Yamen Aly 19100944
AbdelRahman Khalil 20101646
Mohamed Maghraby 19102722
Adham Mohamed 19100821

Real Time Face Mask Detection

1 Introduction

A simple weapon has surfaced in the ongoing fight against COVID-19: the face mask. Masks provide protection by obstructing respiratory droplets, safeguarding the wearer and anyone in their vicinity. For those who are particularly susceptible, this two-way protection is extremely important as it slows the virus's transmission. But it can be difficult to guarantee mask conformity. Real-time face mask identification methods can help with that. Imagine security cameras that can recognize people without masks in an instant.

2 Related Work

Researchers have been investigating different models for real time face mask detection. While methods like YOLOv3 and SSD work well for face identification, they have trouble telling masked faces apart from unmasked ones. To solve this, the authors of this work suggest a lightweight deep learning architecture made especially for mask recognition, which achieves accuracy without compromising speed in real-time. By incorporating a mask classification branch and building on pre-existing face detection models such as RetinaNet, this method produces reliable and efficient mask detection across a range of contexts. To further demonstrate the expanding potential of this technique in practical applications, more research is being done on handling occlusions, detecting different types of masks, and enhancing robustness to changes in lighting.

3 Methodology

3.1 Data

The data used was from the 12K image Face Mask dataset by ASHISH JANGRA on Kaggle. The dataset consists of 3 folders train, test and validation all combined of about 12,000 pictures balanced to be 6000 masked and 6000 unmasked.

Figure 1: A sample image from the dataset of a person wearing a mask.

3.2 Features

- Resizing + Rescaling + Sharpening + Median Blur
- Resizing + Rescaling + Sharpening + Median Blur + Brightness/Contrast + Colors
- Resizing + Rescaling + Sharpening + Median Blur + Brightness/Contrast + Colors + Inverse Transform + Histogram Equalization

3.3 Code

The offered Python code creates a strong machine learning model for picture categorization by integrating multiple potent libraries, such as TensorFlow, NumPy, Pandas, and OpenCV. The code also makes use of cv2.dnn from OpenCV and the MobileNetV2 model from TensorFlow's Keras applications. Using Pandas for effective data processing, TensorFlow for deep learning features, and NumPy for numerical calculations, the code illustrates a thorough method for creating an

Yamen Aly 19100944
AbdelRahman Khalil 20101646
Mohamed Maghraby 19102722
Adham Mohamed 19100821

intricate image categorization system. Matplotlib is also used to visualize loss and accuracy graphs. To guarantee a well-organized and effective machine learning pipeline, the code also includes features like train-test splitting and data preprocessing. All in all, this code demonstrates a careful blending of many libraries, emphasizing the adaptability and joint potential of these instruments in the creation of sophisticated machine learning solutions.

3.4 Classification Model

We used four models with varying data enhancements and head model complexity to fine-tune mask detection accuracy. As a baseline, the first model was trained using unprocessed, raw data. More advanced features were added to later models: color manipulation, brightness/contrast modifications, median blur, sharpening, rescaling, and resizing. To improve the data representation, our final model also included an inverse transform and histogram equalization. All models leveraged transfer learning, with our proprietary head model coming in second to MobileNetV2 as the basis. A dropout layer for regularization, two dense layers (128 and 2 neurons), flattening, an average pooling layer, a ReLU activation function for the first dense layer, and a softmax activation for binary classification (masked vs. unmasked) were all included in this model. With this progressive approach, we were able to assess the effects of each improvement and maximize the network's capacity to identify minute characteristics that are crucial for precise mask identification.

4 Analysis and Results

	Test Accuracy	Test Loss
0 Enhancements	96.87%	9.97%
4 Enhancements	96.91%	9.44%
6 Enhancements	96.94%	9.14%
10 Enhancements	96.60%	9.57%

Conclusion

Our tests show a distinct trend: the model's test accuracy continuously rises as we add more picture enhancements to our mask detection dataset. Moving from no improvements to four, six, and eight enhancements, we witness a continuous gain in accuracy from 96.87% to 96.94%, supported by a reduction in test loss. This shows that the model learns more robust features and, eventually, produces better predictions when the data is enhanced with these carefully selected changes. The accuracy slightly decreases with the inclusion of two more enhancements, for a total of ten enhancements. This could be a sign of diminishing returns or the need to fine-tune the particular enhancements employed.

All things considered, these findings provide compelling evidence for the efficacy of picture improvements in raising mask detection accuracy. Potentially, more investigation into other combinations and optimizations could raise the performance even further.

References

“A Deep Learning-based Approach for Real-time Facemask Detection”

https://arxiv.org/abs/2110.08732

Mask and social distancing detection using VGG19, NAGESH SINGH CHAUHAN :

https://www.kaggle.com/code/nageshsingh/mask-and-social-distancing-detection-using-vgg19