Face Mask Detection using TensorFlow in Python

Build an intelligent computer vision system to detect face masks in real-time using deep learning and TensorFlow

Two-Phase Detection System

Our face mask detector follows a structured approach that separates training from deployment, ensuring robust and reliable detection capabilities.

Training Phase

Load the face mask dataset, train a CNN model using Keras and TensorFlow, and serialize the trained detector for deployment

Deployment Phase

Load the trained mask detector, perform face detection on images or video streams, and classify each face as with_mask or without_mask

Dataset Preparation

Initial Dataset Balance

Our starting dataset contains a well-balanced distribution of masked and unmasked face images, providing a solid foundation for training.

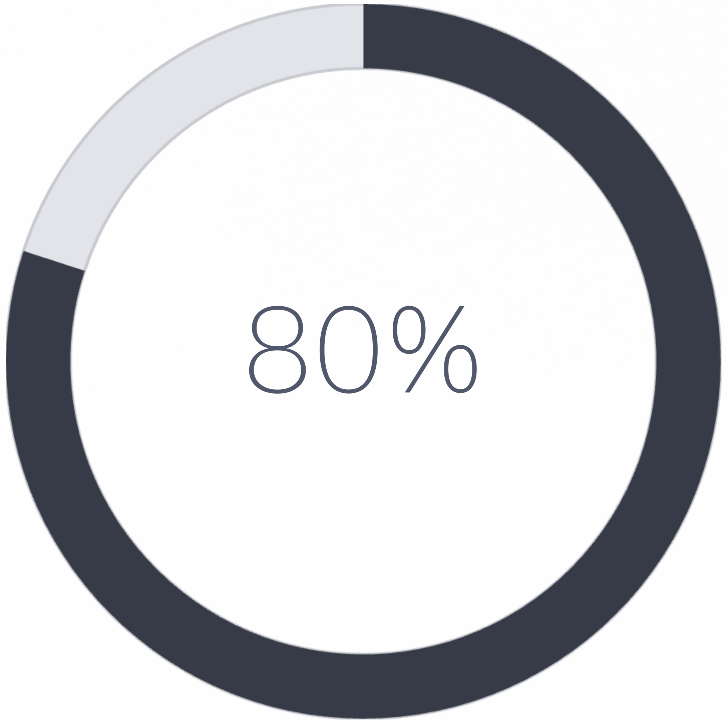
- ***690 images** with face masks (labeled 'yes')*
- ***686 images** without masks (labeled 'no')*
- *Nearly equal distribution ensures unbiased learning*

Data Augmentation

We expand our dataset through rotation and flipping techniques, increasing robustness and generalization capabilities.

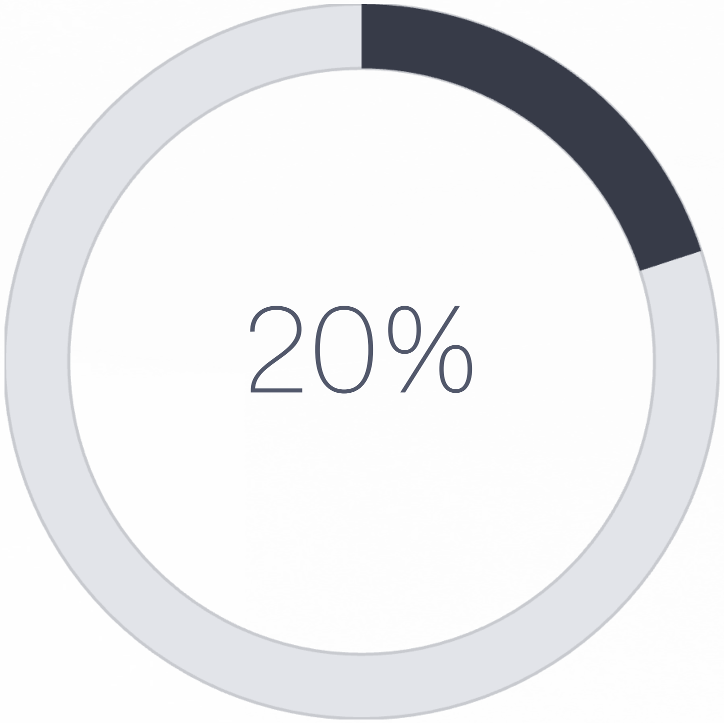
- ***2,751 total images** after augmentation*
- *1,380 images with masks (50.16%)*
- *1,371 images without masks (49.84%)*

Train-Test Split Strategy



Training Set

2,200 images used to train the CNN model and learn mask detection patterns



Test Set

551 images reserved for evaluating model performance on unseen data

This 80-20 split ensures our model has sufficient training data while maintaining a robust test set for validation.

Building the CNN Architecture

Our Sequential CNN model leverages multiple convolutional and pooling layers to extract hierarchical features from face images.

01

Convolutional Layers

Two Conv2D layers with 100 filters and ReLU activation extract spatial features from 150×150 input images

02

Pooling Layers

MaxPooling2D layers reduce spatial dimensions while preserving important features

03

Flatten & Dropout

Flatten layer converts 2D features to 1D, while 50% dropout prevents overfitting

04

Dense Layers

Two Dense layers with softmax activation output probability scores for mask classification

*The model uses **Adam optimizer** with **binary crossentropy loss** for efficient two-class classification. Consider using MobileNetV2 for improved accuracy on mobile devices.*

Training Results

30

Training Epochs

Model iterations to optimize mask detection accuracy

98.86%

Training Accuracy

Final accuracy achieved on the training dataset

96.19%

Test Accuracy

Validation accuracy on unseen test images

The close alignment between training and test accuracy indicates excellent model generalization without overfitting. Additional epochs could improve performance, but careful monitoring prevents degradation.

Face Detection Integration

Haar Cascade Classifier

We integrate OpenCV's Haar Feature-based Cascade Classifier for robust frontal face detection in real-time video streams.

The classifier is pre-trained on thousands of positive and negative face images, enabling fast and accurate facial feature detection.

- *Uses **haarcascade_frontalface_default.xml** file*
- *Detects faces across various angles and lighting conditions*
- *Provides bounding box coordinates for each detected face*

Color-Coded Results

Visual feedback system for instant recognition:

- *GREEN rectangle: Mask detected (label: 1)*
- *RED rectangle: No mask (label: 0)*

Real-Time Detection Pipeline

1 Video Capture

Initialize webcam stream using OpenCV's VideoCapture function

2 Face Detection

Apply Haar Cascade Classifier to locate faces in each frame

3 Preprocessing

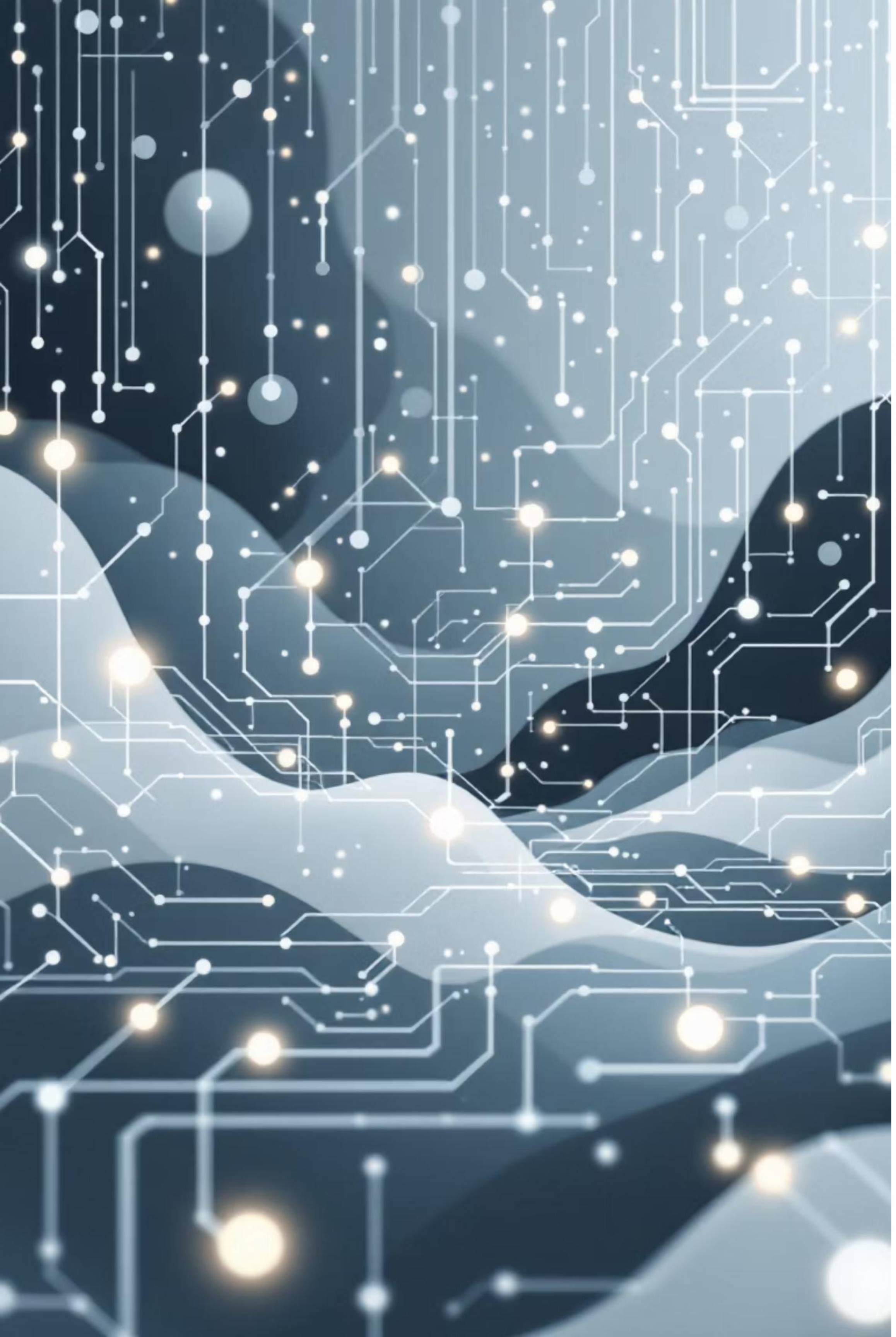
Extract face ROI, convert to RGB, resize to 224×224, and normalize pixel values

4 Batch Prediction

Run CNN model on all detected faces simultaneously for efficient inference

5 Visualization

Draw color-coded bounding boxes with confidence scores on detected faces



Deployment & Next Steps

Ready for Production

Your face mask detector is now trained and ready to deploy across multiple platforms.

- *Works with PC webcams and smartphone cameras*
- *Processes real-time video streams efficiently*
- *Provides instant visual feedback with confidence scores*
- *Press 'q' key to exit the detection loop*

Enhancement Opportunities

- *Integrate **MobileNetV2** architecture for better mobile performance*
- *Add multi-angle face detection support*
- *Implement distance monitoring for social distancing*
- *Deploy on edge devices for offline operation*
- *Log detection statistics for compliance reporting*