# **Overview**

This system is developed to identify the staff (with name tag) among all the person in a particular area. The objective is to retrieve the frame in which staff appears and their respective bounding box (x1, y1, x2, y2). Two models will be implemented in this system, object detection model (yolov7) and classification model (Swin Transformer).

# **Requirements & Constraints**

A GitHub repo is required to be clone in the environment. The necessary libraries have to be installed as well in order to run the system. Unfortunately, it is unable to achieve a real-time detection system since each frame will go through two models: object detection and classification. So, the results will be printed and retrieved after the video is successfully processed by the 2 models within the system.

**Requirements:**

1. **Clone the repository of yolov7 from GitHub.**
   1. git clone <https://github.com/WongKinYiu/yolov7.git>
2. **Install the necessary libraries with Python = 3.7.16:**
   1. torch==1.12.1+cu116
   2. torchvision==0.13.1+cu116
   3. opencv-python==4.7.0.72
   4. pandas==1.3.5
   5. tqdm==4.65.0
   6. PyYAML==6.0
   7. matplotlib==3.5.3
   8. seaborn==0.12.2
   9. scipy==1.7.3
3. **Add environment directory to path**
   1. C:\Users\USERNAME\anaconda3\Library\bin
   2. C:\Users\USERNAME\anaconda3\Scripts
   3. C:\Users\USERNAME\anaconda3\condabin

# **Methodology**

## **3.1 Problem Statement**

The method works relatively simply: the Yolo v7 model will identify every individual in each frame, and it will be followed by a Swin Transformer model to determine whether that individual is a member of a staff (identified by a name tag).

There is no data provided (staff images) to be trained for the staff detection system, but however, a sample of a staff image with name tag is given as shown in Sample 1.



*Sample 1: Staff (with name tag)*

## **3.2 Data Preparation**

Sample creation is necessary in this case in order to train a model to classify them because no data is available for the classification model's training. In the step, we are trying to generate samples like Sample 1 with self-create name tag image as shown as Sample 2.



*Sample 2: Self-create name tag image*

The image act as the background for the name tag image will be the data set from the [Human-Aligned Bounding Boxes from Overhead Fisheye cameras dataset (HABBOF)](https://vip.bu.edu/projects/vsns/cossy/datasets/habbof/). The final product will be shown as Sample 3 in which we put a name tag on those images that consist of people.



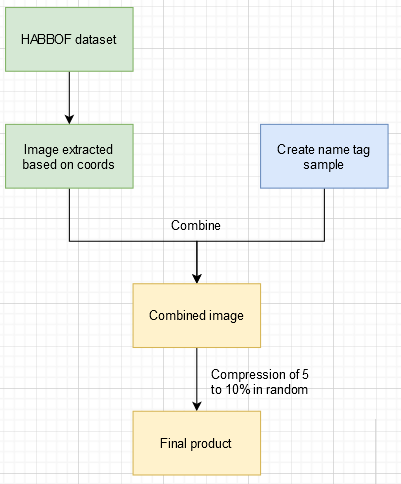
*Sample 3: Combined image*

Compression is required as well due to the high resolution of our dataset while the video has a very low quality. Without compression, the classification model most likely will be failed due to the difference of the resolutions and the pixels in the name tag generated.

*Sample 4: Compressed image Sample 5: Compressed image*

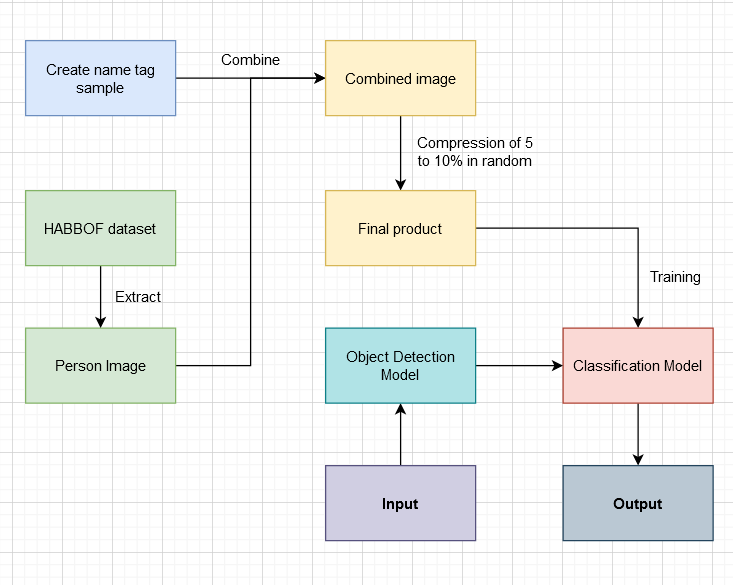
The data preparation flowchart is presented as in Figure 1.



*Figure 1: Flow chart*

## **3.5 System Design**

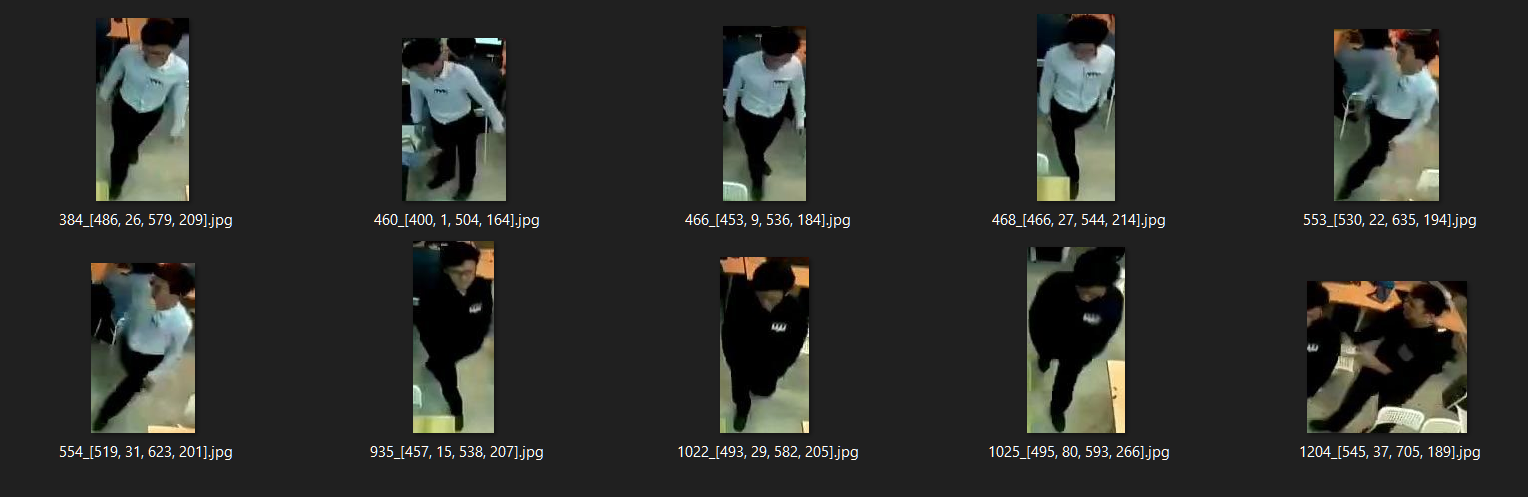
The system design is presented as in Figure 2.



*Figure 2: System Design*

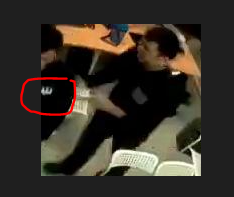
# **3.6 Evaluation**

The result is shown in the diagram 1 below with threshold = 0.5:



*Diagram 1: Results*

The two staff members who are wearing white and black shirts are detected in the sample video. The last image, as shown in diagram 2 below, is detected because a partial area of the staff with the name tag was captured by the first model.



*Figure 6: Missed captured image*

# **Files and command**

* train.py (to run the training of the classification model)
* earlystopping.py (to control the early stopping of the training process)
* transformers.py (class to perform Albumentations during the training process)
* detect.py (to run the system on an input video)
* employee\_detection.py (load yolov7 model performing person detection and return the bounding box)
* id\_classificaiton.py (classification model to identify whether he/she is a staff)
* image\_augmentation.ipynb (to produce the name tag and extract the image of a person)
* yolov7.pt (person detection model)
* swin\_b\_staff\_classification.pt (classification models)
* sample.mp4 (input video)

Command to run the test:

*python detect.py --detection\_model yolov7.pt --video sample.mp4 --thres 0.5 --staff\_model swin\_b\_staff\_classification.pt*

The results will store under runs/detect.