**Title: Facial Mask Detection System**

**Audience: Eng. Yazan Kheiry**

**Course Colleagues**

**CEO Managers.**

**Employers**

**Co. Hr. Managers**

# Abstract

COVID-19 pandemic has rapidly increased health crises globally and is affecting our day-to-day lifestyle. A motive for survival recommendations is to wear a safe facemask, stay protected against the transmission of the virus. Manual monitoring of correct wearing of facemasks is difficult task to be managed in open and closed areas. Therefore, the working on this project aimed to provide an end to end face mask detection system and provide a user friendly model to monitor correct wearing of facemasks especially for crowded areas. Also, it allow users to get summary report showing the percentage of compliance in wearing facemasks. Using Kaggle datasets, the proposed model is trained and tested. The system runs in real-time and detect if an individual face has facemask, if not, the system notify the security and monitor man through visual representation shown on the monitor screen. Additionally, the features of the system allow processing archived images from different sources and providing results on compliance of wearing facemasks. [1]

# Background

The recent pandemic of COVID-19 has called for very strict health protocols overall the world. According to the World Health Organization (WHO), COVID-19 define as “infectious disease caused by the most recently discovered coronavirus” [2]. This new virus were unknown before the outbreak began in Wuhan, China, in December 2019. COVID-19 presents a risk at areas such as markets and public buildings where people are still attending in person and physically close to each other.

The spread of COVID-19 virus is through close contact with the people and in crowded public areas. The guidelines listed by the WHO, primary precaution should be taken to prevent the spread of virus is to wear facemask [3].

In this project, a real-time facemask detection model was developed by fine tuning VGG19 Pre-trained model for image classification and recognition. The developed system can be used for indoor and outdoor facilities such as schools, universities, shopping malls, multiplex etc. It can help in monitoring individuals automatically and check whether they are wearing facemask. Additionally, it can help to break the chain of spreading of virus when in close contact and reduces the positive cases which are rapidly increasing day-by-day overall the globe.

# Project Objectives

The main goal of this system is “to build an end to end face mask detection Model that can help in minimizing the risk of COVID-19 spread between individuals, through detecting visitors' compliance in wearing the face mask correctly in public and private outdoor and indoor places”.

The specific objectives of this system are:

* Allow beneficiaries/users to apply real-time monitoring of visitors and check their compliance in wearing face masks.
* Facilitate beneficiaries/users to follow up of customers compliance with national regulations associated to wearing face masks.
* Provide daily report summarizing the percentage of compliance visitors (wearing face masks). Which in result, can be used, when needed, to adapt the security procedures for more compliance.
* Save and retrieve data using local SQLite database (during no internet connection) and cloud Firebase.
* For archive files, the system can check images to predict those complying with correct wearing of face masks vs. others.

# System Development

## Summary

The system operates by run a live video of visitors in specific area (such as building), this usually obtained from the monitoring cameras installed within the facility. Then, the system take shots (images) each second and sending them through Api web service to the server. A pre-processing of shots is then take place to extract the faces. The extracted faces from the previous step are then send to the model for facemasks detection to identify those without masks, incorrect wearing of masks, and those correctly wearing the masks. Afterwards, the system return outputs summarizing results of facemasks detection, time, and location of the face in the originally taken shot. The outputs are then stored on local database and cloud database. Final results of mask detection are presented on the monitor showing correct wearing of masks in green color, incorrect wearing of mask in blue color, and faces without masks in red color. Reports are saved automatically on the DVR on daily bases.

Using the same model, the system allow processing archived images from different sources and providing results on compliance of wearing facemasks.

Add flow chart

First the system will provide a live video (in real time) face mask estimation of face mask detection plus it takes a shot every 1 second then sends the image to Api server which detects the facial in each shot and classify each face in a single shot if it is with mask or without mask or wearing mask incorrectly, return the result to the client in a nice visualization way in the real time camera monitor by showing a green rectangle around the section of the face indicating that the person is wearing a mask, a red rectangle indicates that the person is not wearing a mask, and a blue rectangle shows that the person is not wearing a mask, this allows the security man to easily detect the customers with no mask wearing. Plus the ability to send a shot manually by the user to the model to detect the faces and return the detection output to the client visualized in the same image. In real time monitoring system, at the end of the day when the security man shutdown the system, it automatically returns a nice formatted readable daily report as csv file and send a copy by e-mail to the user for future checking purposes. The system also saves the report in the local SQLite database and in the cloud Firebase in synchronized way. So the user can easily save and retrieve the data automatically even if there is no internet connections available.

## System Components

Inputs to the system

Client monitoring cameras (security cameras) are used as a source of input to the system. The live streaming videos are processed to take images every second for the purpose of face mask detection.

Dataset

The dataset was used to train the model for classifying different ways of wearing face masks, and to make diverse and unbiased detection. It consists of 3 classes: a) Incorrect mask b) With mask C) without mask.

The used dataset for this system was the FMD\_DATASET [4], which has a total of 14535 images. This dataset is chosen because it is universal dataset and thus allows to build a face mask detection system that can detect almost all types of face masks with different orientations.

The classification of images within this dataset were as the following:

* incorrect masked class consists of 5000 images, of which 2500 are Mask on Chin and 2500 are Mask on Mouth and Chin.
* With mask class has 4789 images, of which 4000 are simple with mask and 789 are complex  
  with mask images.
* without mask has 4746 images, of which 4000 are simple and 746 are complex images.

The definition of each classification categories as follows:

• "Mask OnChin" images: These are the images in which masks are put on a chin only. The mouth and the nose of a person are visible.  
• "MaskOnChinMouth" images: In this, the mask is covering the chin and the mouth area. The nose of a person is not covered.  
• "Simple WithMask" images:- It consists of data samples of face masks without any texture, logos, etc.  
• "Complex WithMask" images: It includes the images of the sophisticated face masks with textures, logos, or designs printed on them.  
• "Simple WithoutMask images: These are images without any occlusion.  
• "complex WithoutMask" images: It consists of faces with occlusion, such as beard, hair, and hands covering the face.

Model:

The heart of the system is the detection part or (Model detector) which consists of two detection stages: the first stage, was a prebuilt Mediapipe libraries which used to detect faces in the taken images (shots). The second stage, was developed based on fine tuning of VGG 19 model to detect way of wearing face masks. This model was built and trained using transfer learning techniques to fit our desired detection output. Using this technique allow us to easily and rapidly reach high accuracy and precession (validation accuracy= 98% in our model), the system applies some preprocessing (resize the image and encoding the labels) and post processing (label decoding, detection output visualization, report format) to make the data suitable for our model.

Application Programming Interface (API):

API is a software intermediary that allows two applications to communicate to each other. API architecture is usually explained in terms of client and server. The application sending the request is called the client, and the application sending the response is called the server [5].

In our system, the client send the Image through API to the server and the server then returns the detection output for the user as a response. API is also used to send requests to save the data locally and in the firebase cloud, in addition to make a request to retrieve data that have been saved.

Database:

Both, firebase cloud and SQlite3 database were used to periodically save detection data. The SQlite3 database used to save data locally to enable the client to easily save and retrieve data during periods of no internet connection. While the firebase cloud used as a backup. Automatic synchronization between the two databases performed by the system to keep the data continuously updated.

Monitor:

The output of the system which is the faces location and the mask detection presented on the security monitor to allow the user to easily and frequently configure the customers’ compliance. The monitor showing correct wearing of masks in green color, incorrect wearing of mask in blue color, and faces without masks in red color.

System Software

API Code

Model Code

Client Code

Testing and Evaluation

Results and Conclusions

**2.Introduction**

**2.2 Background**

**2.2.1 End to end solution system**

**2.2.1** **The difference between image classification and object detection**

**2.2.2 What is Transfer Learning?**

Transfer learning is the task of using a pre-trained model and applying it to a new task, i.e., transferring the knowledge learned from one task to another. This is useful because the model doesn’t have to learn from scratch and can achieve higher accuracy in less time as compared to models that don’t use transfer learning

**The use of transfer learning in the machine learning domain has surged in the last few years. The following are the top reasons:**

* **Growth in the ML community and knowledge sharing: The research and investments by top universities and tech companies have grown exponentially in the last few years and there is also a strong desire to share state-of-the-art models and datasets with the community. This allows people to utilize pre-trained models in a specific area bootstrap quickly.**
* **Common sub-problems: Another key motivator is that many problems share common sub-problems, e.g., in all visual understanding and prediction areas, tasks such as finding edges, boundaries, and background are common sub-problems. Similarly, in the text domain, the semantic understanding of textual terms can be helpful in almost all problems where the user is represented by text terms, including search, recommendation systems, ads, etc.**
* **Limited supervised learning data and training resources: Many real-world applications are still mapped onto supervised learning problems where the model is asked to predict a label. One key problem is the limited amount of training data available for models to generalize well. One key advantage of doing transfer learning is that we have the ability to start learning from pre-trained models, and hence, we can utilize the knowledge from similar domains.**

**Techniques for transfer learning utilization**

**The transfer learning technique can be utilized in the following ways: Extract features from useful layers**

* **Keep the initial layers of the pre-trained model and remove the final layers. Add the new layer to the remaining chunk and train them for final classification. Fine-tuning**
* **Change or tune the existing parameters in a pre-trained network, i.e., optimizing the model parameters during training for the supervised prediction task. A key question with fine-tuning the model is to see how many layers can we freeze and how many final layers we want to fine-tune. This requires understanding the network structure of the model and role of each layer, e.g., for the image classification model we used in the Image data example, once we understand the convolution, pooling, and fully connected layers, we can decide how many final layers we need to fine-tune for our model training process.** [2]

**Deep learning models are widely used for object detection in computer vision systems, which are pre-trained to detect several common objects using large datasets [7]. Transfer learning, one of the most efficient ways to use these models for custom object detection, is used**

**here. Mask detection is one of the most efficient ways to prevent COVID infection, entitling its need as essential. Therefore, there is a requirement to monitor people with and without masks. Social distancing comes as an added aid for prevention wherein the detected people in the image can be measured using the detection parameters like the bounding box. The dataset needed to train any model must be diverse and suitable for real-time monitoring scenarios, crowds with masks, and the like. A dataset for masks is used here, where it has a diverse set of images in terms of race, with and without crowds, blurred and clear, modelling multiple instances of real situations from a surveillance point of view [8]. This dataset, the prime focus of this paper, was used to train and test several models that are fast and accurate, well suited for real time object detection and recognition problems. Social distancing and mask detection have been implemented individually in several state-of-the-art publications. However, a pipeline for mask detection and social distancing is rare.**

**In our project we will perform fine-tuning with deep learning following these steps:**

   
Remove the fully connected nodes at the end of the network (i.e., where the actual class label predictions are made).

 Replace the fully connected nodes with freshly initialized ones.

 Freeze earlier CONV layers earlier in the network (ensuring that any previous robust features learned by the CNN are not destroyed).

 Start training, *but only train the FC layer heads*.

[3]

First, we’ll load the VGG19 architecture (with pre-trained ImageNet weights) from disk, leaving off the fully connected layers By omitting the fully connected layers, we have effectively put the network in a guillotine to behead our network, From there, we define a new fully connected layer head

**2.2.3**

**2.2.4 learning rate Decaying Schedule:**

 the learning rate, \alpha, controls the “step” we make along the gradient. Larger values of \alpha imply that we are taking bigger steps. While smaller values of \alpha will make tiny steps. If \alpha is zero the network cannot make any steps at all (since the gradient multiplied by zero is zero).

When training our network, we are trying to find some location along our loss landscape where the network obtains reasonable accuracy. **It doesn’t have to be a global minima or even a local minima, but in practice, simply finding an area of the loss landscape with reasonably low loss is “good enough”.**

**Using Decaying in learning rate decrease our learning rate, thereby allowing our network to take smaller steps** — this decreased learning rate enables our network to descend into areas of the loss landscape that are “more optimal” and would have otherwise been missed entirely by our learning rate learning.

The Keras library ships with a time-based learning rate scheduler — it is controlled via the decay parameter of the optimizer class (such as SGD, Adam, etc.).

Internally, Keras applies the following learning rate schedule to adjust the learning rate after every batch update — it is a **misconception** that Keras updates the standard decay after every epoch.

The update formula follows: lr = init\_lr * \frac{1.0}{1.0 + decay * iterations}

Using polynomial learning rate decaying schedule, our **learning rate is decayed to zero over a fixed number of epochs**.

The rate in which the learning rate is decayed is based on the parameters to the polynomial function. A smaller exponent/power to the polynomial will cause the learning rate to decay “more slowly”, whereas larger exponents decay the learning rate “more quickly”. [4]

**2.2.5**  Evaluation metrics:

In order to have a fast and accurate model for real-time object detection, it is essential to measure the performance of the model and compare the model with certain metrics. mAP and FPS are used here to choose the optimal model for mask detection. Thus accuracy is measured in terms of detection and speed of detection in terms of Frames inferred per second. In order to identify the mAP, precision and recall has to be computed. The precision and recall are computed by identifying the True positive (TP), True negative (TN), False Positive (FP) and False Negative (FP). The True positive in general terms are referred for images that where labelled true and the prediction produced true. The True negative are for images that are labelled true but predicted as false. False positive on the other hand are results that are labelled as false but predicted as false and False negative are the images that are labelled false but predicted as true. In the context of mask detection, due to multiple objects and multiple classes in one image itself, TP, FN,FP and TN were all measured for each detection based on the objectness of the detection resultant from the intersection of Union measurement which is as stated below.

IoU: Intersection of Union:

The Intersection of Union (IoU), a commonly used evaluation metric which estimates regression quality. It is the IoU between a predicted bounding box and its corresponding assignment ground truth box [38]. The overlap of two boundaries is measured. The real object boundary is the ground truth to the predicted object boundary. In object detection models, True Positive (TP), False Positives (FP), False Negative (FN) and True Negative (TN) are defined by setting an IoU threshold. This paper uses a standard threshold of 50 percent match to classify the true positive or false positives in detection [41]. IoU which equals to or over 50% is set to True positive (TP) that is a correct detection, and IoU below 50% is False positive, a wrong detection. A FN is determined by counting an object not detected.

Precision and Recall:

Precision is the percentage of the time your prediction is correct. Recall measures how well all the positive predictions are found. Eq. (5), and Eq. (6) are used to calculate precision and recall [41]. 𝑃 𝑟𝑒𝑐𝑖𝑠𝑖𝑜𝑛 = 𝑇 𝑃 𝑇 𝑃 + 𝐹 𝑃 (5) 𝑅𝑒𝑐𝑎𝑙𝑙 = 𝑇 𝑃 𝑇 𝑃 + 𝐹 𝑁 (6)

Average Precision (AP):

Precision is the percentage of the time your prediction is correct. Recall measures how well all the positive predictions are found. Eq. (5), and Eq. (6) are used to calculate precision and recall [41,46]. The average precision can be calculated as the area enclosed under the Precision–Recall curve plotted on the coordinate axis. It is denoted as 𝐴𝑃 = ∫ 1 0 (𝑃 𝑅) 𝑑𝑅 In terms of object detection models the mean of the AP is calculated at different recall values.

Mean Average Precision:

mAP, or mean Average Precision, is the metric used to determine performance on various object detection models. The classification and localization of the image are specifically determined [41]. The ground truth of object detection models, the class of the objects, and the bounding box of each object serve as parameters for calculating the mAP. The correctness of each image is determined by a bounding box through intersection of union (IoU), which is the ratio between the intersection and union of the predicted boxes and ground truth. A confidence threshold is kept, determining positive and negative boxes. From this, the mAP is calculated with a fixed confidence threshold and IoU value, typically 50% for Pascal VOC datasets. Different confidence thresholds are chosen in such a way that recall varies from 0 to 1, which makes AP, the mean of precision values at different recall values. Therefore, mAP is the average precision of all the classes in the dataset [39,41] [5]

**2.2.6** [Cross-Validation](https://scikit-learn.org/stable/modules/cross_validation.html)

[**Scikit-Learn’s K-fold cross-validation**](https://scikit-learn.org/stable/modules/cross_validation.html)**feature randomly splits the training set into K distinct subsets called folds. Then it trains and evaluates the model K times, picking a different fold for evaluation every time and training on the other K-1 folds.**

**Methodology:**

**1. define the problem statement**

**2.Collecting datafrom Kaggle website**

**3.Load the data and represent it and study the main features , check if the data un balanced**

**4. selecting the labels and apply the specific encoding method**

**5.splitting the data**

# 6. Data Preparation and preprocessing data

# 7. Selecting and Training Machine Learning Models

# 8. Evaluating model

# 9. Enhance the model by cross validation and fine-tuning the hyper parameters to obtain a high performance

# 10. Evaluate the Entire System

# 11. save the model into a file to be further loaded and used by the web service

# 12. A web service — that gives a purpose for your model to be used in practice. We’ll use Flask to develop this service.

# Developing a web service

# The next step is to package this model into a web service that, when given the image through a POST request, returns the Mask predictions as a response. By using the Flask web framework, a commonly used lightweight framework for developing web services in Python.

# 13. A cloud and local database to save the output

**Results:**

**Model performance can be validated in different ways. One of the popular methods is using the confusion matrix. Diagonal values of the matrix indicate correct predictions for each class, whereas other cell values indicate a number of wrong predictions.**

**model\_report** = **ClassificationInterpretation**.**from\_learner(Model)**

**model\_report**.**plot\_confusion\_matrix()**

Conclusion

An efficient system for social distancing and mask detection was chosen by experimental analysis and comparison to state-of-art litera-ture with two datasets, the MOXA3K and the ViDMask dataset. Transfer learning was performed on five deep learning models. YOLOV5 was found to be efficient in terms of mAP and FPS for mask detection compared to state-of-art literature. The weights for this model are used for social distance measurement. The efficiency of social distance monitoring is thereby tagged to the efficiency of face mask detection, achieving intelligent mask detection and social distancing surveillance. A new, diverse and challenging video dataset, VIDMASK, containing more than 60 videos for face mask detection, was used for training and inference. This dataset produced high mean average precision for YOLOR with added noise and image rotation. This was a challenge for YOLOv4, YOLOv5 and Fast Mask RCNN with Resnet-101 backbone. VidMask can further be explored for mask detection and social distancing on the edge by compressing and quantizing the models or identifying lighter models with less computational complexity requiring only CPU usage. Besides the evident requirement for COVID situation, this can be further used for epidemics that require masks or environments that require proper social distancing and mask wearing suchas laboratories that may have contaminants. Due to different cameraangles used, it not only has a surveillance perspective, the model can be mounted on ground devices such as robots. This dataset can be further used for crowd counting applications for crowd that wear masks as some of the videos are of crowded scenes. Domain adaptation is a deep

learning training process that can be used to adapt the model to specific domains thus can improve the model for diverse purposes requiring less computational complexity. In addition to the above, it can be used as a tool for social experiments where effectiveness of masks and social distancing can be confirmed.

Acknowledgement

This project work was made possible by the support of Upskilling program held by HTU in Amman/Jordan (2022) with support of GIZ.

We would like to thank Eng. yazan Kheiry support the instructor of Data Science course for his helpful feedback and discussion.