HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

Biometric authentication system - IT4432E

2D-Face authorization

Members:

- Chu Trung Anh 20225564
- Bui Duy Anh 20225563
- Pham Minh Tien 20225555

Supervisor:

- Dr.Ngo Thanh Trung
- Dr.Tran Nguyen Ngoc

ONE LOVE. ONE FUTURE.

HUST

hust.edu.vn fb.com/dhbkhn

Content

1. Introduction
2. Dataset
3. Our pipeline
4. Evaluation
5. Application
6. Future improvement

1. Introduction

In today's world, strong security is more important than ever because traditional methods like passwords can be easily hacked

Pros:

- Easier and fast way to verify access

Cons:

- Can be inefficient due to noise or face spoofing

Facial authentication is an advanced solution that uses your unique facial features to verify your identity

1. Introduction

Reason why we choose this approach to the problem:

- Easy to implemented in real-life situation
- High universality and acceptability
- Security enhancement
- Broad application: in company, government, individual,...
- Secure physical spaces(room, safe, building,...)
- Secure digital system(laptop, phone,...)

1. Introduction

Dependenices

- Backend:

TensorFlow™

PyTorch

scikit learn

OpenCV

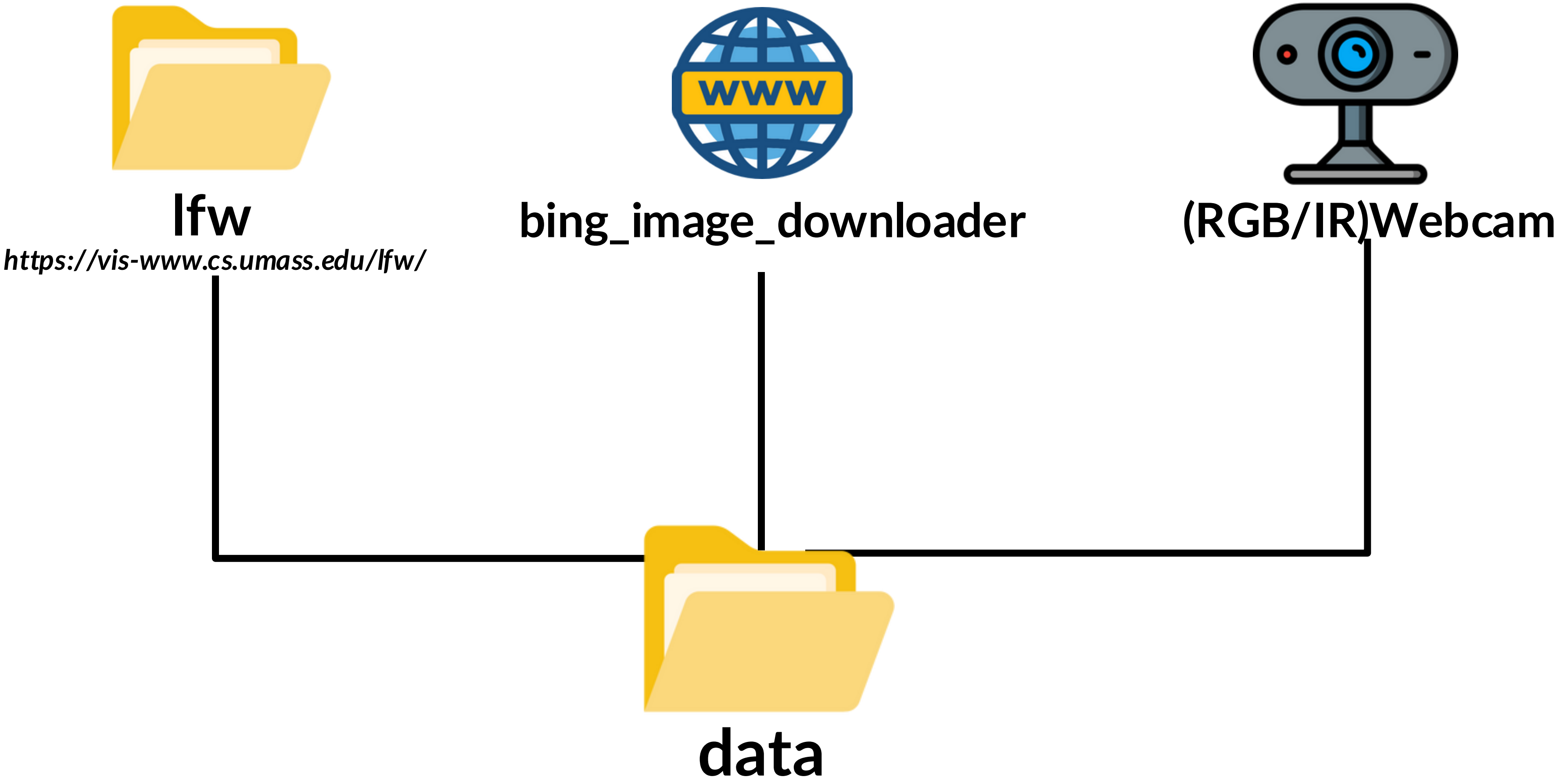
- GUI:

PYQT6

kivy

2. Dataset

- Our project used three source of data:



lfw

https://vis-www.cs.umass.edu/lfw/

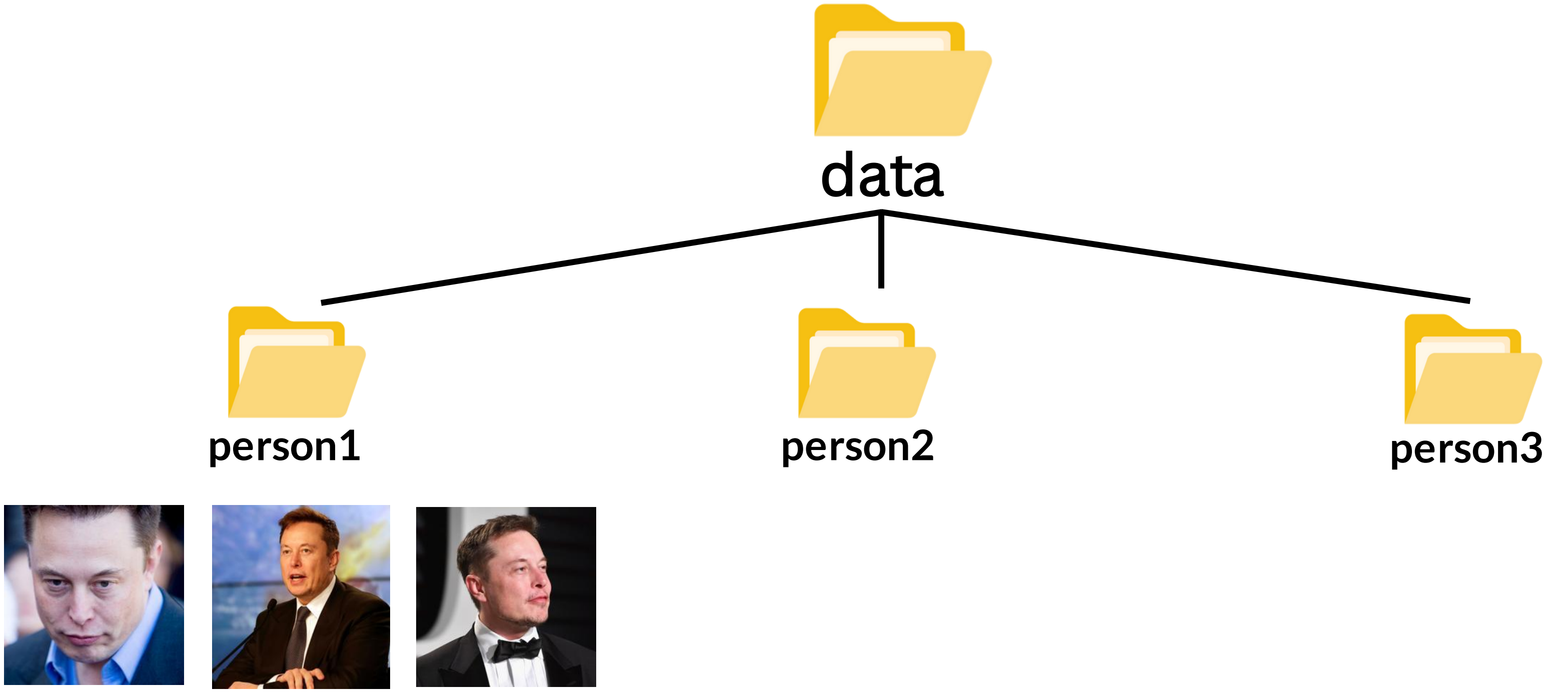
bing_image_downloader

(RGB/IR)Webcam

data

2. Dataset

- Data structure:



data

person1

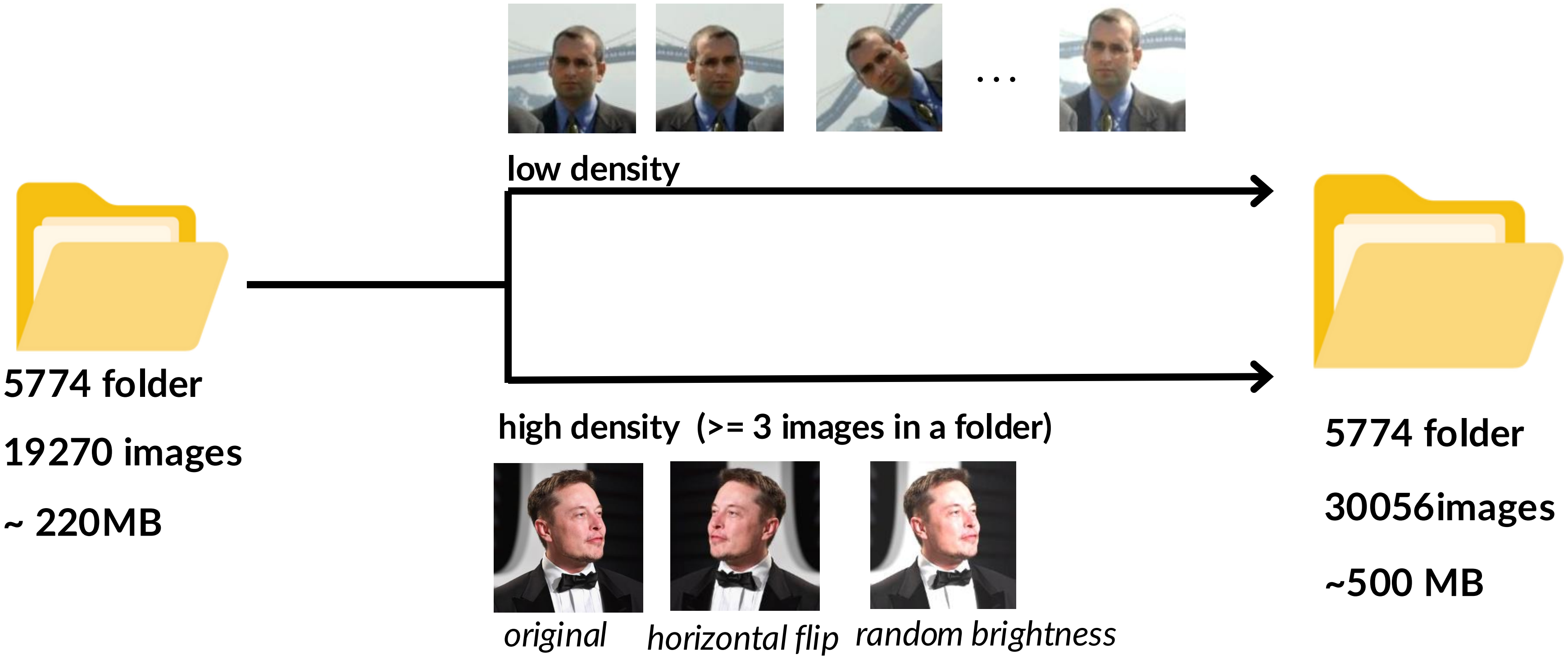
person2

person3

- 250x250 jpg format

2. Dataset

- Data argumentation:



low density

. . .

5774 folder

19270 images

~ 220MB

high density (>= 3 images in a folder)

original

horizontal flip

random brightness

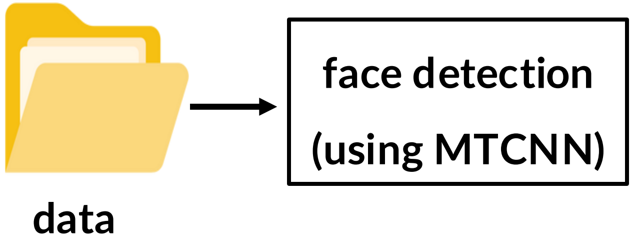
5774 folder

30056images

~500 MB

3. Pipeline

3.1 Train model



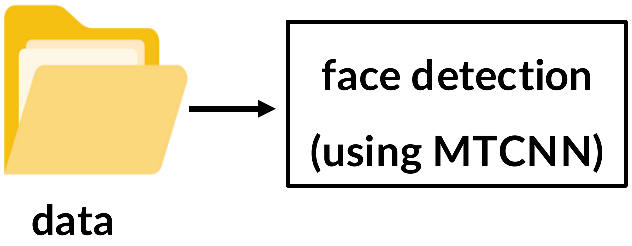
data

face detection

(using MTCNN)

3. Pipeline

3.1 Train model



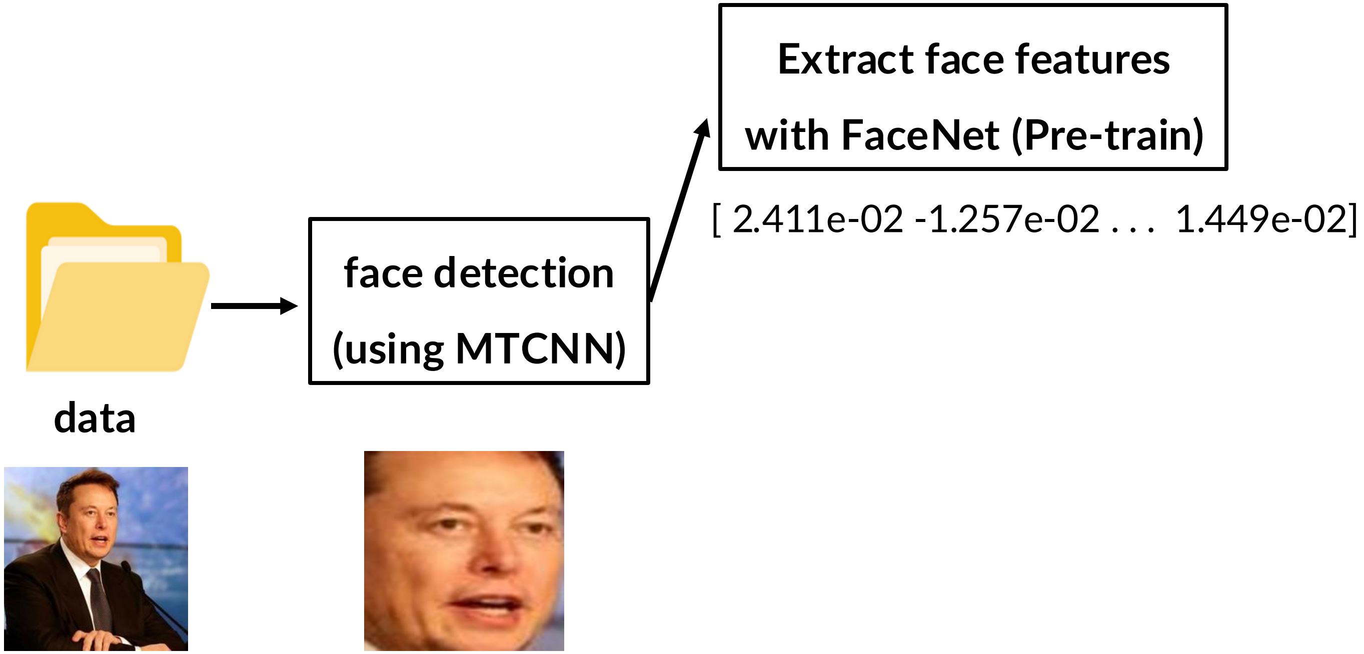
data

face detection

(using MTCNN)

3. Pipeline

3.1 Train model



data

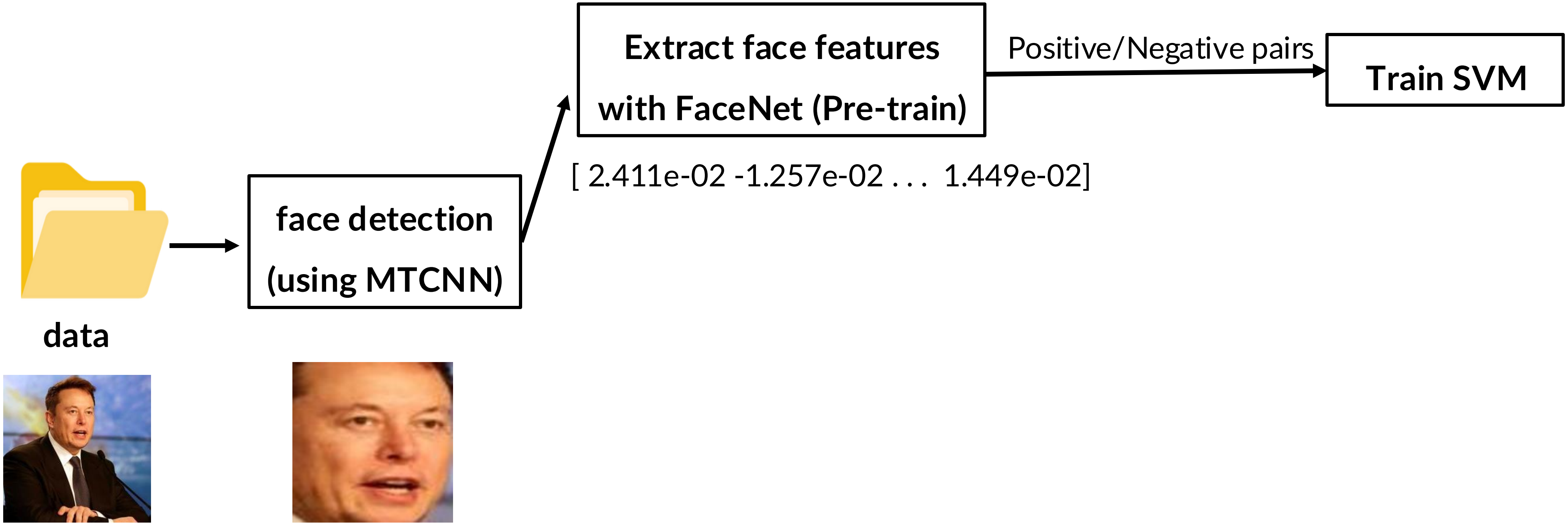
face detection (using MTCNN)

Extract face features with FaceNet (Pre-train)

[2.411e-02 -1.257e-02 . . . 1.449e-02]

3. Pipeline

3.1 Train model



data

face detection (using MTCNN)

Extract face features with FaceNet (Pre-train)

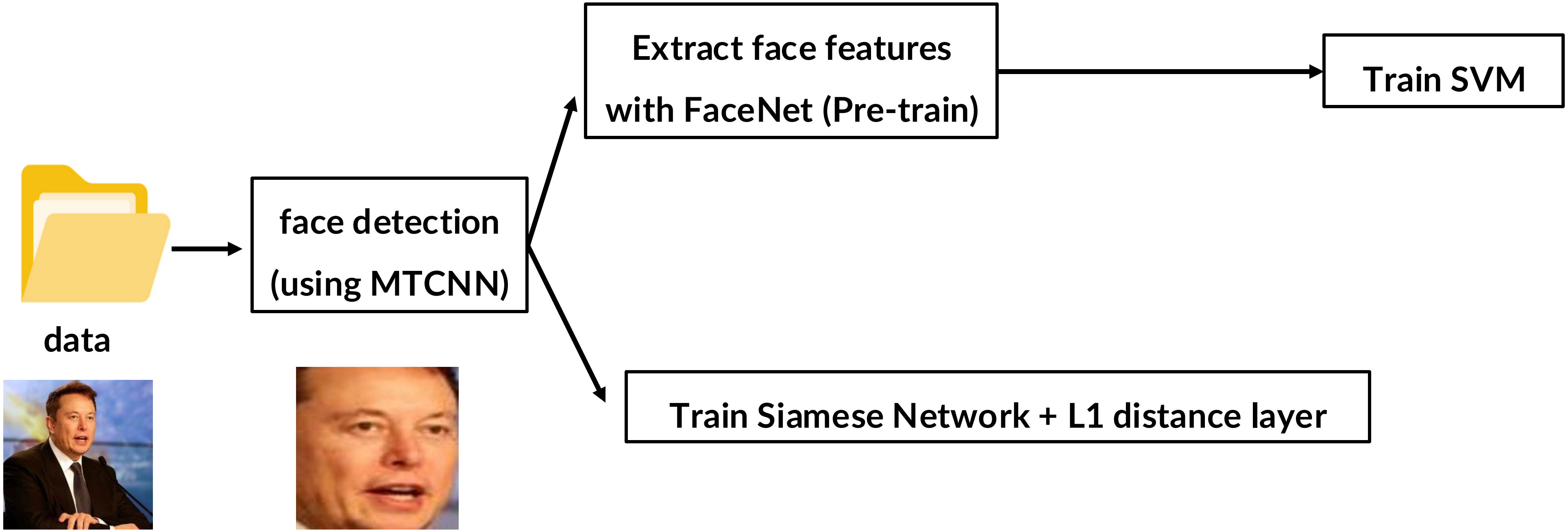
[2.411e-02 -1.257e-02 . . . 1.449e-02]

Positive/Negative pairs

Train SVM

3. Pipeline

3.1 Train model



data

face detection

(using MTCNN)

Extract face features

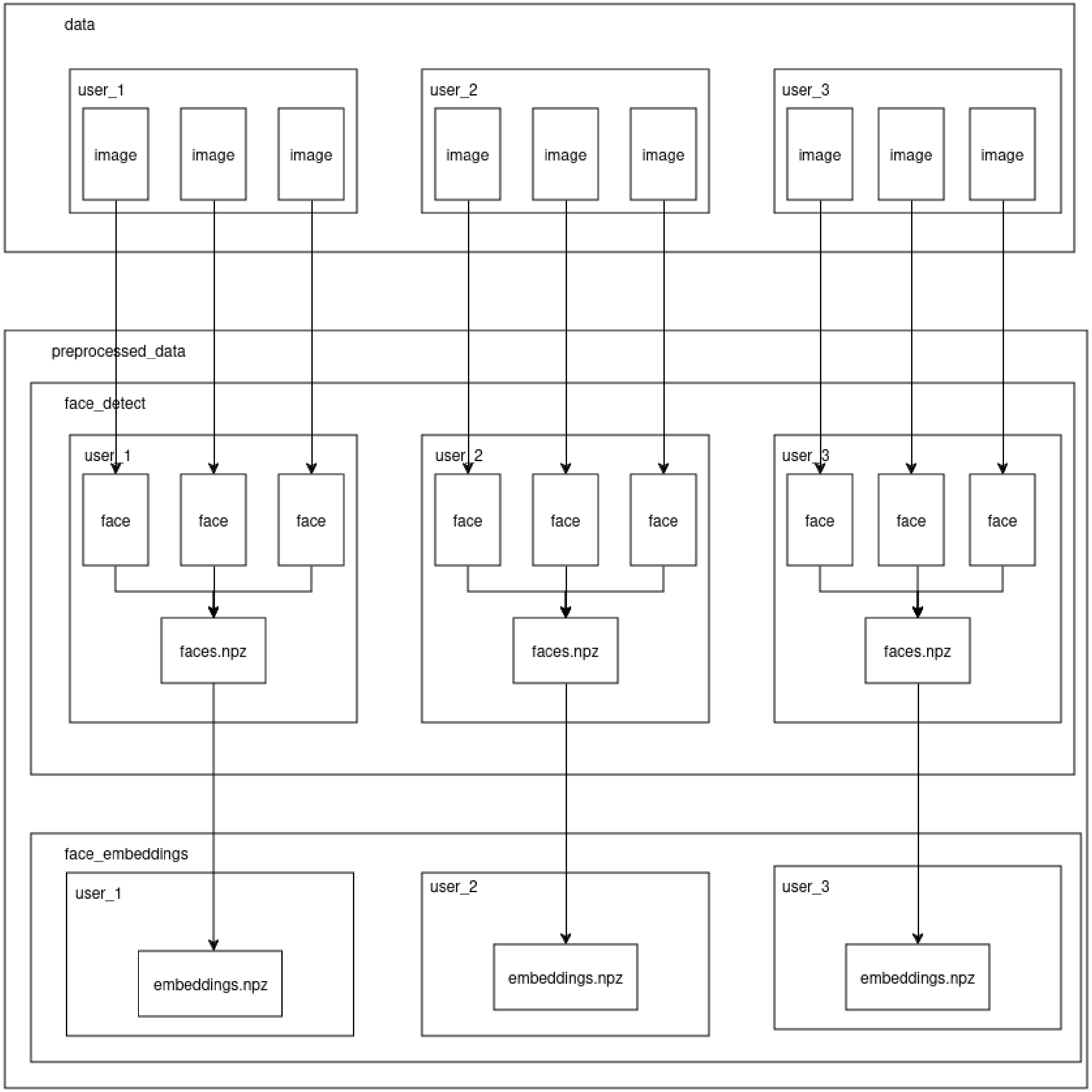
with FaceNet (Pre-train)

Train SVM

Train Siamese Network + L1 distance layer

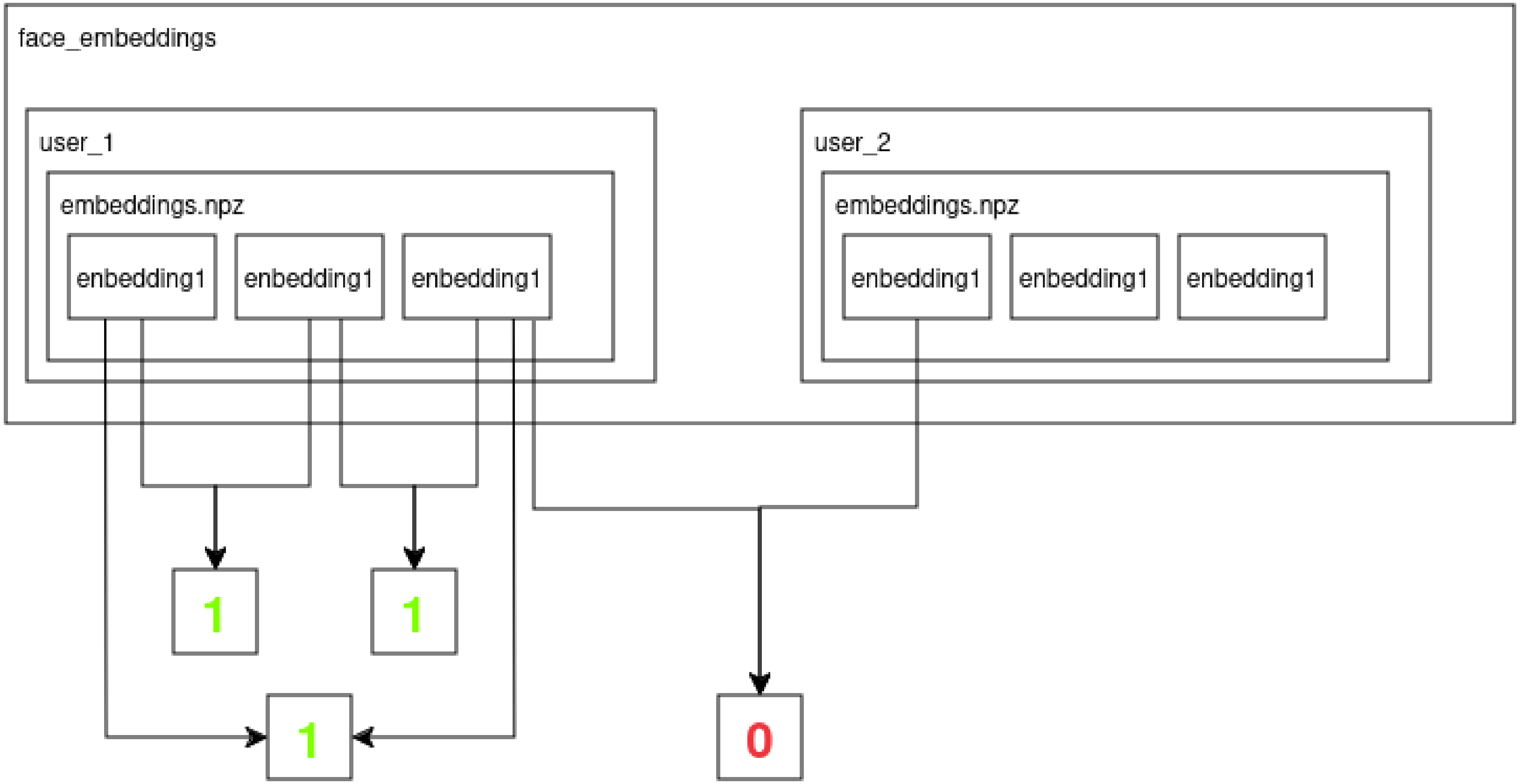
3. Pipeline

3.1.1 Train model – pipeline 1



3. Pipeline

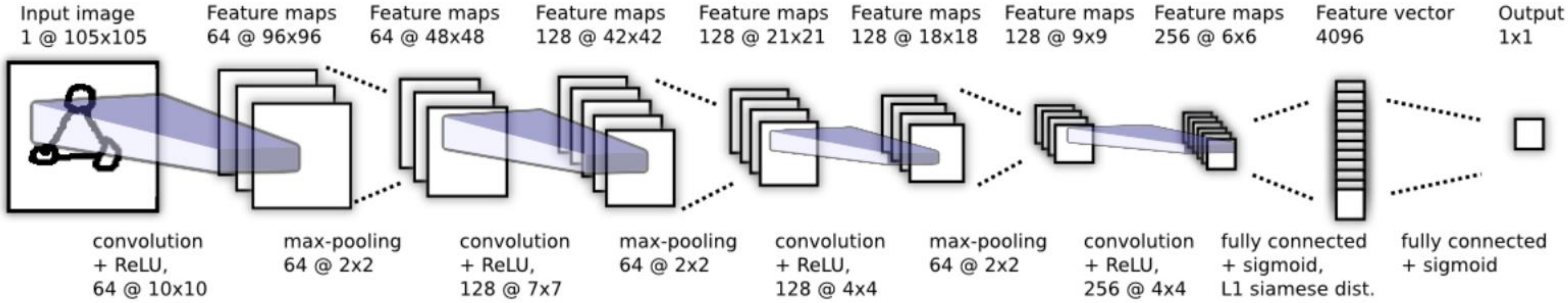
3.1.1 Train model – pipeline 1



3. Pipeline

3.1.2 Train model – pipeline 2

- Our network architecture:

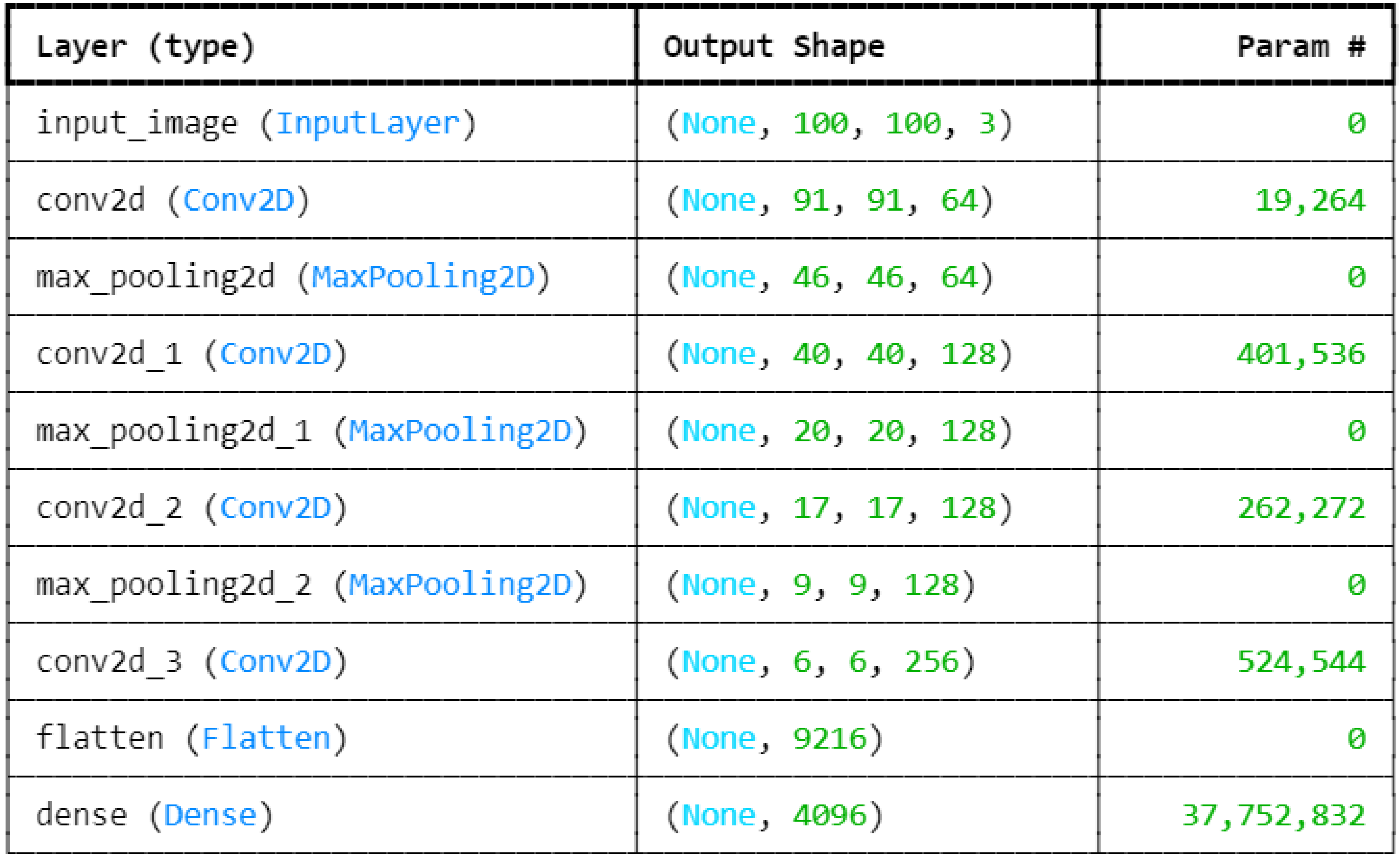


https://www.cs.cmu.edu/~rsalakhu/papers/oneshot1.pdf

3. Pipeline

3.1.2 Train model – pipeline 2

- Our network architecture:



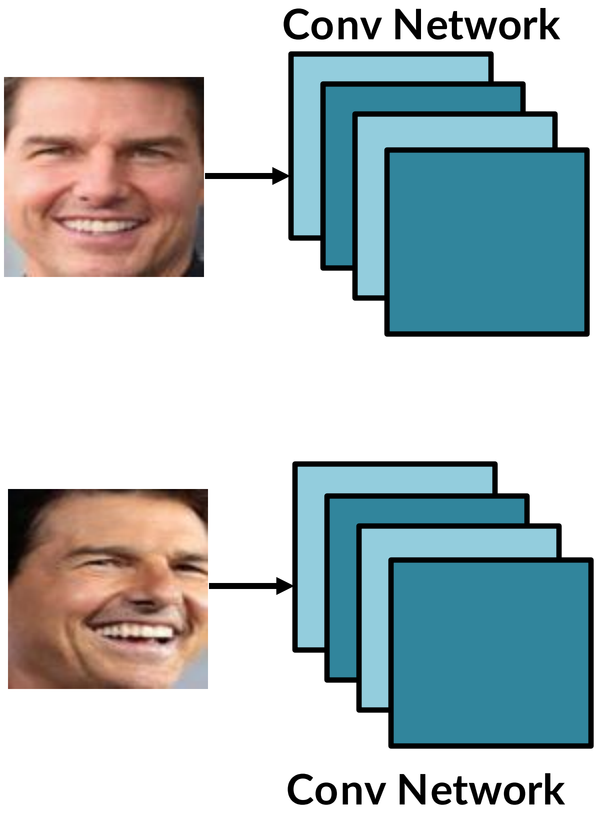
Layer (type)	Output Shape	Param #
input_image (InputLayer)	(None, 100, 100, 3)	0
conv2d (Conv2D)	(None, 91, 91, 64)	19,264
max_pooling2d (MaxPooling2D)	(None, 46, 46, 64)	0
conv2d_1 (Conv2D)	(None, 40, 40, 128)	401,536
max_pooling2d_1 (MaxPooling2D)	(None, 20, 20, 128)	0
conv2d_2 (Conv2D)	(None, 17, 17, 128)	262,272
max_pooling2d_2 (MaxPooling2D)	(None, 9, 9, 128)	0
conv2d_3 (Conv2D)	(None, 6, 6, 256)	524,544
flatten (Flatten)	(None, 9216)	0
dense (Dense)	(None, 4096)	37,752,832

3. Pipeline

3.1.2 Train model – pipeline 2

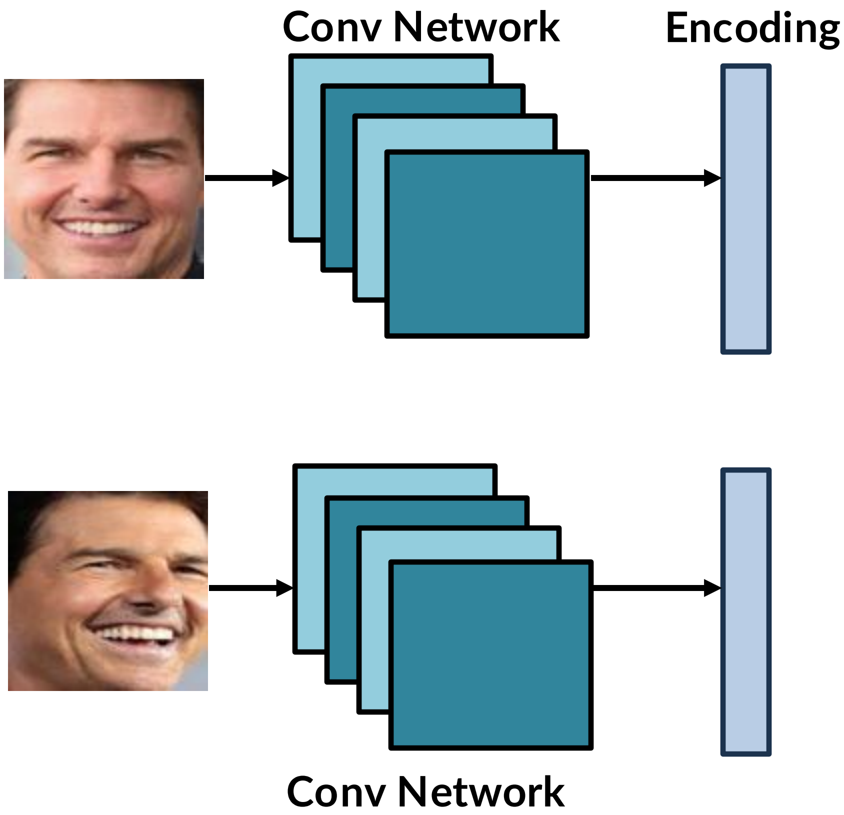
3. Pipeline

3.1.2 Train model – pipeline 2



3. Pipeline

3.1.2 Train model – pipeline 2



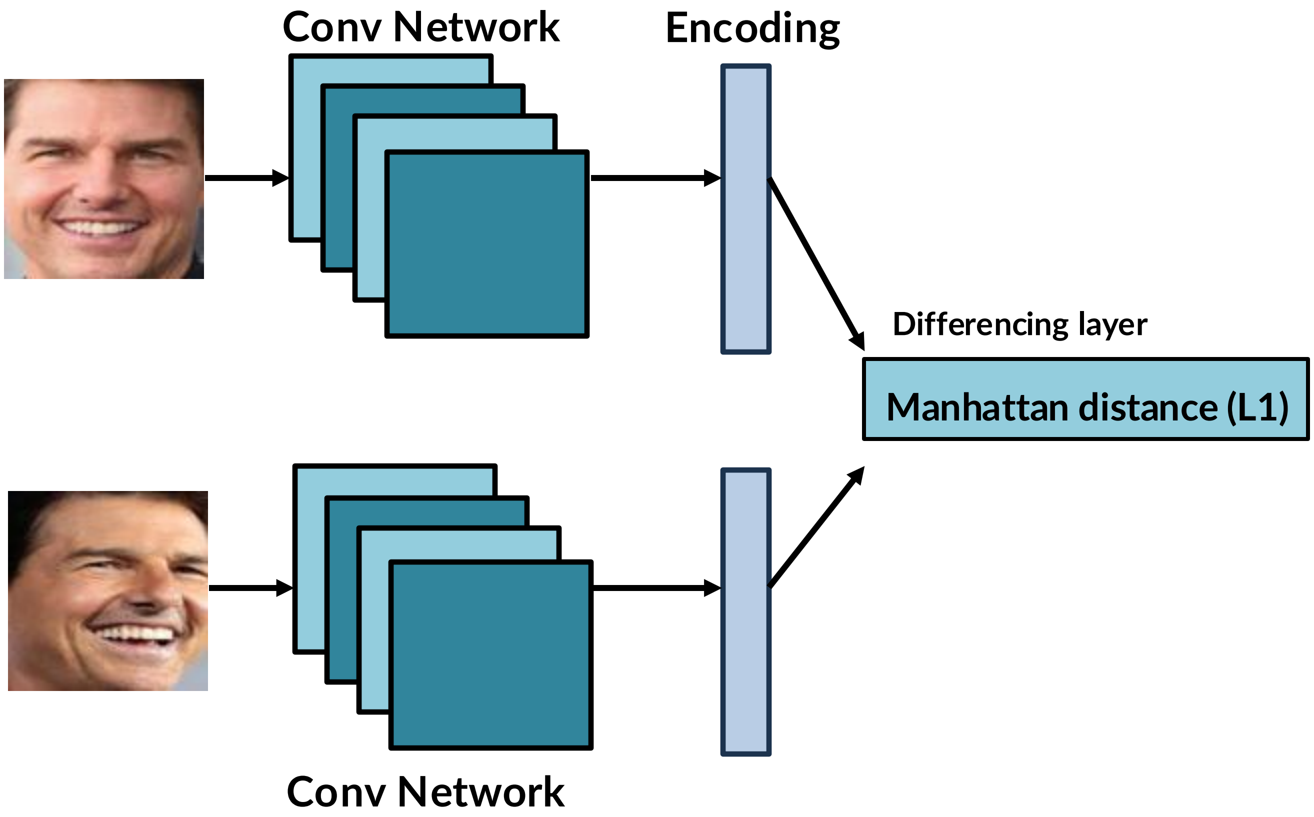
Conv Network

Encoding

Conv Network

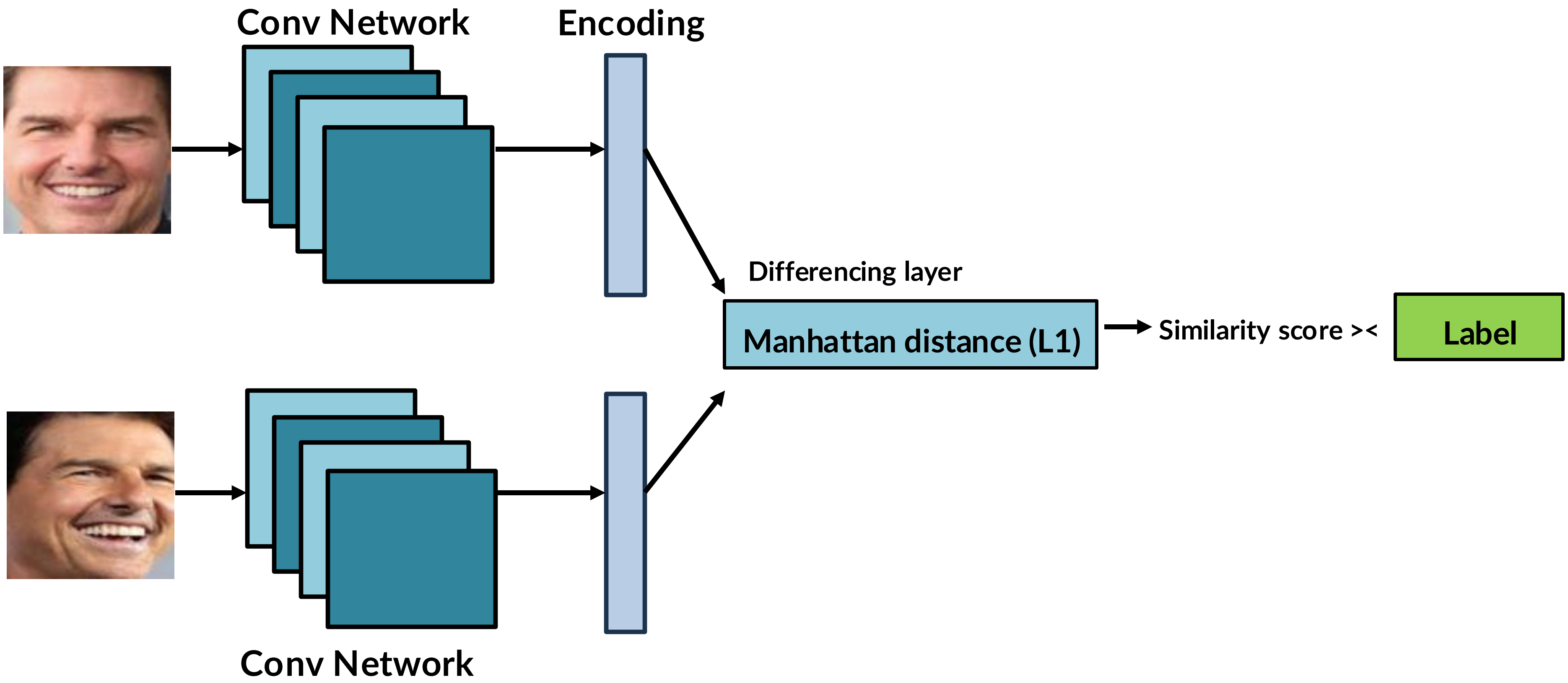
3. Pipeline

3.1.2 Train model – pipeline 2



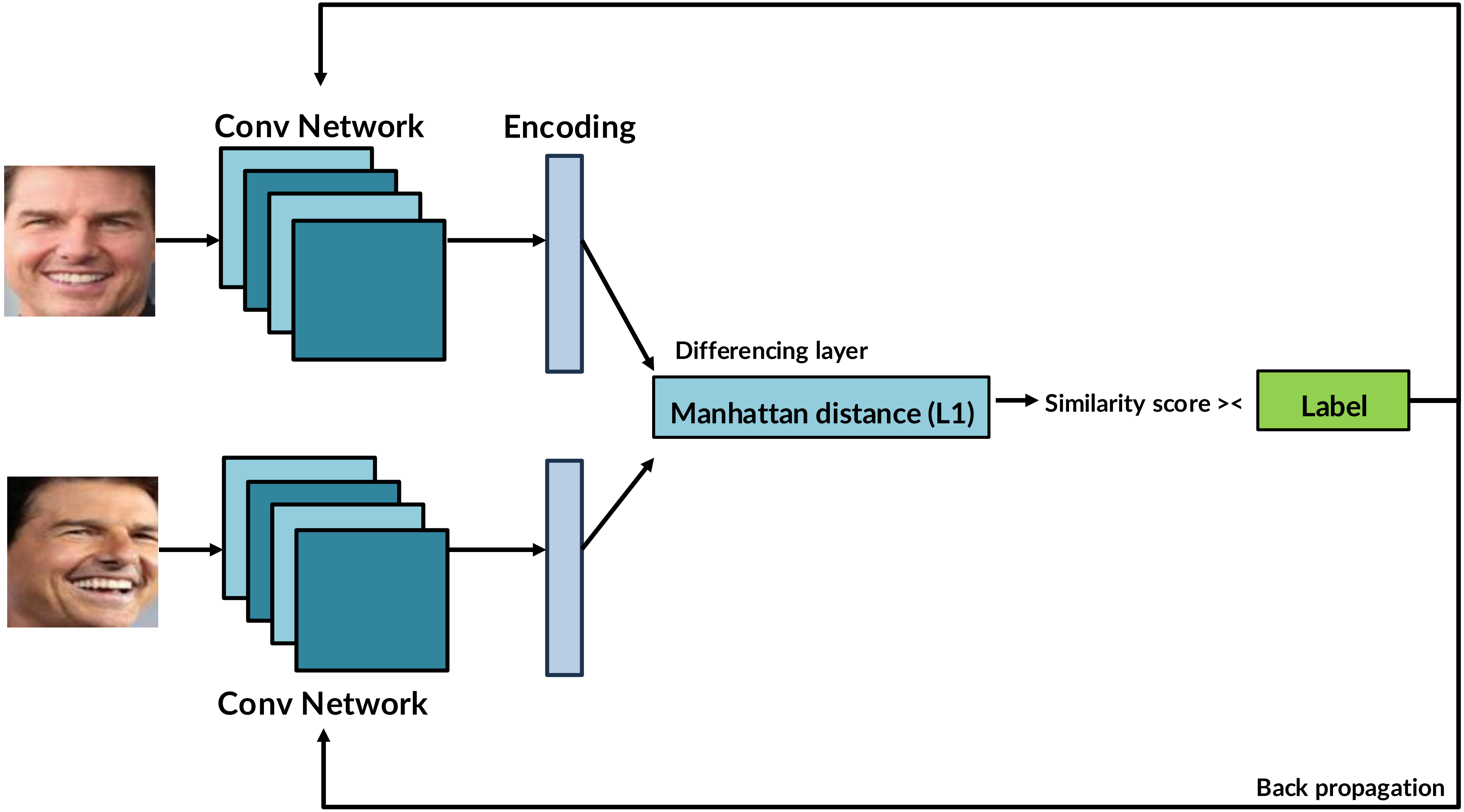
3. Pipeline

3.1.2 Train model – pipeline 2



3. Pipeline

3.1.2 Train model – pipeline 2



3. Pipeline

3.2 Enrollment/ Login

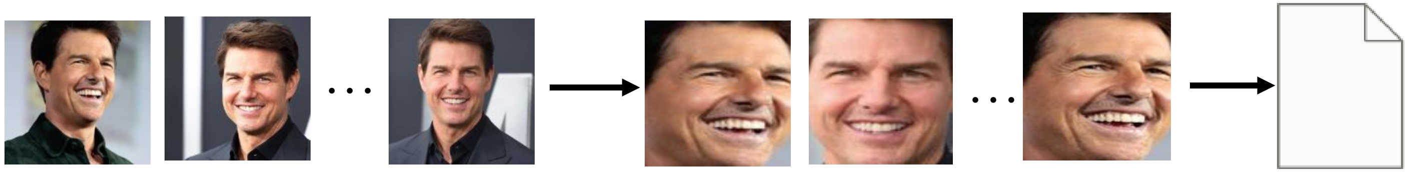
application_data/validation_images/tom cruise

User scan images

3. Pipeline

3.2 Enrollment/ Login

application_data/validation_images/tom cruise



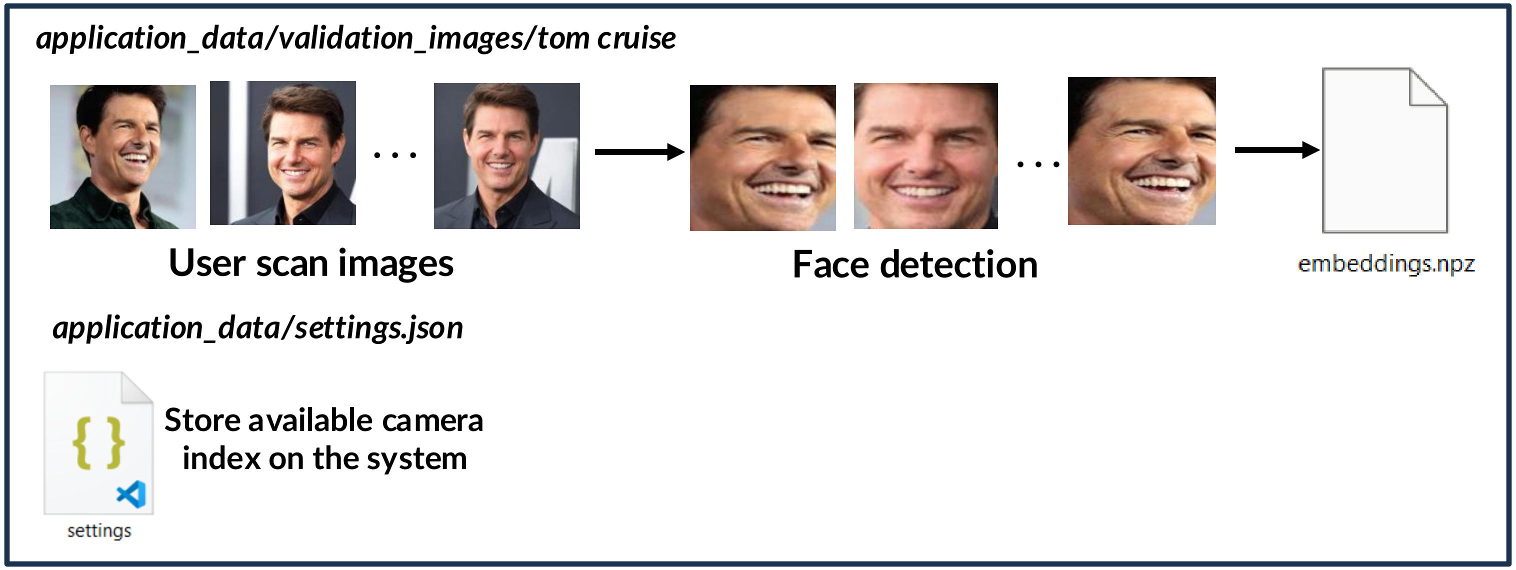
User scan images

Face detection

embeddings.npz

3. Pipeline

3.2 Enrollment/ Login



application_data/validation_images/tom cruise

User scan images

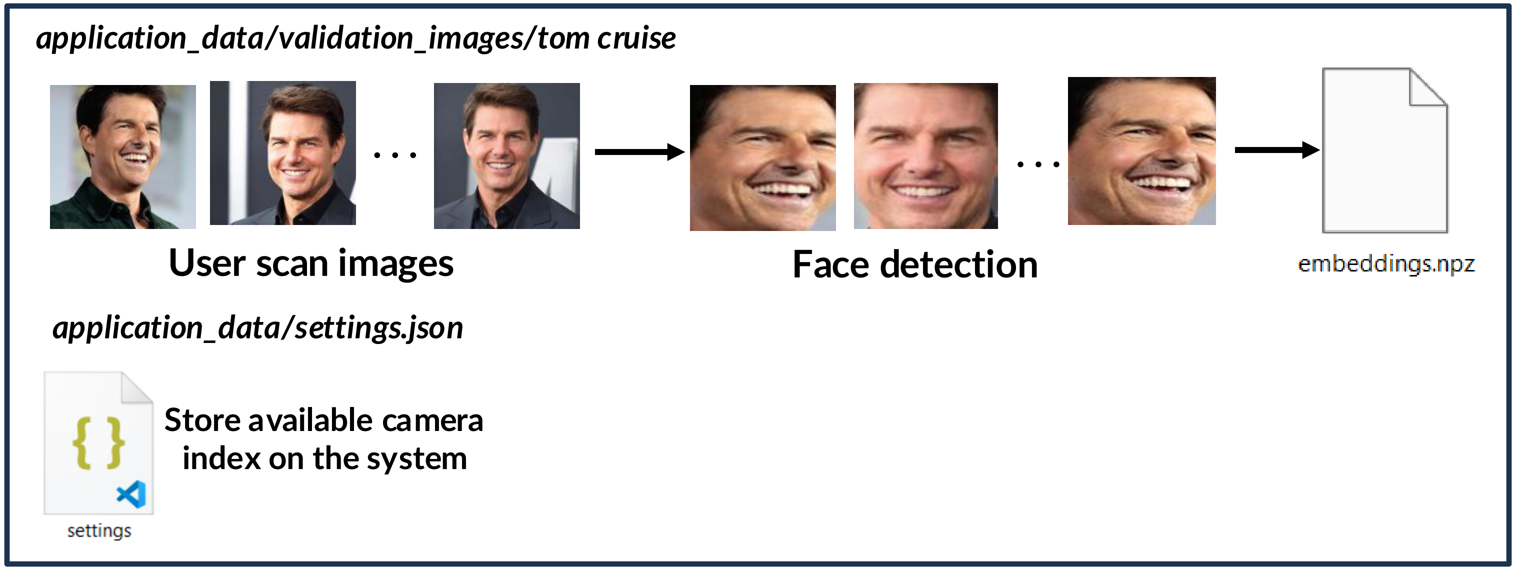
Face detection

application_data/settings.json

Store available camera index on the system

3. Pipeline

3.2 Enrollment/ Login



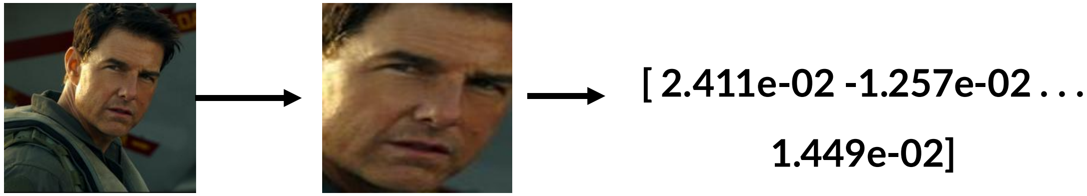
application_data/validation_images/tom cruise

User scan images

Face detection

application_data/settings.json

Store available camera index on the system

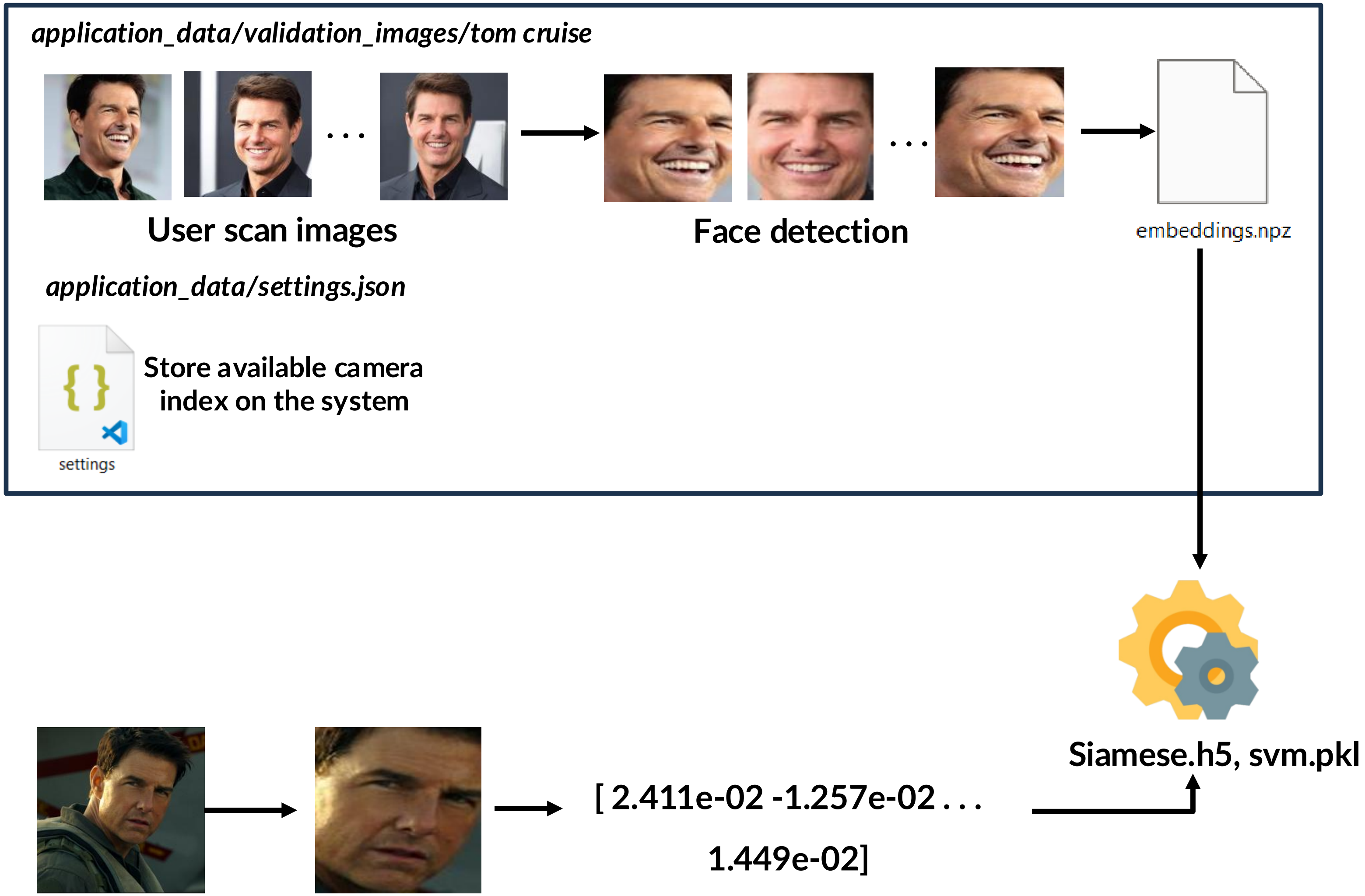


[2.411e-02 -1.257e-02 . . .

1.449e-02]

3. Pipeline

3.2 Enrollment/ Login



application_data/validation_images/tom cruise

User scan images

Face detection

application_data/settings.json

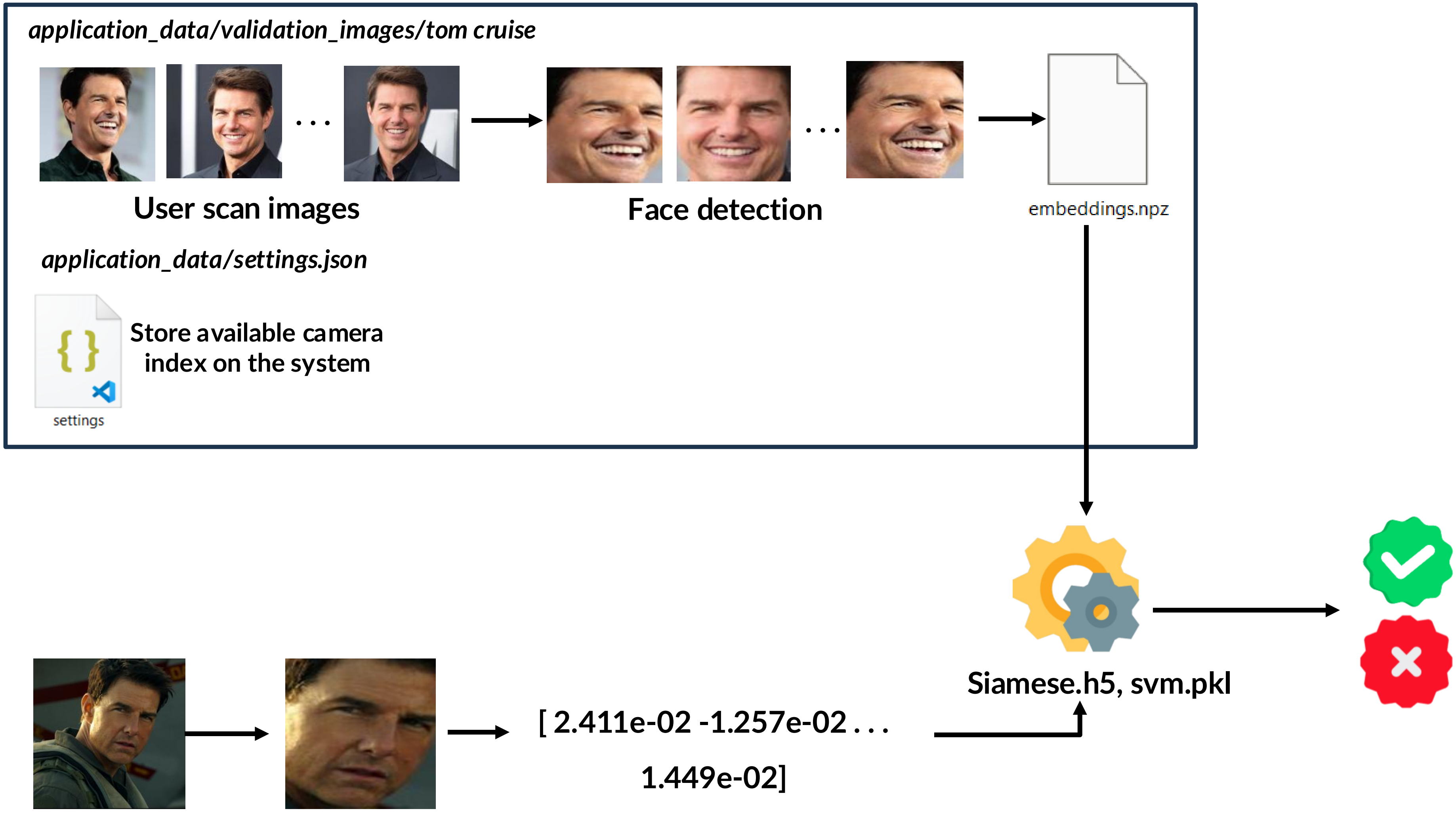
Store available camera index on the system

Siamese.h5, svm.pkl

[2.411e-02 -1.257e-02 . . . 1.449e-02]

3. Pipeline

3.2 Enrollment/ Login



application_data/validation_images/tom cruise

User scan images

Face detection

application_data/settings.json

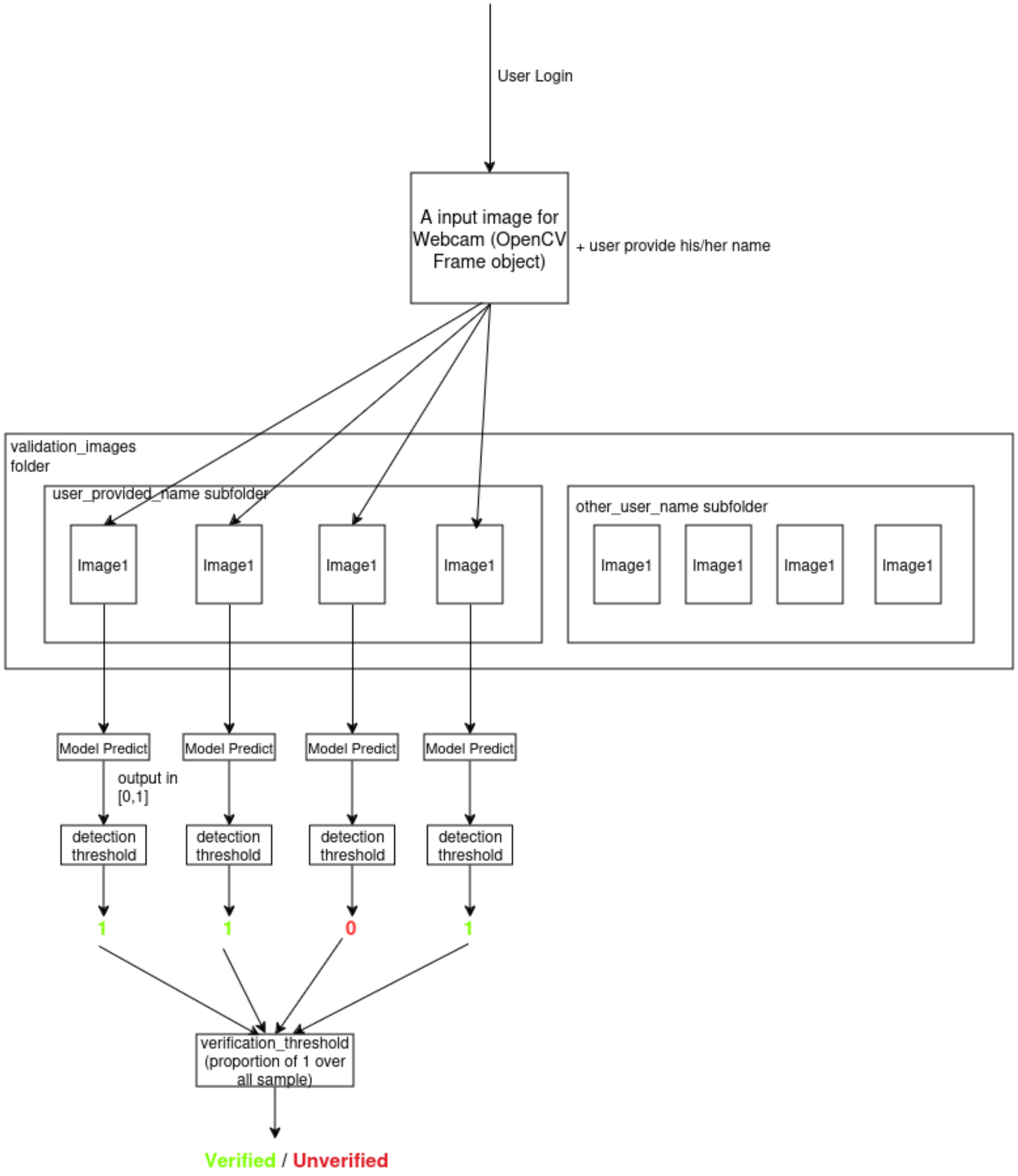
Store available camera index on the system

Siamese.h5, svm.pkl

[2.411e-02 -1.257e-02 . . . 1.449e-02]

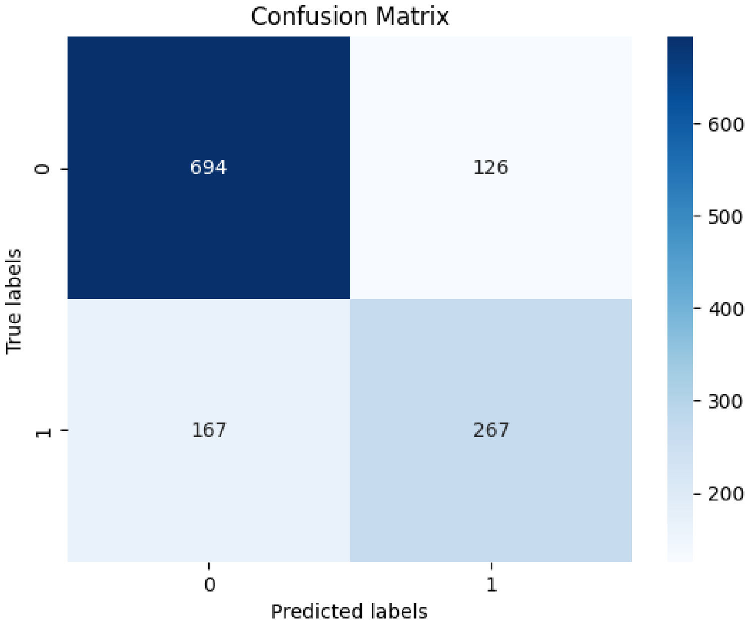
3. Pipeline

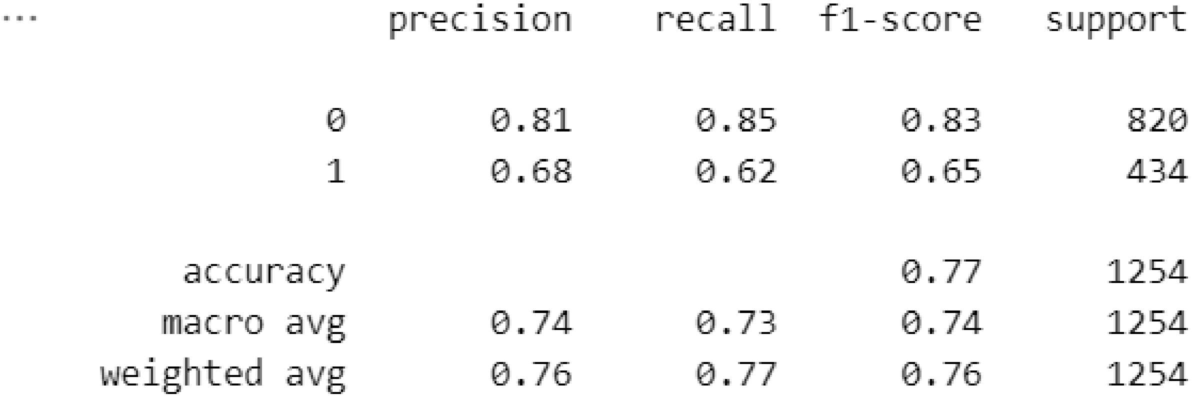
3.2 Enrollment/ Login



4. Evaluation

4.1 Pipeline 1

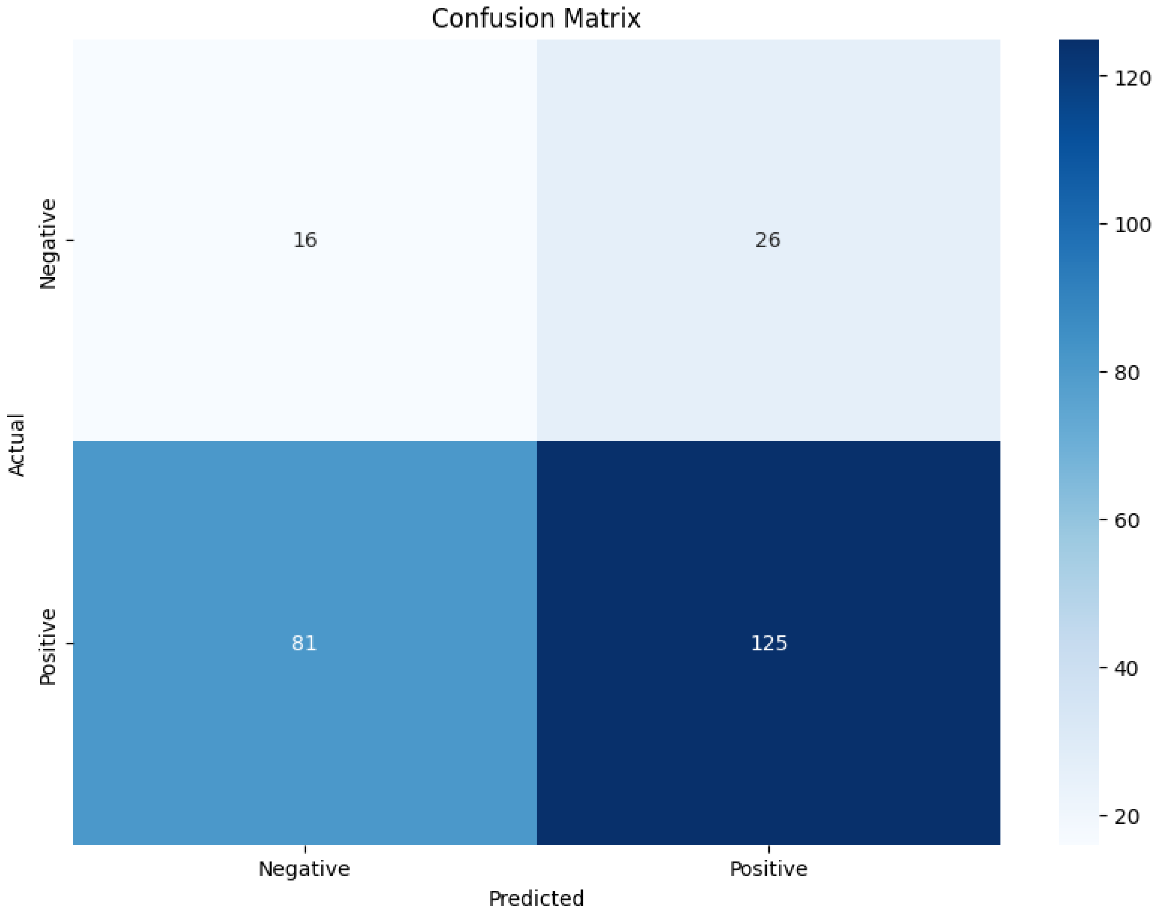


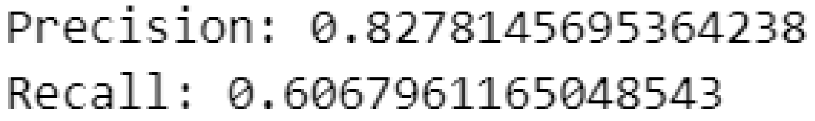


```
...
              precision    recall  f1-score   support

           0       0.81      0.85      0.83       820
           1       0.68      0.62      0.65       434

    accuracy                           0.77      1254
   macro avg       0.74      0.73      0.74      1254
weighted avg       0.76      0.77      0.76      1254
```

4. Evaluation

4.2 Pipeline 2

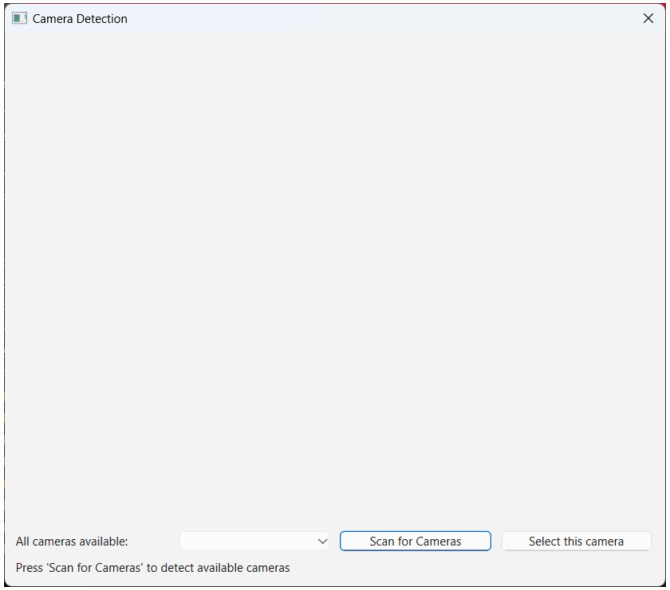




```
Precision: 0.8278145695364238
Recall: 0.6067961165048543
```

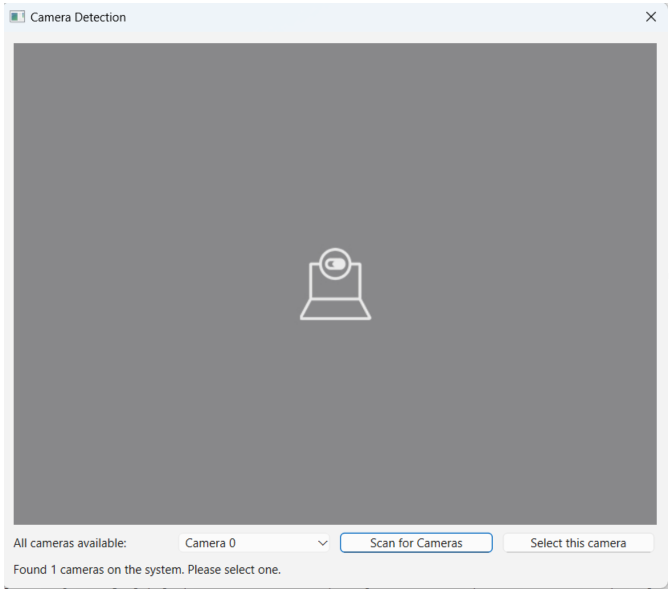
5. Demo

5.1 Camera detection



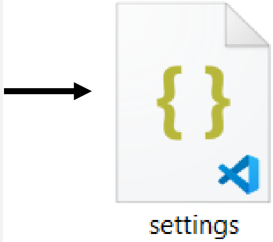
All cameras available:

Press 'Scan for Cameras' to detect available cameras

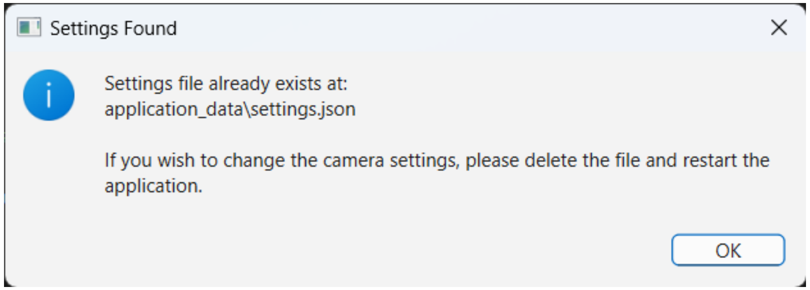


All cameras available: Camera 0

Found 1 cameras on the system. Please select one.



```
{"camera_list": [0,2],
"preferred_camera": 0 }
```



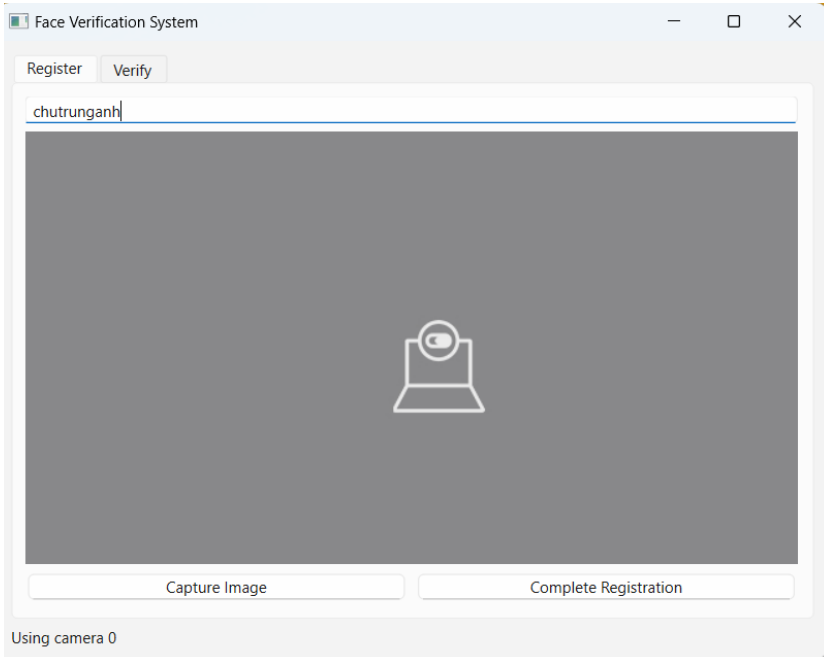
Settings file already exists at:
application_data\settings.json

If you wish to change the camera settings, please delete the file and restart the application.

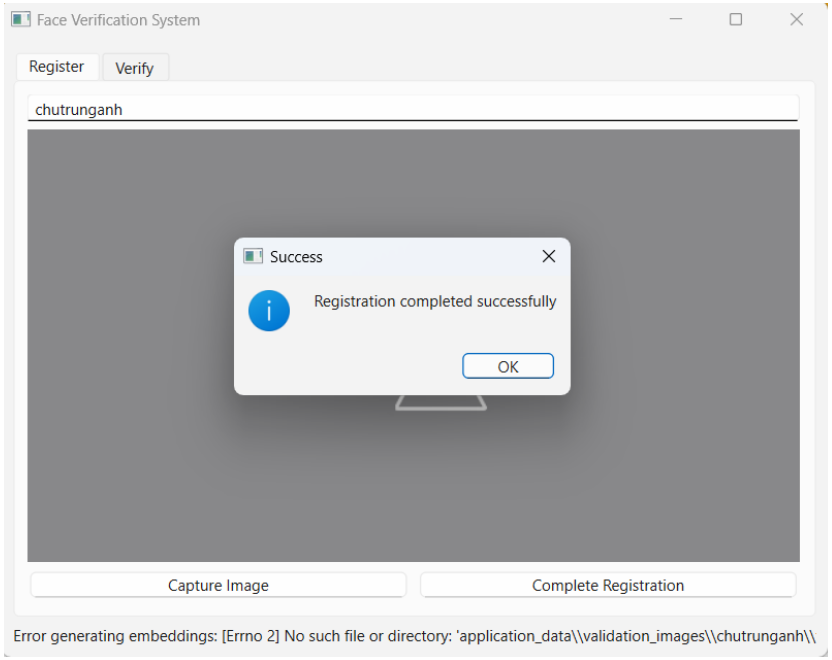
If file settings. json already existed, no not camera detection window

5. Demo

5.2 Enrollment



Using camera 0

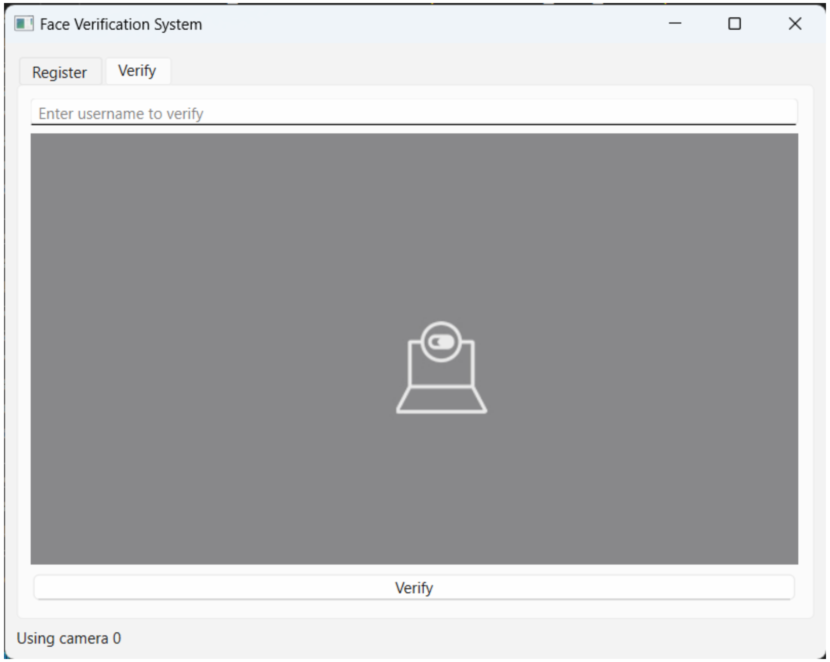


Registration completed successfully

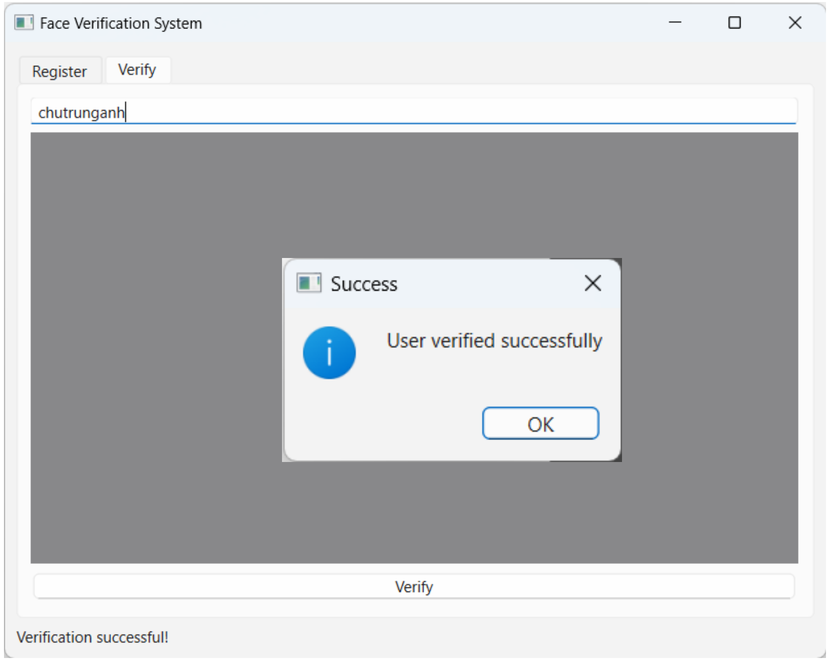
Error generating embeddings: [Errno 2] No such file or directory: 'application_data\\validation_images\\chutrunganh\\

5. Demo

5.3 Login



Using camera 0



chutrunganh

User verified successfully

Verification successful!

6. Future Improvement

- Trying other models to optimize the price per performance (time reduce; requires less powerful resources like camera, processor,...
- Build a more user-friendly UI, easier to interact.
- Deploy using docker, package in .exe, .deb for end-user
- Collect more quality data to reduce biases
- Handle variations like glass, mask, make-up,...
- Quality control: Reject image that's too bright, dark , unable to recognize
- Implement multi –instance verification of login images
- Implement multi –thread for faster response time

HUST

hust.edu.vn fb.com/dhbkhn

Thank you for your listening !