OBJECT DETECTION FOR HOME SECURITY

CSE573- Computer Vision and Image Processing

Name: Venkata Mani Sai Praneeth Chidella UBIT:50560459 UBName: vchidell

Name: Reddy Kamal Teja Thota UBIT:50560696 UBName: reddykam

Application:

The goal of this is to create a Home Security System that work in real-time, identifying and categorizing objects like people, pets, packages, or any unusual items through security cameras. The system will continuously monitor the surroundings and send alerts in case of suspicious activity. This approach will be beneficial for homeowners, apartment residents, and businesses looking for an automated way to enhance security.

State of Art:

Existing security systems using motion or simple object detection often trigger false alarms from shadows, leaves, or pets. We enhance the accuracy with YOLO for real-time object detection, distinguishing between critical and non-critical such as difference between person and a small animal object.

Similar projects: (1) "YOLO-based Real-Time Object Detection for Surveillance" (IEEE) – Uses YOLO for security monitoring. (2) GitHub:ultralytics/yolov8 – Offers pre-trained YOLOv8 models for object detection.

Inputs and Outputs:

- Inputs: Live video feed from a laptop webcam or security cameras or any recorded footage.
- Intermediate Outputs: Frames with bounding boxes, detected object labels.
- Final Outputs: Labeled objects in video feed, logs of detected objects, and optional alert notifications.

Required Data:

- Pre-trained Model Data: This project will use the COCO dataset, which contains labeled images of common objects like people, animals, and vehicles for training and testing. we will be using an existing data set (coco dataset)
- data acquisition: Using pre-trained YOLO models on the COCO dataset. if needed we will collect real-world webcam footage for model fine-tuning for better results.
- Scale of Data: COCO dataset contains over 118,000 training images across 80 object categories.

Coding Resource Requirements:

We are going to usePython as the primary programming language, utilizing libraries such as OpenCV, PyTorch, YOLOv8, and NumPy and we are going to use TensorFlow/PyTorch framework. I am

leveraging pre-trained YOLO models from the Ultralytics GitHub repository while customizing detection thresholds, bounding box filtering, and alert mechanisms to enhance performance. Additionally, I am independently contributing to the implementation of real-time video feed processing using OpenCV, developing an alert mechanism for notifications via email or SMS, and optimizing inference speed to ensure efficient real-time execution.

Computational Resource and Effort Requirements:

The project will require a capable development machine with a GPU for efficient training and real-time inference. Edge computing solutions like Raspberry Pi may be used for deployment, potentially enhanced with a TPU for improved performance. Training on large datasets like COCO can take a considerable amount of time, while real-time object detection will be much faster on a GPU compared to a CPU.

The project is expected to require approximately **2-3 days of GPU time** for model training and fine-tuning. In terms of effort, the estimated workload per person is around **90-120 hours**, covering tasks such as data preprocessing, model customization, testing, and performance optimization to ensure real-time detection capabilities.

Evaluation:

The success of the project will be defined by achieving accurate real-time detection of humans, animals, and packages with minimal false positives and false negatives. The system should maintain an FPS of 30+ for real-time performance. Evaluation will be based on metrics like mAP (Mean Average Precision) for overall accuracy, precision & recall assessing detection correctness, and FPS to ensure real-time capability. Testing will cover both simple scenarios with clear visibility and challenging cases such as low-light conditions, occluded objects, and small objects in the background.

Project Expectations:

we are excited in using YOLO models for accurate and fast object detection in security systems. This will help improve the effectiveness of real-world security applications by making them more reliable and responsive. This project has the potential to enhance various security applications, making them smarter and more capable of handling complex scenarios.

we hope to gain a deep understanding of real-time object detection, explore optimization techniques for low-latency processing, and learn how to develop effective real-world security solutions. Additionally, we aim to improve our skills in integrating machine learning models into practical applications. This project will provide valuable insights into building efficient, scalable systems that can handle complex tasks in real-time environments.

Additional Guidelines:

In this project, we use YOLOv8 for real-time object detection, which offers fast inference but has limitations depending on image complexity. In low-light conditions, YOLOv8 may struggle with accuracy, such as missing detections at night. Occlusions where objects are partially hidden can also cause inaccuracies, like failing to detect a person behind a car. Additionally, small object detection becomes challenging, especially for distant objects, like a small package on the ground. To address these issues, we can use infrared cameras for low-light situations, fine-tune models with challenging images, and employ higher-resolution cameras to improve detection accuracy and performance across different conditions.