La détection des gens non masqués

Nassih Mohamed

Professeur encadrant : Elouadih Elmostafa

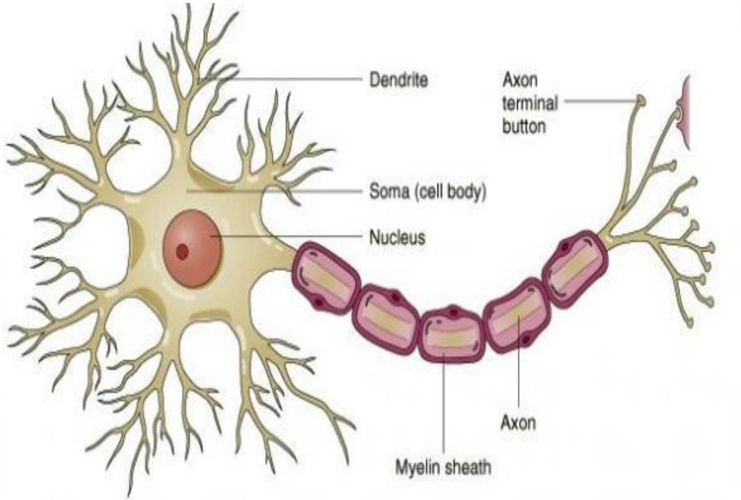
Plan :

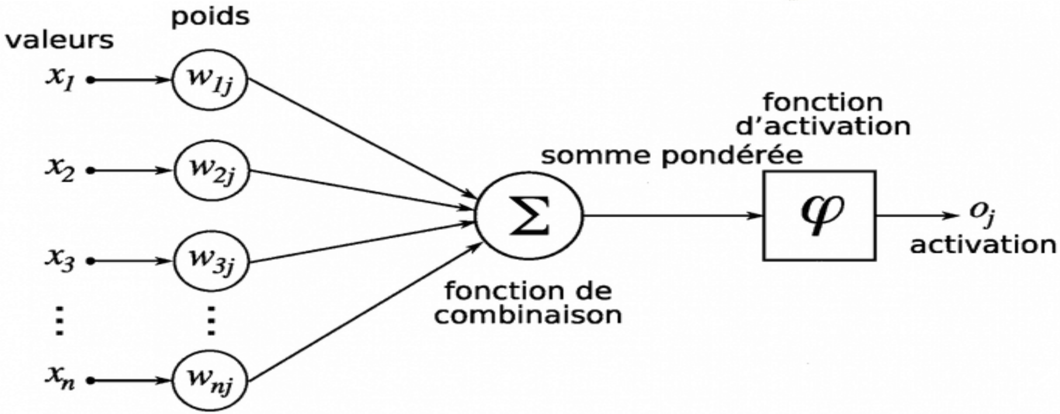
- Problématique
- Réseau neuronal convolutif
- La reconnaissance facial
- La détection de port du masque

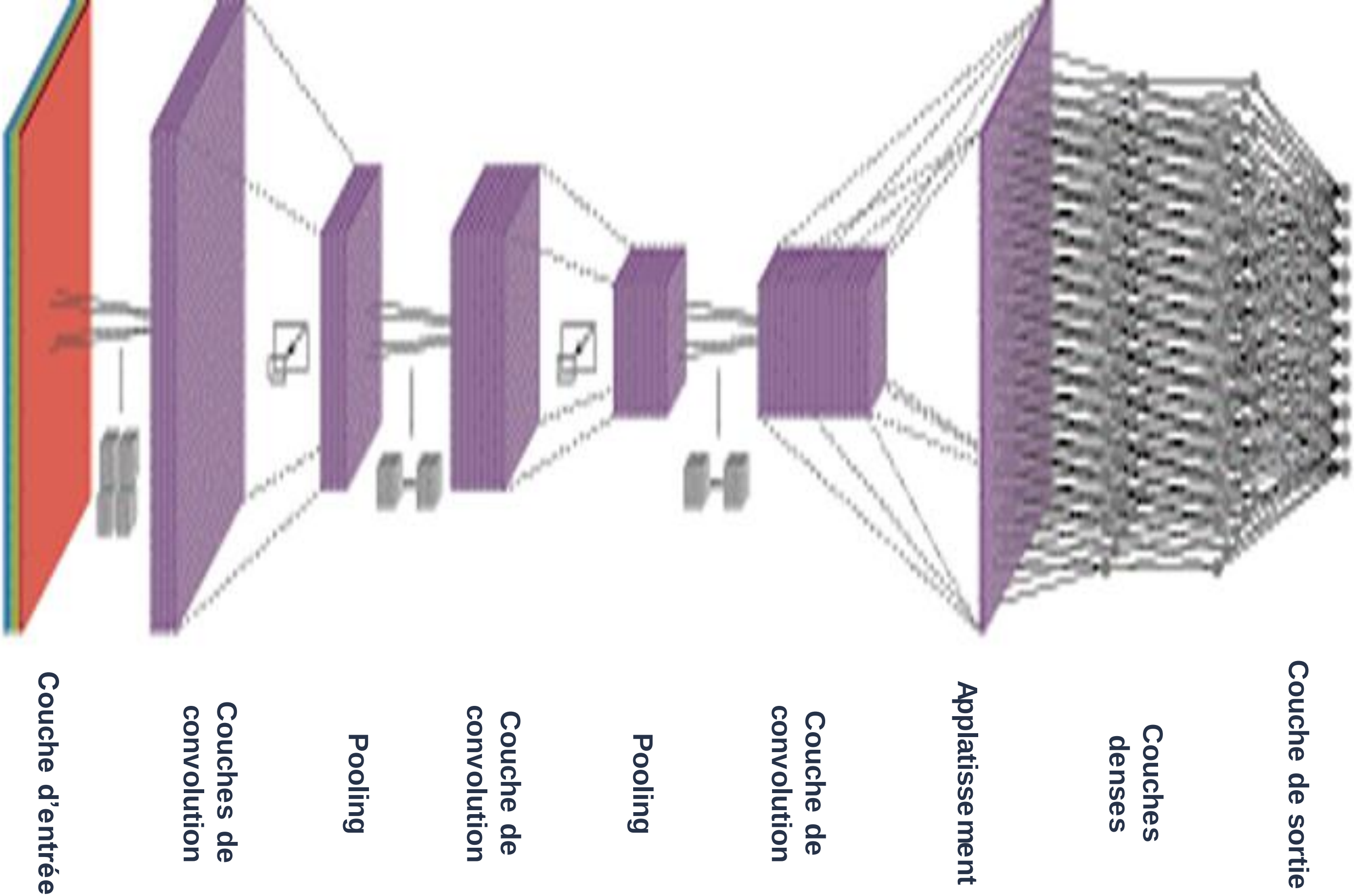
“ Les algorithmes informatiques actuelles développés par les programmeurs n’atteignent pas le niveau de précision élevé pour identifier les gens non masqués dans une grande foule.

1

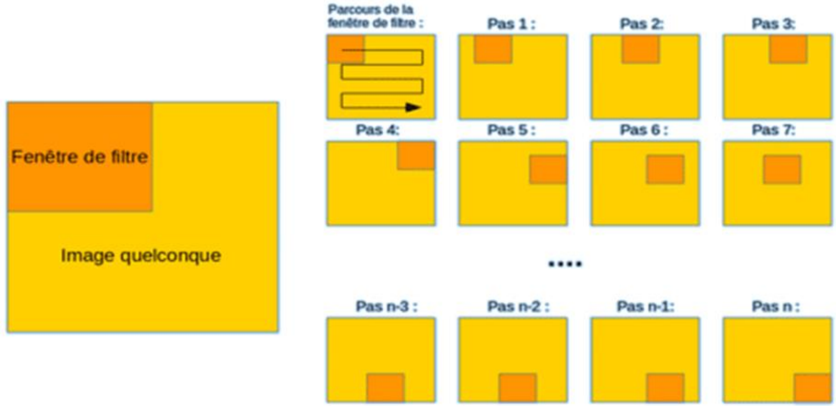
Réseau neuronal convolutif :

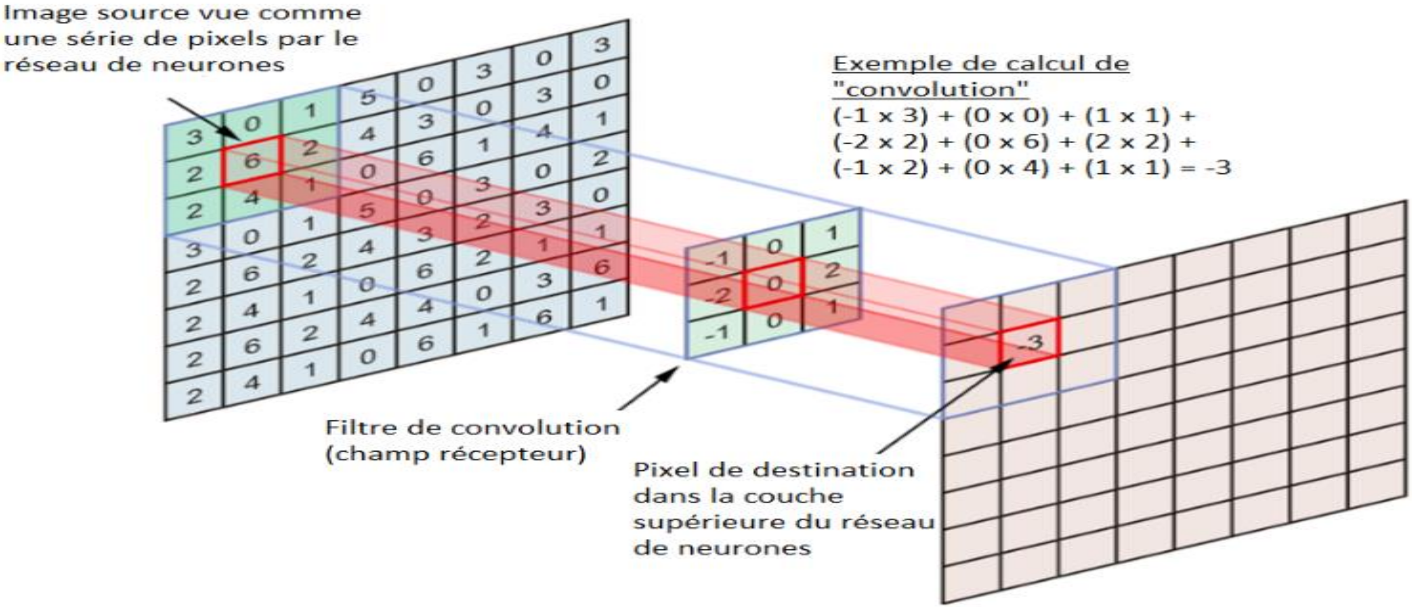

Dendrite
Axon terminal button
Soma (cell body)
Nucleus
Axon
Myelin sheath


poids
valeurs
x_1
w_{1j}
x_2
w_{2j}
x_3
w_{3j}
x_n
w_{nj}
fonction d'activation
somme pondérée
Σ
φ
o_j
activation
fonction de combinaison

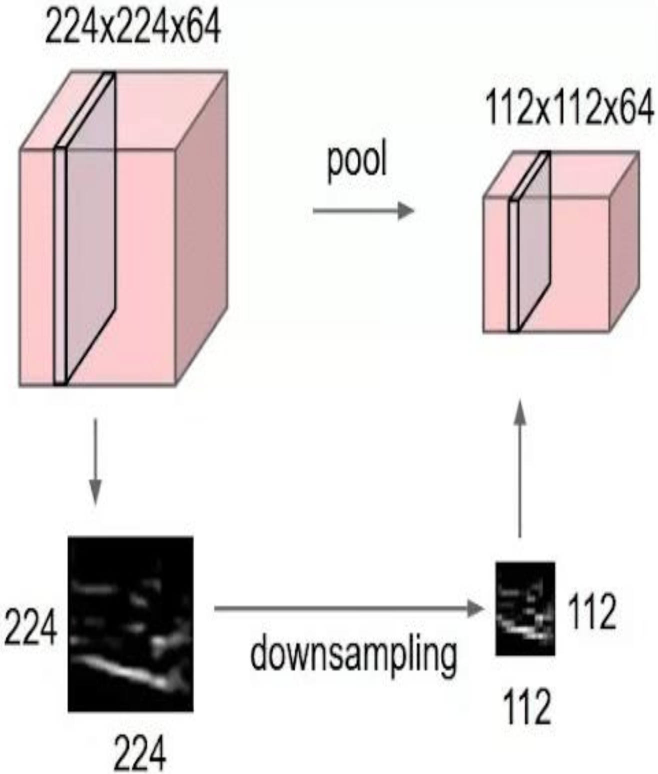

Couche d'entrée
Couches de convolution
Pooling
Couche de convolution
Pooling
Couche de convolution
Applatissement
Couches denses
Couche de sortie

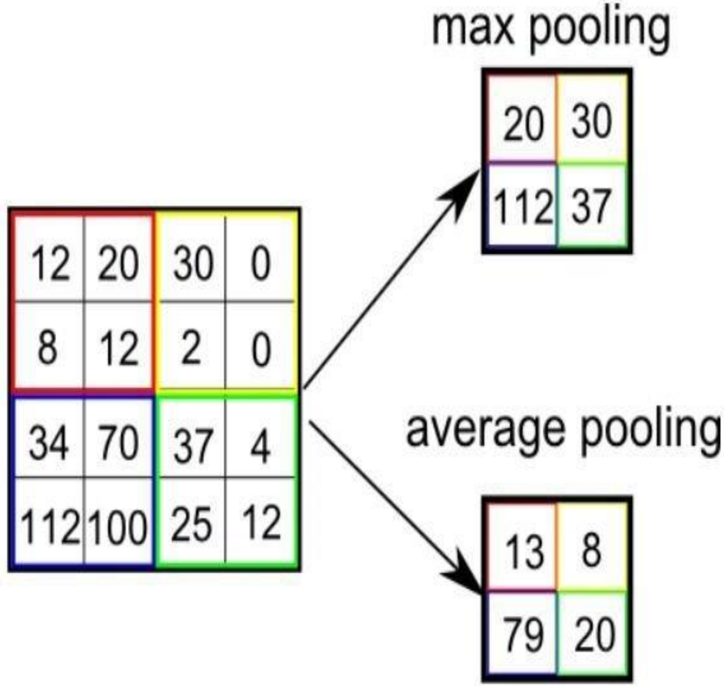
La couche de convolution :



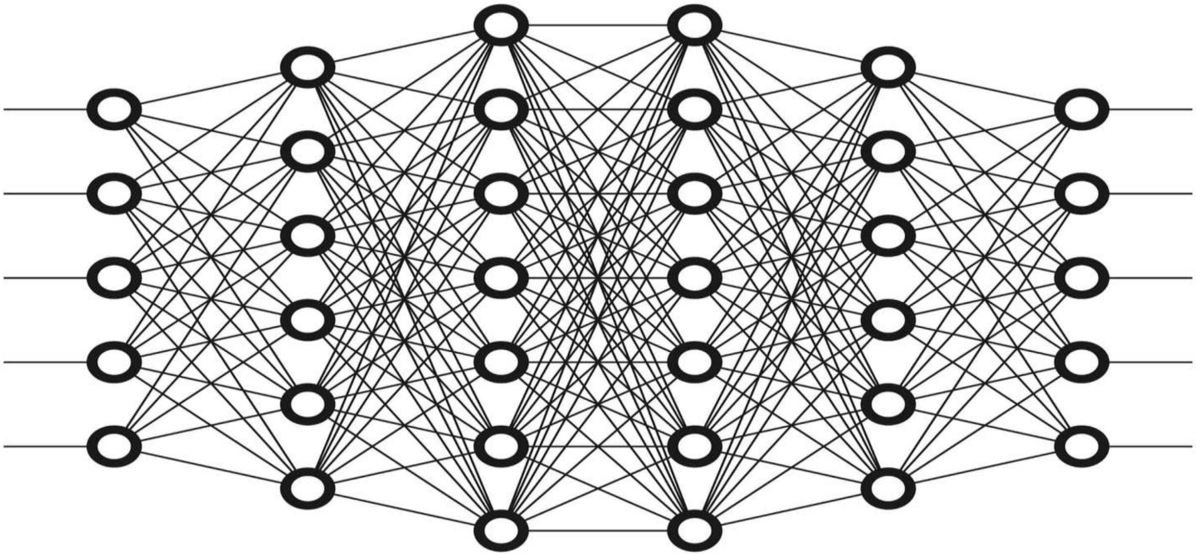


couche d’échantillonage :





Classification :



2

La reconnaissance faciale:

Les images de références :

$$\mathbf{I_i} = \begin{bmatrix} \mathbf{a_{1,1}} & \cdots & \mathbf{a_{1,l}} \\ \vdots & \ddots & \vdots \\ \mathbf{a_{h,1}} & \cdots & \mathbf{a_{h,l}} \end{bmatrix}_{\mathbf{h\ x\ l}} \longrightarrow \mathbf{\Gamma_i} = [\mathbf{a_{1,1}} \quad \cdots \quad \mathbf{a_{h,l}}]_{\mathbf{1xh*l}}$$

$$\downarrow$$

$$\mathbf{\Gamma} = [\mathbf{\Gamma_1}, \cdots\cdots\cdots, \mathbf{\Gamma_M}]_{\mathbf{Mxh*l}}$$

```
from imageio.v2 import imread
import glob
import numpy as np
pngs = glob.glob( './gallery/*.png')
M=len(pngs)
image = imread(pngs[0])
h = len(image)
l= len(image[0])
imgs = np.array([imread(i).flatten()  for i in pngs])
```

Le visage moyen et le visage propre :

$$\boldsymbol{\Gamma} = [\Gamma_1, \cdots \cdots \cdots, \Gamma_M]_{Mxh*l} \longrightarrow \boldsymbol{\Gamma} = \begin{bmatrix} a_{1,1} & \cdots & a_{h,l} \\ \vdots & \ddots & \vdots \\ z_{1,1} & \cdots & z_{h,l} \end{bmatrix}_{Mxh*l}$$

$$\boldsymbol{\Psi} = \frac{1}{M}\sum_{i=1}^{M}\Gamma_i = [m_{1,1} \quad \cdots \quad m_{h,l}]_{1xh*l}$$

```
moyenne = np.mean(imgs,0)
```

$$\phi = \Gamma - \Psi = [\varphi_{i,j}]_{Mxh*l}$$

```
phi = imgs-moyenne
```

La matrice de covariance :

$$C = \phi^T . \phi = [c_{i,j}]_{h*l \text{ x } h*l}$$

Problème:

Si par exemple on a 31 images de résolution 125x150

Alors la matrice de covariance va être de taille 18750x18750

Quelle est la solution?

$$\mathbf{L} = \phi . \phi^{T} = \left[\mathbf{L}_{i,j}\right]_{M \times M}$$

covr = np.dot(phi , phi.transpose())

$$\mathbf{C} . \mathbf{e} = \lambda . \mathbf{e}$$

$$\mathbf{L} . \mathbf{v} = \mu . \mathbf{v}$$

```
from scipy import linalg
vap , vepr = np.linalg.eig(covr)
```

$$\begin{cases} \mathbf{e} = \phi^{T} . \mathbf{v} \\ \lambda = \mu \end{cases}$$

```
vep = np.dot( phi.transpose() , vepr )
```

Démonstration :

$$\mathbf{L}\,.\,\mathbf{v} = \mu.\,\mathbf{v}$$

$$\Rightarrow \boldsymbol{\phi} \cdot \boldsymbol{\phi}^{\mathbf{T}} \cdot \mathbf{v} = \mu.\,\mathbf{v}$$

$$\Rightarrow \boldsymbol{\phi}^{\mathbf{T}}.\,\boldsymbol{\phi}.\,\boldsymbol{\phi}^{\mathbf{T}}.\,\mathbf{v} = \boldsymbol{\phi}^{\mathbf{T}}.\,\mu.\,\mathbf{v}$$

$$\Rightarrow \mathbf{C}.\,\boldsymbol{\phi}^{\mathbf{T}}.\,\mathbf{v} = \boldsymbol{\phi}^{\mathbf{T}}.\,\mu.\,\mathbf{v}$$

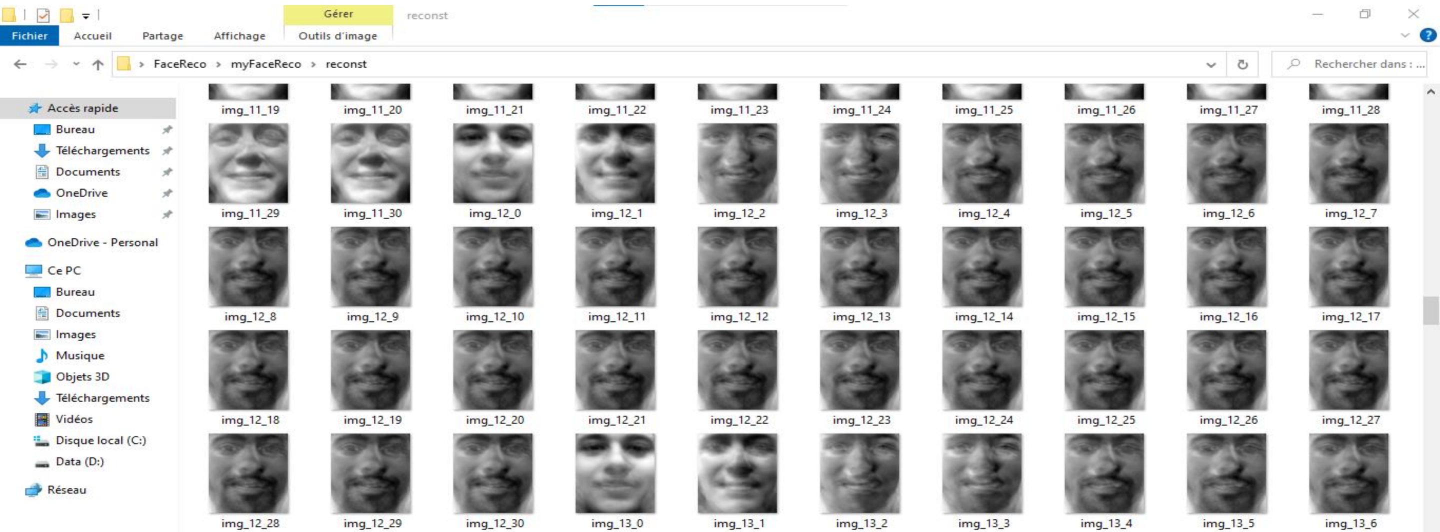
$$\Rightarrow \mathbf{C}.\left(\boldsymbol{\phi}^{\mathbf{T}}.\,\mathbf{v}\right) = \mu.\left(\boldsymbol{\phi}^{\mathbf{T}}.\,\mathbf{v}\right)$$

$$\mathbf{e} = \boldsymbol{\phi}^{\mathbf{T}}.\,\mathbf{v} = \left[\mathbf{e}_{\mathbf{i},\mathbf{j}}\right]_{\mathbf{h}*\mathbf{l}\;\mathbf{x}\;\mathbf{M}}$$

Le poids :

$$\Omega = \phi \, . \, e = [\Omega_{i,j}]_{MxM}$$

```
poids = np.dot( phi , vep )
```




```
for k in range(M):
    for i in range(M):
        recon = moyenne + np.dot( poids[k,:i] , vectp[:,:i].transpose() )
        imsave("reconst/img_"+str(k)+"_"+str(i)+".png", recon.reshape( h, l) )
```

L'identification d'une personne :

$$\Gamma = [\tau_i]_{1 \times h*l}$$

```
import time
import matplotlib.pyplot as plt
t1 = time.time()
image_a_trouver = np.array ( imread( "a_tester/1.png" ).flatten() )
a_trouver = plt.figure (1)
plt.imshow ( image_a_trouver.reshape ( h, l ) )
plt.title ( "A trouver")
```

$$\phi_\Gamma = \Gamma - \Psi = [\varphi_i]_{1x\, h*l}$$

```
phi2 = image_a_trouver - moyenne
```

$$\Omega_\Gamma = \phi_\Gamma \cdot e = [\omega_i]_{1x\, M}$$

```
poids2= np.dot(phi2,vectp)
```

La distance euclidienne :

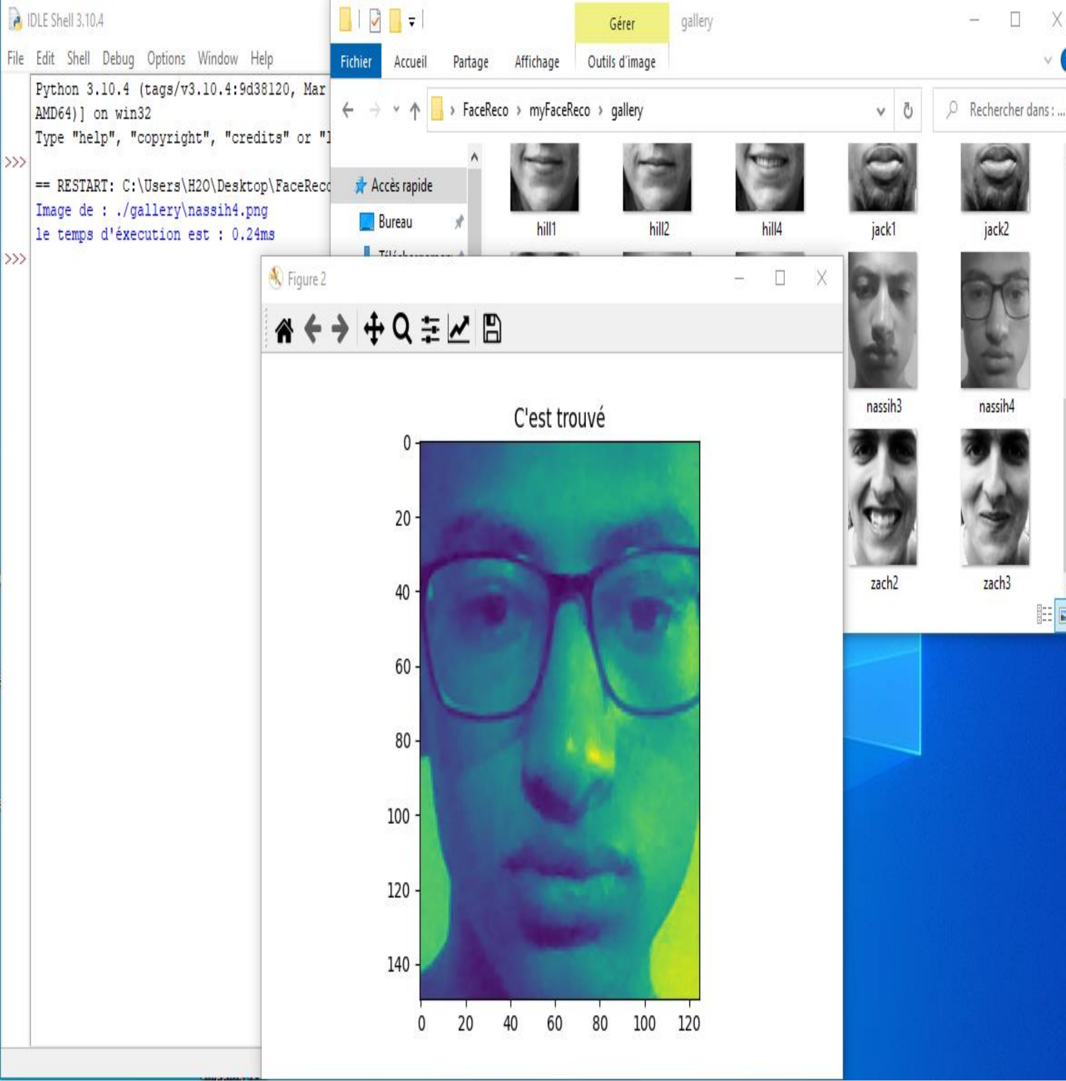
$$\mathcal{E} = \|\boldsymbol{\Omega} - \boldsymbol{\Omega}_{\boldsymbol{\Gamma}}\|^2 = [\mathcal{E}_i]_{\mathbf{1\ x\ M}}$$

```
dist = np.min( (poids - poids2)**2, axis=1 )
```

$$\mathcal{E}_{\mathbf{k}} = \min_{\mathbf{i}\in[\![\mathbf{1},\mathbf{M}]\!]} \mathcal{E}_{\mathbf{i}}$$

```
indiceImg = np.argmin( dist )
```

```
print( "Image de : "+ pngs[indiceImg] )
trouvé = plt.figure(2)
plt.imshow( imgs[indiceImg].reshape( h, l))
plt.title("C'est trouvé")
trouvé.show()
t2 = time.time()
t = t2 - t1
print(f'le temps d\'éxecution est : {t:.2}ms')
```

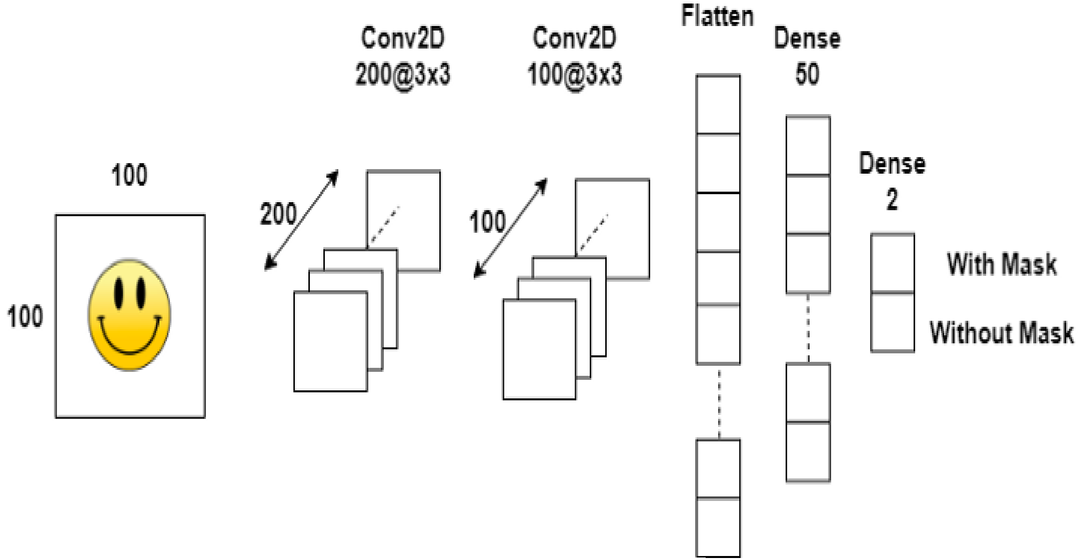

IDLE Shell 3.10.4
File Edit Shell Debug Options Window Help
Python 3.10.4 (tags/v3.10.4:9d38120, Mar
AMD64)] on win32
Type "help", "copyright", "credits" or "l
>>>
== RESTART: C:\Users\H20\Desktop\FaceReco
Image de : ./gallery\nassih4.png
le temps d'éxecution est : 0.24ms
>>>
Gérer
gallery
Fichier
Accueil
Partage
Affichage
Outils d'image
FaceReco › myFaceReco › gallery
Rechercher dans : ...
Accès rapide
Bureau
hill1
hill2
hill4
jack1
jack2
nassih3
nassih4
zach2
zach3
Figure 2
C'est trouvé
0
20
40
60
80
100
120
140
0
20
40
60
80
100
120

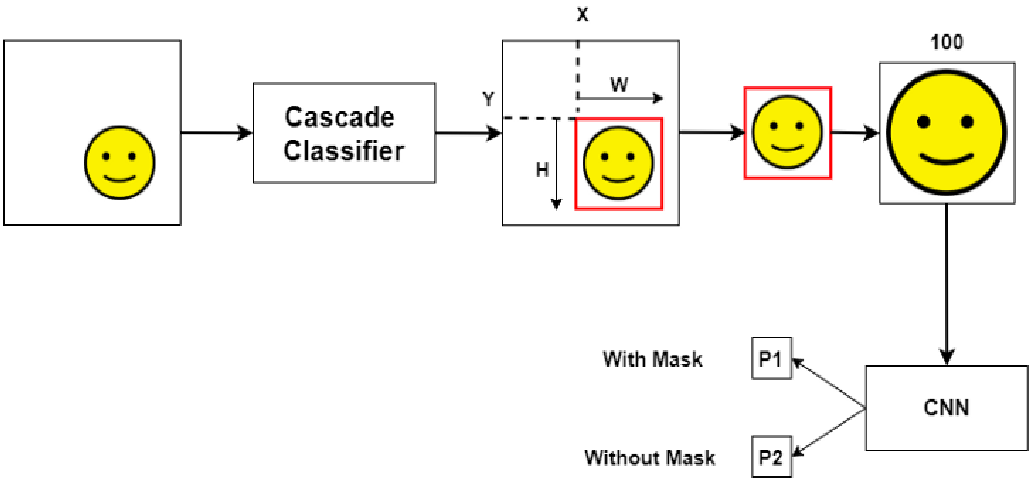
"1.png"

3

La détection du port de masque

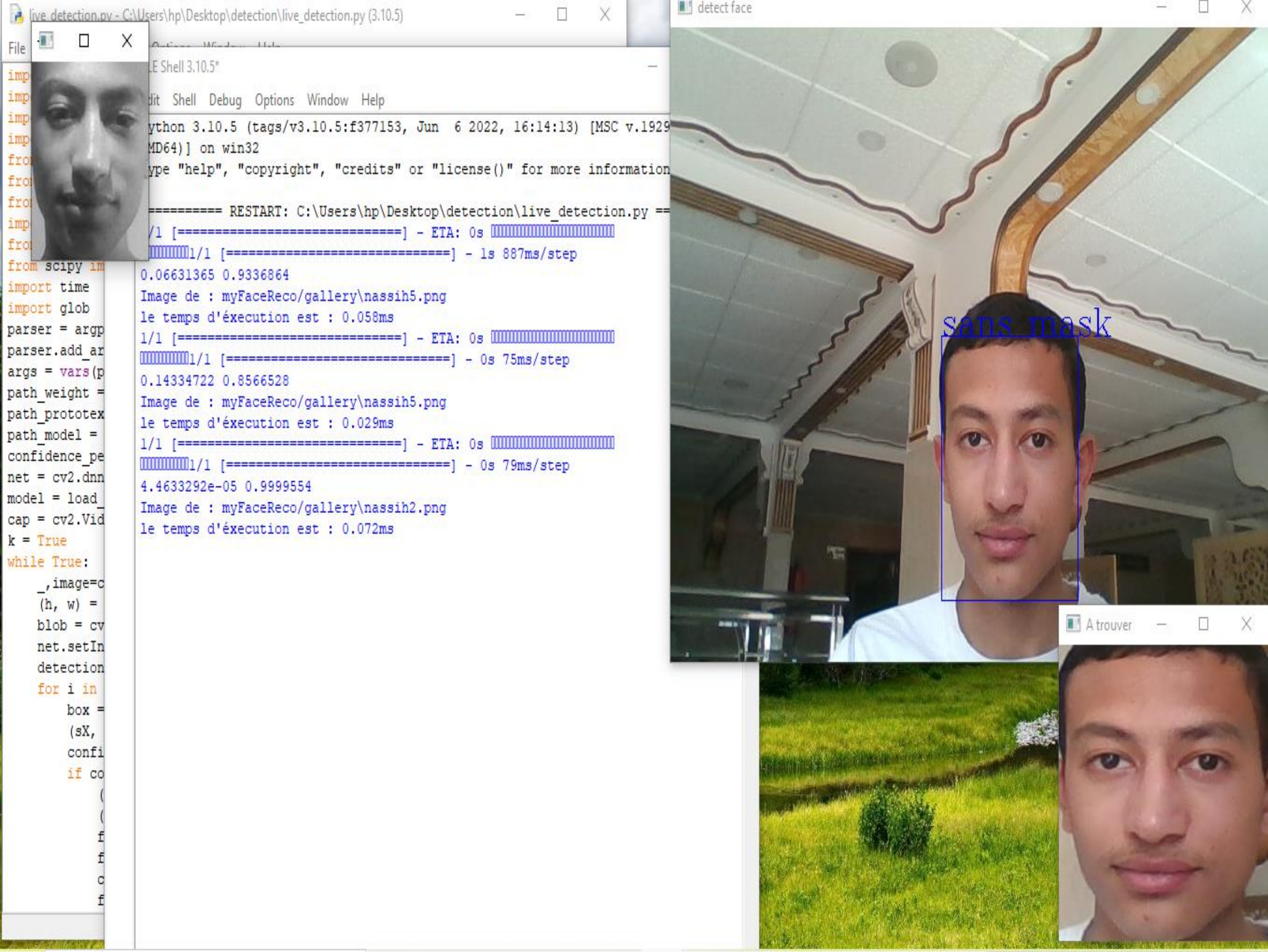
Dataset :

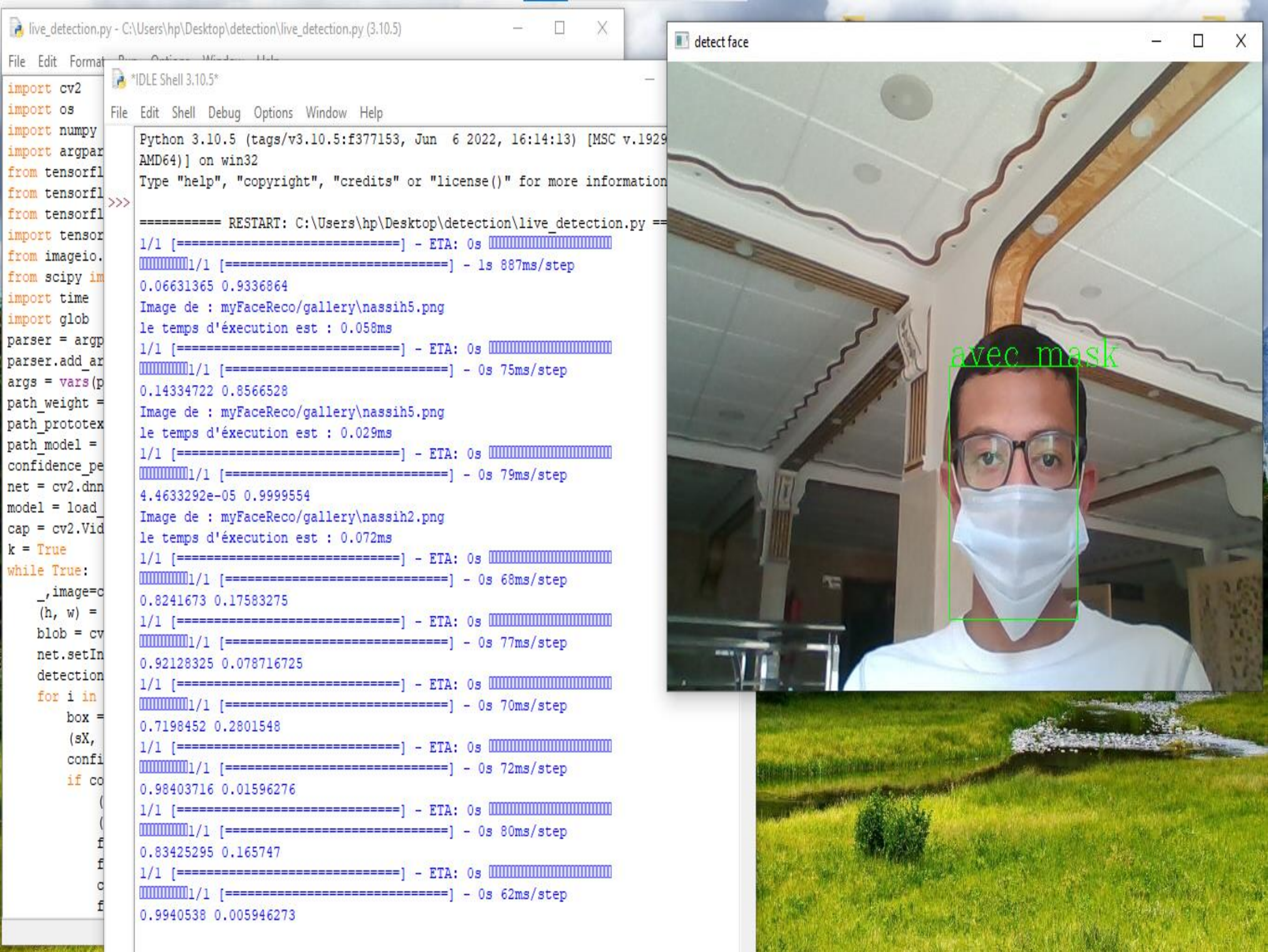

Conv2D
200@3x3
Conv2D
100@3x3
Flatten
Dense
50
Dense
2
100
100
200
100
With Mask
Without Mask


Cascade
Classifier
X
Y
W
H
100
With Mask
Without Mask
P1
P2
CNN

```
from tensorflow import keras

model = keras.models.Sequential()
model.add( keras.layers.Input(  ( 100, 100, 1)))

model.add( keras.layers.Conv2D(  200, ( 3, 3), activation='relu'))
model.add( keras.layers.MaxPooling2D(  ( 2, 2)))
model.add( keras.layers.Dropout(  0.2))

model.add( keras.layers.Conv2D(  100, ( 3, 3), activation='relu'))
model.add( keras.layers.MaxPooling2D(  ( 2, 2)))
model.add( keras.layers.Dropout(  0.2))

model.add( keras.layers.Flatten())
model.add( keras.layers.Dense(  50, activation='relu'))
model.add( keras.layers.Dropout(  0.5))

model.add( keras.layers.Dense(  2, activation='softmax'))

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 98, 98, 200)       2000

 max_pooling2d (MaxPooling2D  (None, 49, 49, 200)      0
 )

 dropout (Dropout)           (None, 49, 49, 200)       0

 conv2d_1 (Conv2D)           (None, 47, 47, 100)       180100

 max_pooling2d_1 (MaxPooling  (None, 23, 23, 100)      0
 2D)

 dropout_1 (Dropout)         (None, 23, 23, 100)       0

 flatten (Flatten)           (None, 52900)             0

 dense (Dense)               (None, 50)                2645050

 dropout_2 (Dropout)         (None, 50)                0

 dense_1 (Dense)             (None, 2)                 102

=================================================================
Total params: 2,827,252
Trainable params: 2,827,252
Non-trainable params: 0
_________________________________________________________________
```

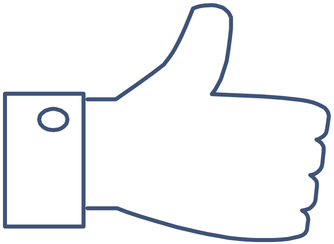

live_detection.py - C:\Users\hp\Desktop\detection\live_detection.py (3.10.5)
detect face
File
LE Shell 3.10.5*
dit Shell Debug Options Window Help
ython 3.10.5 (tags/v3.10.5:f377153, Jun 6 2022, 16:14:13) [MSC v.1929
MD64)] on win32
ype "help", "copyright", "credits" or "license()" for more information
========== RESTART: C:\Users\hp\Desktop\detection\live_detection.py ==
/1 [==============================] - ETA: 0s
1/1 [==============================] - 1s 887ms/step
0.06631365 0.9336864
Image de : myFaceReco/gallery\nassih5.png
le temps d'éxecution est : 0.058ms
1/1 [==============================] - ETA: 0s
1/1 [==============================] - 0s 75ms/step
0.14334722 0.8566528
Image de : myFaceReco/gallery\nassih5.png
le temps d'éxecution est : 0.029ms
1/1 [==============================] - ETA: 0s
1/1 [==============================] - 0s 79ms/step
4.4633292e-05 0.9999554
Image de : myFaceReco/gallery\nassih2.png
le temps d'éxecution est : 0.072ms
from scipy im
import time
import glob
parser = argp
parser.add_ar
args = vars(p
path_weight =
path_prototex
path_model =
confidence_pe
net = cv2.dnn
model = load_
cap = cv2.Vid
k = True
while True:
_,image=c
(h, w) =
blob = cv
net.setIn
detection
for i in
box =
(sX,
confi
if co
sans mask
A trouver



live_detection.py - C:\Users\hp\Desktop\detection\live_detection.py (3.10.5)
File Edit Format
import cv2
import os
import numpy
import argpar
from tensorfl
from tensorfl
from tensorfl
import tensor
from imageio.
from scipy im
import time
import glob
parser = argp
parser.add_ar
args = vars(p
path_weight =
path_prototex
path_model =
confidence_pe
net = cv2.dnn
model = load_
cap = cv2.Vid
k = True
while True:
_,image=c
(h, w) =
blob = cv
net.setIn
detection
for i in
box =
(sX,
confi
if co
IDLE Shell 3.10.5
File Edit Shell Debug Options Window Help
Python 3.10.5 (tags/v3.10.5:f377153, Jun 6 2022, 16:14:13) [MSC v.1929
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information
>>>
=========== RESTART: C:\Users\hp\Desktop\detection\live_detection.py ==
1/1 [==============================] - ETA: 0s
1/1 [==============================] - 1s 887ms/step
0.06631365 0.9336864
Image de : myFaceReco/gallery\nassih5.png
le temps d'éxecution est : 0.058ms
1/1 [==============================] - ETA: 0s
1/1 [==============================] - 0s 75ms/step
0.14334722 0.8566528
Image de : myFaceReco/gallery\nassih5.png
le temps d'éxecution est : 0.029ms
1/1 [==============================] - ETA: 0s
1/1 [==============================] - 0s 79ms/step
4.4633292e-05 0.9999554
Image de : myFaceReco/gallery\nassih2.png
le temps d'éxecution est : 0.072ms
1/1 [==============================] - ETA: 0s
1/1 [==============================] - 0s 68ms/step
0.8241673 0.17583275
1/1 [==============================] - ETA: 0s
1/1 [==============================] - 0s 77ms/step
0.92128325 0.078716725
1/1 [==============================] - ETA: 0s
1/1 [==============================] - 0s 70ms/step
0.7198452 0.2801548
1/1 [==============================] - ETA: 0s
1/1 [==============================] - 0s 72ms/step
0.98403716 0.01596276
1/1 [==============================] - ETA: 0s
1/1 [==============================] - 0s 80ms/step
0.83425295 0.165747
1/1 [==============================] - ETA: 0s
1/1 [==============================] - 0s 62ms/step
0.9940538 0.005946273
detect face
avec mask

"

Conclusion :



Merci beaucoup pour votre attention