



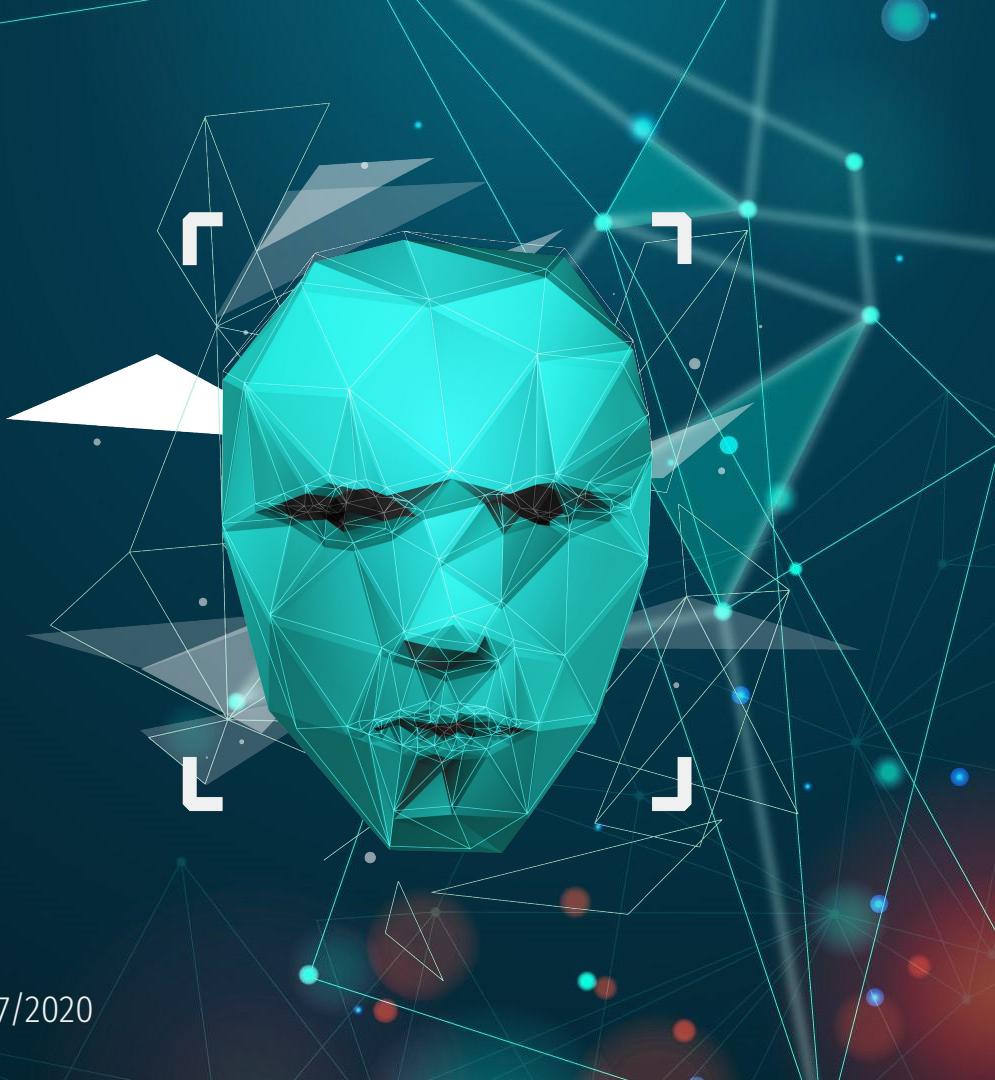
UATTENDED

# UAttended

## Attendance Checker Based On Face Recognition

Hsin-Ting Hsieh  
Theresia Baier  
Yuxin Wang

29/07/2020





UATTENDED

# Contents

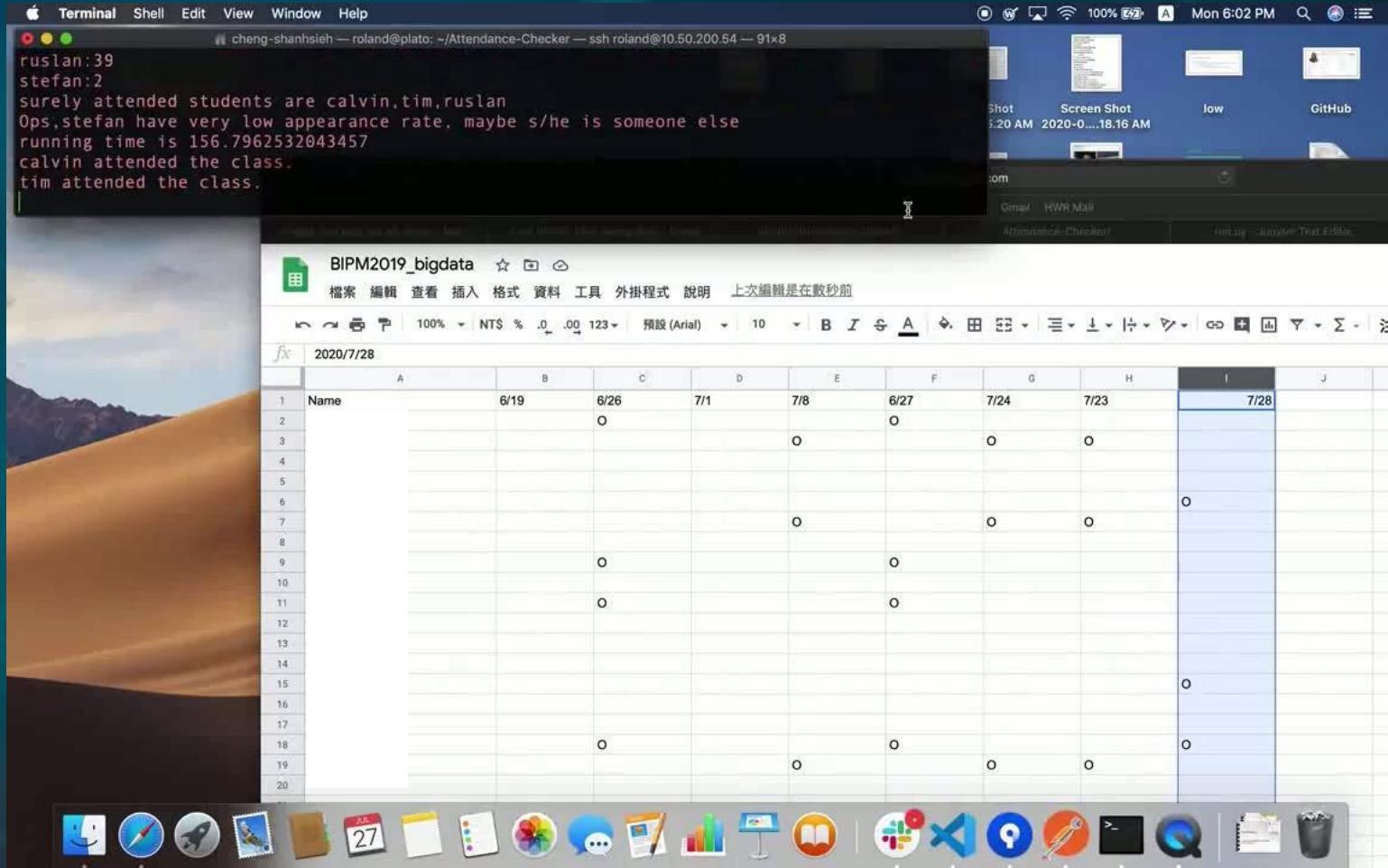
- 01** **Show Time!**  
**UAttended Demo**  
The user journey of generating  
an attendance list
- 03** **Implementation**  
Preparation, training, application,  
evaluation, deployment

- 02** **Techniques &  
Methodology**  
Face recognition methods &  
tools, GPU, API
- 04** **Conclusion &  
Outlook**  
Summary, reflection & further  
work

# Show Time! UAttended Demo

# 01

The user journey of generating an attendance list



# Motivation



## **Attendance Check Is Everywhere**

- An important action to evaluate students' dedication, the popularity of lectures/events
- Schools, companies, events all need attendance check

## **Online Learning Rises Up Dramatically**

- Over 1 billion students (pre-primary to tertiary) were affected due to COVID-19
- More and more distant learning programs

## **Filling-in Attendance List Is Forgettable**

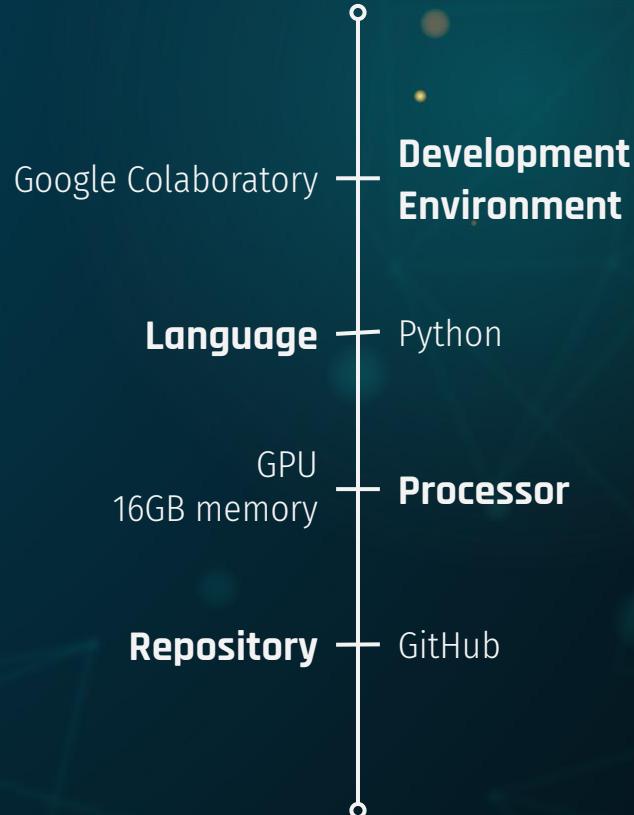
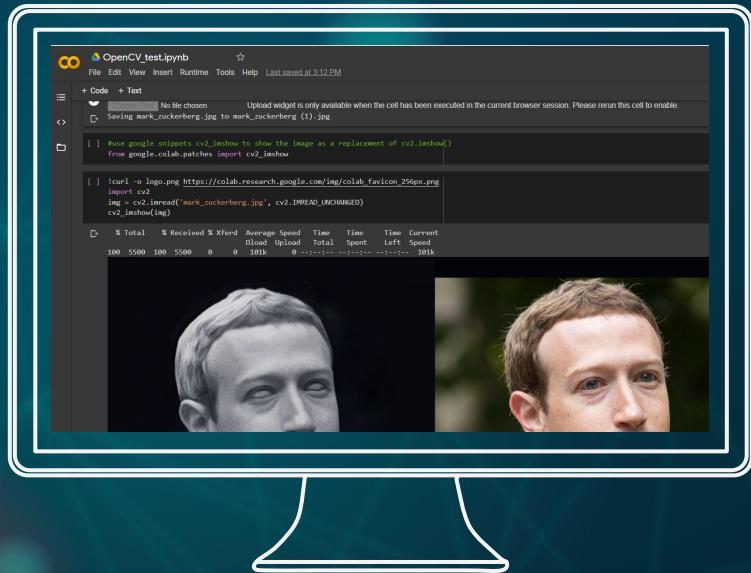
- Check-in in Excel every time for each lecture adds administrative efforts to professors and students
- Too many topics to focus on to remember check-in every time

# Techniques & Methodology

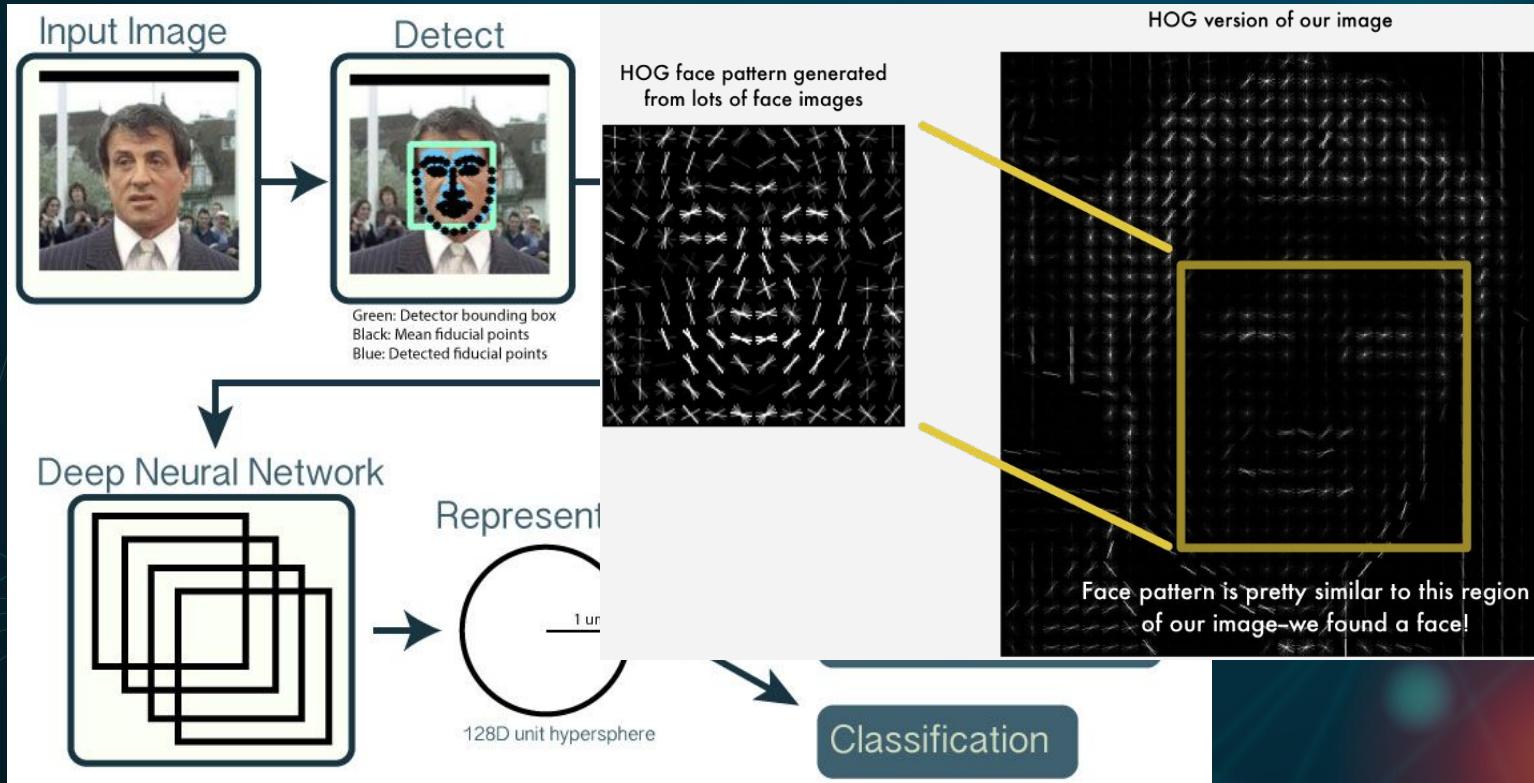
# 02

Face recognition methods & tools,  
GPU, API

# Working Environment



# Face Recognition



# Computer Vision Library - OpenCV

## Cascades

- About 6,000 classifiers for a face & thousands of data blocks
- Break the detection into 30 to 50 stages
- Only go to the further stage for detection if the block passes the current stage



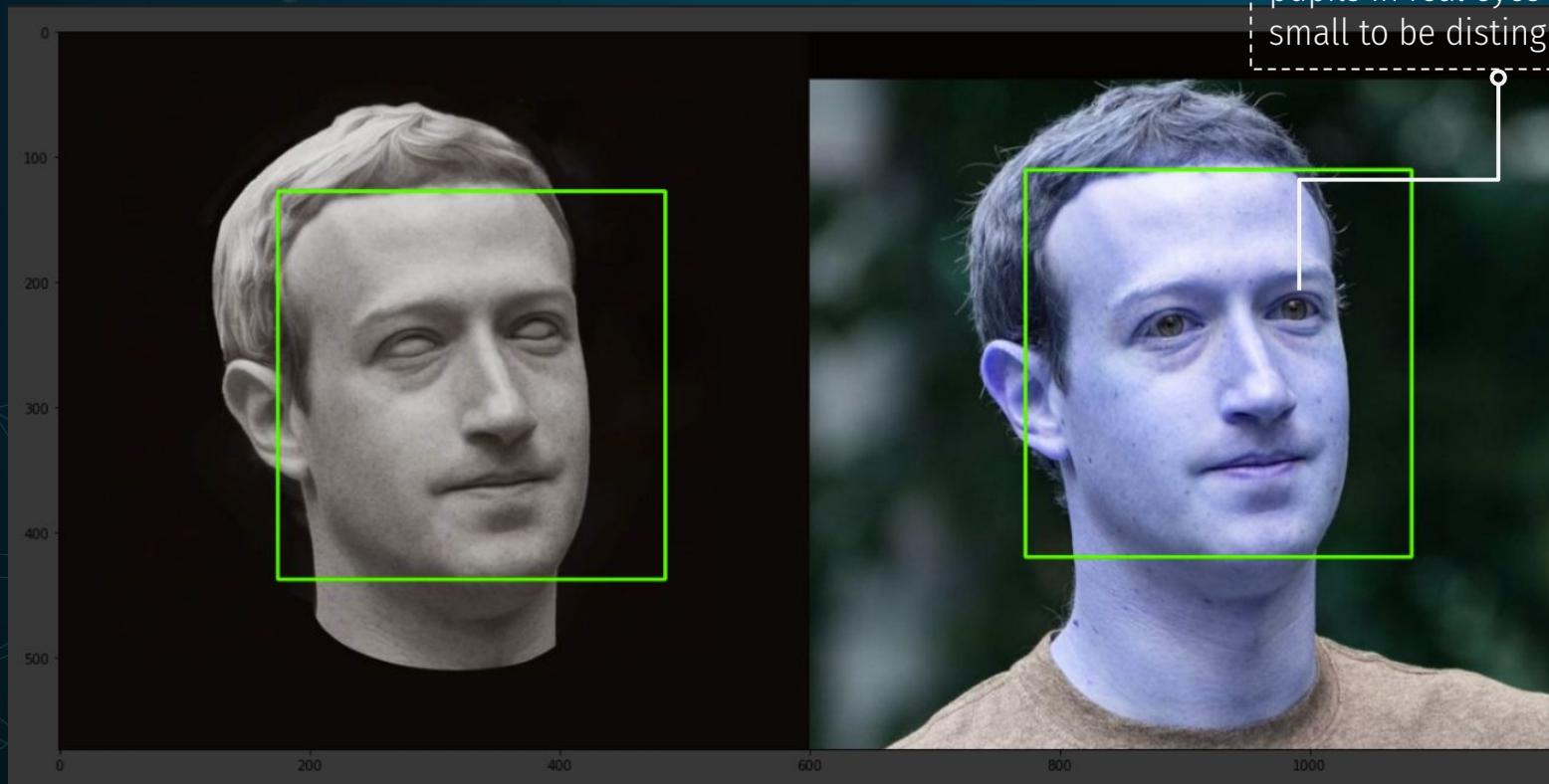
## Image Capture & Detection

- Google snippets for `cv2_imshow()` is needed
- Load Cascades xml and apply
- Draw a rectangle around the face

## Video Capture

- Via webcam or video files
- capture the video frame by frame.

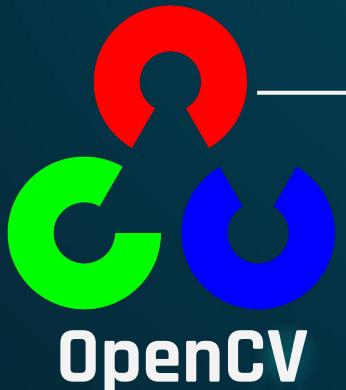
# OpenCV Face Detection Result



# Computer Vision Library - OpenCV

## Cascades

- About 6,000 classifiers for a face & thousands of data blocks
- Break the detection into 30 to 50 stages
- Only go to the further stage for detection if the block passes the current stage



## Image Capture & Detection

- Google snippets for `cv2_imshow()` is needed
- Load Cascades xml and apply
- Draw a rectangle around the face

## Video Capture

- Via webcam or video files
- capture the video frame by frame.

# Google Sheets API



- **Google API library (291 public available)**
- **Authentication and authorization**
- **Wide range of functionalities**

```
values = [
    [
        # Cell values ...
    ],
    # Additional rows ...
]
body = {
    'values': values
}
result = service.spreadsheets().values().update(
    spreadsheetId=spreadsheet_id, range=range_name,
    valueInputOption=value_input_option, body=body).execute()
print('{0} cells updated.'.format(result.get('updatedCells')))
```

# gspread



- **A Python API for Google Sheets.**
  - Integrate with pandas and Numpy packages

## Quick Example

```
import gspread

gc = gspread.service_account()

# Open a sheet from a spreadsheet in one go
wks = gc.open("Where is the money Lebowski?").sheet1

# Update a range of cells using the top left corner address
wks.update('A1', [[1, 2], [3, 4]])

# Or update a single cell
wks.update('B42', "it's down there somewhere, let me take another look.")

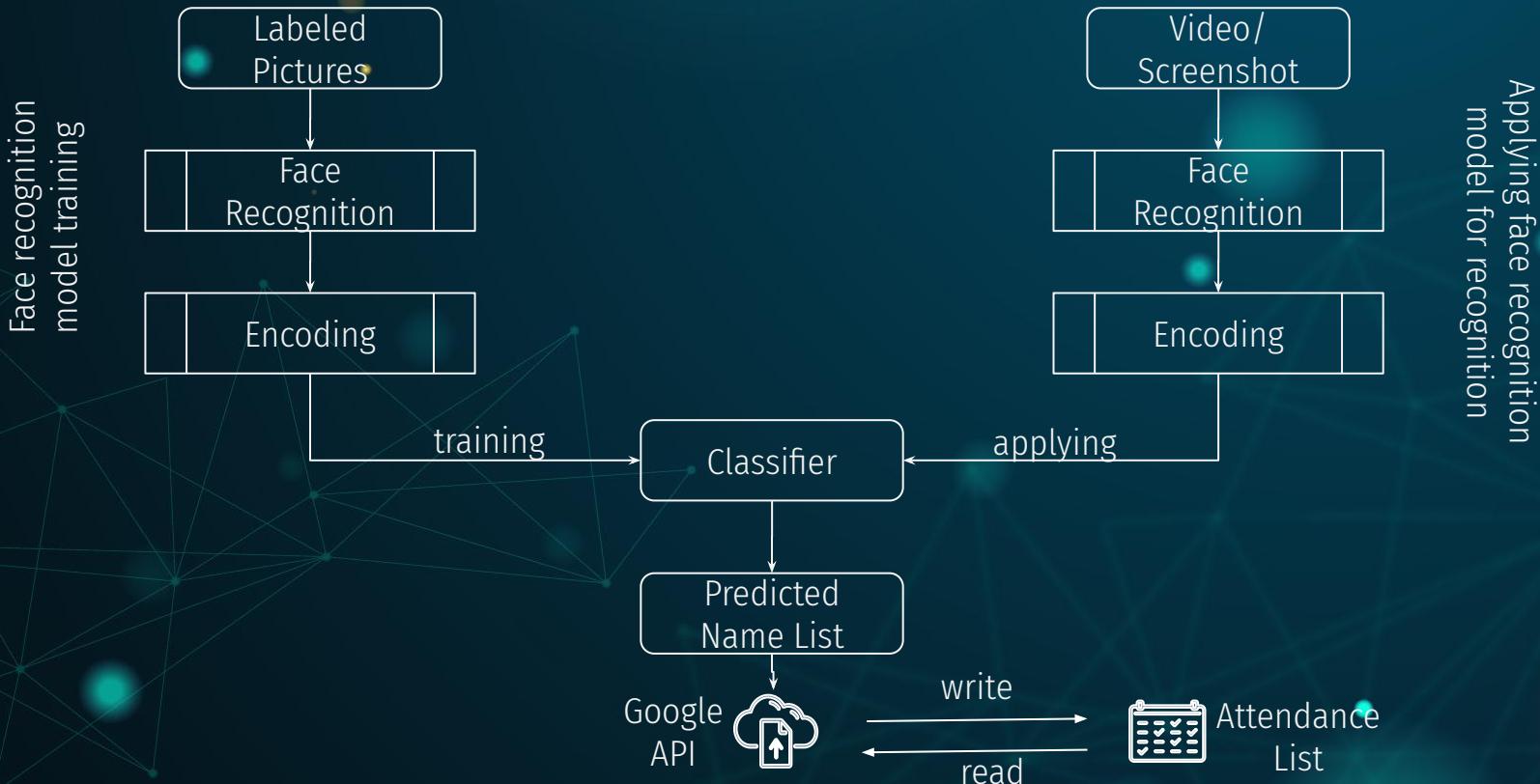
# Format the header
wks.format('A1:B1', {'textFormat': {'bold': True}})
```

# Implementation

# 03

Preparation, training, application,  
evaluation, deployment

# UAttended Architecture



# Preparation



Structure:

```
<test_image>.jpg  
<train_dir>/  
  <person_1>/  
    <person_1_face-1>.jpg  
    <person_1_face-2>.jpg  
    .  
    .  
    <person_1_face-n>.jpg  
  <person_2>/  
    <person_2_face-1>.jpg  
    <person_2_face-2>.jpg  
    .  
    .  
    <person_2_face-n>.jpg  
  .  
  .  
  <person_n>/  
    <person_n_face-1>.jpg  
    <person_n_face-2>.jpg  
    .  
    .  
    <person_n_face-n>.jpg
```

# Training Face Recognition Model



# Testing Evaluation

01

Testing with Individual Pictures

	precision	recall	f1-score	support
1	1.00	1.00	1.00	9
2	1.00	1.00	1.00	11
3	1.00	1.00	1.00	9
4	1.00	1.00	1.00	5
5	1.00	1.00	1.00	8
6	1.00	1.00	1.00	7
7	1.00	1.00	1.00	5
8	1.00	1.00	1.00	1
9	1.00	1.00	1.00	3
10	1.00	1.00	1.00	10
11	1.00	1.00	1.00	6
12	1.00	1.00	1.00	1
13	1.00	1.00	1.00	3
14	1.00	1.00	1.00	5
15	1.00	1.00	1.00	4
16	1.00	1.00	1.00	4
17	1.00	1.00	1.00	8
accuracy	100 %	1.00	1.00	103
macro avg	1.00	1.00	1.00	103
weighted avg	1.00	1.00	1.00	103

02

Random Tests with Class Pictures

class (11) true	detected once	class (11) prediction
TRUE		
TRUE		
FALSE		
TRUE		
FALSE		
FALSE		
TRUE		
TRUE		
TRUE		
FALSE		
TRUE		
FALSE		
TRUE		
total	16	13
wrong/missing	4	
75,00% accuracy		

Mean accuracy across  
four examples

88.84 %

# Scenarios

## 01 Recognize with Screenshot

**Pros:** Fast and simple

**Cons:** need to choose the proper moment in case some students are late



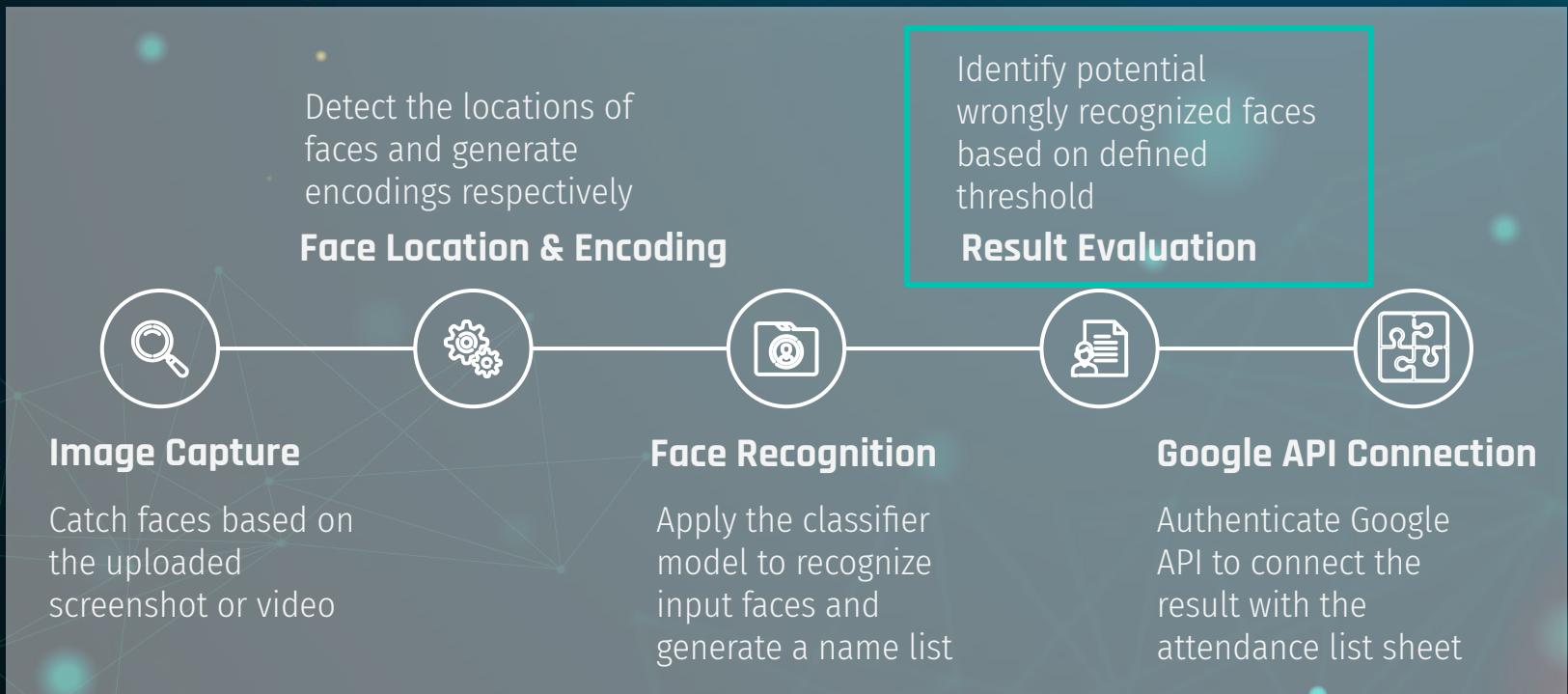
## 02 Recognize with Videos/Webcam

**Pros:** No interruption during the lecture, can extract more info, not miss late students

**Cons:** Complicated computation, privacy concern



# Face Recognition Application



# Result Evaluation

01

## Recognize with Screenshot

identify names  
that appear more  
than once

Calculate the percentage  
of names that could be  
wrongly recognized are  
calculated

Number of faces detected: 13  
students attended are  
anelia,saner,calvin,theresia,yuxin,hsin-ting,  
ruslan,  
angela,chen,immo,amirhossein,tim,stefan.  
Ops! Someone looks too similar to  
ana\_luisa and was wrongly recognized!  
7.6% of faces are recognized duplicitely.  
running time is 0.008185625076293945.

02

## Recognize with Videos/Webcam

Generate a dictionary  
{key =name: value =  
the frequency  
appearing in all  
frames}

The frequency of names in all frames are:  
yuxin: 63  
hsin-ting: 89  
theresia 89  
ana\_luisa: 2  
Surely attended students are  
yuxin,hsin-ting, theresia.  
Ops, ana\_luisa has very low appearance  
rate, maybe s/he is someone else!

Threshold:  
if the frequency of a  
name( $v_i$ )/max(v) < 10%  
the name could potentially  
be wrongly recognized.

# UAttended Deployment on GitHub

hsiehkl / Attendance-Checker

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master 1 branch 0 tags Go to file Add file Code

hsiehkl Update README.md 827577b 15 minutes ago 25 commits

cred	Add folders	yesterday
material	Add more images for api setup	4 days ago
models	Add folders	yesterday
raw_data	Add folders	yesterday
train_dir	Add folders	yesterday
.gitignore	Update .gitignore	yesterday
README.md	Update README.md	15 minutes ago
api_connector.py	Check empty name list	yesterday
google_sheets.json	Add google_sheets.json example	4 days ago
predict.py	Fix typo	yesterday
run.py	Update run.py	yesterday
train.py	Update train.py	yesterday

About No description, website, or topics provided.

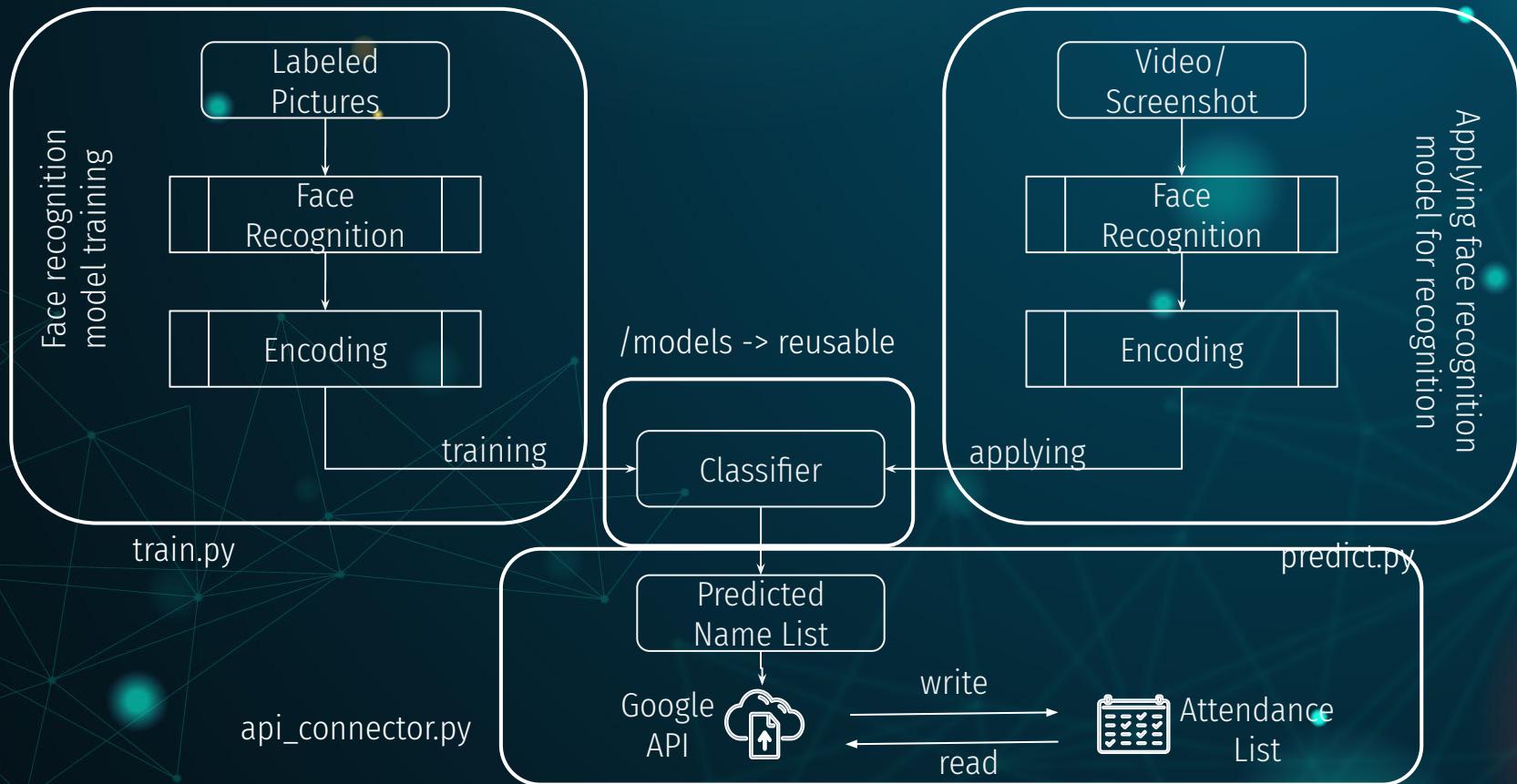
Readme

Releases No releases published Create a new release

Packages No packages published Publish your first package

Languages Python 100.0%

# UAttended Main Components



# UAttended Deployment on GitHub

<https://github.com/hsiehkl/Attendance-Checker>

# Conclusion & Outlook

# 04

Summary, reflection & further work

**LET AI DO  
THE JOB**

# Challenges



- **Adaption To Google Colaboratory**

- File reading

- Display the video and image

- **Limited GPU memory**

- Can't reliably apply CNN model

- **<100% Accuracy**

- Fairness

- Subtle manual adjustments needed

- Unknown faces can't be recognized as unknown

- **Data privacy**

- Dissent of students

# Misclassification

```
[8] # Load the test image with unknown faces into a numpy array
test_image = face_recognition.load_image_file(data_path_in + '/class/obama.jpg')

# Find all the faces in the test image using the default HOG-based model
face_locations = face_recognition.face_locations(test_image)
no = len(face_locations)
print("Number of faces detected: ", no)

# Predict all the faces in the test image using the trained classifier
name_list = []
print("Found:")
for i in range(no):
    test_image_enc = face_recognition.face_encodings(test_image)[i]
    name = model.predict([test_image_enc])
    name_list.append(*name)
    print(*name)
print(name_list)

⇨ Number of faces detected: 1
Found:
yasmin
['yasmin']
```

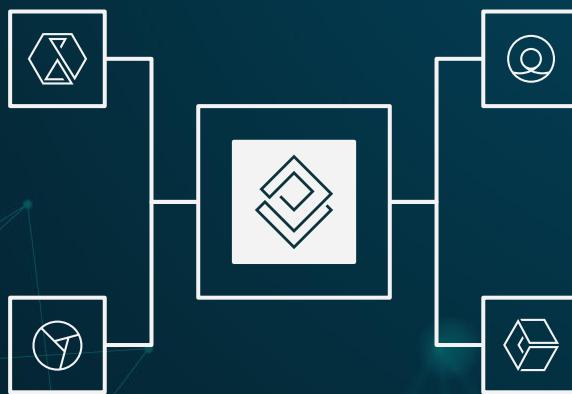


# Further Work

Get more powerful GPU  
Apply CNN face recognition model  
Tune classifier model  
Reach 100% accuracy rate

## Model Optimization

**More Functions**  
Capture check-in and check-out time  
Emotion Analysis  
Lecture quality analysis



More user-friendly  
Use the service directly without the necessity to download specific packages

## Frontend

**Privacy Security**  
Strategy and usage rules to ensure the secure environment and privacy protection

# THANKS!

Do you have any questions ?



UATTENDED

# Sources

Summary of Face Recognition Steps: Amos, B. (2015, September 24). Summary [Digital image]. Retrieved July 27, 2020, from <https://github.com/cmusatyalab/openface/blob/master/images/summary.jpg>

HOG Face Patterns: [HOG Face Patterns]. (n.d.). Retrieved July 27, 2020, from [https://miro.medium.com/max/875/1\\*6xgev0r-qn4oR88FrW6fiA.png](https://miro.medium.com/max/875/1*6xgev0r-qn4oR88FrW6fiA.png)

Folder Structure: Baburajan, A. (2019, December 03). Face\_recognition/face\_recognition\_svm.py at master · ageitgey/face\_recognition. Retrieved July 26, 2020, from [https://github.com/ageitgey/face\\_recognition/blob/master/examples/face\\_recognition\\_svm.py](https://github.com/ageitgey/face_recognition/blob/master/examples/face_recognition_svm.py)

Obama: Obama [Digital image]. (n.d.). Retrieved July 26, 2020, from [https://raw.githubusercontent.com/ageitgey/face\\_recognition/master/examples/obama.jpg](https://raw.githubusercontent.com/ageitgey/face_recognition/master/examples/obama.jpg)