Real- Time Face Mask Detection

Raja Solanki and James Hopham

July 16, 2024

Introduction

The COVID-19 pandemic has underscored the critical need for effective public health measures to mitigate the spread of the virus. One such measure is the widespread use of face masks, which has been proven to reduce transmission rates significantly. However, ensuring compliance in public spaces poses a significant challenge. This project aims to address this problem by developing an automated face mask detection system using machine learning techniques.

The primary objective is to create a robust and accurate model that can identify whether individuals in an image or video are wearing masks correctly. Leveraging the power of deep learning, specifically CNN, this project focuses on detecting various types of face masks, including surgical masks, cloth masks, and N95 respirators. The model will not only recognize the presence of a mask but also differentiate between proper and improper mask usage.

By deploying this system in real-time applications, such as surveillance systems in public transportation, airports, and retail environments, we can significantly enhance public safety measures. The ultimate goal is to provide a scalable and efficient solution that assists authorities in monitoring and promoting adherence to mask-wearing guidelines, thereby contributing to the collective effort in controlling the pandemic.

Architecture

The architecture of the face mask detection solution is designed to efficiently process and analyze images or video streams to identify individuals wearing face masks. The primary components of the architecture include data collection, preprocessing, model training, and deployment. Below is a detailed description of each component:

I. **Data Collection**:

- **Dataset**: The dataset comprises images with annotations indicating the presence and type of face masks (e.g., surgical masks, cloth masks, no mask). This dataset is essential for training the machine learning model.
- **Data Sources**: Images are sourced from public datasets, surveillance footage, and manually annotated data to ensure a diverse and representative dataset.

II. **Data Preprocessing**:

- **Image Augmentation**: Techniques such as rotation, scaling, flipping, and brightness adjustment are applied to increase the diversity of the training data and improve model generalization.
- **Bounding Box Normalization**: The annotated bounding boxes are normalized to a consistent format suitable for input into the neural network.

III. **Model Architecture**:

- **Input Layer**:
  - Takes in images of size 124x124 with 3 color channels (RGB).
- **Convolutional Layers**:
  - **First Layer**: 32 filters, (3, 3) kernel, ReLU activation, "same" padding.
  - **Second Layer**: 64 filters, (3, 3) kernel, ReLU activation.
  - **Third Layer**: 128 filters, (3, 3) kernel, ReLU activation.
- **Pooling Layer**:
  - **MaxPooling2D** with (2, 2) pool size to reduce spatial dimensions and computational cost.
- **Dropout Layers**:
  - **First Dropout**: 0.25 rate.
  - **Second Dropout**: 0.5 rate.
  - **Third Dropout**: 0.5 rate to prevent overfitting.
- **Flattening Layer**:
  - **Flatten**: Converts 2D feature matrices into a 1D vector.
- **Dense Layers**:
  - **First Dense Layer**: 50 units, ReLU activation.

- **Output Layer**: 1 unit, Sigmoid activation for binary classification.

IV **Compilation:**

- **Loss Function**: Binary Cross-Entropy for binary classification.
- **Optimizer**: Adam for adaptive learning rate.
- **Metrics**: Accuracy for evaluating model performance during training and testing.

V. **Model Training**:

- **Loss Function**: The model is trained using a multi-task loss that combines classification loss (e.g., cross-entropy loss for mask detection) and localization loss (e.g., smooth L1 loss for bounding box regression).
- **Optimization**: Stochastic gradient descent (SGD) or Adam optimizer is used to minimize the loss function and update the model weights.

VI. **Model Deployment**:

- **Inference Engine**: The trained model is integrated into an inference engine that processes real-time video streams or static images. OpenCV and TensorFlow Serving are used for efficient deployment.
- **Edge Deployment**: For real-time applications, the model can be deployed on edge devices with GPUs or TPUs to reduce latency and ensure rapid detection.
- **Alert System**: An alert system is implemented to notify authorities or trigger actions when individuals without masks or improperly worn masks are detected.

Results

Our model demonstrated strong performance across our classification scores; notably performing the best in terms of correctly identifying images where subjects truly were wearing a mask with a recall of $\approx .932$. Although still acceptable, the model performed relatively weaker in correctly identifying true positives among those who were labeled to be wearing a mask. However, it is preferable to prioritize recall in this scenario because we want to prioritize identifying if people are wearing masks rather than focusing on improving the precision score at the cost of consequently labeling too many images as false negatives. Overall, the F1-Score of $\approx .881$ demonstrates strong performance of our image classification model.

Figure 1: Classification Scores

Precision	.8347826
Recall	.9320388
F1-Score	.8807339

Figure 2: Confusion Matrix

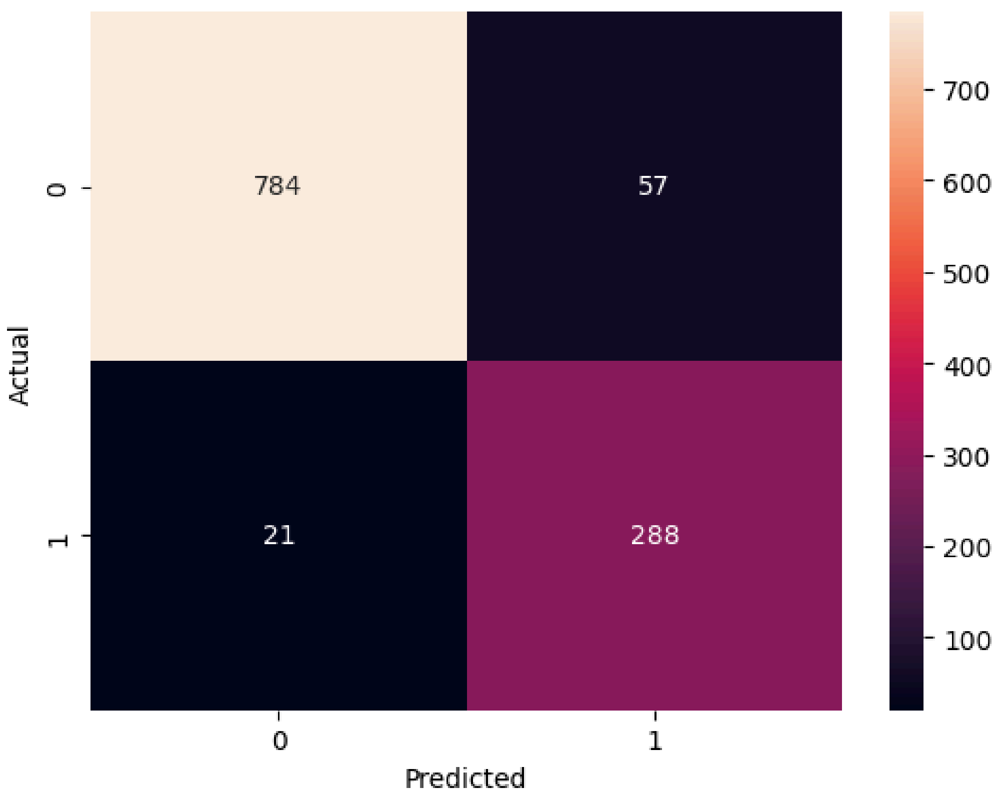


Figure 3: ROC Curve

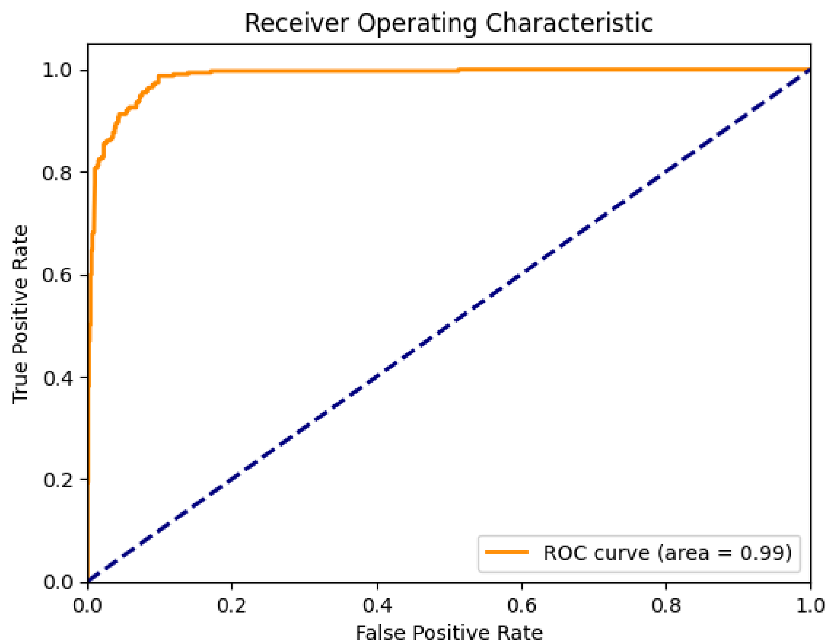
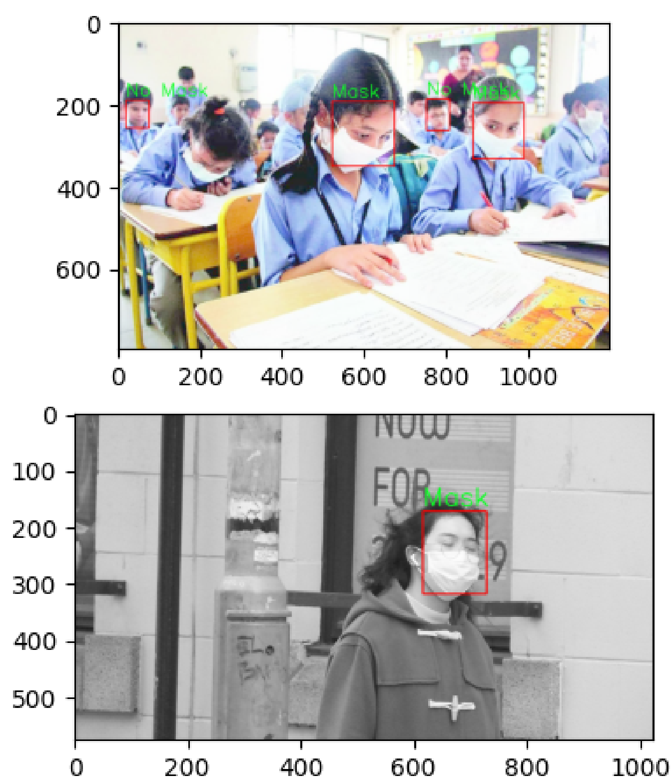


Figure 4: Mask Detection Example Output



Conclusion

Our model successfully demonstrated strong performance in classifying a diverse dataset of images for detecting features to indicate whether the subject in the image was wearing a mask.