WK7 Report

Pre-trained Face Detector

Yunting Chiu
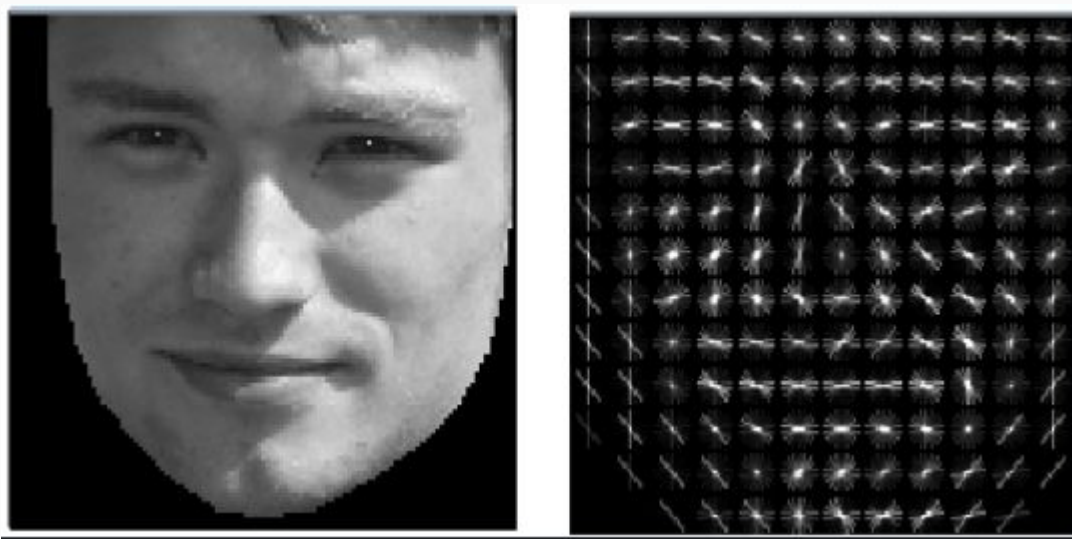
OpenFace

OpenFace 2.2.0: a facial behavior analysis toolkit

Facial Landmark Detection

Facial Feature Extraction

Source:
https://github.com/TadasBaltrusaitis/OpenFace

WHY MTCNN

- Faces can be captured with > 99% accuracy using MTCNN model.

- Then, unify the image size (e.g. 224, 224, 3)

```python
[8]    1 from PIL import Image
       2 import numpy as np
       3
       4 def extract_face(filename, required_size=(224, 224)):
       5   # load image from file
       6   pixels = pyplot.imread(filename)
       7   # create the detector, using default weights
       8   detector = MTCNN()
       9   # detect faces in the image
      10   results = detector.detect_faces(pixels)
      11   # extract the bounding box from the first face
      12   x1, y1, width, height = results[0]['box']
      13   x2, y2 = x1 + width, y1 + height
      14   # extract the face
      15   face = pixels[y1:y2, x1:x2]
      16   # resize pixels to the model size
      17   image = Image.fromarray(face)
      18   image = image.resize(required_size)
      19   face_array = np.asarray(image)
      20   return face_array
      21 a = extract_face("/content/drive/MyDrive/American_Universit
      22 print(a.shape)
```

Extract the faces by using MTCNN

Real
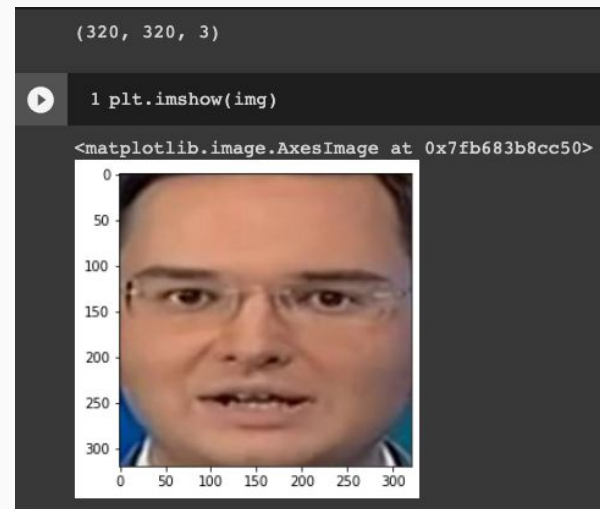
Extract the faces by using MTCNN

Deepfake
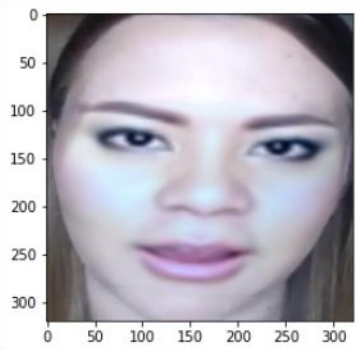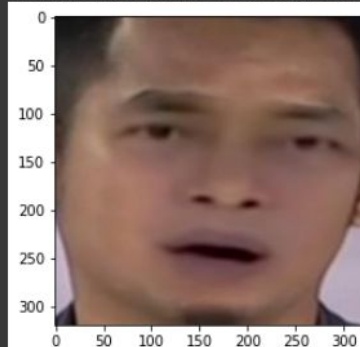
Before building ML models

- numpy.ndarray.flatten()

- Decomposition? (e.g. SVD)

- Normalization? (eg. 0-255 to 0-1)

- Mean subtraction for each pixel?

ML Workflow

X_{train}

$$\begin{bmatrix} fake & \cdots\cdots\cdots & fake \\ \vdots & & \\ fake & & \end{bmatrix}$$

$244 \times 244 \times 3 \times 720$

y_{train}

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$N \times 1$

$$\begin{bmatrix} real & \cdots\cdots\cdots & real \\ \vdots & & \\ real & & \end{bmatrix}$$

$244 \times 244 \times 3 \times 720$

$$\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

$N \times 1$

$$Modelname.fit(X, y).predict(X_{test})$$