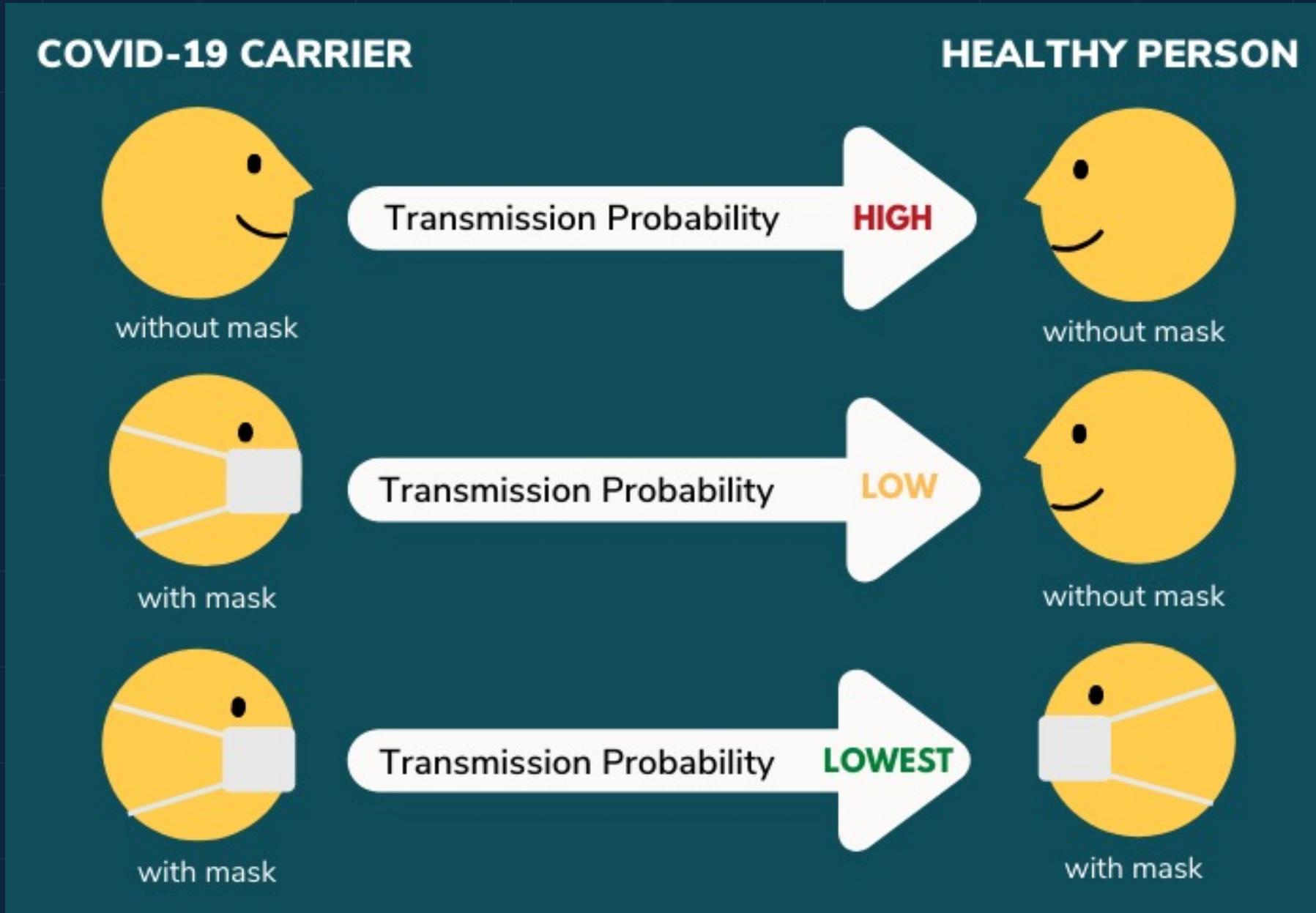


IF YOU MUST MASK

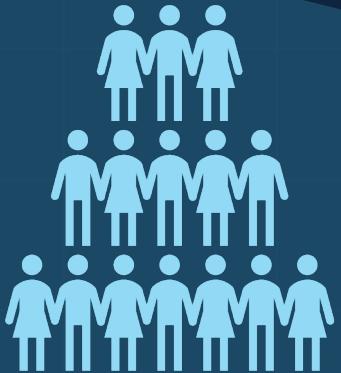
Face Mask Detection with
Deep Learning and Computer Vision

Presented by: Crystal Huang
Metis Bootcamp Project VI

INTRODUCTION



Problem



**Increased crowd
volume as life
returns to normal**



**Mandated mask
at **indoor public**
places (i.e. airports,
hospital, etc.)**



**Difficult and
inefficient to inspect
the large crowd with
labor screening**

GOAL

Build a **face mask detection system** using
deep learning algorithm
and computer vision

Tools

Cloud Computing

- Google Colab
- Python
- Numpy
- Matplotlib
- Seaborn



DL Model

Keras/Tensorflow



Face detection

OpenCV
Haar Cascade



Production

Streamlit
Heroku

10cm

DATA

Face Mask Detection dataset from Kaggle
11,800 Images

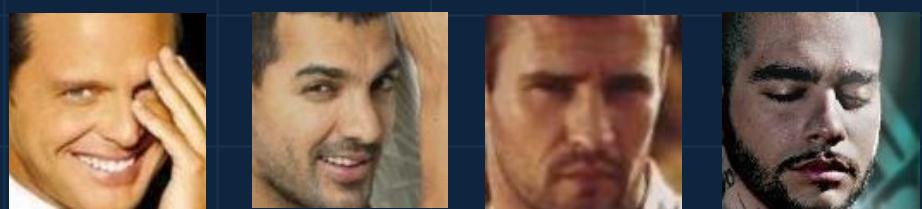


DATA

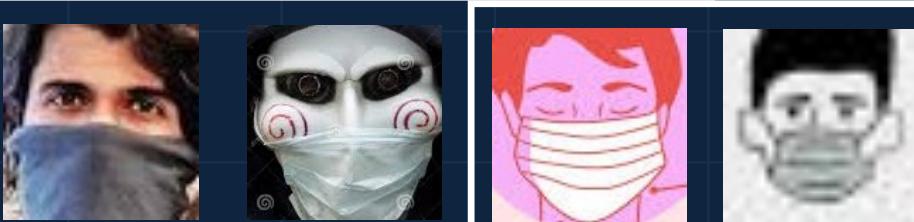
Face Mask Detection dataset from Kaggle

11,800 Images

Without Mask

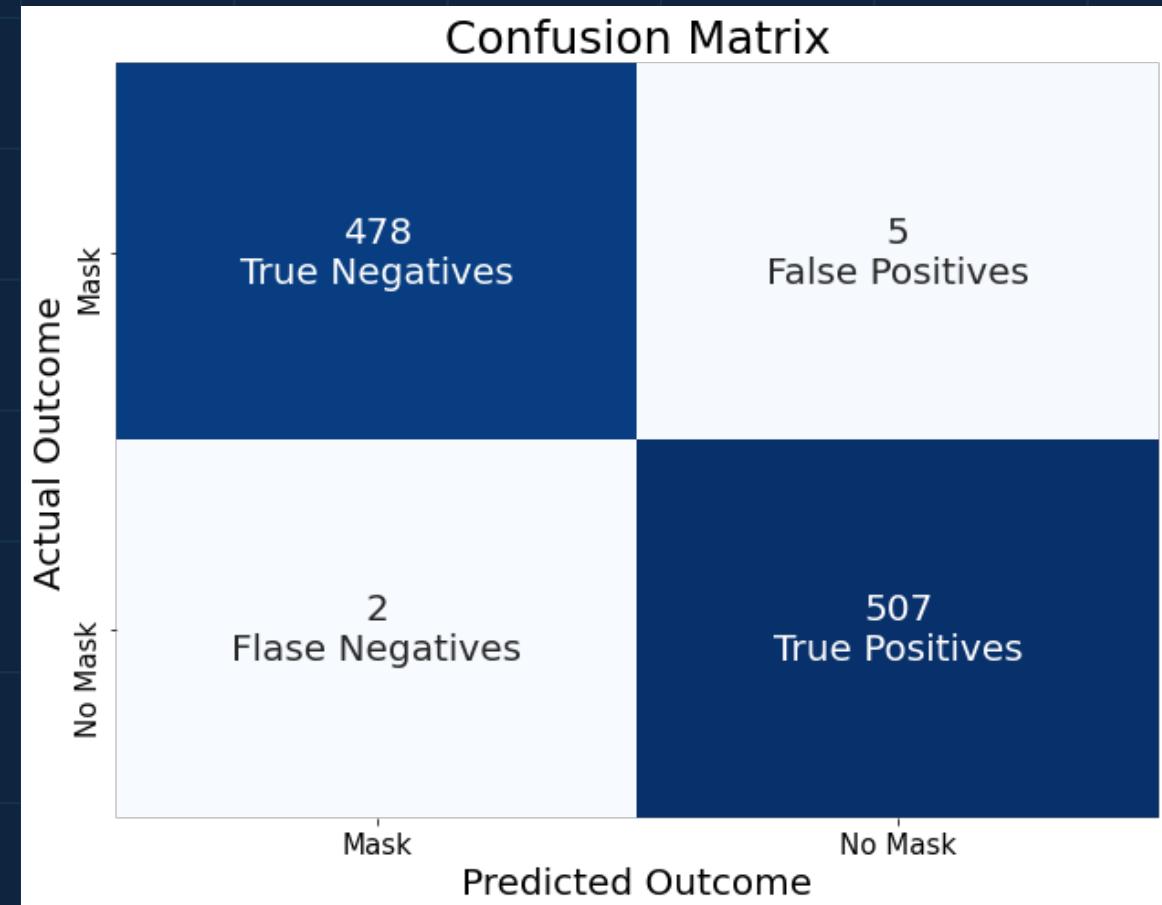
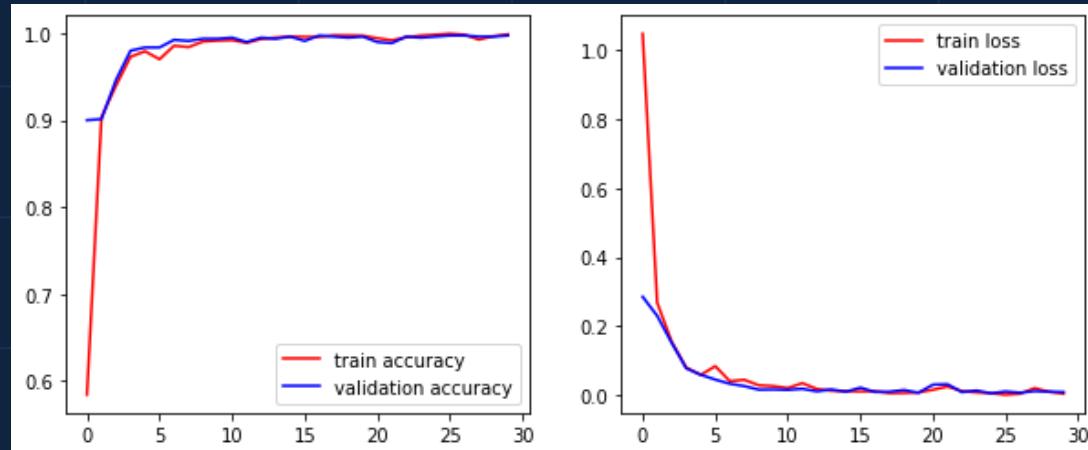


With Mask



Binary Model Performance

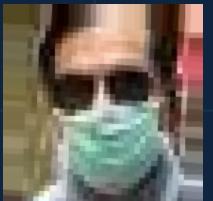
Scores	Binary CNN
Accuracy	0.99
Precision	0.99
Recall	0.99
F1	0.99



Binary Model Examples

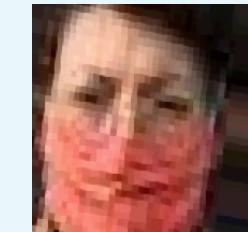
“Good Citizens”

True Negatives



“False Alarm”

False Positives



“Free Pass”
False Negatives



“Gotcha!”
True Positive

Binary Model

1. With Mask
2. Without Mask



3-class Model

1. With Mask
2. Without Mask
3. Incorrect Mask

DATA For 3-Class Model

Previous Data:
11,800 Images



MaskedFace-Net
dataset

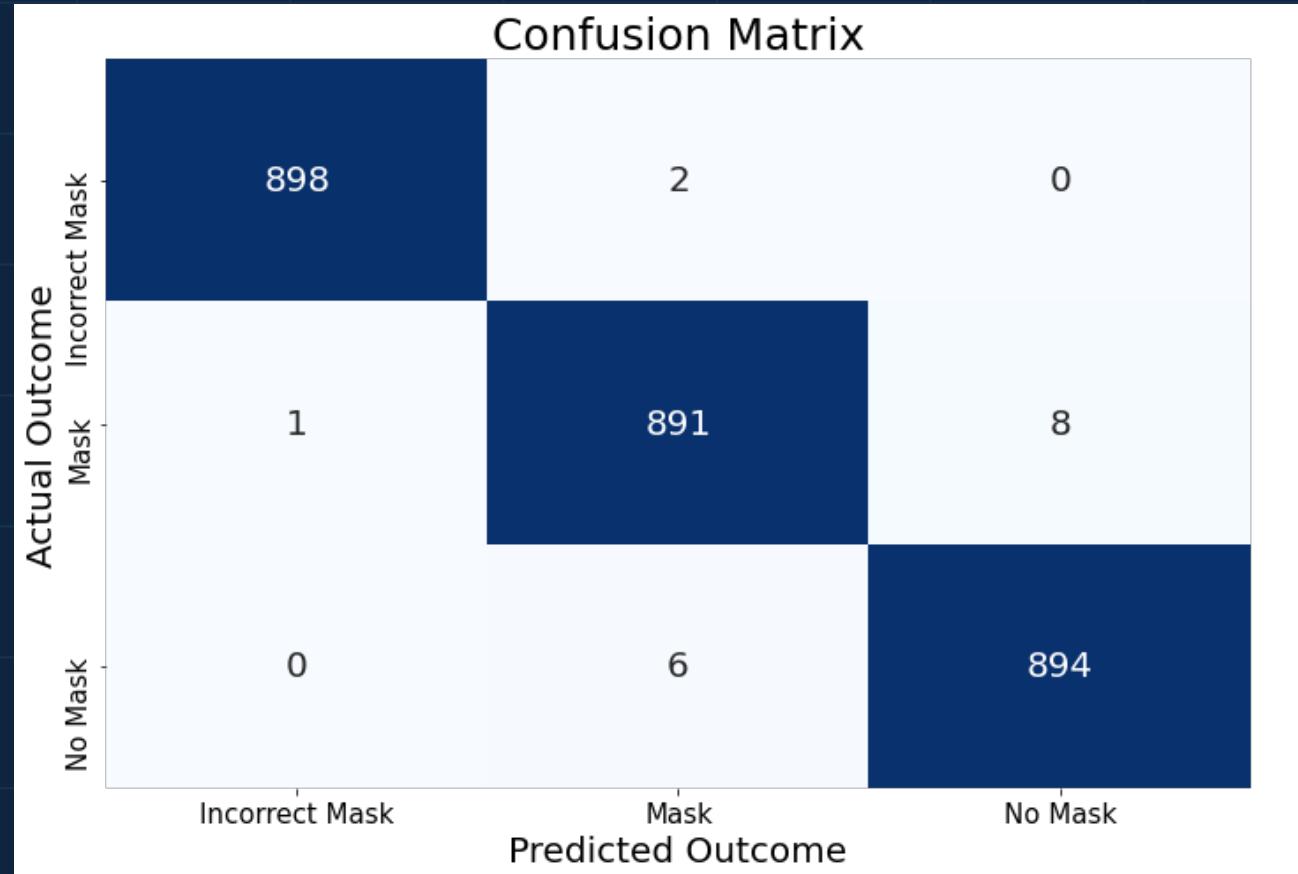
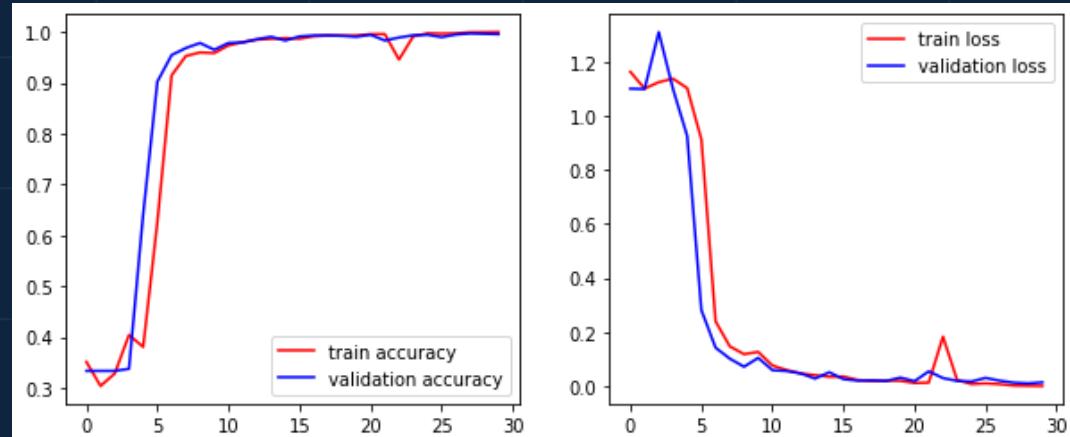
CelebFaces
dataset



New Data:
24,000 Images
(Evenly distribution
among classes)

3-class Model Performance

Scores	Binary CNN	3-Class CNN
Accuracy	0.99	0.99
Precision	0.99	0.99
Recall	0.99	0.99
F1	0.99	0.99



3-class Model Examples

Incorrect Mask

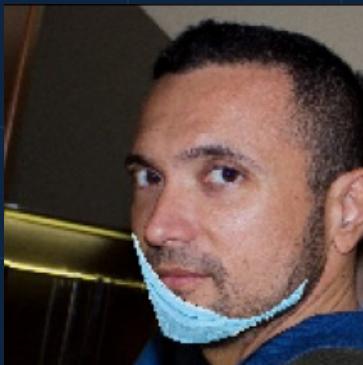
Dataset test set



"Incorrect Mask"



"Incorrect Mask"



"Incorrect Mask"

New images



"With Mask"



"With Mask"



"With Mask"

Insights

- Binary Model – Great performance on both **dataset** and **new real-life images**
- 3-Class Model – Great performance within the dataset, but **not very good on new real-life images** for “incorrect mask”
- Binary Model is more suitable for **real-life application**

Application in Real-Life



Dataset Image

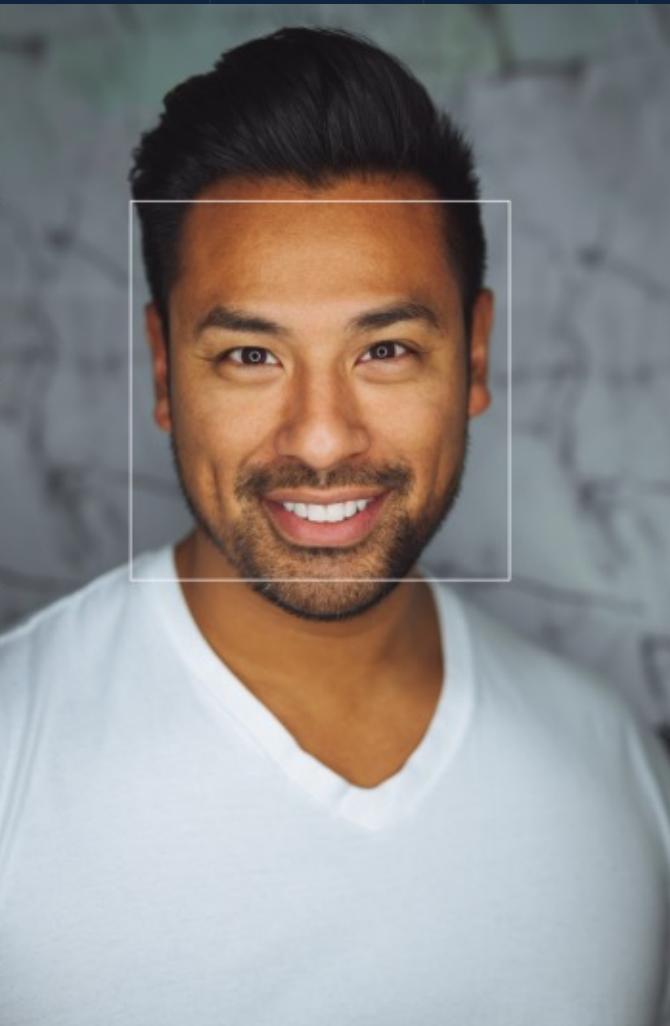


Real-life Image

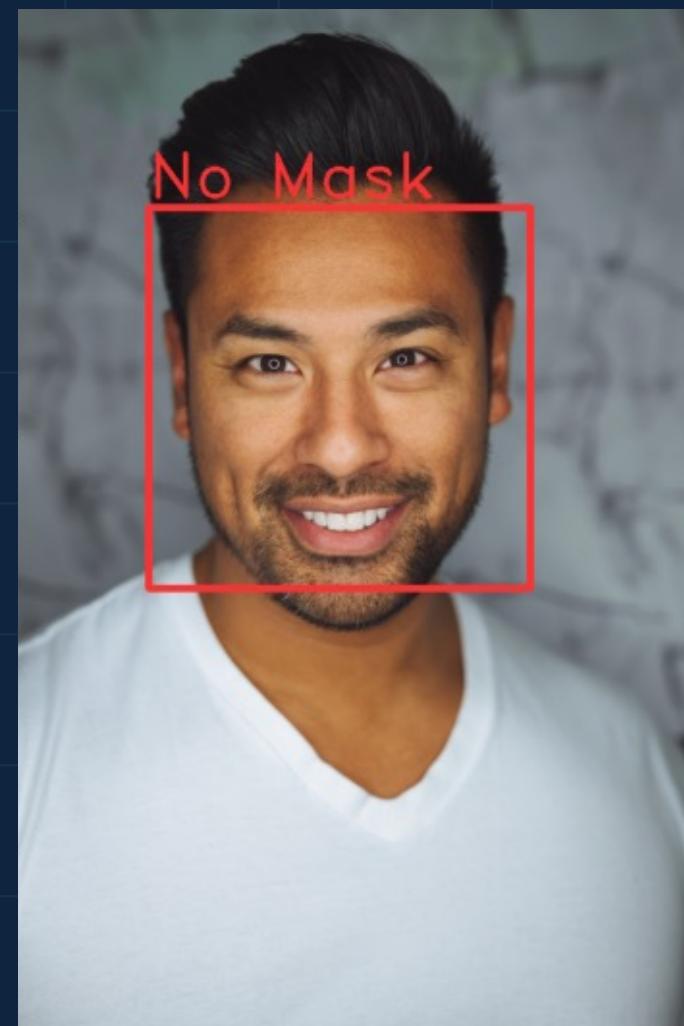
10cm

WORKFLOW For Face mask detection

1 Detect Face



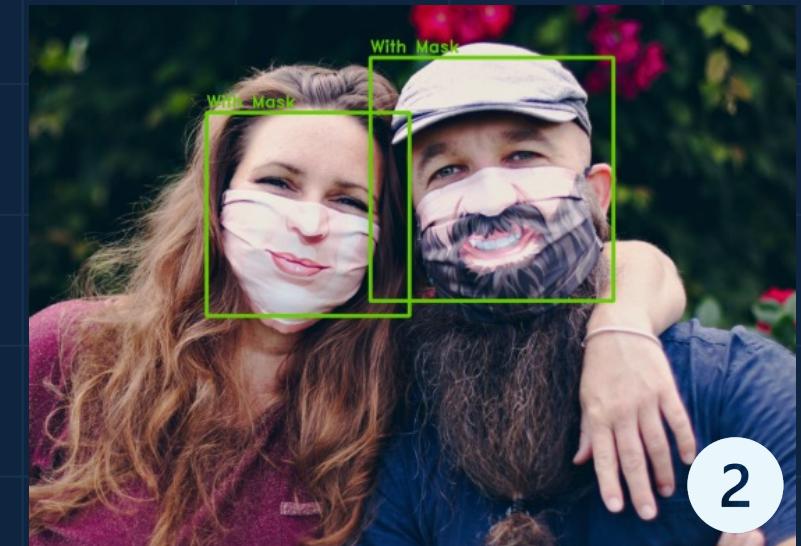
2 Apply Mask Detector



10cm

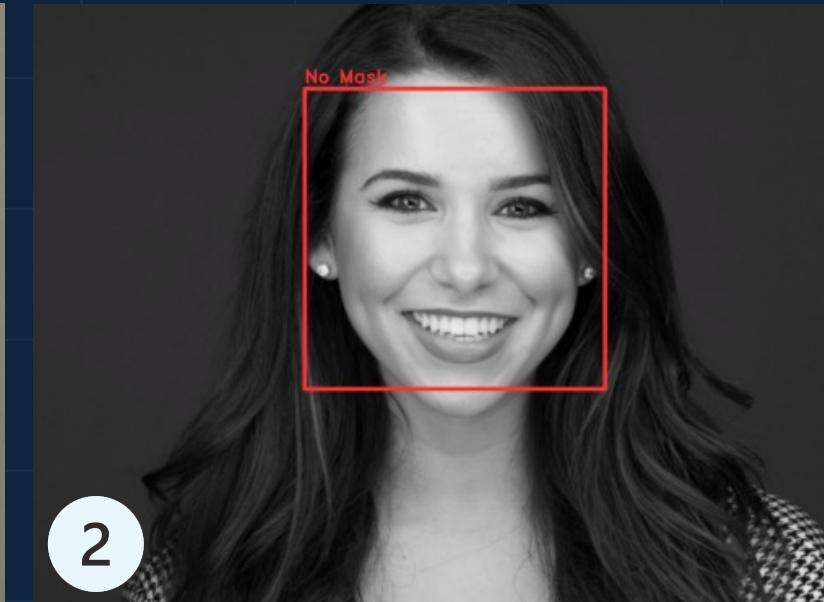
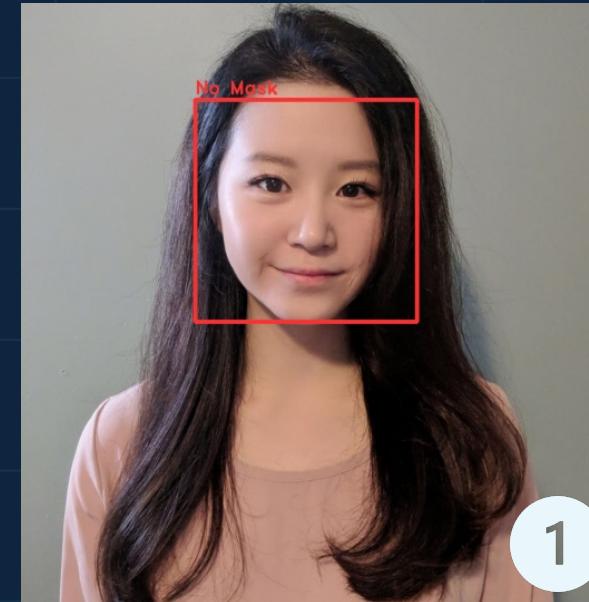
RESULTS

With Mask



RESULTS

No Mask



10cm

RESULTS

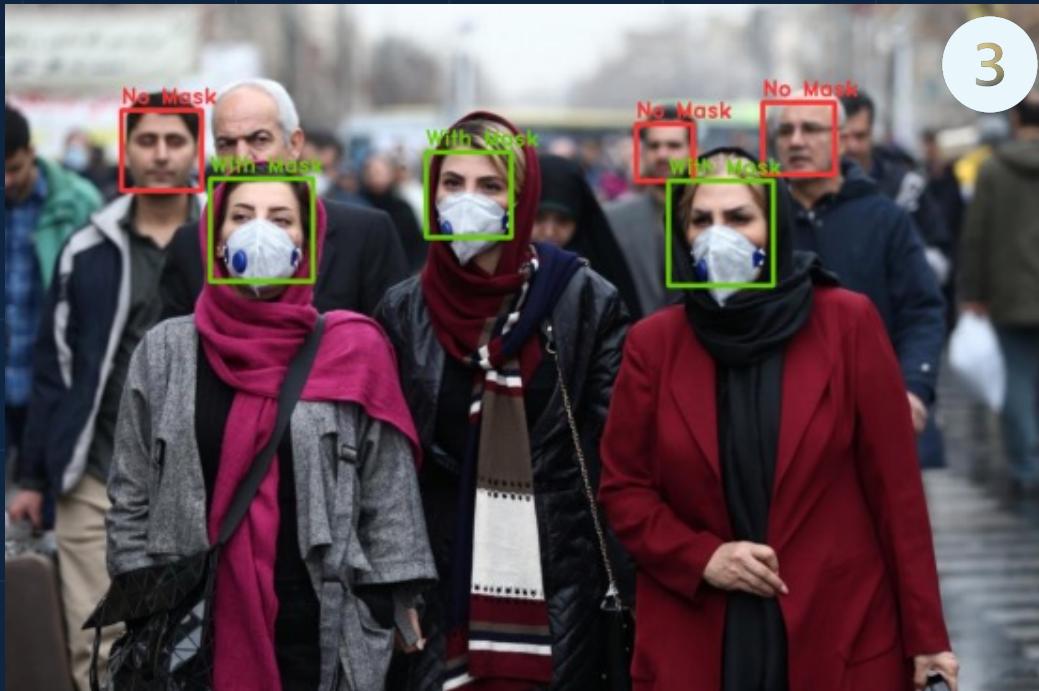
Mixed



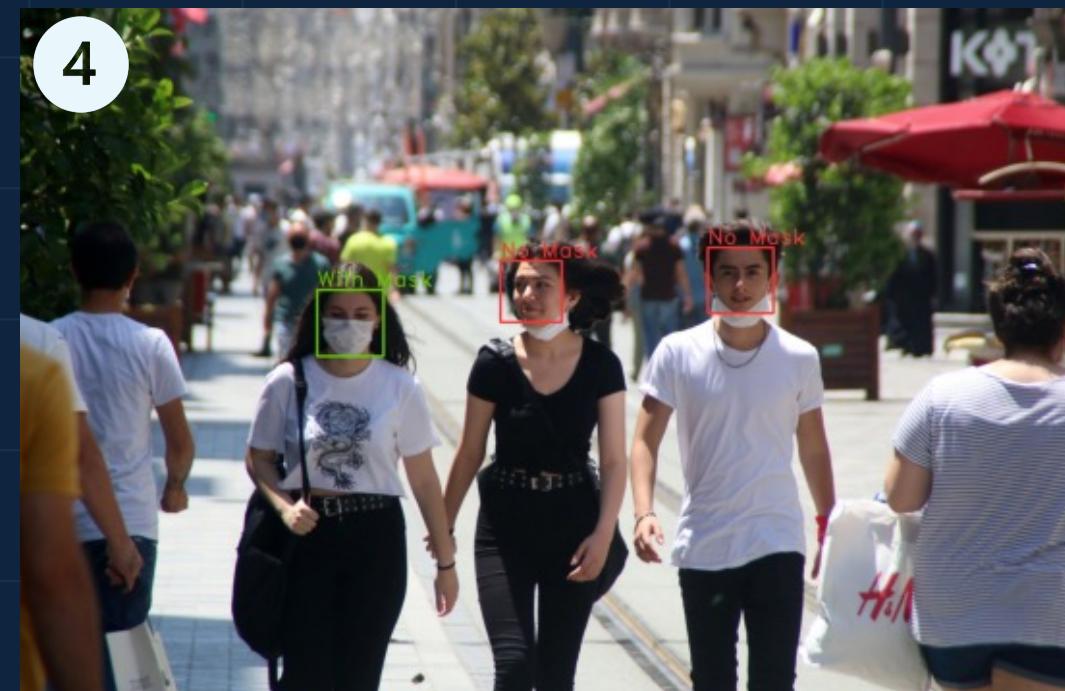
1



2



3



4

10cm

APPLICATION

The screenshot shows a web browser window with the URL `localhost:8501` in the address bar. The page content is as follows:

- Mask detection on:** A dropdown menu showing "Home".
- Mask Patrol**: A section featuring three cartoon faces wearing masks (green, yellow, red) and the text "If you must **mask**, I shall answer...".
- A Face Mask Detection System**: A large heading.
- Built with OpenCV and Keras/TensorFlow leveraging Deep Learning and Computer Vision Concepts to detect face mask in still images as well as in real-time webcam streaming.**
- You can choose the options from the left.**
- Upcoming Features:**
 - Webcam Mask Detection
 - Detecting Incorrect Mask

FUTURE WORK

- ① Process 3-Class model images
Crop to face area with Haar Cascade Classifier
- ② Build a robust 3-Class model
That can be used outside of the dataset
- ③ Feature Extraction, Data Augmentation
Generalize images



THANK
YOU

Mask any
Question?



CrystalHuang-ds

<https://mask-patrol.herokuapp.com/>

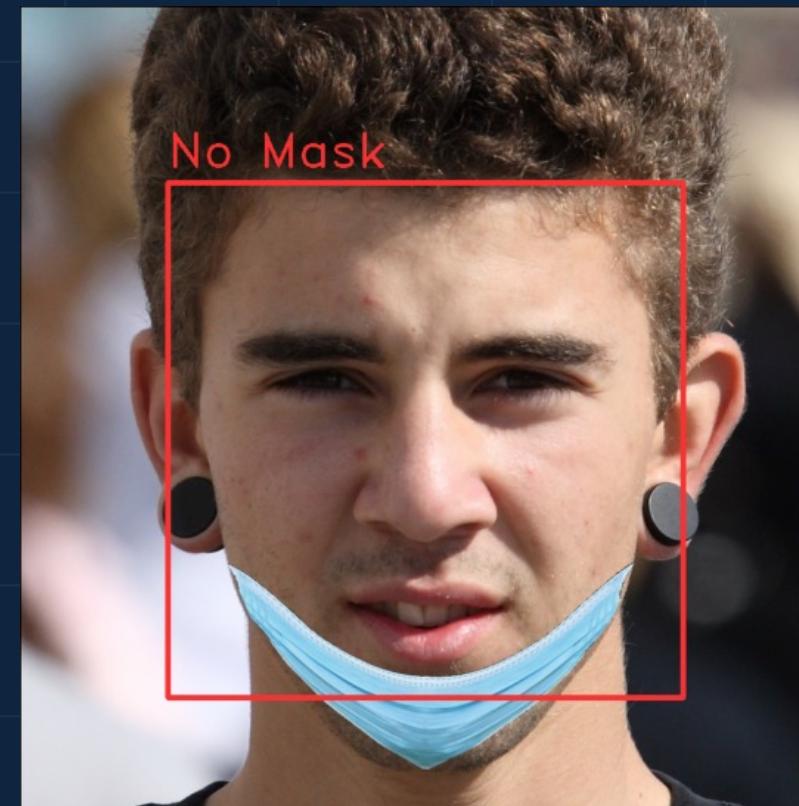
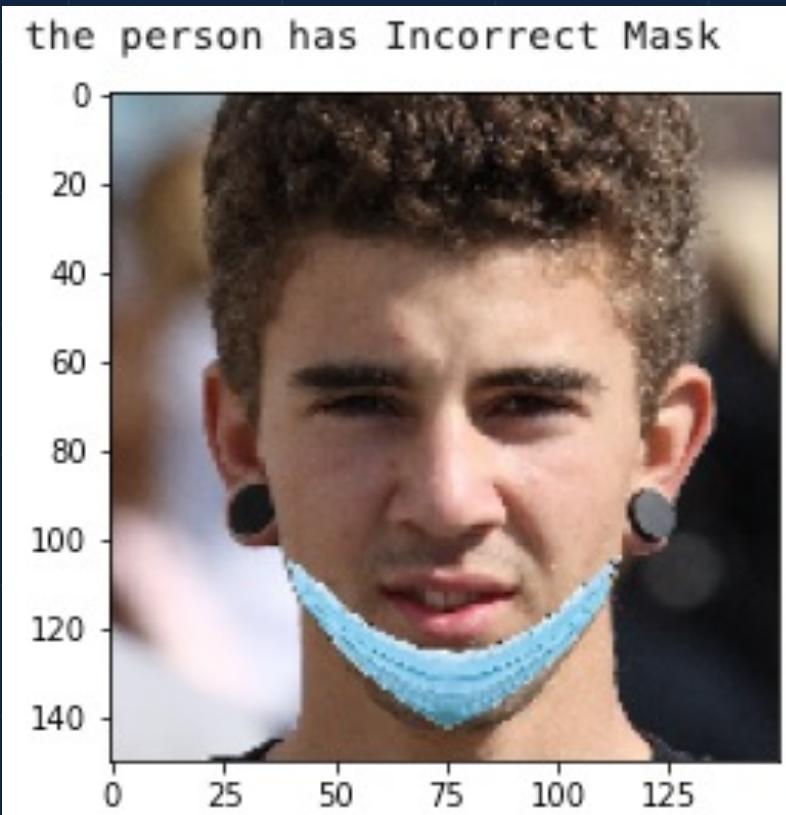


crystal-ctrl

References

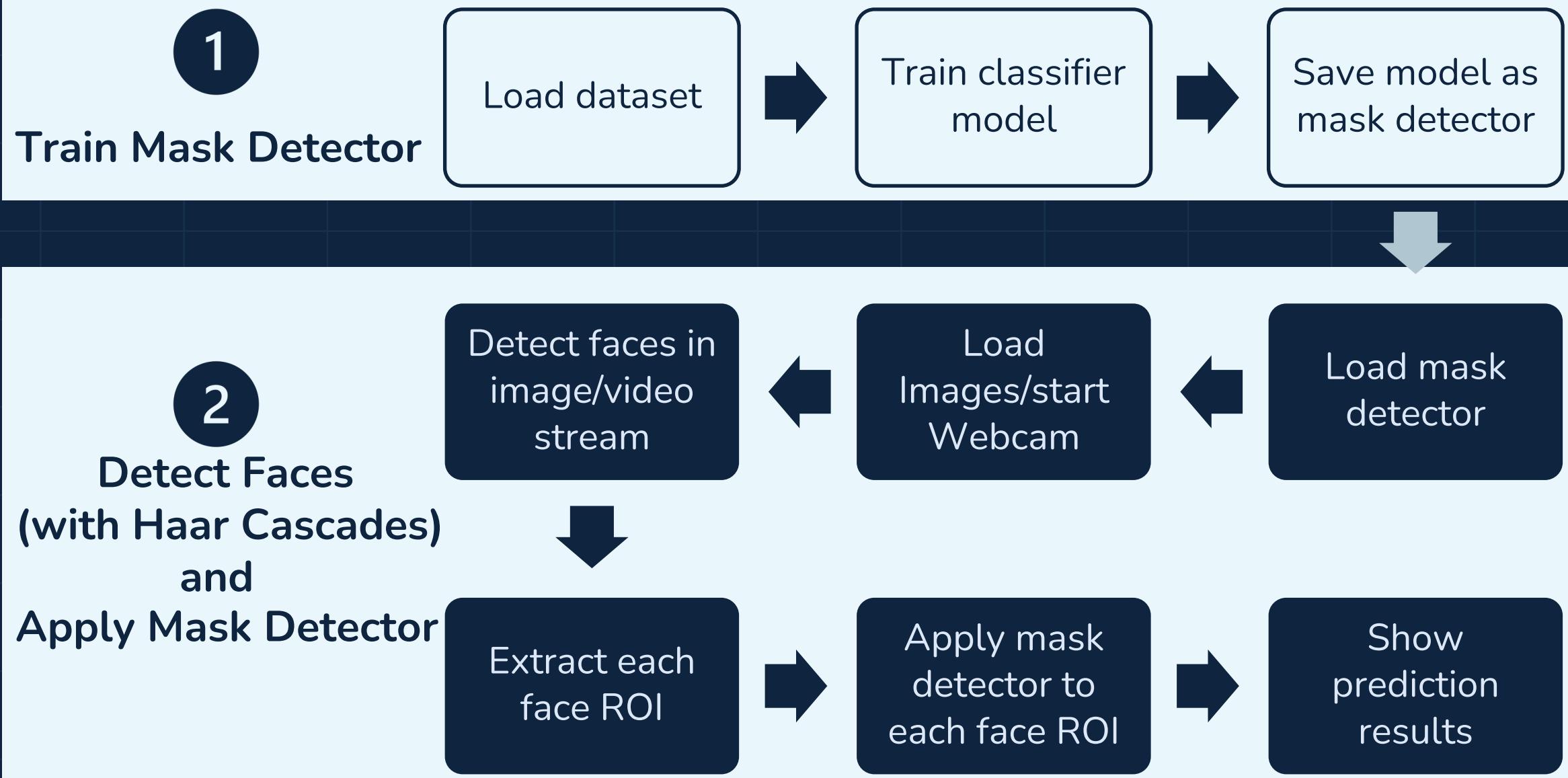
1. Mittal, A. 2020. “Haar Cascades, Explained.” *Analytics Vidhya. Medium.* <https://medium.com/analytics-vidhya/haar-cascades-explained-38210e57970d>
2. Menon, A. 2019. “Face Detection in 2 Minutes using OpenCV & Python.” *Towards Data Science. Medium.* <https://towardsdatascience.com/face-detection-in-2-minutes-using-opencv-python-90f89d7c0f81>
3. Rosebrock, A. 2018. “OpenCV Face Recognition”. *Pyimagesearch.* <https://www.pyimagesearch.com/2018/09/24/opencv-face-recognition/>

Appendix 1



10cm

Appendix 2 - WORKFLOW



Appendix 3-1

Load the model

```
model = load_model('binary_model.h5')

# load the test set for prediction
test_dir = 'Face Mask Dataset/Test'
image_gen = ImageDataGenerator(rescale=1./255)
test_generator = image_gen.flow_from_directory(test_dir,
                                              target_size=(150,150),
                                              batch_size=80,
                                              seed=42,
                                              shuffle=False,
                                              class_mode='binary')

prediction = model.predict_classes(test_generator)
y_true = test_generator.labels
```



Found 992 images belonging to 2 classes.

Appendix 3-2

Map Files to Prediction Results ¶

```
[17]: filenames = [x for x in test_generator.filenames]
label = [x for x in y_true]
predict = [x[0] for x in prediction]

import pandas as pd
df = pd.DataFrame(list(zip(filenames, label, predict)), columns=['Filenames', 'Label', 'Predict'])
df.head()
```

```
[17]:      Filenames  Label  Predict
 0  WithMask/1163.png    0      1
 1  WithMask/1174.png    0      0
 2  WithMask/1175.png    0      0
 3  WithMask/1203.png    0      0
 4  WithMask/1361.png    0      0
```

Appendix 3-3

```
[18]: # function to add "Outcome" column
def predict_outcome(row):
    if (row['Label'] == 0) & (row['Predict'] == 0):
        return "TN"
    elif (row['Label'] == 0) & (row['Predict'] == 1):
        return "FP"
    elif (row['Label'] == 1) & (row['Predict'] == 1):
        return "TP"
    elif (row['Label'] == 1) & (row['Predict'] == 0):
        return "FN"

# applying function
df['Outcome'] = df.apply (lambda row: predict_outcome(row), axis=1)

[56]: # Sanity check
df.sample(5)
```

	Filenames	Label	Predict	Outcome
73	WithMask/86.png	0	0	TN
748	WithoutMask/3945.png	1	1	TP
460	WithMask/Augmented_854_5503218.png	0	0	TN
826	WithoutMask/4597.png	1	1	TP
675	WithoutMask/3012.png	1	1	TP

```
[58]: df.Outcome.value_counts()
```

```
[58]: TP      507
TN      478
FP       5
FN       2
Name: Outcome, dtype: int64
```

Appendix 3-4

False Positive

Predict = No Mask

Actual = With Mask

```
[20]: df[df["Outcome"]=="FP"]
```

		Filenames	Label	Predict	Outcome
	0	WithMask/1163.png	0	1	FP
	48	WithMask/45.png	0	1	FP
	379	WithMask/Augmented_689_9677847.png	0	1	FP
	448	WithMask/Augmented_824_6606851.png	0	1	FP
	482	WithMask/Augmented_99_5504654.png	0	1	FP