Title – Detection of face mask and helmet using computationally efficient neural network.

Team Member and Student Id – Kunal Sanghvi, 011809708

Abstract – This is a company-sponsored project. I will be in touch with a company named UtopiaTech Pvt. Ltd. in India and solve a problem statement that will be a component of their final product. The motivation behind choosing this topic is that I want to develop some applications that can solve real-world problems. The unique advantage my project will provide is that it will identify the best neural network for face mask detection, which has an almost low computation cost and can be easily deployed on hardware such as the Raspberry Pi. This project will help me thoroughly understand every concept of neural networks.

Introduction – The development of this project will be from the perspective of a real-world application. This application is going to be installed in an ATM facility. The camera will always be at the top-end corner of the facility, so achieving a similar kind of dataset will be a big task. Yet, we expect to generate our dataset, which can be as close as possible to the real-world setting. I will be in touch with the co-founder, Mr. Mitesh Bajaria, regarding the project, which in turn will help me with any support required in terms of hardware or software.

There are multiple models available, such as Mobilenet SSD and Yolo, with several versions along with a tiny version to deploy on hardware. However, these models provide less accuracy at a high computational cost. Hence, my aim while developing the project will be to identify the best configuration of models that provide the maximum accuracy and have the least computational cost when deployed on hardware, thus giving us a fair FPS to process and work on.

The plan is divided into four steps as follows:

1. Literature review to identify all the possible models which can be used for the application.
2. Generate and gather the dataset.
3. Train a bunch of models and find an accurate setting that has the most accuracy and least computational cost.
4. Try to deploy the model on a hardware setting and get a proof of concept of the application.

Once this application is developed, we can increase our dataset with multiple applications, train our model, and compare its efficiency. Thus, this will create a complete use case for a real-world application.

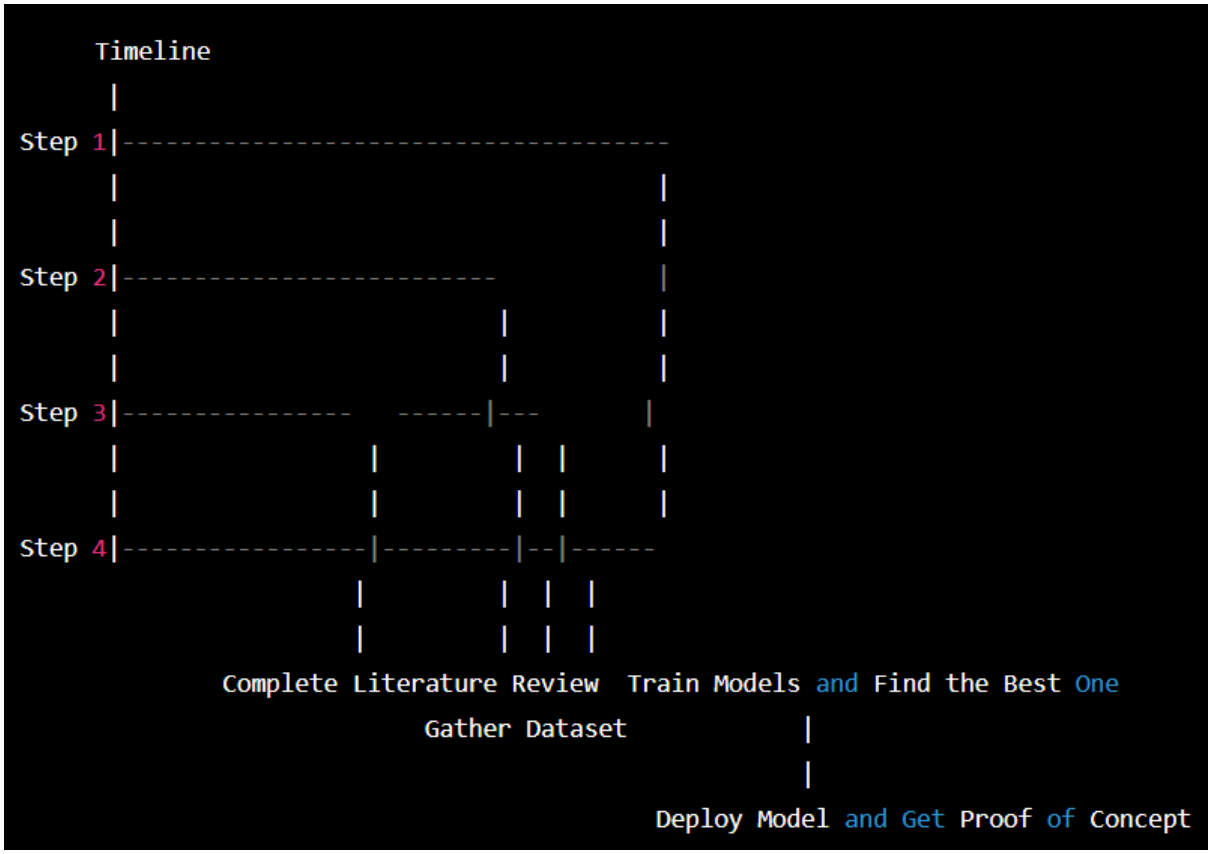
Literature Review – The paper presents a solution for real-time face mask detection using Tensorflow Lite on a Raspberry Pi device, with AI models deployed in the cloud, which is referred to as an AIoT (Artificial Intelligence of Things) application. This solution is intended to help enforce compliance with face mask requirements in public places during the COVID-19 pandemic. The proposed solution consists of two components: a client-side component running on the Raspberry Pi, and a server-side component running in the cloud. The client-side component uses a camera module to capture live video feed and then uses Tensorflow Lite to run a pre-trained AI model for detecting faces and classifying whether they are wearing masks. The server-side component uses Flask, a Python web framework, to expose a RESTful API for storing and retrieving trained AI models. The client-side component can use this API to download the latest version of the AI model for face mask detection. The authors trained two different AI models: one based on a pre-trained MobileNet model for face detection, and another based on a custom-built convolutional neural network (CNN) for face mask classification. The latter model achieved an accuracy of 98.7% on a test set of 1,500 images. The authors conducted experiments to evaluate the performance of the proposed solution in terms of accuracy, latency, and resource usage. They found that the MobileNet-based model for face detection had an accuracy of 96.8% and a latency of 67 milliseconds on average, while the custom-built CNN-based model for face mask classification had an accuracy of 98.2% and a latency of 34 milliseconds on average. They also found that the solution consumed an average of 40% CPU and 25% RAM on the Raspberry Pi. Overall, the authors demonstrate that the proposed solution can provide real-time face mask detection with high accuracy and low latency, using readily available hardware and AI models deployed in the cloud. This makes it a potentially useful tool for enforcing face mask requirements in public places during the COVID-19 pandemic.[1]

This paper proposes a deep learning-enabled face mask detection system for access/egress control using TensorFlow Lite-based edge deployment on a Raspberry Pi. The system is designed to help enforce compliance with face mask requirements in public places during the COVID-19 pandemic. The proposed system uses a camera module connected to a Raspberry Pi to capture a live video feed, which is processed using TensorFlow Lite to detect faces and classify whether they are wearing masks. The authors trained two different models: one based on a pre-trained MobileNet model for face detection, and another based on a custom-built convolutional neural network (CNN) for face mask classification. The CNN model achieved an accuracy of 98.86% on a test set of 1,000 images. The authors conducted experiments to evaluate the performance of the system in terms of accuracy, latency, and resource usage. They found that the MobileNet-based model for face detection had an accuracy of 96.7% and a latency of 45 milliseconds on average, while the custom-built CNN-based model for face mask classification had an accuracy of 98.9% and a latency of 67 milliseconds

on average. They also found that the system consumed an average of 28.7% CPU and 32.5% RAM on the Raspberry Pi. The authors also implemented an access control system using the proposed face mask detection system, which allows or denies access to a designated area based on the presence of a face mask. The system achieved a recognition rate of 97.4% for individuals wearing masks and 94.2% for individuals not wearing masks. Overall, the authors demonstrate that the proposed system can provide accurate and efficient face mask detection using readily available hardware and deep learning models deployed on the edge. The system has potential applications for access/egress control in public places during the COVID-19 pandemic, as well as other applications such as surveillance and security.[2]

This paper proposes a system for mask detection and social distance identification using the Internet of Things (IoT) and the Faster R-CNN algorithm. The system is designed to help enforce compliance with face mask requirements and social distancing guidelines in public places during the COVID-19 pandemic. The proposed system consists of three components: a camera module connected to a Raspberry Pi, an IoT module, and a cloud-based server. The camera module captures live video feed, which is processed using the Faster R-CNN algorithm to detect faces and classify whether they are wearing masks. The system also uses computer vision techniques to identify the distance between individuals in the video feed. The IoT module is used to collect and transmit data from the camera module to the cloud-based server, where it is analyzed and visualized in real time. The authors developed a web-based dashboard to display the data, which includes the number of individuals wearing masks, the number of individuals not wearing masks, and the distance between individuals. The authors conducted experiments to evaluate the performance of the system in terms of accuracy, latency, and resource usage. They found that the Faster R-CNN algorithm achieved an accuracy of 94.8% on a test set of 1,000 images for mask detection and a mean average precision of 88.6% for object detection. They also found that the system achieved an average latency of 160 milliseconds and consumed an average of 30% CPU and 22% RAM on the Raspberry Pi. The authors demonstrate that the proposed system can provide accurate and efficient mask detection and social distance identification using readily available hardware and computer vision algorithms. The system has potential applications for enforcing face mask requirements and social distancing guidelines in public places during the COVID-19 pandemic, as well as other applications such as surveillance and security. The web-based dashboard also provides a useful tool for monitoring and visualizing the data collected by the system in real time.[3]

Technical Plan –



The horizontal axis represents the timeline of the project, and the vertical axis represents the four steps of the plan. Each step is represented by a horizontal bar, with the start and end points of the bar indicating the start and end of the step in the project timeline.

Step 1, Literature review, is the first step and is represented by a single bar. Step 2, Generate and gather the dataset, begins before step 1 is completed and ends before step 3 begins. Step 3, Train a bunch of models and find an accurate setting, overlaps with Step 2 and starts before Step 2 ends. Step 4, Try to deploy the model and get a proof of concept, which starts after step 3 is completed.

Dataset – We generated our own dataset consisting of 1500 images and used data augmentation which is an inbuilt function in yolo and hence tried to increase the possible number of images in the dataset.

Model – We trained a lot of models ranging from TensorFlow, TensorFlow Lite, XNOR, PyTorch, Darknet, MXNet, and Caffe, and finally having a perfect setting we ended up with Yolov3-tiny where we achieved 91.3% train accuracy and 85.7% test accuracy. We used cross-validation to separate test data from the train data, hence providing us with some options to test the model. Also, we used pruning while having a set of various

parameter options. The final setting, we are using after thousands of variations are 38 layers, 8669002 parameters, and 0 gradients.

Deployment – The model was deployed on a Raspberry Pi and a laptop and the FPS which we got was quite near 1.5 on both devices.

Results – The model when trained was deployed on-site on a laptop with a webcam interfaced with the code and the video was then processed to detect a helmet present in the video it was bounded with a box stating the detection and the following figures show it as follows:

Fig 1: Helmet Detection with confidence 0.56

Fig 2: Helmet Detection with confidence 0.80

Fig 3: Helmet Not Detected

The model trained was deployed on my laptop using the inbuilt camera and the video was then processed to detect a mask present in the video it was bounded with a box stating the detection and the following figures show it as follows:

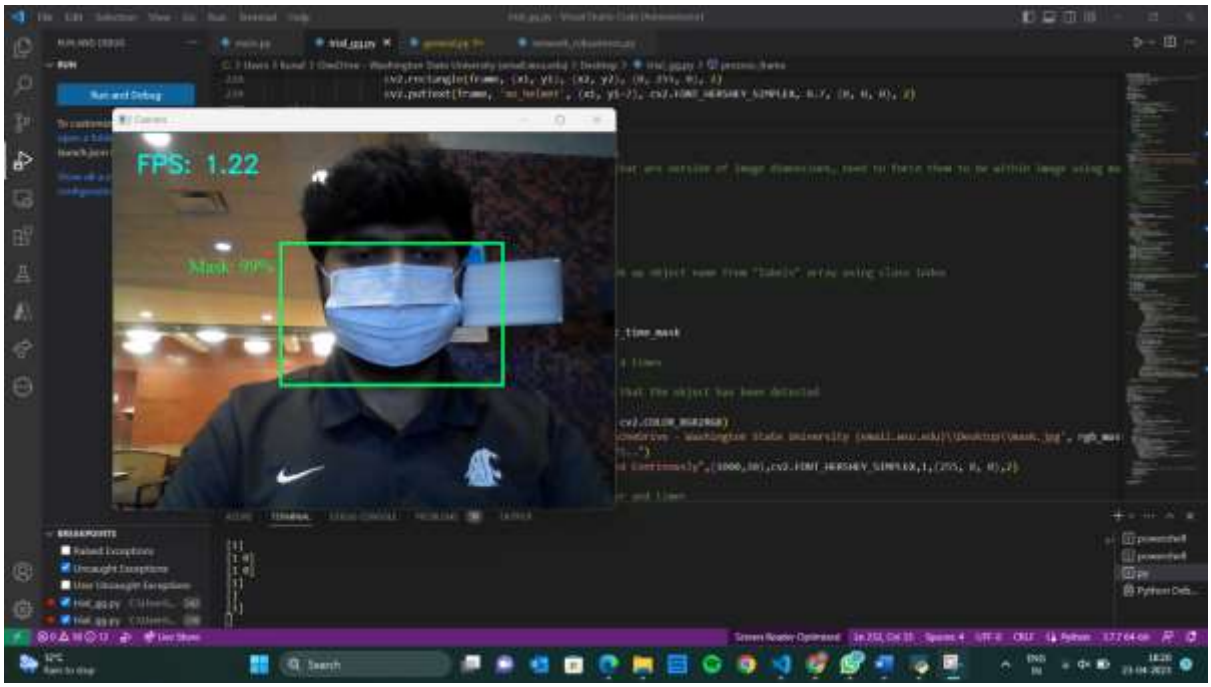


Fig 4: Mask Detection

More about helmet detection can be found on the video link which was shot in a simulated environment on-site –

https://emailwsu-my.sharepoint.com/:v:/g/personal/kunal_sanghvi_wsu_edu/Eetsbdc5Z6NDpVyugV94c9AB8HVbCYhGzsmA3pzpY5Ulvw?e=aOf5WE

Future Work –

This topic can be further extended as a research option. After training a lot of models still, we could not achieve good accuracy in detecting helmets as you can see in Fig 3 the helmet is still worn by the person, yet it is not getting detected based on the camera image quality and its setting since it is in the dark part. We feel the number of images in the dataset like the situation can be increased which could be the future work to increase the dataset with all the possible situations. Mask detection worked well. Also, we can research ways to decrease the computational time which can further increase the FPS when deployed on mobile hardware like Raspberry Pi. Research on these areas provides a great opportunity for future work.

References –

1. P. Subramanian, S. Nagarajan, and S. Sivapriya, “Real-Time Face Mask Detection using Tensorflow Lite on Raspberry Pi using AI models deployed in the cloud: An AIoT application,” in Proceedings of the 4th International Conference on Internet of Things and Connected Technologies (ICIoTCT), 2019, pp. 175–181.
2. S. K. Dutta, P. Saikia, and K. Sarma, “On the use of Deep Learning Enabled Face Mask Detection For Access/Egress Control Using TensorFlow Lite Based Edge Deployment on a Raspberry Pi,” in Proceedings of the 2021 International Conference on Electronics, Information, and Communication (ICEIC), 2021, pp. 1–5.
3. S. Gopi, R. Raghavendra, K. J. Raja, and V. N. Bhadran, “Mask Detection and Social Distance Identification Using Internet of Things and Faster R-CNN Algorithm,” in Proceedings of the 2020 International Conference on Smart Electronics and Communication (ICOSEC), 2020, pp. 436–441.