2021 6th International Conference for Convergence in Technology (I2CT)
Pune, India. Apr 02-04, 2021

Multi-Stage CNN Architecture for Face Mask Detection

Amit Chavda
iPing Data Labs LLP
Mumbai, India
amit.chavda@iping.in

Jason Dsouza
iPing Data Labs LLP
Mumbai, India
jason.dsouza@iping.in

Sumeet Badgujar
iPing Data Labs LLP
Mumbai, India
sumeet.badgujar@iping.in

Ankit Damani
iPing Data Labs LLP
Mumbai, India
ankit.damani@iping.in

Abstract* - Coronavirus Disease 2019 (COVID-19) broke out at the end of 2019, and it's still wreaking havoc on millions of people's lives and businesses in 2020. There is an upsurge of uneasiness among people who plan to return to their daily activities in person, as the world recovers from the pandemic and plans to get back to a state of regularity. *Wearing a face mask significantly reduces the risk of viral transmission and provides a sense of protection, according to several studies. However, manually tracking the implementation of this policy is not possible. The key here is technology. We present a Convolutional Neural Network (CNN) based architecture for detecting instances of improper use of face masks. Our system uses two-stage CNN architecture that can detect both masked and unmasked faces and is compatible with CCTV cameras. This will aid in the tracking of safety violations, the promotion of face mask use, and the creation of a safe working environment.

Keywords - Face Masks; CNN; Object Detection; COVID-19; Object Tracking

I. INTRODUCTION

Accelerated advances in technology have propelled us to a point where we can now accomplish feats that seemed impossible only a few decades ago. Machine Learning and AI have made life a lot easier and presented solutions to a variety of complex problems in various fields. In visual perception tasks, Machine Learning and Deep Learning algorithms are approaching human-level performance. Technology is acting as a lifesaver in the fight against the Coronavirus Disease (COVID-19) pandemic. Work from home has replaced our normal work routines and has become a part of our daily lives thanks to technological advancements. However, some industries are unable to adapt to this new norm.

Individuals are still hesitant to return to work as the pandemic settles and such sectors become eager to resume in-person work. Returning to work has become a source of anxiety for 65 percent of employees [1]. Face masks have been shown in multiple studies [2] and [3], to reduce the risk of viral transmission, while also providing a sense of protection. However, manually enforcing such a policy on large premises and tracking any violations is impractical. A better alternative is to use Computer Vision. We developed an effective system that can detect the use of face masks by people in images and videos using a combination of image classification, object detection, object tracking, and video analysis.

We present a dual-stage CNN architecture in which Stage 1 detects human faces, and Stage 2 uses a lightweight image classifier to classify and localize the faces detected by Stage 1 as either 'Mask' or 'No_Mask'. This method has an advantage over other object detectors, in that it makes use of transfer learning and pre-trained models. As a result, the proposed system achieves high accuracy while using less training data. This eliminates the need for a large human-annotated dataset and compute resources, which are required by most of the modern object detectors. In comparison to other object detectors, the inference time is significantly shorter. To track safety violations, promote the use of face masks, and ensure a safe working environment, this system can be integrated with an image or video capturing device, such as a CCTV camera. To extend our application to videos, we combined an object tracker with our face mask detection system. The detected faces are then tracked between frames, resulting in robust and consistent video feed detections that are immune to motion blur, jitter, frame transition, frame drop, and other common video detection challenges. This tracking algorithm improves video performance significantly without sacrificing quality or any processing time overhead

II. RELATED WORK

A classical object detection model can deal with the issue of detecting multiple masked and unmasked faces in pictures. Object detection primarily includes localizing the objects inn images and then classifying them. Traditional algorithms like Haar Cascade [4] and Histogram Of Gradients [5] have proven to be effective for this purpose, but they depend heavily on Feature Engineering, and offer an overall inferior performance as contrasted to modern techniques like Neural Networks in terms of speed and accuracy.

Object detection algorithms that use CNNs as their backbone are now widely used for such tasks. They are divided into 2 broad categories: Multi-Stage Detectors - In these, the detection process is broken into several steps. A dual-stage detector like RCNN [6] first examines and comes up with a list of regions of interest using selective search. The CNN feature vectors are then withdrawn from each region one-by-one. Single-Stage Detectors - A single-stage detector performs detections in a single pass. These algorithms do not use the region proposal phase used in multi-stage detectors, and thus have higher speed, at the cost of some loss of accuracy. One of the most popular single-stage methods, You Only Look Once (YOLO) [7], is widely

2021 6th International Conference for Convergence in Technology (I2CT) | 978-1-7281-8876-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/I2CT51068.2021.9418207

used for object detection and can achieve nearly real-time performance. As many countries began enforcing precautionary efforts against the Covid-19, many implementations of Face Mask Detection systems came forth.

The authors of [8] have produced facial recognition on masked and unmasked faces applying Principal Component Analysis (PCA). However, if the recognized face is masked then the recognition accuracy falls below 70%.

The authors of [8] have produced facial recognition on masked and unmasked faces applying Principal Component Analysis (PCA). However, if the recognized face is masked then the recognition accuracy falls below 70%.

The work in [9] proposed an approach to determine face mask wearing conditions. They split the face mask covering conditions into three classes: correct face mask covering, incorrect face mask covering, and no face mask. Their process selects a picture, detects and crops faces, and later runs SRCNet [10] to implement image super-resolution and classify them.

[11] suggested an approach that identifies the presence or absence of a medical mask. The main target of this approach was to set off an alert particularly for medical staff who do not wear a surgical mask and to reduce as many false-positive face detections as feasible, without missing any medical mask detections.

The writers of [12] introduced a model that comprises two components. The initial component uses ResNet50 [13] for feature extraction. An ensemble of classical Machine Learning algorithms is then used in the next stage for face mask classification. The authors assessed their system and determined that Deep Transfer Learning methods would bring about better results. The authors came into such a conclusion since the building, comparing, and choosing the best model among a collection of classical Machine Learning models is a time-consuming process. We strongly believe that our proposed system covers up all the drawbacks mentioned in this research. Our two stage CNN based system eliminates the need of a large annotated data (which is required to train most modern object detectors), through the smart use of pre-trained models. Our system also breaks down the challenging problem of face mask detection and localization, into a face detection (much easier problem) and masked face classification, which offers faster inference as well as ensures that no masked faces are missed for classification.

III. Proposed Methodology

We introduce a dual stage model architecture which is capable of identifying and localizing faces as masked or non masked

A. Architecture Overview

The system architecture we propose (input image from the dataset in [14]). There are two major stages to it. A Face Detector is included in the stage 1 of our architecture, which pinpoints multiple faces in images of various sizes and detects faces even in interlaced scenarios. The detected faces (roi) are then batched together and passed to the stage 2 of our architecture, a CNN model for Face Classification as masked or non masked. The results from Stage 2 are post processed, and the final result is an image with all of the faces correctly localized and categorized as masked faces or unmasked faces.

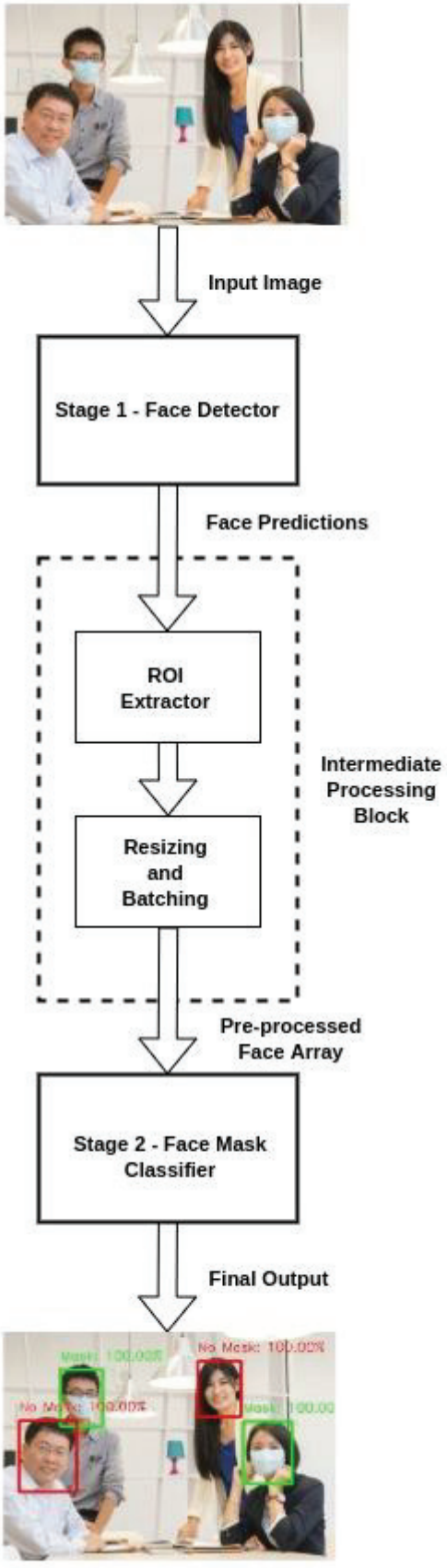

Fig. 1. Architecture Block Diagram

B. *Stage 1 - Face Detector*

The first stage of our system is a face detector. This stage is fed a raw RGB image as an input. All of the faces detected in the image are extracted and output by the face detector, along with their box coordinates. The process of accurately detecting faces is critical for our architecture. An accurate face detector needs a significant amount of good quality data, ample time, and appropriate compute resources. We selected a pre-trained model that is trained on a big dataset, for quick generalization and stable detection. RetinaFace [15] was chosen as our Stage 1 model, and its performance was compared to that of two other popular Face Detection Models, Dlib [16] and MTCNN [17]:

RetinaFace - It is a pixel-wise localization one stage model that utilizes a multi-task learning strategy to predict bounding box coordinates , prediction score, and keypoints for all faces, all at the same time.

Dlib - The Dlib face detector is based on C++.It outperforms its predecessor, the Dlib HOG-based face detector, by a wide margin.

MTCNN - For detecting and localising faces and facial keypoints, it employs a three stage cascade architecture.

The detection process is difficult for the model used at this stage because it must detect human faces that may be obscured by masks. In section IV.B, we cover the experimentation and comparative analysis for the first stage model.

C. *Intermediate Processing Block*

This block processes the detected faces and groups them together for classification which is done by Stage 2. The bounding boxes for the faces are generated by the detector from Stage 1. Stage 2 necessitates a thorough examination of the person's entire head in order to accurately classify the faces as masked or unmasked. Most cases consist of a first step, in which the bounding boxes are expanded up to 120% of their original area, which contains the required Region of Interest, without much overlap with any other faces. Extracting out the image's expanded bounding boxes to obtain the ROI for each detected face is the next step. Stage 2 requires that the extracted faces be resized and normalized. We batch the face ROIs, so that we can reduce processing time by using batch processing instead of detecting faces in a loop.

D. *Stage 2 - Face Mask Classifier*

In Stage 2, we have a module containing a CNN based Face Mask Classifier. The processed ROI from the previous stage is sent to Stage 2 for classification as a masked or non-masked face. For this stage, a CNN classifier was trained using three image classification models: MobileNetV2 [18], DenseNet121 [19], and NASNet [20]. These models feature a light architecture that provides high performance with low latency, making them ideal for video analysis. This stage produces an image (or frame) with localized faces with a bounding box that highlights if the face has a mask or its non masked.

E. *Dataset*

We collected the images for masked faces and unmasked faces and created our datasets, from other datasets publicly accessible, along with some data gathered by scraping data online. Masked images were taken from the Real-world Masked Face Recognition Dataset [21] and Face Mask Detection dataset on Kaggle, by Larxel [14].

Fig. 2. Masked Face Images: RMFRDs

Fig. 3. Masked Face Images: Larxel (Kaggle)

The images from RMFRD are biased towards Asian faces. Therefore, masked images from the Larxel (Kaggle) were combined to the dataset to get rid of this bias. RMFRD also contains images for unmasked faces. However, as discussed before, they were heavily biased towards Asian faces. Hence, we chose not to use these images. The Flickr-Faces-HQ dataset introduced in [22] was used for unmasked images.

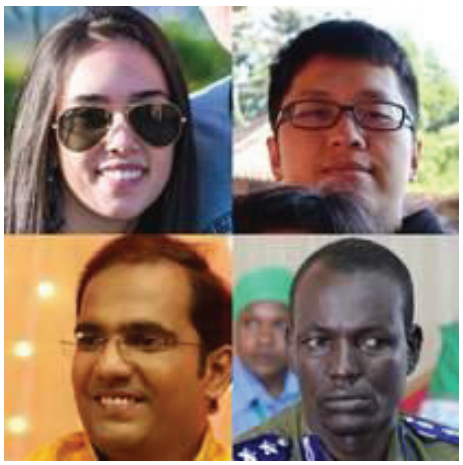

Fig. 4. Non-Masked Face Images:Flickr Faces HQ

Our dataset also consists of pictures of improperly put on face masks or palms masking the face, which become labeled as non-masked faces.

Fig. 5. Incorrectly worn face masks or face covered by hand

We ran the assembled RAW data through Stage 1 (Face Detector) and the Intermediate Processing Block of the architecture. We carried out this process to ensure that the distribution and nature of training data for Stage 2 match the required input for Stage 2 during the final deployment.

Fig. 6. Extracted ROIs from RAW Base Images

The final dataset has 7855 images, belonging to two classes:

TABLE I. FACE MASK CLASSIFIER DATASET

Class Name	Description	No. of images
Mask	Faces with masks correctly used	3440
No_Mask	Faces with no masks or masks incorrectly used	4415

F. Face Mask Classifier Model Training

Three CNN classifiers were trained for the second stage to classify images as masked or unmasked face . We built our models using Keras [23]. We used ImageNet weights as our initial weights, instead of using Xavier Initialization [25]. Using a ratio of 80:10:10, the dataset was divided into training, cross validation, and testing set. The ImageDataGenerator class in Keras was used to augment the data. The input image was set to 224 x 224 pixels. We chose a learning rate of 1e-4 as the starting point. Aside from that, the training process included saving the weights for the lowest validation loss, decreasing the learning rate when the training reached a plateau, and stop the training using the early stopping callback.

IV. EXPERIMENTAL ANALYSIS

(Images used in this section are self-obtained or belong to the dataset in [14])

A. Face Mask Classifier Training Statistics

TABLE II. TRAINING METRICS FOR STAGE 2 CLASSIFIER

Model Name	Training		Validation		Test Acc. (%)
	Acc. (%)	Loss	Acc. (%)	Loss	
NASNetMobile	**99.82**	**0.001**	**99.45**	**0.018**	99.23
Dense Net121	99.49	0.016	98.73	0.031	**99.49**
Mobile NetV2	99.42	0.018	99.36	0.03	99.23

We can conclude from Table II that all three models have excellent statistics. Overall, the NASNetMobile model performs slightly better than the other models.

TABLE III. EVALUATION METRICS FOR STAGE 2 CLASSIFIER

Model Name	Precision	Recall	F1-Score
NASNetMobile	98.28	**100**	99.13
DenseNet121	**99.70**	99.12	**99.40**
MobileNetV2	99.12	99.12	99.12

DenseNet121 has the highest F1-Score, as shown in Table III. The other models, on the other hand, are not far behind. As a result, other aspects of performance comparison, such as inference speed and model size, had to be measured in order to select the final Face Mask Classifier Model.

B. Face Detector Comparison

We compared RetinaFace's performance to that of Dlib DNN and MTCNN. Based on a set of images(mask and no mask), the average inference times for each of the models were calculated. The RetinaFace model is the most effective of the three.

TABLE IV. INFERENCE STATISTICS OF STAGE 1

Model Name	Inference Time per Frame (in sec)		
	480p	720p	1080p
Dlib	3.62	8.20	14.65
MTCNN	0.57	0.68	1.26
RetinaFace	0.09	0.11	0.19

On images taken from a very short distance with no more than two people in the image, all three models produce good results. However, it was discovered that as the number of people in the images grows, Dlib's performance suffers. Dlib also has trouble detecting masked or hidden faces.

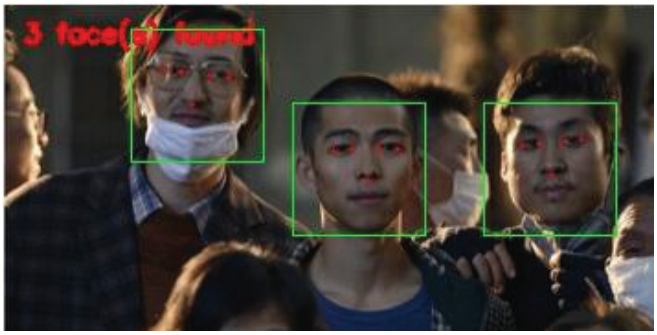

Fig. 7. (a). Good performance by Dlib on normal faces; (b). Poor performance by dlib on faces covered by face masks

RetinaFace and MTCNN outperform Dlib in terms of detecting multiple faces in images. Both of them are capable of detecting masked faces in image. MTCNN has a good performance while detecting face sin front view. However, when detecting faces from the side view, it exhibits a poor performance.

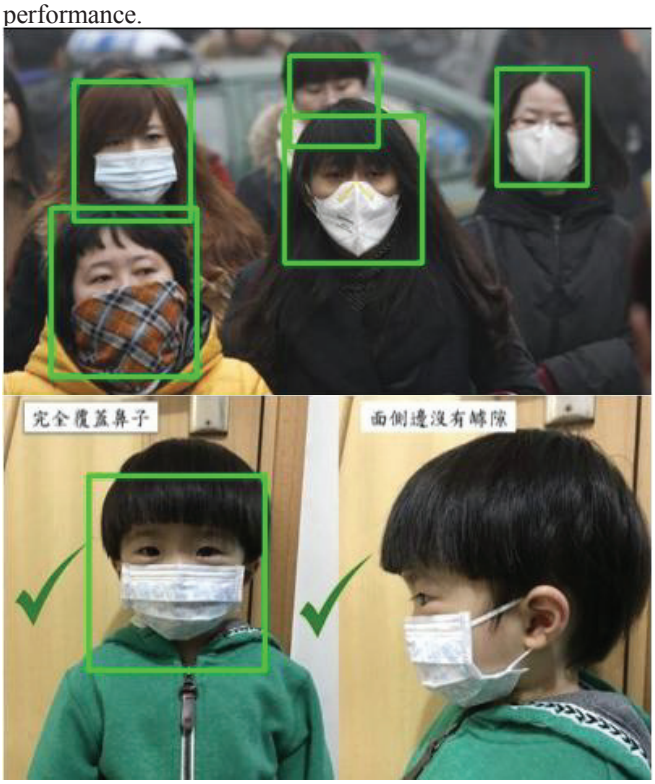

Fig. 8. (a). Good performance by MTCNN on covered faces; (b). Low performance by MTCNN on side faces

Conversely, side view faces were detected with a high accuracy by RetinaNet. RetinaFace reduces the failure rate from 26.31 percent to 9.37 percent (the NME threshold is 10 percent) when compared to MTCNN [15, Page 6]. As a result, for our first stage, we chose RetinaFace for detecting and localizing faces.

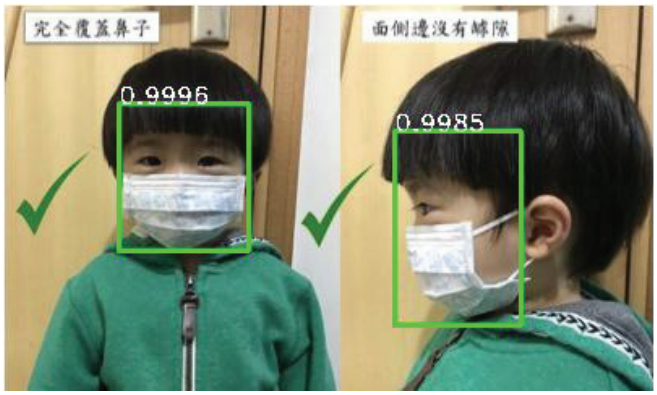

Fig. 9. (a). Excellent performance by RetinaFace on covered faces; (b) Excellent performance by RetinaFace on faces in side v

C. *Face Mask Classifier Comparison*

TABLE V. INFERENCE SPEED AND MODEL SIZE COMPARISON OF FACE MASK CLASSIFIER

Model Name	Average Inference Time (in seconds)	No. of model parameters (in millions)
NASNetMobile	0.295	4.88
DenseNet121	0.353	8.52
MobileNetV2	0.118	4.07

NASNetMobile and DenseNet121 produce better results than MobileNetV2 and are nearly identical. The results in Table V show that NASNet is significantly faster than DenseNet121. NASNet also has a smaller model size than DenseNet121 (due to a smaller number of parameters). As a result, the model loads faster during inference. As a result of these factors, NASNetMobile is far better suited for real-time applications than DenseNet121. As a result, NASNetMobile was chosen as the Face Mask Classifier's final model.

D. *Final Results*

Incorporating all the stages of our architecture, we therefore obtain an exceptionally accurate and robust Face

Mask Detection System. RetinaFace was chosen as our Face Detector in Stage 1, while the NASNetMobile based model was chosen as our Face Mask Classifier in Stage 2. The resultant system indicates tremendous performance and can recognize face masks in pictures with multiple faces over a vast spread of angles.

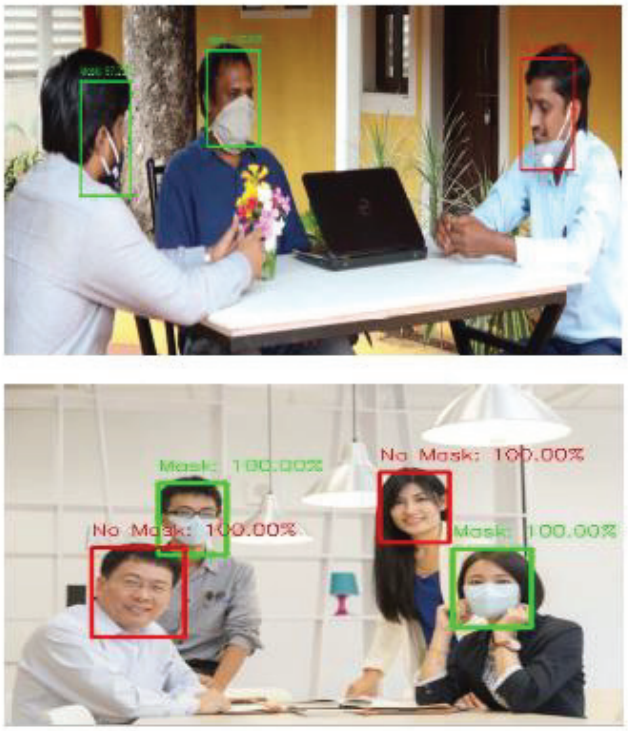

Fig. 10. Final Detection Results

E. Video Analysis

For real-world scenarios, it is useful to extend an object detection system to run on videos as. Motion blur, dynamic focus and frame transition are some of the challenges faced in video feed analysis. However, in order to ensure that the detection is consistent, the model performance between consecutive frames must be robust. Most object detection models trade accuracy for processing time. Furthermore, training such a highly accurate model would require a large amount of data, time and compute resources. Even then, the model might face issues due to the aforementioned challenges faced in video detection. Instead of training a more accurate but computationally expensive detection model, we used the process of Object Tracking. Not only does this bolster the performance in video detection, but it is also computationally less expensive and gives faster and consistent results in videos, and this makes it a suitable candidate for real-time applications.

A modified version of Centroid Tracking, inspired by [26] by us, in order to track the detected faces between consecutive frames. The video frame 'n' is passed through the Stage 1 of the system, which returns the bounding box coordinates of all the detected faces. The centroids of the faces detected by Stage 1 are then calculated by the Centroid Tracking algorithm. These centroids and their respective bounding box coordinates are cached in memory, and the current frame 'n' along with its cached data is passed to the subsequent stages to generate the results for that frame. Now, the next frame 'n + 1' is processed similarly by the Stage 1 and the tracking algorithm calculates the centroids for this frame. The spatial distance between all the combinations of the cached centroids and the current centroids (frame 'n + 1') are calculated. Each cached centroid and respective box coordinates are replaced by its closest centroid from frame 'n + 1' and box coordinates respectively, if the distance between the centroids is less than D_{thresh}. If any centroid in frame 'n + 1' does not replace any cached centroid, then it is attributed to a newly detected face and cached in memory along with its box coordinates. The newly cached centroids and box coordinates are sent to the subsequent stages in the system and detections are generated. This process is repeated for every frame in the video stream. If any cached centroid does not get updated for a predefined number of consecutive frames (f_{thresh}), then it is assumed that the detected face is no longer present in the video stream, and the cached information is cleared from memory. This ensures that our detection algorithm is resilient to the several challenges usually faced in videos, like motion blur and noise, where other algorithms could fail to detect some objects.

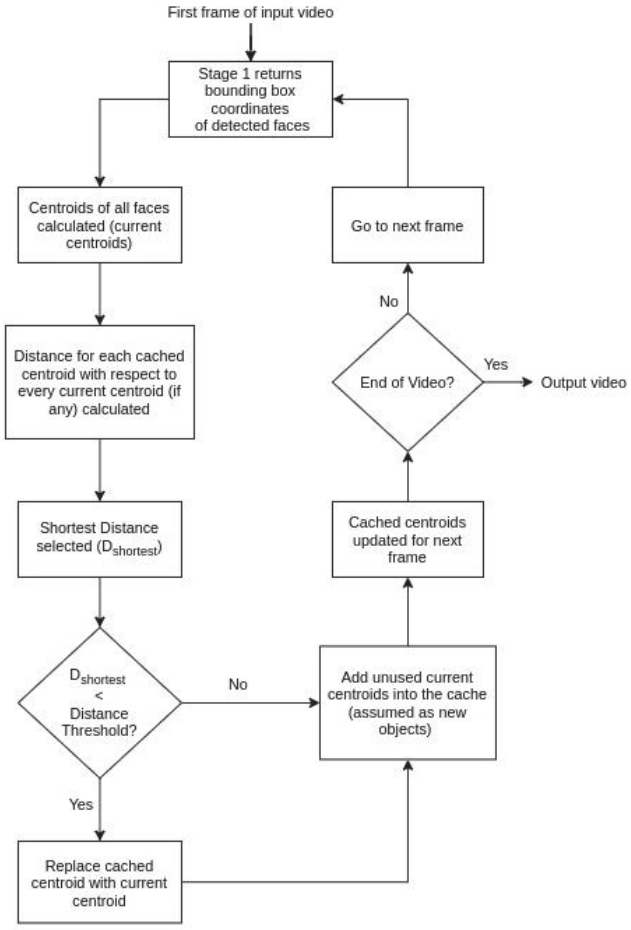

Fig. 11. Implementation of Centroid Tracking in Our System

The tracker ensures that even if a previously detected face is missed by the detector in the consecutive frame during frame transition, the ROI coordinates for that face are still stored in the cache, thus ensuring robust detection over video streams. Based on experimentation, for a 30 FPS video, we found that a f_{thresh} value of 15 gave good results . A substantial improvement was observed in face mask detection in video streams after using this method. The

results were very stable and the bounding boxes were steady during frame transition.

The figures below show the difference in results with and without the inclusion of the Centroid Tracking module:

Fig. 12. (a). Results without Object Tracking; (b). Results for the Same Frames with Object Tracking

V. Discussion

A. Conclusions

In this paper, a dual-stage Face Mask Detection architecture was introduced. The first stage comprises of a pre-trained RetinaFace model for stable face detection, after comparing its results with other models like Dlib and MTCNN. We created a dataset of masked and unmasked faces was that contained no biased images. For the second stage, we trained three different Face Mask Classifier models on the generated unbiased dataset, and based on several evaluation metrics, the NASNetMobile model was finalized for classifying the detected faces as masked faces or non-masked faces. Moreover, a technique called Centroid Tracking was added to our system, which helped improve the overall stability and results on video data. As the COVID-19 pandemic recedes, and the world looks to return to a state of normalcy, this system can be easily deployed for automated monitoring of the proper usage of face masks at public places and workplaces, which will help make them secure.

B. Future Scope

We will be working on a few things in the near future:

• On a CPU, the model currently provides a 5 FPS inference speed. We hope to improve this to 15 FPS in the future, making our solution suitable for CCTV feeds without the use of a GPU acceleration.

• Machine Learning is becoming increasingly popular in the field of mobile deployment. As a result, we intend to convert our models to TensorFlow Lite versions.

• Our architecture can achieve better inference performance on edge devices and make our models multi-threading efficient, by using TFRT acceleration.

• In the future, improved models that provide better accuracy and lower latency can easily replace Stage 1 and Stage 2 models.

References

[1] Woods, A., BDaily News, Jun 2020, Britain faces an anxiety crisis as people return to work, https://bdaily.co.uk/articles/2020/06/22/britain-faces-an-anxiety-crisis-as-people-return-to-work.

[2] Howard, J., Huang, A., Li, Z., Tufekci, Z., Zdimal, V., van der Westhuizen, H., von Delft, A., Price, A., Fridman, L., Tang, L., Tang, V., Watson, G.L., Bax, C.E., Shaikh, R., Questier, F., Hernandez, D., Chu, L.F., Ramirez, C.M., Rimoin, A.W., Face Masks Against COVID-19: An Evidence Review, Preprints, 2020, 2020040203, (doi: 10.20944/preprints202004.0203.v1).

[3] Verma, S., Dhanak, M., & Frankenfield, J. (2020), Visualizing the effectiveness of face masks in obstructing respiratory jets, Physics of fluids (Woodbury, N.Y. : 1994), 32(6), 061708. https://doi.org/10.1063/5.0016018.

[4] Viola, P., and Jones, M., Rapid object detection using a boosted cascade of simple features, Computer Vision and Pattern Recognition, 2001, CVPR 2001, Proceedings of the 2001 IEEE Computer Society Conference on, vol. 1. IEEE,2001, pp. I–I.

[5] Dalal, N., and Triggs, B., Histograms of oriented gradients for human detection, CVPR '05, 2005.

[6] Girshick, R., Donahue, J., Darrell, T., and Malik, J., Rich feature hierarchies for accurate object detection and semantic segmentation, Proceedings of the IEEE Conference on Computer Vision and pattern recognition, 2014, pp. 580–587.

[7] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A., You Only Look Once: Unified real-time object detection, IEEE Conference Computer Vision and Pattern Recognition (CVPR), 2016.

[8] Ejaz, M.S., Islam, M.R., Sifatullah, M., Sarker, A., Implementation of principal component analysis on masked and non-masked face recognition, 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), 2019, pp. 1–5.

[9] Qin, B., and Li, D., Identifying facemask-wearing condition using image super-resolution with classification network to prevent COVID-19 (2020), Unpublished results, doi: 10.21203/rs.3.rs-28668/v1.

[10] Dong, C., Loy, C.C., Tang, X., Accelerating the Super-Resolution Convolutional Neural Network, in Proceedings of European Conference on Computer Vision (ECCV), 2016.

[11] Nieto-Rodríguez, A., Mucientes, M., Brea, V.M., (2015), System for Medical Mask Detection in the Operating Room Through Facial Attributes. In: Paredes R., Cardoso J., Pardo X. (eds) Pattern

Recognition and Image Analysis. IbPRIA 2015. Lecture Notes in Computer Science, vol 9117. Springer, Cham. https://doi.org/10.1007/978-3-319-19390-8_16.

[12] Loey, M., Manogaran, G., Taha, M. H. N., & Khalifa, N. E. M. (2021) (in press), A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the COVID-19 pandemic, Measurement: Journal of the International Measurement Confederation, 167. https://doi.org/10.1016/j.measurement.2020.108288.

[13] He, K., Zhang, X., Ren, S., and Sun, J., Deep residual learning for image recognition, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[14] Larxel, 2020 May, Face Mask Detection, Kaggle - Version 1, https://www.kaggle.com/andrewmvd/face-mask-detection.

[15] Deng, J., Guo, J., Ververas, E., Kotsia, I., and Zafeiriou, S., RetinaFace: Single-Shot Multi-Level Face Localisation in the Wild, 2020, IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 5202-5211, doi: 10.1109/CVPR42600.2020.00525.

[16] Sharma, S., Shanmugasundaram, K., and Ramasamy, S.K., FAREC — CNN based efficient face recognition technique using Dlib, 2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), Ramanathapuram, 2016, pp. 192-195, doi: 10.1109/ICACCCT.2016.7831628.

[17] Zhang, K., Zhang, Z., Li, Z., and Qiao, Y., (2016), Joint face detection and alignment using multitask cascaded convolutional networks, IEEE Signal Processing Letters, 23(10):1499–1503.

[18] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L., MobileNetV2: Inverted Residuals and Linear Bottlenecks, 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, 2018, pp. 4510-4520, doi: 10.1109/CVPR.2018.00474.

[19] Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q., Densely Connected Convolutional Networks, 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 2261-2269, doi: 10.1109/CVPR.2017.243.

[20] Zoph, B., Vasudevan, V., Shlens, J., & Le, Q., (2018), Learning Transferable Architectures for Scalable Image Recognition, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[21] Wang, Z., Wang, G, Huang, B., Xiong, Z., Hong, Q., Wu, H., Yi, P., Jiang, K., Wang, N., Pei, Y., Chen, H., Miao, Y., Huang, Z., & Liang, J., 2020, Masked Face Recognition Dataset and Application.

[22] Karras, T., Laine, S., and Aila, T., A Style-Based Generator Architecture for Generative Adversarial Networks, 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019, pp. 4396-4405, doi: 10.1109/CVPR.2019.00453.

[23] Chollet, F., & others, (2015), Keras, https://keras.io.

[24] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L., ImageNet: A Large-Scale Hierarchical Image Database, IEEE Computer Vision and Pattern Recognition (CVPR), 2009.

[25] Glorot, X., Bengio, Y., Understanding the difficulty of training deep feedforward neural networks, Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS), 2010.

[26] Nascimento, J. C., Abrantes, A. J., and Marques, J. S., An algorithm for centroid-based tracking of moving objects, 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258), Phoenix, AZ, USA, 1999, pp.3305-3308, doi: 10.1109/ICASSP.1999.757548.