Intel® Unnati Industrial Training – Summer 2023

Social Distancing using Computer Vision and Deep Learning

Sahil Patil

Manipal Institute of Technology, MAHE

27-06-2023

Abstract

This project proposes a solution for social distancing detection using Computer Vision and the OpenVINO toolkit. The proposed system uses a pre-trained YOLOv5 object detection model to detect people in a video stream. The model is then used to calculate the distance between each pair of detected people. If the distance between any two people is less than a predefined threshold, the system will issue an alert. OpenVINO toolkit is used for further performance upgrades.

Introduction

YOLO (You Only Look Once) is an object detection model pre-trained on COCO dataset consisting of 80 different classes. There are different variations of YOLOv5 based on the size. The one used for this project is YOLO v5x. The model is tested on a video stream wherein it detects people in a frame and calculates the distance between their centres, and if it is lesser than a predefined threshold it alerts the viewer by change in colour; Green denoting low/no risk & Red denoting high risk.

The overall performance of the model is boosted by converting the model weights to OpenVINO Intermediate Representation format (IR) and running inference. The performance measures used to evaluate the model are speed, inference time and non-max suppression time per frame of the video.

Motivation

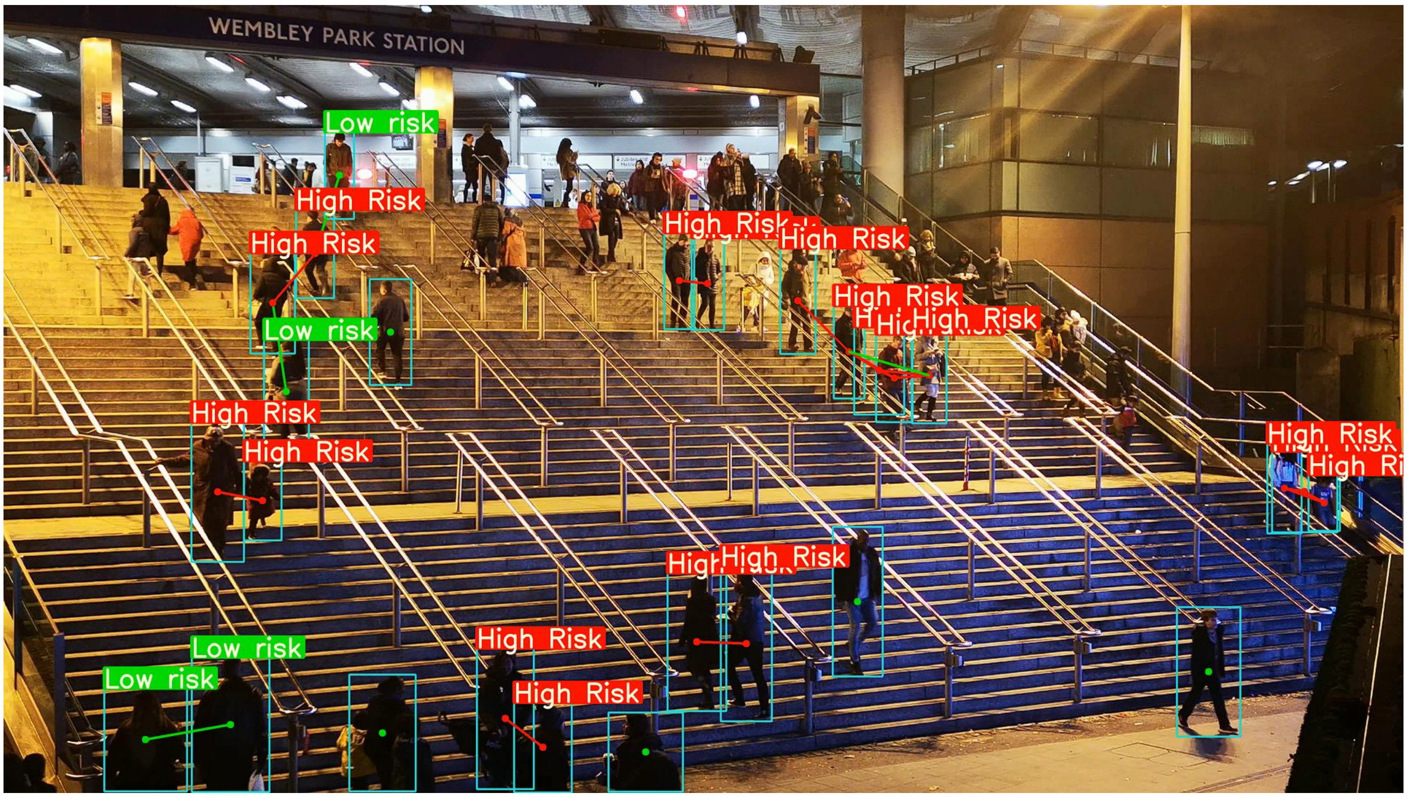
Maintaining a physical distance of specific threshold between people, is one of the best preventive measures against the coronavirus. Social distancing norms must be followed if we want lockdowns to not happen in future. Computer vision technology with Deep Learning can be of great help as one can use it to build a system that estimates the distance between any two individuals in any given frame. Thus, this project offers a solution to enforce compliance of social distancing norms.

Approach

The model is pre-trained in COCO and is tested and evaluated based on different metrics. The detect.py file runs inference in the input video stream, and export.py file converts the model weights to OpenVINO IR format consisting of .xml and .bin files. Usage of OpenVINO toolkit helps in better performance and lesser inference time. The approach used is that people in a frame are detected and a bounding box is drawn around the object, in this case a person. The Euclidean distance between the centres of bounding box is computed and if its below a user-defined/predefined threshold, it alerts by a change in colour.

Results

The following project was run on a local CPU



A frame of output video stream [Image shape: (1,3,640,640)]

Model: AMD Ryzen 5 3450U with Radeon Vega Mobile Gfx

Without OpenVINO optimization

Pre-process (ms)	Inference (ms)	Non-Maximum suppression (ms)	Throughput (frames/s)
1.7	**2544.3**	**4.6**	**0.38**

With OpenVINO Optimization

Pre-process (ms)	Inference (ms)	Non-Maximum suppression (ms)	Throughput (frames/s)
2.9	**2267.2**	**4.4**	**0.44**

References

https://docs.openvino.ai/2022.3/notebooks/226-yolov7-optimization-with-output.html#verify-model-inference

https://docs.openvino.ai/2022.3/omz_demos_social_distance_demo_cpp.html

https://docs.openvino.ai/2022.3/notebooks/001-hello-world-with-output.html

https://gist.github.com/helena-intel/3bf9e91c605645f7a8af9c704ac105c8

https://github.com/DeepNeuralAI/CV-Social-Distancing-Detector/tree/master

https://github.com/ChargedMonk/Social-Distancing-using-YOLOv5

The steps to execute the project are given in the README.md file in the link provided below.

GitHub Link: https://github.com/S4HILp/Social-Distancing-Detection-using-Computer-Vision