Project 2 - Digital Forensics

Face Recognition Project

Francesco L. De Faveri - ID. 2057069

Università degli Studi di Padova

17^{th} of June 2022

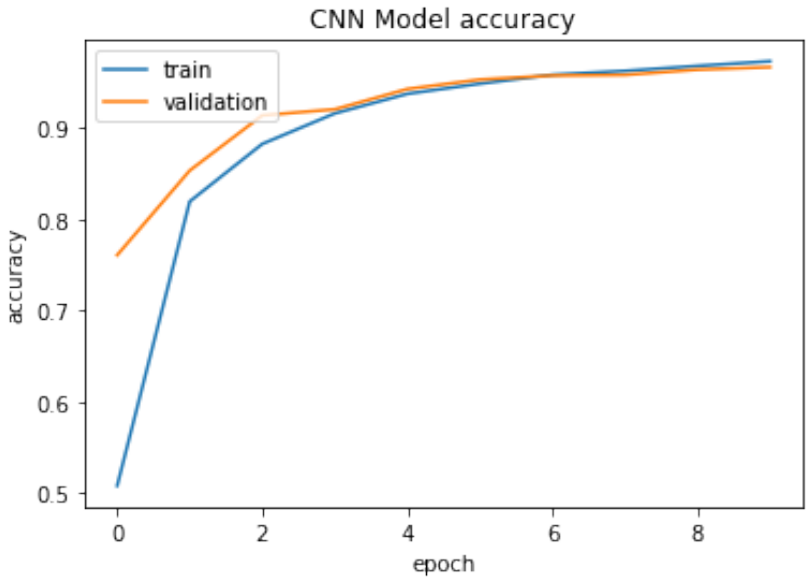
Structure of the project

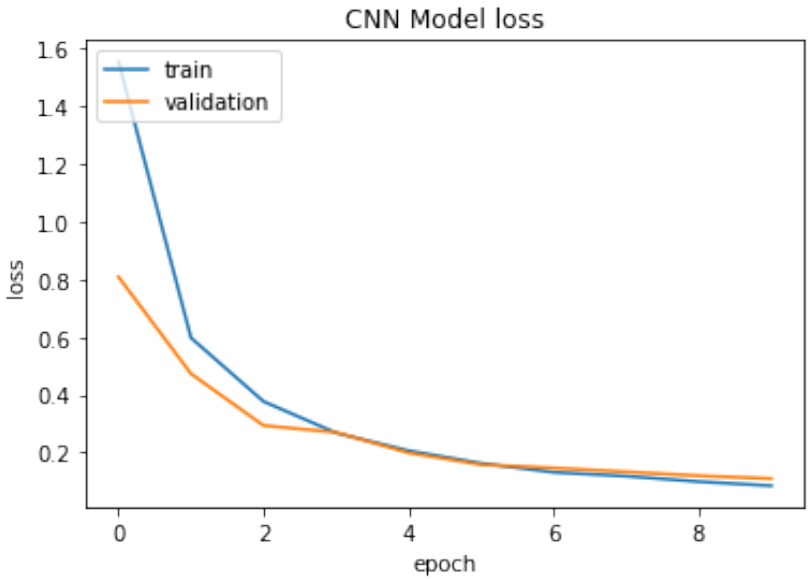
- **Section 1**, a possible solution for the experience in Lab. 4, implementing a simple model based on *keras.Conv2D* layers. The model has been tested also on partially covered faces, a.e. with glasses or facial mask.
- **Section 2**, extension of the face detection task through video. The model has been tested also on partially covered face. Also implementation of pretrained filters of OpenCV for detecting a face from a photo or live through the notebook camera.
- **Section 3**, implementation, training and testing of AlexNet architecture on the dataset.

CNN model Architecture

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 128, 128, 32)      320

 max_pooling2d (MaxPooling2D  (None, 64, 64, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 62, 62, 8)         2312

 max_pooling2d_1 (MaxPooling  (None, 31, 31, 8)        0
 2D)

 flatten (Flatten)           (None, 7688)              0

 dropout (Dropout)           (None, 7688)              0

 dense (Dense)               (None, 13)                99957

=================================================================
Total params: 102,589
Trainable params: 102,589
Non-trainable params: 0
_________________________________________________________________
```

CNN model Training

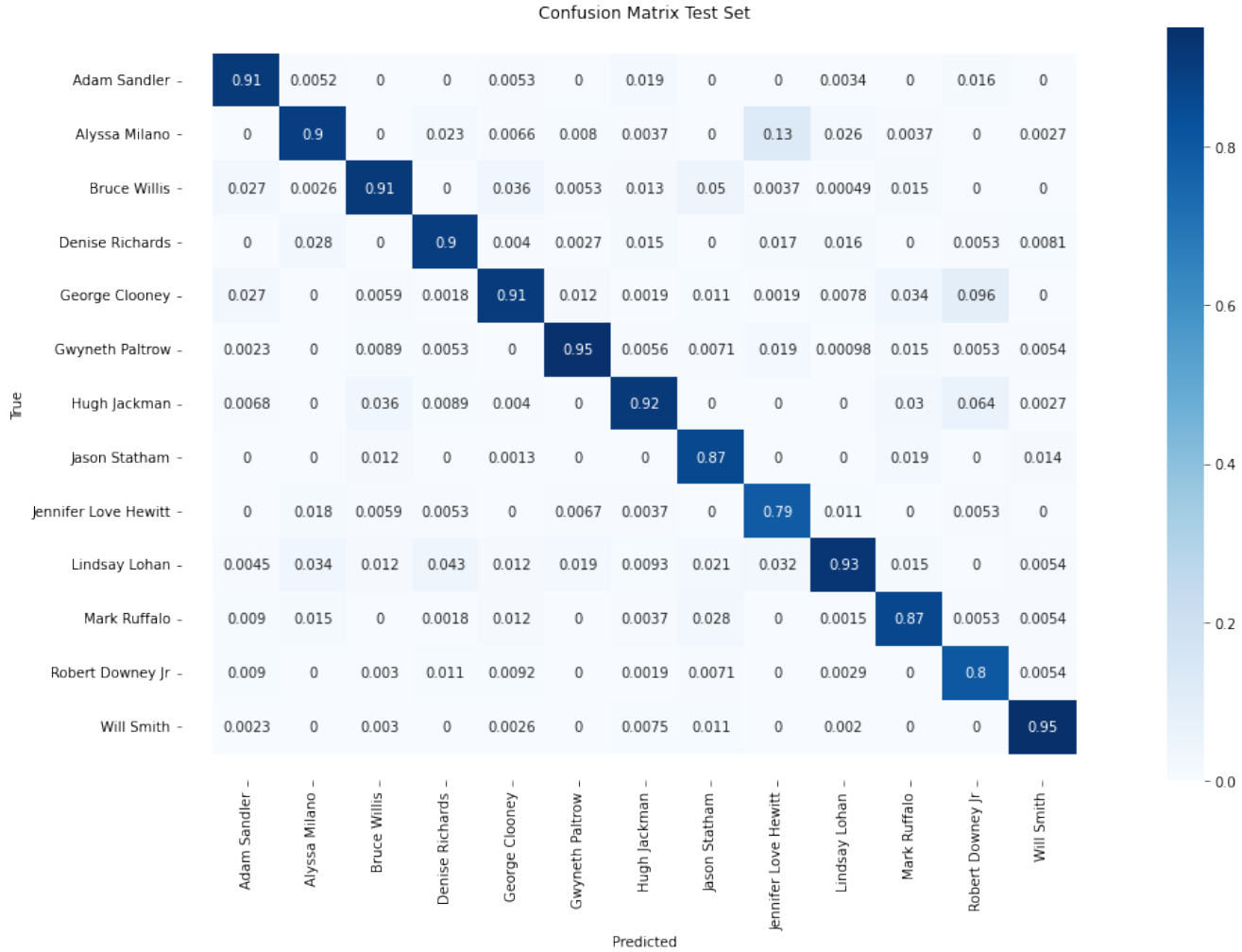




The training has been performed with the following parameters:

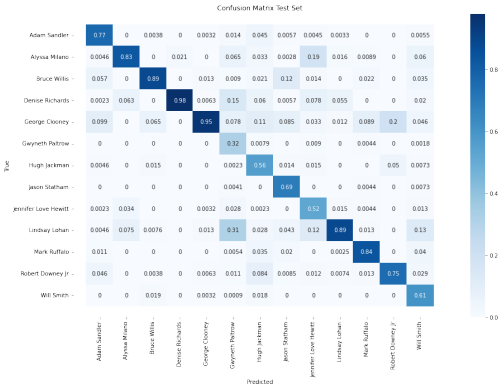
$$\text{batch\_size} = 128,\ \text{epochs} = 10,\ \text{validation\_split} = 0.2$$

CNN model Testing

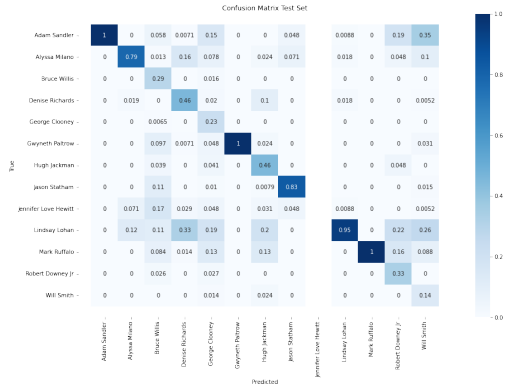


The overall test_accuracy is equal to 91%

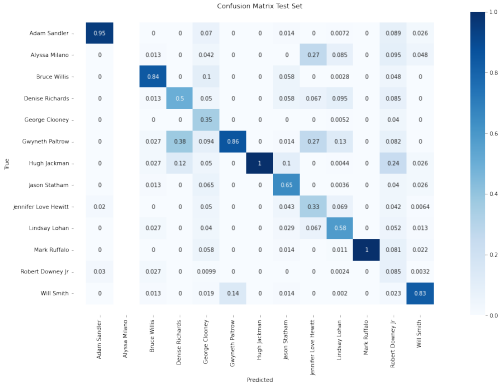
CNN on different test set



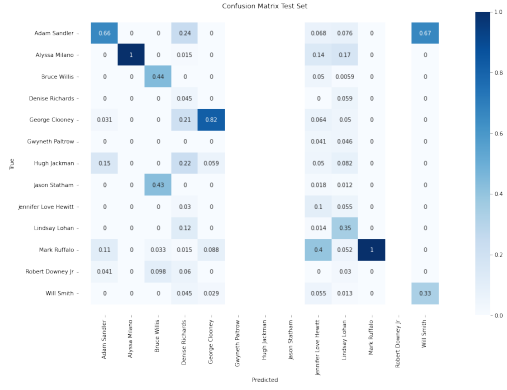
Faces with glasses: 61%



Faces with mask: 36%



Faces with sunglasses: 42%

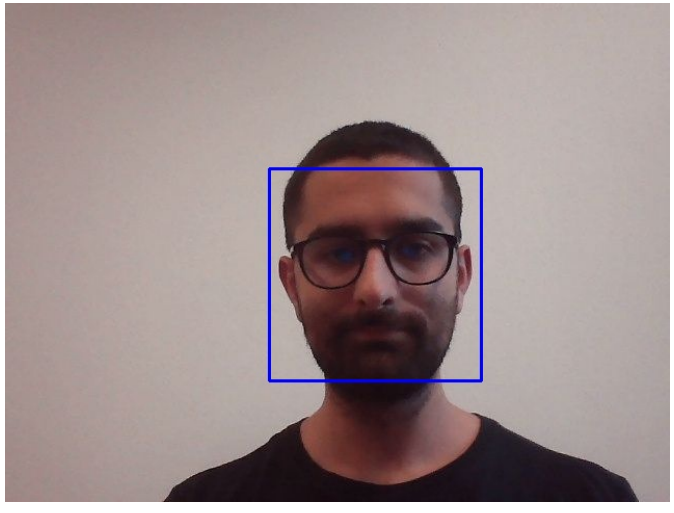


Faces with facial mask: 34%

Haar-cascade filters in OpenCV

```
#Snippet pre writte in Colab taken to access the camera of the notebook from the Colab app
def take_photo(filename='photo.jpg', quality=0.8):
  js = Javascript('''
    async function takePhoto(quality) {
      const div = document.createElement('div');
      const capture = document.createElement('button');
      capture.textContent = 'Click here to take the photo';
      div.appendChild(capture);

      const video = document.createElement('video');
      video.style.display = 'block';
      const stream = await navigator.mediaDevices.getUserMedia({video: true});

      document.body.appendChild(div);
      div.appendChild(video);
      video.srcObject = stream;
      await video.play();

      // Resize the output to fit the video element.
      google.colab.output.setIframeHeight(document.documentElement.scrollHeight, true);

      // Capture on click button
      await new Promise((resolve) => capture.onclick = resolve);

      const canvas = document.createElement('canvas');
      canvas.width = video.videoWidth;
      canvas.height = video.videoHeight;
      canvas.getContext('2d').drawImage(video, 0, 0);
      stream.getVideoTracks()[0].stop();
      div.remove();
      return canvas.toDataURL('image/jpeg', quality);
    }
    ''')
  display(js)

## Part of the code for the detection of a face using Haar Cascades

  # get photo data
  data = eval_js('takePhoto({})'.format(quality))

  # get OpenCV format image
  img = js_to_image(data)

  # grayscale img
  gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
  print(gray.shape)

  # get face bounding box coordinates using Haar Cascade
  faces = face_cascade.detectMultiScale(gray)

  # draw face bounding box on image
  for (x,y,width,height) in faces:
      img = cv2.rectangle(img,(x,y),(x+width,y+height),(255,0,0),2)

  # save image
  cv2.imwrite(filename, img)

  return filename
```

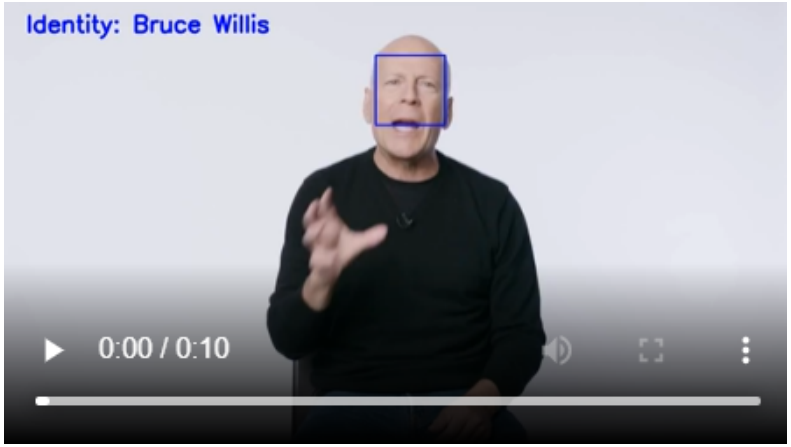
Snippet of the code



Output of the code

Video extension - Clear face

Input video



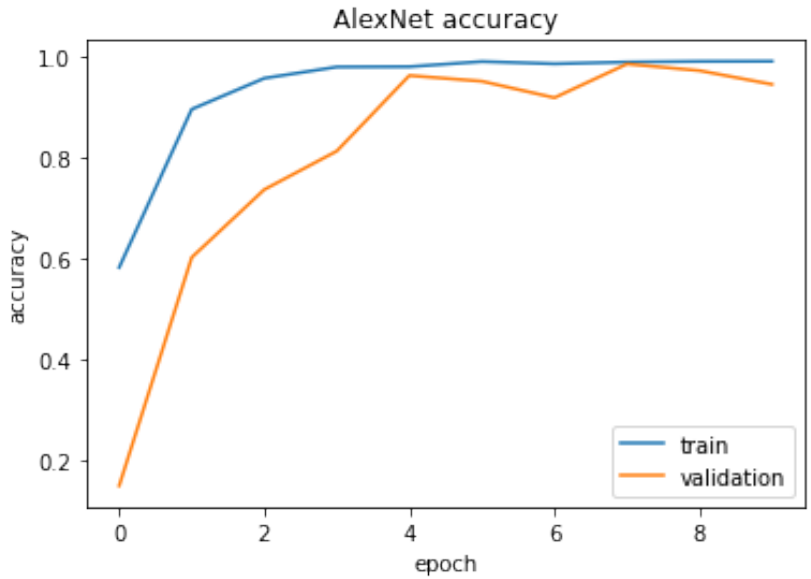
Output video

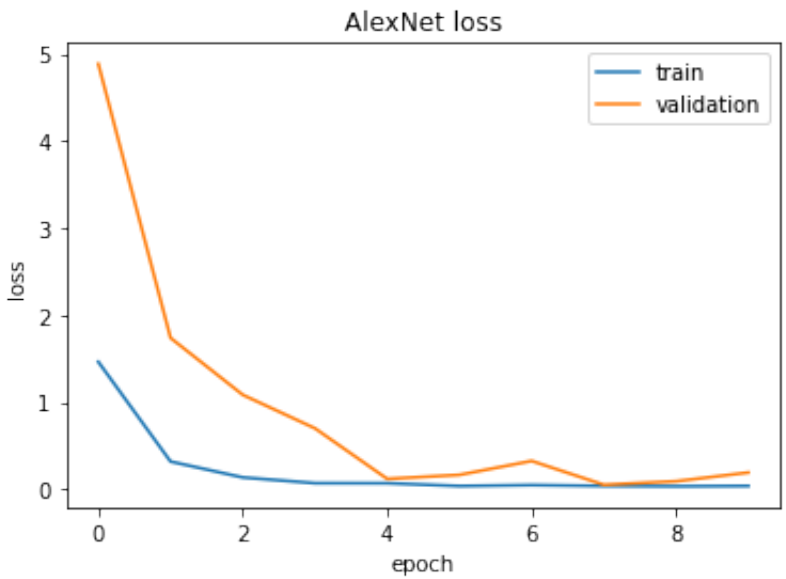
Video extension - Covered face

Input video

Output video

AlexNet Training

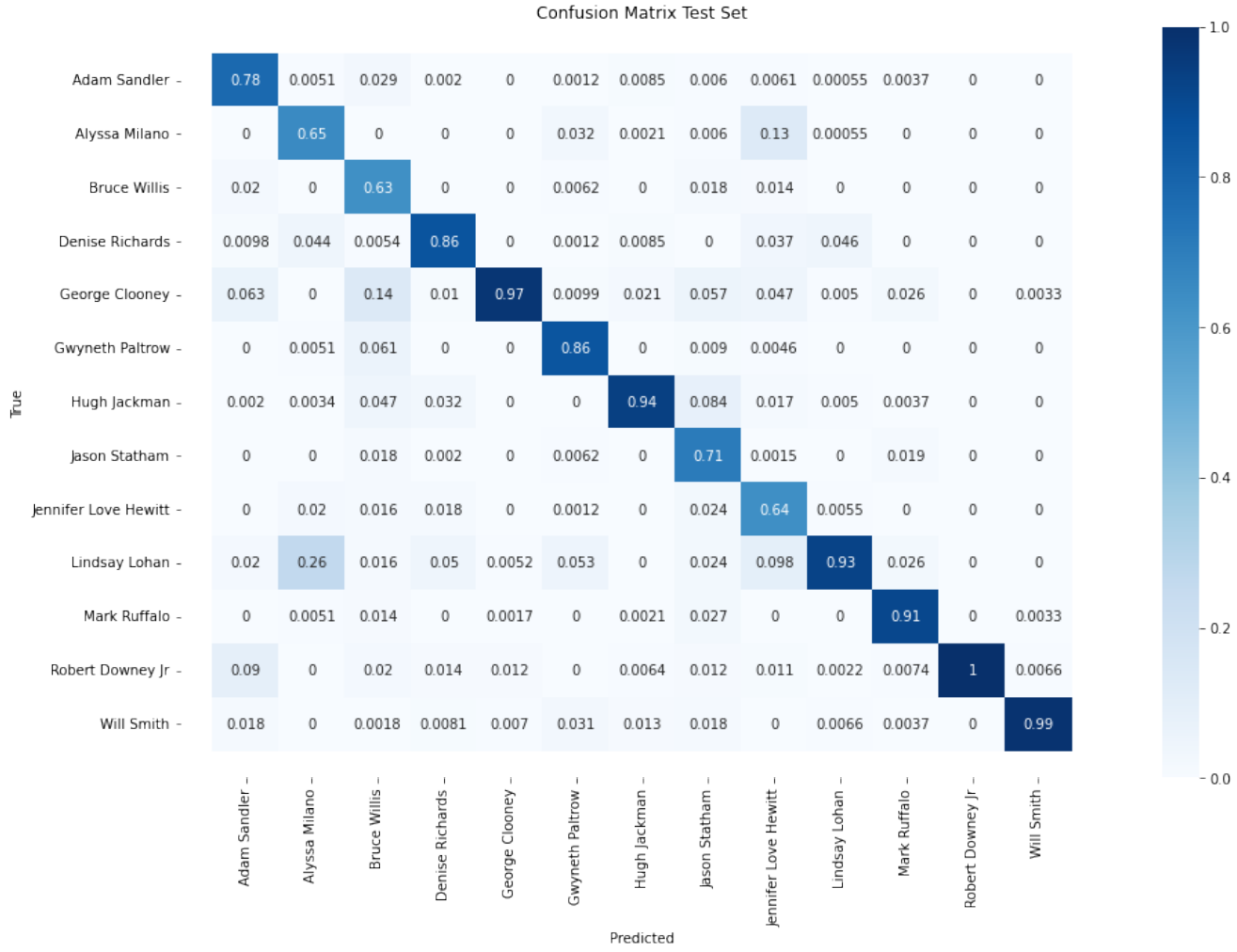




The training has been performed with the following parameters:

$$\text{batch\_size} = 128,\ \text{epochs} = 10,\ \text{validation\_split} = 0.2$$

AlexNet Testing



The overall test_accuracy is equal to 83%

References

1. **Section 1**: Lab4 Experience
2. **Section 2**: docs.opencv.org & theAIGuysCode/webcam
3. **Section 3**: top-4-pre-trained-models & Paper.pdf