary Python packages in Google Colab, we then uploaded a sample video file using Python's files.upload() method, from there we used the OpenCV to read the video frame by frame.

For each frame:

- We passed it through the **YOLOv8** small model (**yolov8s.pt**) to detect the objects.
- We used YOLO's built-in **plot()** function to draw bounding boxes and labels.
- We calculated the average confidence levels and estimated FPS for each frame.
- We printed detection summaries in the console.
- For the first few frames , we also displayed the annotated frames in the notebook.

Finally, we wrote the annotated frames into an output video file called **output, mp4**.

Results:

We are totally satisfied with the results of this model. The model detected objects like humans, cars, and other objects with an extremely good accuracy and confidence score.
The confidence score in the majority of the frames ranged frames ranged from 0.70 to 0.95, which is kind of impressive for a pre-trained model.

We also monitored the estimated FPS and noticed that it was quite consistent. One problem that we faced was working with the high-resolution video, which sometimes paused in Colab, utilizing the light model variant (YOLOv8s) made sure that performance was still smooth.

Conclusion:

Development of this gave us a deeper understanding of how real-time object detection works and how to deploy them with cutting-edge model's like YOLOv8. I was able to create an end-to-end video analysis pipeline that detects and labels objects with great accuracy,
One of the key takeaways from this project for us was realizing that advanced computer vision tasks don't always require expensive hardware but clever use of available tools and optimized models can go a long way. This project also strengthened our skills in Python, OpenCV, and working with deep learning models in practice. And in the future, we would like to enhance this project by integrating live video streams, using custom-trained YOLO models for specialized tasks, and potentially deploying the solution as part of a larger real-time monitoring system.
In the end, we would like to express our sincere gratitude to our honorable faculty Dr. Mohammad Mahmudul Alam sir, whose encouragement feedbacks explanation helped us a lot to stay on track to complete this project. We also appreciate the resources and learning environment provided by him, which made it possible to carry out this project successfully.