Home Assignment - Black Rover

By Shunit Truzman

A. Person Identity Catalogue (cross-clip)

The Solution Approach:

1. Detect & track people

The first step was to detect and track each person in a clip, assigning them a local ID. For detection, I used ***"yolo11m.pt",*** and for tracking, given limited GPU resources, I used the built-in different **YOLO trackers** options: "***ByteTrack"*** and **"*BoT-SORT"***. Among them, *BoT-SORT* achieved the most robust and accurate results in assigning consistent IDs, especially in scenarios where people appear close to each other. After selecting the tracker, I tuned the parameters (confidence and IoU thresholds) manually through trial and error on the provided dataset to achieve optimal performance.

Notes, Limitations & Possible Improvements:

- **"*DeepSORT*"** is another possible tracker, but it is computationally heavier since it requires running two models instead of one. For computational efficiency, I preferred to use the built-in **YOLO trackers**, as I worked on *Google Colab* with limited GPU resources. Since I don't have access to a local GPU, working with *Google Colab* was the only available option for accessing one.

2. Assign Global IDs

The second step was to map the local IDs (LIDs) obtained from the tracker to global IDs (GIDs) across all clips. For this task, I used the "***Torchreid"*** library - a popular framework for **Person Re-Identification (ReID)**, and I tested several pretrained models designed to extract embeddings of a person's appearance. Ultimately, I selected the pretrained model ***"osnet_ibn_x1_0",*** which provided the best trade-off between accuracy and complexity compared to other models I checked (such as "*osnet_x1_0*" and "*osnet_ain_x1_0*").

The assignment algorithm from LIDs to GIDs works as follows:

1. If an LID already appeared in the same video, it keeps its previous GID and updates its embedding.
2. Otherwise, its feature vector is compared against all existing global embeddings to build a similarity matrix.
3. The Hungarian algorithm finds the optimal one-to-one assignment between detections and existing GIDs based on cosine similarity.
4. Only pairs above the tuned similarity threshold are matched and updated; any unmatched detection is assigned with a new GID.

Notes, Limitations & Possible Improvements:

- The models could be further optimized to achieve better results; however, due to the limited GPU resources available in *Google Colab* and the long runtime required for each configuration, it was challenging to perform proper optimizations.

- With additional time and GPU power, **transformer-based embedding models** (e.g., ***Swin Transformer, Vision Transformer***) could be explored, as they typically yield strong person representations over time.

- ❖ Person re-identification accuracy could be further enhanced with **multi-modal fusion**: combining face recognition (within and across clips), clothing detection (within a single clip), and full-body embeddings. Unfortunately, I didn't have time to explore that.
- ❖ The order in which clips were processed was important - sorting them by name ensured consistent embedding assignment. Without a fixed order, results could vary depending on which clip was processed first.

B. Scene Labelling (per-clip)

The Solution Approach:

I began by analyzing the dataset to understand the actions and visual characteristics of each scene to decide how to treat and classify them.

My initial idea was to use a pretrained model from "***Hugging Face***" - VideoMAE large – finetuned on UCF-Crime. This model is specifically adapted for crime vs. normal video classification (e.g., robbery, shoplifting, stealing, etc.). However, it misclassified both the "normal" clips and the "shoplifting" clip, leading me to explore alternative methods.

Next, I turned to **optical flow–based methods,** aiming to detect abrupt movements (e.g., pulling out a weapon, shoplifting, etc.). I implemented the classical "***Farneback optical flow algorithm***" (OpenCV) and extracted statistical features such as the max, mean, and median of motion magnitudes. The classical approach was chosen due to its speed, ease of implementation, and lack of GPU requirements. The goal was to test whether noticeable distinctions in motion magnitude could be detected, and eventually, this method proved effective and was chosen as the main algorithm.

Initially, I applied the optical flow algorithm to the entire frame, but the results were too noisy. Then, I focused on arms (detected via "***YOLO pose***"), assuming that in shoplifting and robbery scenarios, hand movement is faster. Finally, analyzing the entire **person bounding box** (detected via the "***YOLO tracker***") provided the best results, as it also captured full-body movements such as running (e.g., in Clip 4, where the robber runs away).

After running the algorithm across all clips, I tuned thresholds and selected the maximum magnitude as the main feature for classification. This method successfully flagged suspicious behavior in the robbery clip (Clip 4), but failed to detect the shoplifting scene (Clip 1), because the hand movements were relatively slow.

Notes, Limitations & Possible Improvements:

- ❖ The current methods rely on lightweight models due to hardware constraints. With a more powerful GPU, **transformer-based embeddings,** and **advanced optical flow models** (e.g., "***FlowNet", "RAFT"***) could significantly improve results.
- ❖ To better detect subtle actions like shoplifting, I suggest training a dedicated model specialized for that behavior. One promising approach is a **Siamese Neural Network (SNN)**. An SNN consists of identical subnetworks ("twins") with shared weights that produce embeddings for different inputs. These embeddings are compared using a distance metric to determine similarity. Training such a network on normal vs. shoplifting examples could yield a more accurate and specialized classifier.