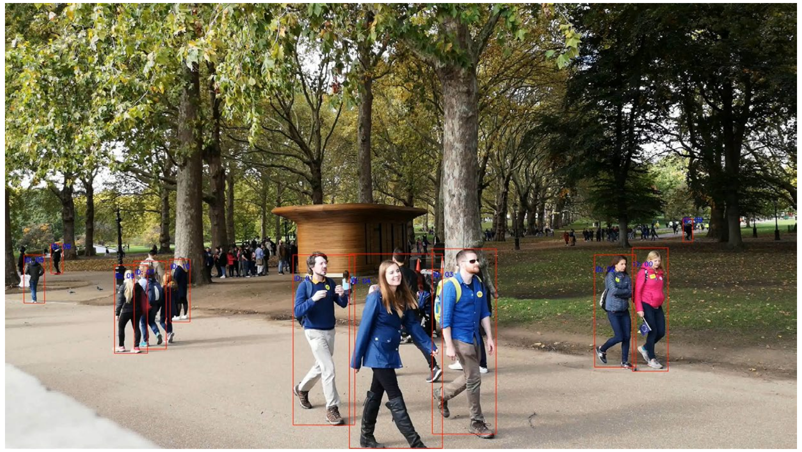
Person Reidentification

Table of Contents:

Methodology Overview

Methodology

YOLOv5 Object Detection

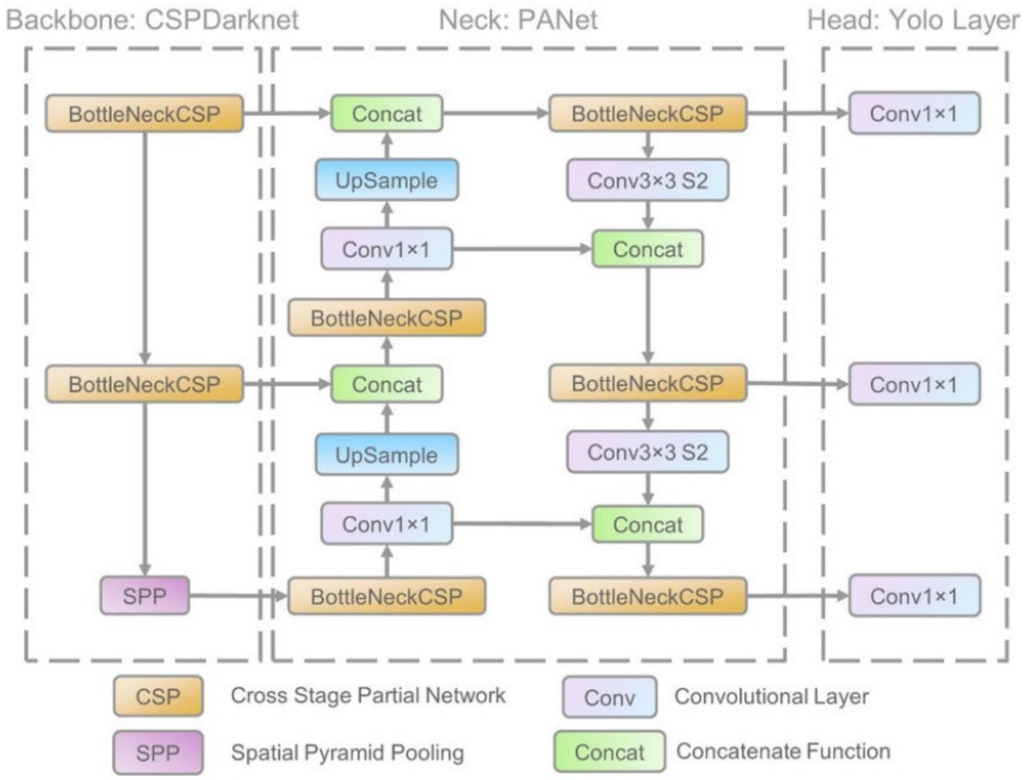


Centroid Triple Loss-based ReID

Algorithms Selection

YOLOv5 is selected as object detection algorithm

1. **YOLO** family has seen consistent improvement since its debut in year 2016.
2. It achieves **good speed-accuracy tradeoff**. One of its smallest models (**YOLOv5s**) achieves
   - mAP 56.8%
   - #Parameters: 7.2M
   - 16.5 GFLOPs
3. Primarily, opting for YOLOv5 is an **experience-based decision** where it gives exemplary performance, either in engineering-related problems (corrosion & leakage detection) or application that is close to our life (crowd detection)

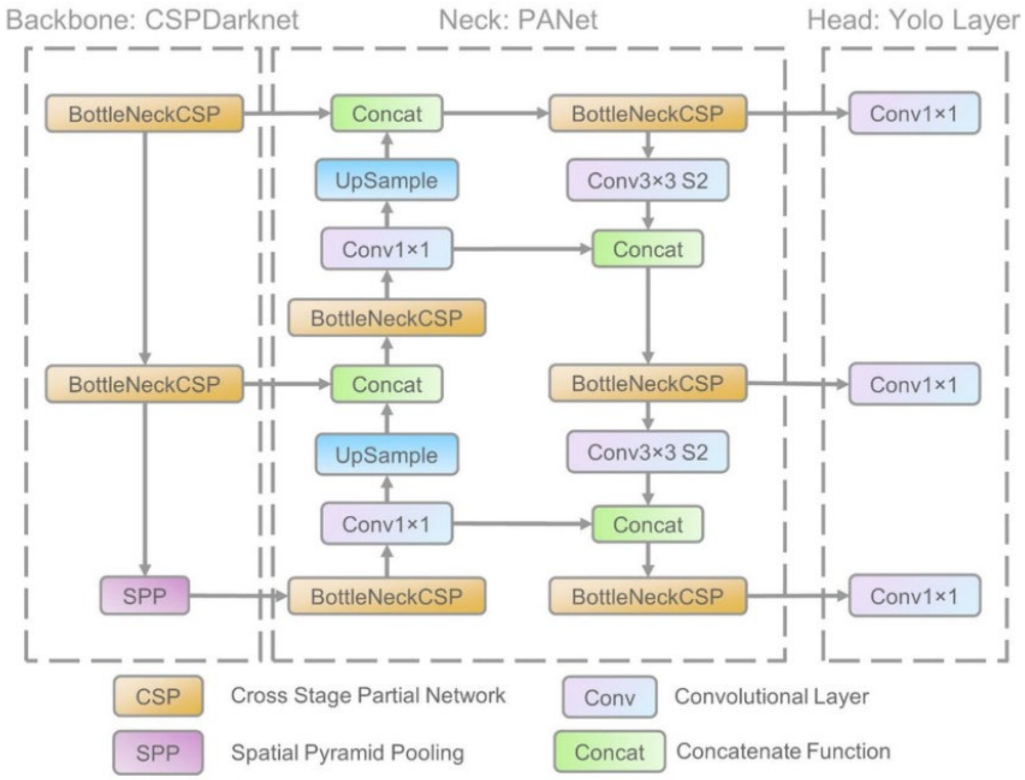


YOLOv5 Architecture

Source: [The network architecture of Yolov5. It consists of three parts: (1)... | Download Scientific Diagram (researchgate.net)]

YOLOv5 is selected as object detection algorithm (Cont'd)

1. YOLOv5 consists of the following:
   - Backbone: Cross-Stage Partial Network
   - Neck: Path Aggregation Network which features additional bottom-up for path augmentation as compared to Feature Pyramid Network for multi-scale features extraction
   - Object Detection Head
2. Loss function
   - Bounding box regression loss for prediction of bounding box coordinates
   - Binary cross entropy for prediction of objects
   - Classification loss for prediction of class



YOLOv5 Architecture

Source: [The network architecture of Yolov5. It consists of three parts: (1)... | Download Scientific Diagram (researchgate.net)]

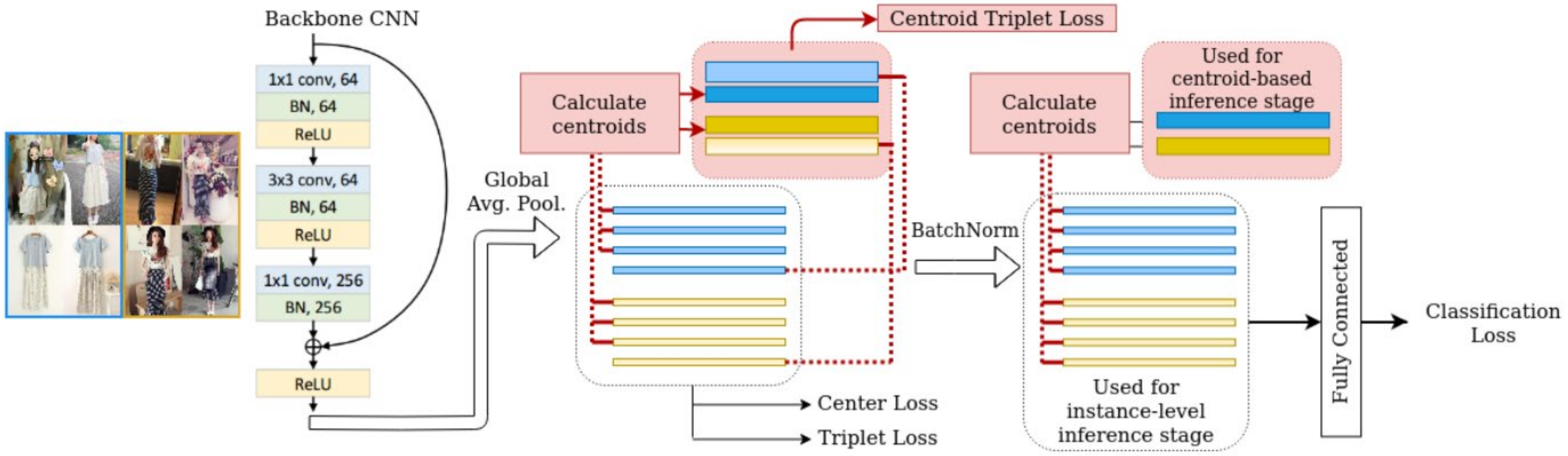
Centroid Triplet Loss-based ResNet50 is opted as person reidentification algorithm

1. Market1501 is the one of the most **popular** public dataset in person reidentification domain with **3490 citations** as recorded by Google Scholar
2. **Centroid Triplet Loss (CTL)-based ResNet50** produces **most astounding** results based on the leaderboard in paperswithcode
   - mAP 98.3%
   - Top1 Accuracy: 98%
3. Thus, it serves as good **baseline** to kickstart the development and more work can be done to further improvise its performance

Snapshot of CTL Technical Paper

Research Problem

- Although some works adopt **centroid** to learn feature representation, they are **used for training alone** and **discarded during retrieval stage**.
- **Without** employing centroid, the retrieval **time** is **large** when one query image has to be compared against multiple images of the same object for reidentification task



Methodology of CTL

Source: [On the Unreasonable Effectiveness of Centroids in Image Retrieval (arxiv.org)]

Research Objective

- To **aggregate** the representation of an item as single embedding, thus **reducing** search space, saving memory and reducing retrieval times significantly

Methodology

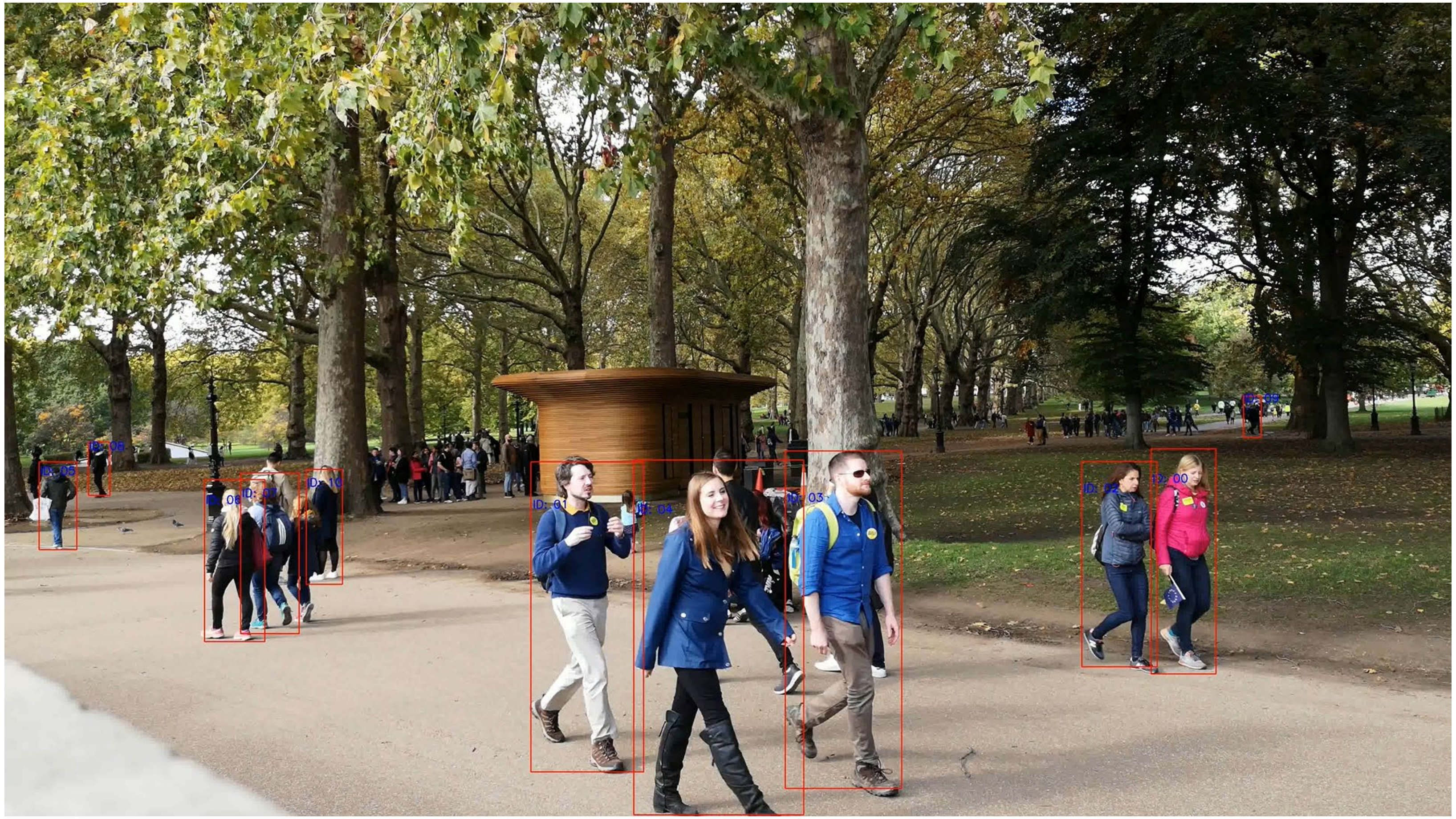
- Training: Adoption of **Centroid Triplet Loss (CTL)** as additional loss by considering distance between embeddings of anchor, centroid of positive class and centroid of negative class
- CTL **eliminates shortcoming** of triplet loss where it produces 0 loss if a sample appears closer to a randomly sampled data point from positive class as compared to that of negative class by coincidence

Results

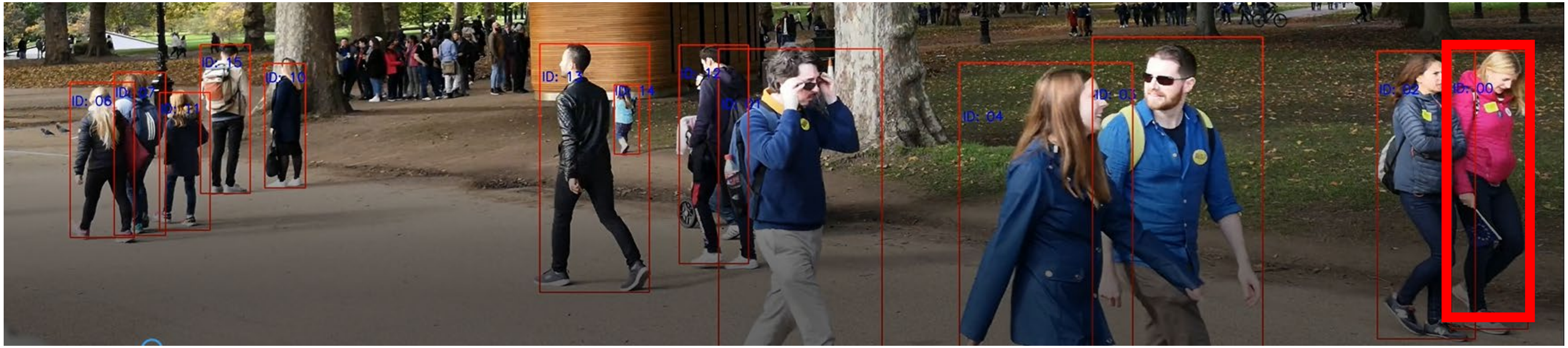
A sample of output video can be found at Google Drive

Findings

People further away from camera is not detected



Tracking based on person reid is not impeccable. 1 ID swapping is seen.



Solution has 3.61 FPS

Machine Specification

- Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz   2.59 GHz
- 32GB RAM
- Nvidia Quadro T1000 – 4GB

Comparison with Existing Trackers

Algorithm	Description	Weakness	Experience
Centroid Tracker	• **Motion-based** tracker • Tracks based on Euclidean distance between two consecutive frame	• Fast object motion will fail the tracking	• **People Counter** solution (#people entering & leaving) for a building
Deep Simple Online Realtime Tracking (Deep SORT)	• **Motion-** and **deep features embeddings-based** tracker • An improvement over SORT • **Deep** Part: Adopt **CNN** to extract feature embeddings • **SORT** Part contains **Kalman filter** and **Hungarian** algorithm • The purpose of Kalman filter to remove sensor noise. Adapting Kalman filter in person reid **eliminates detection noise** e.g. unstable detection • **Hungarian algorithm** associates each bounding box with an identity	• Assume **linear** motion i.e. consistent velocity direction	• **People Counter** • **Hardhat Detection**

Comparison with Existing Trackers (Cont'd)

Algorithm	Description	Weakness	Experience
CTL	• **Deep features embeddings-based** tracker • Kindly refer slide #8 for more info	• Depends solely on feature embeddings	-
FairMOT	• Eliminate the factor of poor detection on reidentification task by **training** both tasks **together** • Backbone: DLA or HRNet • Detection branch: CenterNet optimized through heatmap loss and box offset loss • Reid Branch: Optimized using cross entropy loss • Perform **better** than Deep SORT on MOT16 challenge	-	-

Comparison with Existing Trackers (Cont'd)

Algorithm	Description	Weakness	Experience
Observation Centric Simple Online Realtime Tracking (OCSORT)	• **Motion-based** tracker • **Robust** to occlusion and non-linear motion • **Observation-centric Online Smoothing (OOS)** eliminates error in position prediction by Kalman filter when an object is untracked • **Observation-Centric Momentum (OCM)** reduces noise to allow the assumption of linear motion (consistent velocity direction) holds • **Observation-Centric Recovery (OCR)** associates last seen observation with new detection instead of using predicted position	-	• Piping Integrity Inspection

Suggested Improvements

Suggested Improvements

Increasing Detection Accuracy

- Adopt pretrained model trained on **huge dataset** specifically for people detection
- PeopleNet by NVIDIA are trained on **71 millions** objects for **person** class

Increasing Tracking Accuracy

- Based on self-experience, OCSORT delivers good performance
- Exploration of other state-of-the-arts algorithms

Suggested Improvements (Cont'd)

Enhancing Inference Speed

- Converting native pytorch model to **TensorRT**
- Quantization: Inference on **fp16** and even **int8** but with slight drop in accuracy
- **Model pruning** by removing less important layers using Nvidia TAO Toolkit
- Use NVIDIA Trition Server or NVIDIA Deepstream as inference pipeline for speed optimization

Source Code

[shihao28/person_reid (github.com)](https://github.com/shihao28/person_reid)