

Rapport de stage d'ingénieur

RAPPORT DE STAGE



SFM TELECOM

Projet de reconnaissance faciale

Elaboré par :

Chachia Mohamed Wacef

Encadré par :

Mr Ben Driss Karem

Rapport de stage d'ingénieur

Remerciements

Avant tout développement sur cette expérience professionnelle, il semble opportun de commencer ce rapport de stage en remerciant ceux qui m'ont beaucoup appris durant ce stage.

En hommage à leur sympathie, je tiens à remercier chaleureusement tous les managers et membres de SFM pour leur accueil chaleureux et leurs multitudes d'assistants avec beaucoup de sincérité et de gratitude et qui m'ont donné la force de rendre justice à mon travail et à contribuer de mon mieux pour en faire un succès.

J'exprime également ma gratitude à Mon encadreur Monsieur Ben Driss Karem qui m'a énormément aidé à améliorer mes connaissances et qui m'a fait bénéficier de ses connaissances et compétences.

Pour ma part, j'espère que mon pilotage et mon apprentissage ont laissé une bonne impression de SUP'COM et affirment son image et sa marque.

Je termine ces remerciements en saluant chaleureusement les membres du jury pour l'honneur qu'ils font en acceptant de juger ce modeste ouvrage.

Rapport de stage d'ingénieur

SOMMAIRE

.....	1
Remerciements	2
Introduction générale.....	5
CHAPITRE 1 : Aperçu du projet	6
1.1 Contexte du sujet	6
1.2 Présentation de l'entreprise	6
1.3 Les atouts de SFM.....	7
1.4 Les références de SFM	8
1.5 Les partenaires	8
1.6Environnement du travail	8
CHAPITRE 2 : La reconnaissance faciale.....	9
2.1 Introduction	9
2.2 Définitions	9
2.3 Utilités	10
2.4 Avantages	11
2.5 Désavantages	11
2.6 Probèmes	11
2.7 Conclusion	12
CHAPITRE 3 : Image numérique	13
3.1 Introduction	13
3.2Traitement d'image avec Python	14
CHAPITRE 4 : Détection de visage	19
4.1 Introduction	19
4.2 Haar face based cascade classificateur.....	19
4.2.1 Calcul des caractéristiques de Haar	20
4.2.2 Création d'images d'intégrales	20
4.2.3 Utilisation d'Adaboost	21
4.2.4 Implémentation des classifieurs en cascade.....	21
4.2.5 Code	22
4.2.6 Conclusion	22
4.3 CNN face detector	23
4.3.1 Introduction	23
4.3.2 Les avantages	23

Rapport de stage d'ingénieur

4.3.3 Les désavantages	23
4.3.4 Code	23
4.4 MTCNN	24
4.4.1 Introduction	25
4.4.2 Explication	25
4.4.3 Code	26
4.5 DNN face detector in OpenCV	26
4.5.1 Introduction	27
4.5.2 Code	27
4.6 Fréquence d'images	27
4.7 Conclusion	28
CHAPITRE 5 : Apprentissage profond	29
5.1 Introduction	29
5.2 CNN	29
5.2.1 Introduction	29
5.2.2 Couche de convolution	30
5.2.3 Couche de mise en commun	31
5.2.4 Couche entièrement connecté	31
5.2.5 Couche de non linéarité	32
5.2.6 Conclusion	33
5.3 Apprentissage par transfert	33
5.3.1 Introduction	33
5.3.2 Processus	34
5.4 VGG16	35
5.4.1 Introduction	35
5.4.2 Architecture	36
5.4.3 Création du modèle	37
5.4.4 Précision et erreur	39
5.4.5 Autres modèles	40
CHAPITRE 6 : La bibliothèque Face_recognition	41
6.1 Introduction	41
6.2 Code	42
6.3 Conclusion	44

Rapport de stage d'ingénieur

CHAPITRE 6 : LBPH	45
6.1 Introduction	45
6.2 Etapes	45
6.3 Code	47
CHAPITRE 7 : HOG&SVM	48
7.1 Introduction	48
7.2 Etapes	49
7.3 Code	51
7.4 Conclusion	52
CHAPITRE 8 : Partie matérielle	52
7.1 Introduction	52
7.2 Caméra IP avec Viewer4	53
7.3 Caméra IP avec Python	53
CHAPITRE 7 : Les outils informatiques	45
7.1 Editeurs de code	55
7.2 Langage de programmation	56
7.3 Les bibliothéques	56
Conclusion	62

Rapport de stage d'ingénieur

Introduction générale

La reconnaissance faciale est une technologie biométrique populaire, elle est largement utilisée dans les applications d'authentification, de contrôle d'accès et de vidéosurveillance, il existe plusieurs méthodes globales, locales et hybrides de reconnaissance faciale.

Il existe des catégories principales pour les techniques biométriques citons : les techniques biologiques, comportementales, morphologiques : (empreintes digitales, forme de la main, traits du visage, dessin du réseau veineux de l'œil...). Ces derniers ont l'avantage d'être stables dans la vie d'un individu et ne subissent pas autant les effets du stress par exemple, que l'on retrouve dans l'identification comportementale.

Et parmi toutes les technologies biométriques qui existent, la reconnaissance faciale est l'une des technologies les plus utilisées et les plus adaptées car elle permet d'exploiter beaucoup d'informations relatives à une personne.

Dans ce projet, nous nous sommes concentrés sur ce sujet très utile et très demandé dans le domaine de la sécurité des réseaux, de la sécurité nationale et même internationale. Nous avons proposé de mettre en œuvre une application de reconnaissance faciale utilisant la technique la plus performante de l'état de l'art.

Rapport de stage d'ingénieur

APERÇU DU PROJET

1-Contexte du stage

Le projet consiste à développer un algorithme d'analyse vidéo et de reconnaissance faciale capable d'identifier des personnes en temps réel sur des caméras de surveillance à partir d'une base de données qui contient les noms ainsi que des photos de chaque membre. Aussi, l'algorithme donne la possibilité de modifier la base de données en ajoutant les données et le nom d'une nouvelle personne détectée par la caméra.

2-Présentation de l'entreprise

SFM est une entreprise créée en 1995, issue du domaine des télécommunications et des réseaux. Son équipe d'experts et d'ingénieurs de haut niveau réalise des missions d'ingénierie et de conseil pour le compte de régulateurs des télécommunications, d'opérateurs, de Ministères des TIC et de bailleurs de fonds (BM, BAD).

C'est au cours de ses missions que SFM a développé des outils, applications et plateformes pour la digitalisation des process d'ingénierie, de suivi et de mesures de QoS/QoE, de contrôle des tarifs, ...

SFM a ainsi acquis une expertise pour la réalisation de produits IT pour le secteur des télécoms mais également pour d'autres secteurs comme celui des assurances, de la banque ainsi que des solutions de gestion intégrées pour les TPE/PME.

Les outils et développements sur-mesure offerts par SFM se déclinent selon 3 axes :

1. **Digitalisation des process des entreprises** (ou *Robotic Process Automation*) avec notamment les produits COSAP, Ticketeazy et IT&M
2. **Cybersécurité** avec un portail d'accès WiFi sécurisé, un SOC et une PKI entreprise
3. **Big Data et Intelligence Artificielle** avec des solutions destinées aux secteurs des télécommunications (détection de fraude et churn), bancaire (estimation de l'évolution des taux de change), ajout de couches de ML et DL dans les produits SFM

Rapport de stage d'ingénieur

Certains outils sont immédiatement utilisables tandis que d'autres sont personnalisés par l'équipe SFM après étude des besoins et de l'environnement de chaque client.

SFM maintient avec tous ses clients un lien permanent permettant de faire évoluer ses produits, d'assurer un service après-vente personnalisé et une assistance sur site ou à distance.

3-Les atouts de SFM

- Expertise théorique et pratique : grâce à la réalisation de nombreuses missions d'ingénierie, de conseil et de formation.
- Transfert de compétences : Dans ses locaux, sur le terrain ou chez ses clients, SFM organise périodiquement des missions de formation sur les techniques de mesures de la QoS. Les intervenants transfèrent naturellement et aisément leur expertise aux équipes du Client.
- Recherche & Développement : Grâce à son équipe de développeurs, SFM innove pour mettre en place de nouveaux outils permettant de répondre de manière personnalisée et spécifique aux besoins de ses Clients.
- Equipe qualifiée : constituée d'experts et spécialistes du domaine des réseaux mobiles et des TIC issus des grandes écoles d'ingénieurs et possédant une expérience inégalée de l'exploitation et de l'optimisation des réseaux.

4-Les références de SFM Telecom



Rapport de stage d'ingénieur

5-Les partenaires

Partenaires stratégiques



Partenaires privilégiés



6-Environnement de travail de SFM

J'ai commencé par analyser l'ensemble du projet avec l'aide de mon superviseur, je l'ai donc divisé en sous-tâches selon la priorité. Après cela, j'ai commencé par préparer mon environnement PC en installant plusieurs logiciels et outils. Ensuite, j'ai fait des recherches pour collecter des données pour les modèles de détection faciale et de reconnaissance faciale. Le mélange des deux modèles a nécessité une intégration et de nombreuses transformations pour adapter les deux algorithmes afin d'atteindre notre objectif final.

Enfin, avec l'aide de mon superviseur, j'ai fait mes choix finaux pour créer un algorithme efficace pour obtenir le meilleur résultat possible de notre projet.

La reconnaissance faciale

1-Introduction

Durant la vie quotidienne chacun de nous identifie tout au long de la journée différents visages. Ainsi lorsque nous rencontrons une personne, notre cerveau va chercher dans notre mémoire et vérifie si cette personne est répertoriée ou non, c'est une tâche aisée pour les humains. En est-il de même pour une machine ?

Dans ce premier chapitre, nous allons justement présenter les grandes lignes de notre travail, nous allons ainsi expliquer des notions générales sur la reconnaissance faciale, le fonctionnement d'un système de reconnaissance faciale, On va aussi aborder quelques techniques utilisées, des exemples et les domaines d'application.

2-Définitions

La reconnaissance faciale est un moyen d'identifier ou de confirmer l'identité d'un individu à l'aide de son visage. Les systèmes de reconnaissance faciale peuvent être utilisés pour identifier des personnes sur des photos, des vidéos ou en temps réel.

Ce sujet est une catégorie de sécurité biométrique. D'autres formes de logiciels biométriques incluent la reconnaissance vocale, la reconnaissance d'empreintes digitales et la reconnaissance de la rétine oculaire ou de l'iris. La technologie est principalement utilisée pour la sécurité et l'application de la loi, bien qu'il y ait un intérêt croissant dans d'autres domaines d'utilisation.

Le marché de la reconnaissance faciale devrait atteindre 7,7 milliards de dollars en 2022, contre 4 milliards de dollars en 2017. C'est parce que la reconnaissance faciale a de nombreuses applications commerciales. Il peut être utilisé pour tout, de la surveillance au marketing.

Rapport de stage d'ingénieur

3-Utilités

- Cette technologie est utilisée à diverses fins. Ceux-ci inclus :
 - Aéroports et contrôle aux frontières
 - Recherche de personnes disparues
 - Déverrouillage des téléphones
 - Réduire la criminalité dans le commerce de détail
 - Forces de l'ordre
 - Marketing et publicité
 - Suivi de la présence des étudiants ou des travailleurs

4-Avantages

- La reconnaissance faciale a de nombreux avantages comme :
 - Sécurité accrue
 - Réduction de la criminalité
 - Supprimer le biais de l'arrêt et de la recherche
 - Plus de commodité
 - Intégration avec d'autres technologies

5-Désavantages

- Atteinte à la vie privée
- Stockage massif de données
- Portée de l'erreur

6-Problèmes

- L'intérêt commercial croissant pour la reconnaissance faciale est encourageant, mais il s'avère également être une entreprise difficile en ce qui concerne les problèmes associés qui ont joué en permanence avec sa qualité de livraison. Ces défis surviennent lorsque les situations ne sont pas coopératives et provoquent des apparences/expressions faciales variées.
- Nous citerons quelques défis qui limitent le potentiel d'un système de reconnaissance faciale pour aller plus loin :

Rapport de stage d'ingénieur

- **Éclairage** : L'éclairage représente les variations de lumière. Le léger changement des conditions d'éclairage pose un défi important pour la reconnaissance faciale automatisée et peut avoir un impact significatif sur ses résultats. Il a été constaté que la différence entre deux mêmes faces avec des éclairages différents est supérieure à deux faces différentes prises sous le même éclairage.



- **Pose** : Les systèmes de reconnaissance faciale sont très sensibles aux variations de pose. La pose d'un visage varie lorsque le mouvement de la tête et l'angle de vue de la personne changent. Il devient difficile d'identifier le vrai visage lorsque l'angle de rotation augmente.



- **Expressions** : Des situations variables provoquent des humeurs différentes qui se traduisent par des émotions diverses et éventuellement par des changements dans les expressions faciales.

Rapport de stage d'ingénieur



- Basse résolution
- Vieillissement
- Complexité du modèle

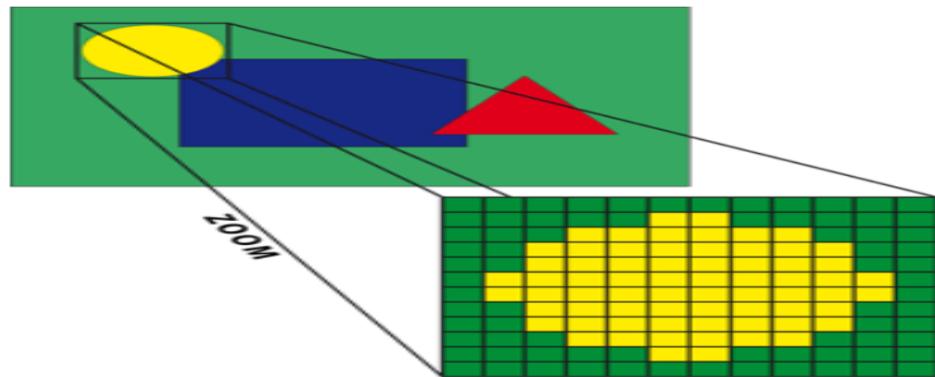
7-Conclusion

- Dans ce chapitre nous avons présenté les grandes lignes de notre travail, comme les notions de base et les techniques utilisées (état de l'art). Nous avons ainsi mis en valeur les points critiques à soigner et dégager les parties à aborder lors des étapes suivantes de la réalisation de notre projet.

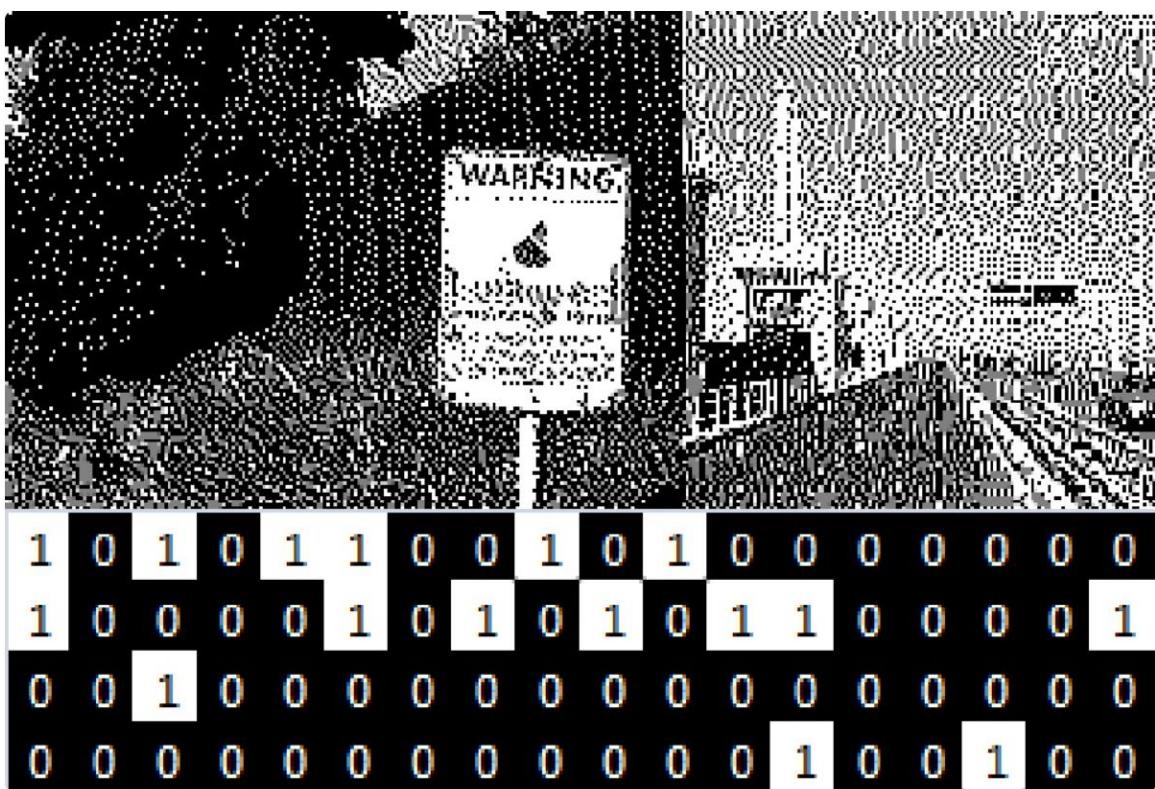
Image numérique

1-Introduction

- Une image numérique (au niveau d'un périphérique comme l'écran) est une **matrice de pixels (px)**. On parle d'image matricielle.



- Pour une **image binaire** (ou image en noir et blanc), les éléments de l'image sont représentés par des pixels noirs (des 0 par exemple) et des pixels blancs (donc des 1).

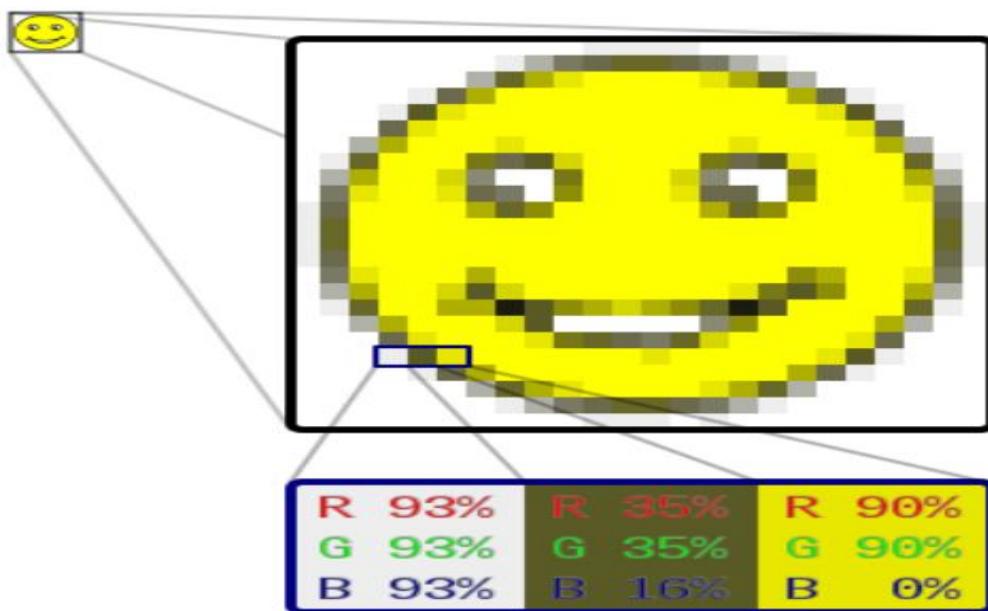


Rapport de stage d'ingénieur

- Pour une image en couleurs, chaque pixel a associé une couleur, usuellement décomposée en trois composantes primaires par synthèse additive : rouge vert bleu (RVB ou **RGB** en anglais). Chaque pixel peut posséder une valeur dans l'intervalle [0 ;255]

R	V	B	Couleur
0	0	0	noir
255	0	0	rouge
0	255	0	vert
0	0	255	bleu
128	128	128	gris moyen
255	255	255	blanc

Exemple de couleurs en RGB



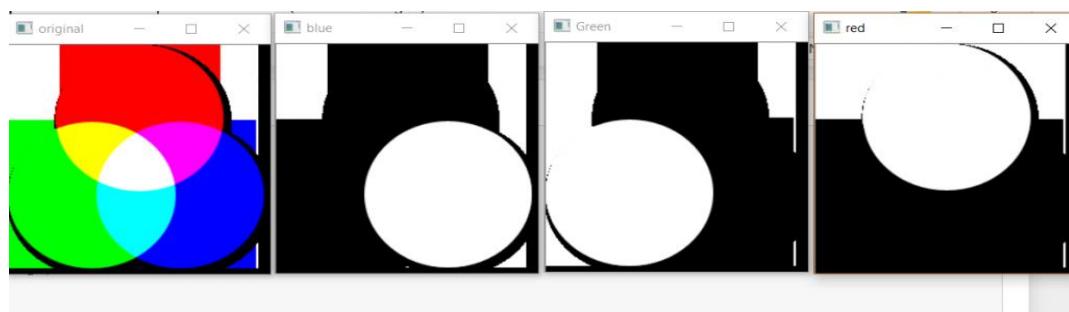
Exemple des composants d'un pixel en RGB

2-Traitement d'image avec Python

- Dans cette partie on va découvrir quelque opération possible qu'on peut exécuter sur des images avec Python.

Rapport de stage d'ingénieur

- Voici une idée sur les fonctions les plus utiles du bibliothèque OpenCV dans notre projet sont :
 - cv2.imread : fonction utilisée pour lire des images.
 - cv2.VideoCapture : utilisée pour lire vidéo à partir de la caméra, une vidéo enregistrée ou un streaming d'une caméra IP.
 - cv2.putText : utilisée pour écrire du texte sur une image
 - cv2.rectangle : permet de dessiner un rectangle sur l'image
 - cv2.imshow : affiche une image
 - cv2.waitKey : permet d'attendre une durée de temps (entrée en milliseconde comme paramètre) pour un événement du clavier.
 - cv2.destroyAllWindows : ferme toutes les fenêtres créées.
 - cv2.imwrite : permet d'enregistrer une image
- On suppose que notre image est enregistrée dans la variable "img"
 - img.shape : donne les dimensions de l'image
 - B, G, R = cv2.split(img) : diviser l'image(colorée) en ces 3 plans RGB



Division d'une image selon les couleurs RGB

- cv2.cvtColor : change l'espace colorimétrique de l'image



BGR to Gray

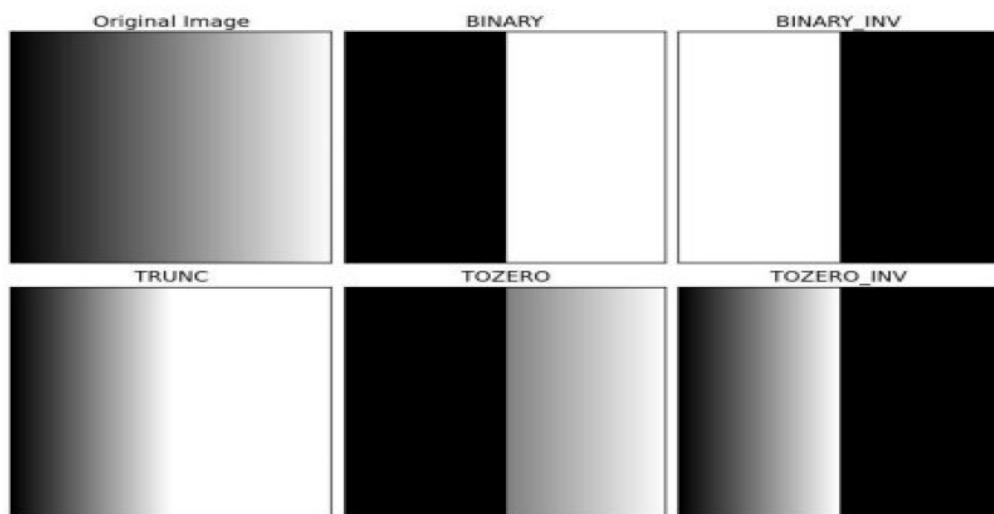
Rapport de stage d'ingénieur

- cv2.inRange : crée un masque pour extraire des pixels spécifiques de l'image selon un seuil.



- cv2.threshold : Si la valeur du pixel est supérieure à une valeur seuil, une valeur lui est attribuée (peut être blanche), sinon une autre valeur lui est attribuée (peut être noire).

Le paramètre style affecte le résultat :



- Lissage de l'image : Le lissage des images aide à la détection des contours en diminuant le bruit de l'image.

Rapport de stage d'ingénieur

- cv2.filter2D : convoluer un noyau avec une image.



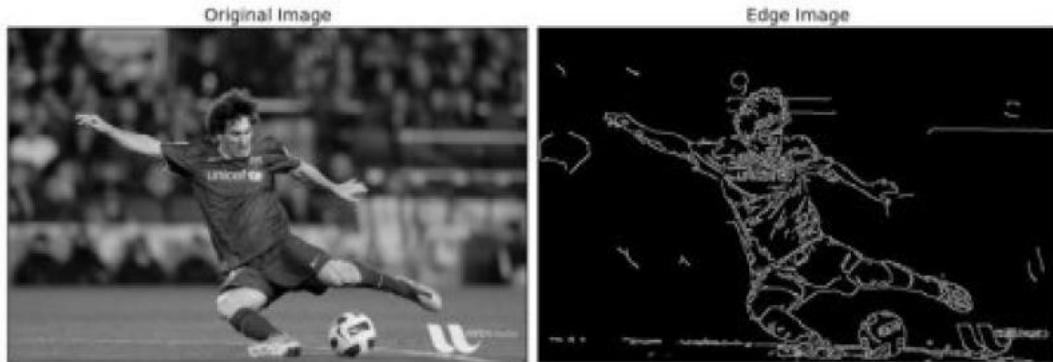
- Autres fonctions de filtrage : cv2.GaussianBlur, cv2.blur, cv2.GaussianBlur, cv2.medianBlur, cv2.bilateralFilter.
- Transformations morphologiques :
 - Érosion : tous les pixels proches des contours seront supprimés en fonction de la taille du noyau. Il supprime les bruits blancs.
 - Dilatation : il augmente la région blanche dans l'image ou la taille de l'objet de premier plan augmente.



- Autres fonctions de transformation possibles : cv2.morphologyEx, cv2.Laplacian, cv2.Sobel.

Rapport de stage d'ingénieur

- Détection des contours
 - cv2.Canny : C'est un algorithme en plusieurs étapes qui permet de détecter les contours.



- Il y a aussi les fonctions suivantes : cv2.cornerHarris, cv2.goodFeaturesToTrack, etc...

Conclusion :

- Voici donc quelques-unes des méthodes de manipulations et traitement d'images utiles et disponibles en Python. Certaines sont assez connues et d'autres peuvent être découverts.

Rapport de stage d'ingénieur

Détection du visage

1-Introduction

- La technologie de détection de visage est utilisée dans de nombreux domaines : divertissement, sécurité, application de la loi, biométrie, etc.
- Il joue un rôle clé dans l'analyse, le suivi et la reconnaissance des visages. En plus, il utilise l'apprentissage automatique et des algorithmes afin d'extraire des visages humains à partir d'images plus grandes ; de telles images contiennent généralement de nombreux objets sans visage, tels que des bâtiments, des paysages et diverses parties du corps.
- Les algorithmes de détection faciale commencent généralement par rechercher les yeux humains, qui sont l'une des caractéristiques faciales les plus faciles à détecter. Ensuite, l'algorithme peut essayer de trouver la bouche, le nez, les sourcils et l'iris. Après avoir identifié ces traits du visage et que l'algorithme conclut qu'il a extrait un visage, il passe ensuite par des tests supplémentaires pour confirmer qu'il s'agit bien d'un visage.

Dans la suite, on va présenter quelques algorithmes utilisés dans la tache de détection de visage.

2-Haar feature-based cascade classifier

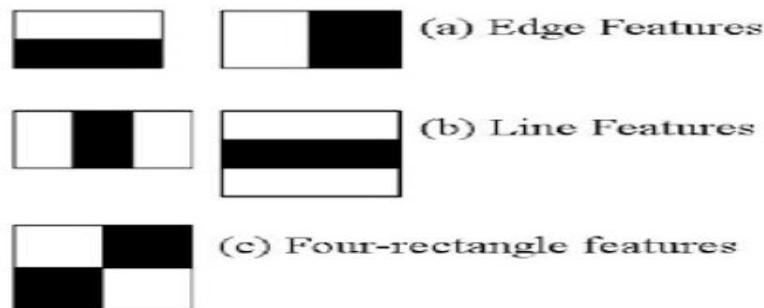
L'algorithme peut être expliqué en quatre étapes :

- Calcul des caractéristiques de Haar
- Crédit d'images intégrales
- Utilisation d'Adaboost
- Implémentation de classificateurs en cascade

Rapport de stage d'ingénieur

2-1Calcul des caractéristiques de Haar

Une caractéristique Haar est essentiellement des calculs qui sont



Types of Haar features. ([Image Source](#))

effectués sur des régions rectangulaires adjacentes à un emplacement spécifique dans une fenêtre de détection. Le calcul consiste à additionner les intensités de pixels dans chaque région et à calculer les différences entre les sommes.

2-2Création d'images intégrales

Au lieu de calculer à chaque pixel, il crée à la place des sous-rectangles et crée des références de tableau pour chacun de ces sous-rectangles.

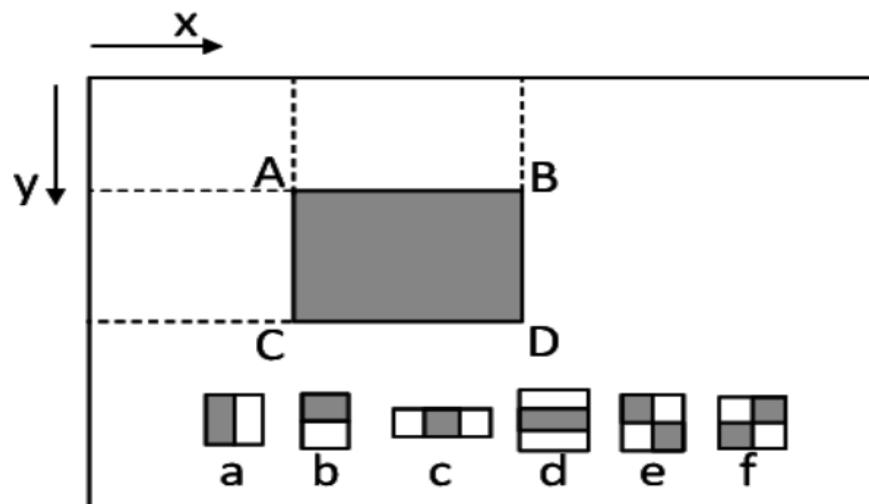
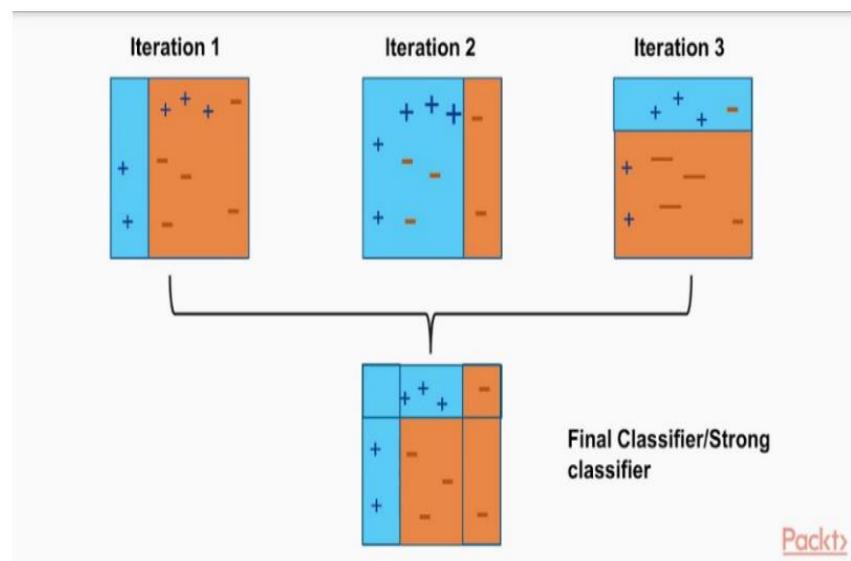


Illustration for how an integral image works. ([Image Source](#))

Rapport de stage d'ingénieur

2-3 Utilisation d'Adaboost

Adaboost choisit essentiellement les meilleures fonctionnalités et forme les classificateurs à les utiliser. Il utilise une combinaison de « classificateurs faibles » pour créer un « classificateur fort » que

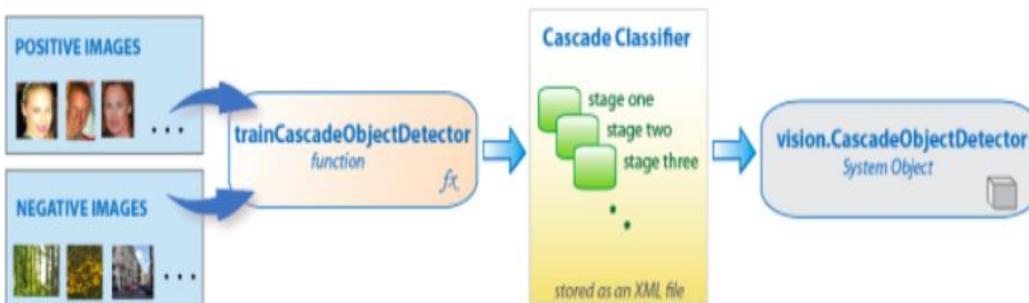


Representation of a boosting algorithm. ([Image Source](#))

L'algorithme peut utiliser pour détecter des objets.

2-4 Implémentation de classificateurs en cascade

Le classificateur en cascade est composé d'une série d'étapes, où chaque étape est une collection d'apprenants faibles.



A flowchart of cascade classifiers. ([Image Source](#))

Rapport de stage d'ingénieur

2-5Code

```

import cv2
import numpy as np

face_classifier =
cv2.CascadeClassifier('/haarcascade_frontalface_default.xml')

gray = cv2.cvtColor(resized, cv2.COLOR_BGR2GRAY)

''' Our classifier returns the ROI of the detected face as a tuple,
It stores the top left coordinate and the bottom right coordinates'''

faces = face_classifier.detectMultiScale(gray, 1.0485258, 6)

'''When no faces detected, face_classifier returns an empty tuple'''
if faces is ():
    print("No faces found")

'''We iterate through our faces array and draw a rectangle over each
face in faces'''
for (x,y,w,h) in faces:
    cv2.rectangle(resized, (x,y), (x+w,y+h), (127,0,255), 2)
    cv2.imshow('Face Detection', resized)
    cv2.waitKey(0)

cv2.destroyAllWindows()

```

2-6Conclusion

Haar Cascade Detection est l'un des algorithmes de détection de visage les plus anciens et les plus puissants inventés. Il existe depuis longtemps, bien avant que Deep Learning ne devienne célèbre. Les fonctionnalités Haar n'étaient pas seulement utilisées pour détecter les visages, mais aussi pour les yeux, les lèvres, les plaques d'immatriculation, etc...

3-CNN face detection

3-1Introduction

CNN (Convolutional Neural Network) est une classe de réseautage profond, il fonctionne très bien pour les visages non frontaux à des angles étranges où le détecteur basé sur HOG n'est pas bon pour cela.

Dlib a une détection de visage basée sur CNN, entraînée avec des millions d'images et stocké le modèle entraîné dans un fichier « .dat »

3-2Les avantages

- La détection des visages basée sur CNN est si précise qu'elle peut trouver des visages étranges
- Cela fonctionne sur le CPU mais doit être plus rapide sur le GPU
- Très facile à utiliser

3-3Les désavantages

- Fonctionne lentement sur le processeur
- Ne détecte pas les petits visages

3-4Code

```
# initialize cnn based face detector with the weights
cnn_face_detector = dlib.cnn_face_detection_model_v1(args.weights)
```

Rapport de stage d'ingénieur

```
start = time.time()

# apply face detection (cnn)
faces_cnn = cnn_face_detector(image, 1)

end = time.time()
print("CNN : ", format(end - start, '.2f'))

# loop over detected faces
for face in faces_cnn:
    x = face.rect.left()
    y = face.rect.top()
    w = face.rect.right() - x
    h = face.rect.bottom() - y

    # draw box over face
    cv2.rectangle(image, (x,y), (x+w,y+h), (0,0,255), 2)
```

4-MTCNN

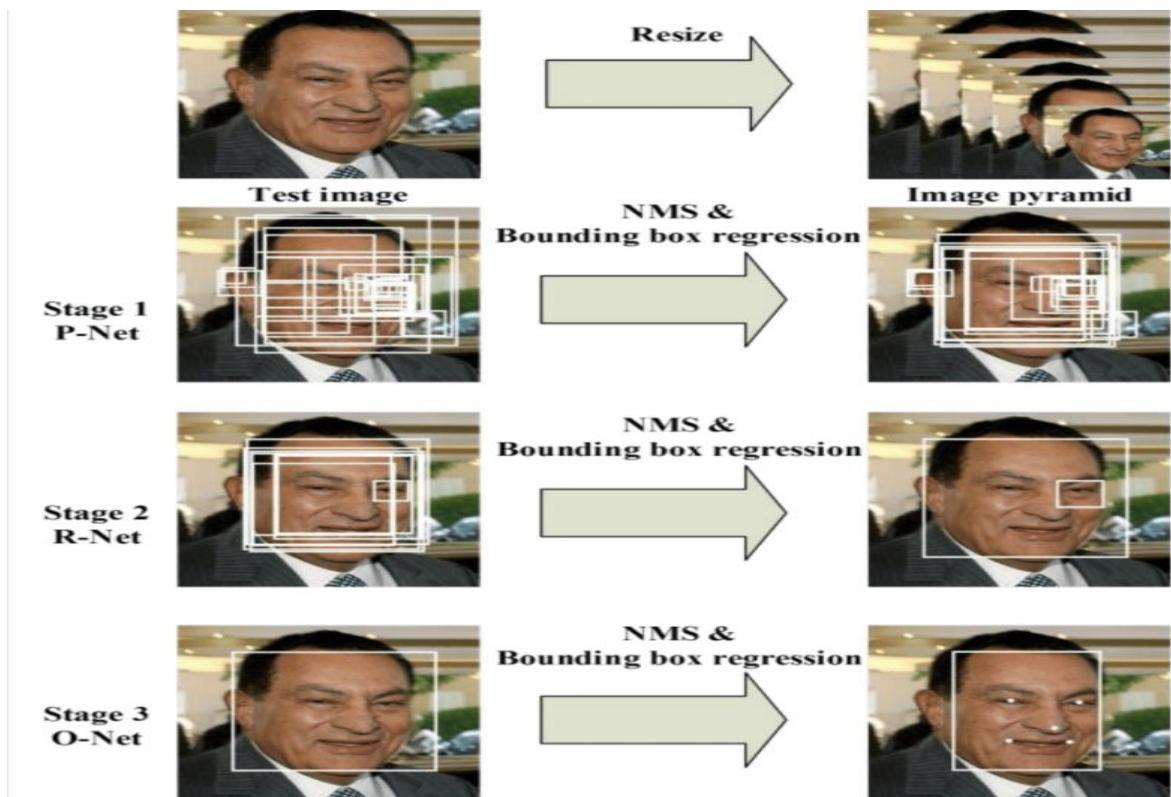
4-1 Introduction

- MTCNN (Multi-Task Cascade Convolutional Neural Network) est un cadre multitâche en cascade profond utilisant différentes fonctionnalités de « sous-modèles » pour chacun augmenter leurs forces de corrélation. Il fonctionne assez rapidement sur un processeur, même si S3FD est toujours plus rapide sur un GPU.

4-2 Explication

Rapport de stage d'ingénieur

- Les CNN proposés se composent de trois étapes. Dans la première étape, il produit rapidement des fenêtres candidates via un CNN peu profond. Ensuite, il affine les fenêtres pour rejeter un grand nombre de fenêtres sans faces à travers un CNN plus complexe. Enfin, il utilise un CNN plus puissant pour affiner le résultat et afficher les positions des repères faciaux.



Etapes de détection de visage avec MTCNN

Une explication plus simple des trois étapes de MTCNN peut être la suivante :

- Dans la première étape, le MTCNN crée plusieurs cadres qui parcourront l'intégralité de l'image en partant du coin supérieur gauche et en progressant finalement vers le coin inférieur droit. Le processus de récupération d'informations est appelé P-Net (Proposal Net) qui est un CNN peu profond et entièrement connecté.
- Dans la deuxième étape, toutes les informations de P-Net sont utilisées comme entrée pour la couche suivante de CNN appelée R-Net (Réseau de raffinement), un CNN complexe entièrement connecté qui rejette la majorité des images qui ne contiennent pas de visages.

Rapport de stage d'ingénieur

- Dans la troisième et dernière étape, un CNN plus puissant et complexe, connu sous le nom d'O-Net (réseau de sortie), qui, comme son nom l'indique, génère la position du repère facial en détectant un visage à partir de l'image/vidéo donnée.

4-2Code

```
# Importing the main MTCNN package
from mtcnn.mtcnn import MTCNN

''' Create an instance of the MTCNN() function/class, using default
weights. '''
detector = MTCNN()

# This internal method detects faces in the image
face_detection = detector.detect_faces(plot_image)

for result in result_list:
    ''' Get the coordinates of the rectangle from result for
    every face detected by MTCNN() '''
    x, y, width, height = result['box']

    ''' Call the rectangular function using above coordinate
    values '''
    traced_rectangle = Rectangle((x, y), width, height,
        fill=False, color='red')

    # Trace the rectangle encasing the faces
    axes.add_patch(traced_rectangle)
```

5-DNN Face Detector in OpenCV

5-1Introduction

- Il s'agit d'un modèle 'Caffe' qui est basé sur le Single Shot-Multibox Detector (SSD) et utilise l'architecture ResNet-10 comme épine dorsale. Il a été introduit après OpenCV 3.3 dans son module de réseau de neurones profonds. Ce modèle fonctionne bien avec l'occlusion, les mouvements rapides de la tête et peut également

Rapport de stage d'ingénieur

identifier les faces latérales. De plus, il a également donné les fps les plus rapides parmi tous.

5-2Code

```
import cv2
import numpy as np

modelFile = "models/res10_300x300_ssd_iter_140000.caffemodel"
configFile = "models/deploy.prototxt.txt"
net = cv2.dnn.readNetFromCaffe(configFile, modelFile)
img = cv2.imread('test.jpg')
h, w = img.shape[:2]
blob = cv2.dnn.blobFromImage(cv2.resize(img, (300, 300)), 1.0,
(300, 300), (104.0, 117.0, 123.0))
net.setInput(blob)
faces = net.forward()
#to draw faces on image
for i in range(faces.shape[2]):
    confidence = faces[0, 0, i, 2]
    if confidence > 0.5:
        box = faces[0, 0, i, 3:7] * np.array([w, h, w, h])
        (x, y, x1, y1) = box.astype("int")
        cv2.rectangle(img, (x, y), (x1, y1), (0, 0, 255), 2)
```

6-Fréquence d'images

Les valeurs rapportées sont obtenues à l'aide d'un processeur Intel i5 7ème génération et la taille d'image transmise est de 640x360 à l'exception du module DNN qui reçoit une image 300x300 comme cela a été fait jusqu'à présent.

Haar — 9,25 fps

Dlib — 5,41 fps

MTCNN — 7,92 fps

Module DNN d'OpenCV — 12.95 fps

Rapport de stage d'ingénieur

7-Conclusion

- Le classificateur Haar Cascade a donné les pires résultats dans la majorité des tests avec beaucoup de faux positifs.
- Dlib et MTCNN ont eu des résultats très similaires avec un léger avantage par rapport à MTCNN, mais Dlib ne peut pas identifier de très petits visages.
- Pour les problèmes généraux de vision par ordinateur, le modèle Caffe d'OpenCV du module DNN est le meilleur.

Apprentissage profond

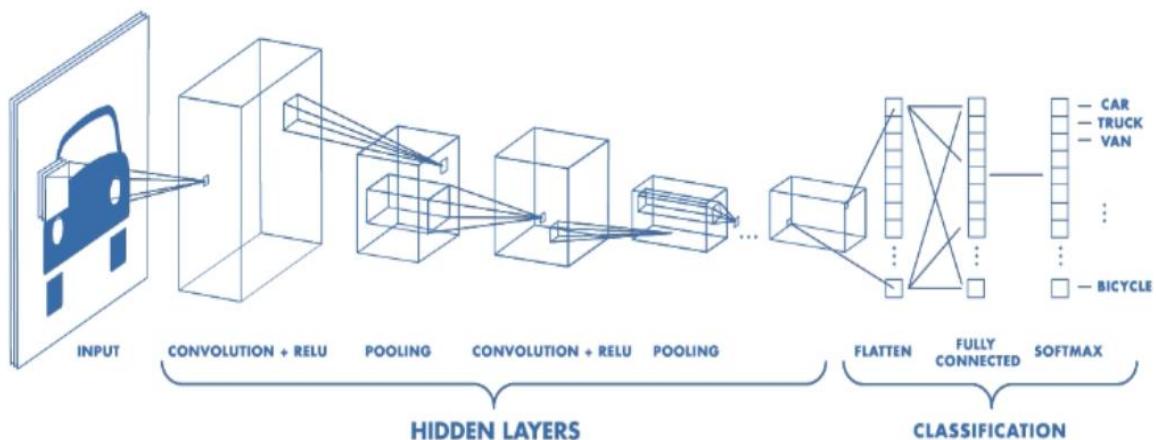
1-Introduction

- L'apprentissage profond est un sous-domaine de l'apprentissage automatique concernant les algorithmes inspirés de la structure et de la fonction du cerveau appelés réseaux de neurones artificiels.
- Les méthodes d'apprentissage en profondeur reposent sur divers programmes complexes pour imiter l'intelligence humaine. Cette méthode particulière apprend aux machines à reconnaître les motifs afin qu'ils puissent être classés en catégories distinctes.
- Parmi les modèles d'apprentissage profond, on trouve le CNN (Convolutional Neural Network) qui est créé pour les données qui ont une topologie de type grille (comme les images).

2-CNN

2-1Introduction

- Un CNN a généralement trois couches : une couche convulsive, une couche de mise en commun et une couche entièrement connectée.



Rapport de stage d'ingénieur

2-2 Couche de convolution (Convolution Layer)

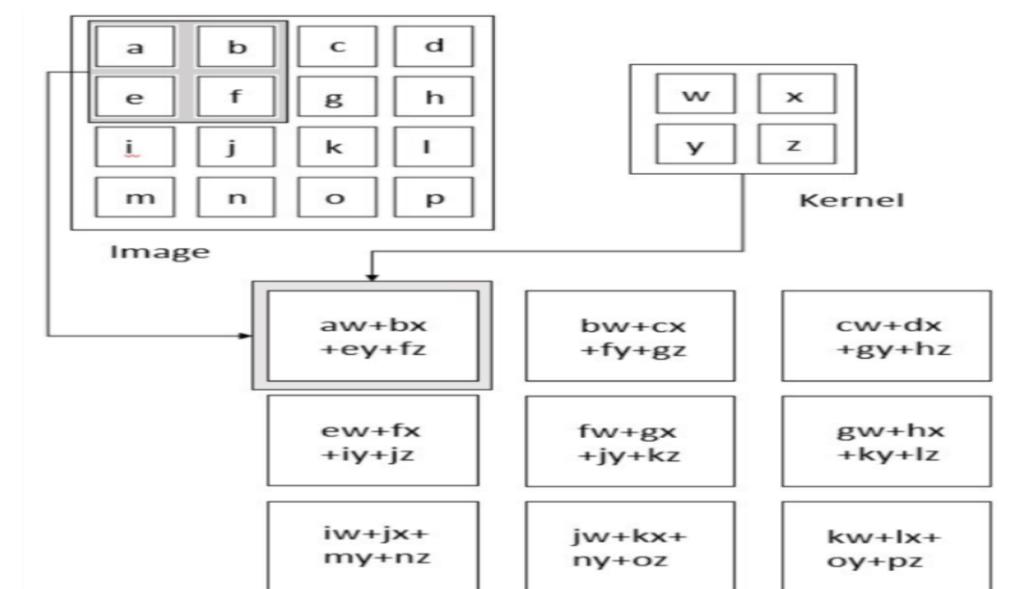
- La couche de convolution est le bloc de construction de base du CNN, cette couche effectue un produit scalaire entre deux matrices, où une matrice est l'ensemble de paramètres apprenables autrement connu sous le nom de noyau, et l'autre matrice est la partie restreinte du champ récepteur.
- La formule de convolution est :

$$W_{out} = \frac{W - F + 2P}{S} + 1$$

W : Width of the input image / F : spatial size of the kernel

S : Stride / P : amount of padding / Wout : Width of the output

Input of size : $W * W * D$ // Output of size : $Wout * Wout * Dout$



L'opération de convolution

Rapport de stage d'ingénieur

2-3 Couche de mise en commun (Pooling Layer)

- La couche de mise en commun remplace la sortie du réseau à certains endroits en dérivant une statistique récapitulative des sorties à proximité.
- Cela aide à réduire la taille spatiale de la représentation, ce qui diminue la quantité requise de calcul et de poids.
- Il existe plusieurs fonctions de mise en commun telles que ‘Max Pooling’ et ‘Average Pooling’, etc...



2-4 Couche entièrement connectée. (Fully connected layer)

- Les neurones de cette couche ont une connectivité complète avec tous les neurones de la couche précédente et suivante. Il aide à cartographier la représentation entre l'entrée et la sortie.

Rapport de stage d'ingénieur

2-5Couches de non-linéarité

- Les couches de non-linéarité sont souvent placées directement après la couche convulsive pour introduire la non-linéarité dans la carte d'activation.
- Les opérations non linéaires les plus populaires sont :
 - **Sigmoid**
 - Son expression est : $\sigma(\kappa) = 1/(1+e^{-\kappa})$
 - **Tanh**
 - Il écrase un nombre à valeur réelle dans la plage [-1, 1]
 - **Relu**
 - Son expression est : $f(\kappa)=\max(0, \kappa)$

2-6Conclusion

- Bien que le CNN ait une grande précision, il faut malheureusement beaucoup de temps pour prédire le résultat ce qui représente un problème dans notre algorithme puisque nous sommes dans un « cas de prédiction en temps réel », c'est pourquoi nous allons utiliser un modèle pré-entraîné et exploiter la technique 'Transfer Learning'.

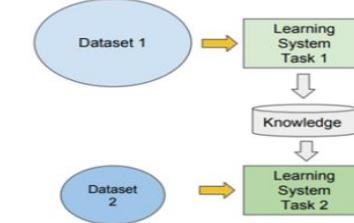
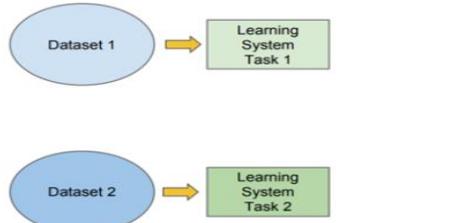
3-L'apprentissage par transfert (Transfer Learning)

3-1Introduction

- Un modèle pré-entraîné est un modèle qui a été entraîné sur un grand ensemble de données de référence pour résoudre un problème similaire à celui que nous voulons résoudre. En conséquence, en raison du coût de calcul de la formation de tels modèles, il est courant d'importer et d'utiliser des modèles de la littérature publiée (par exemple VGG, Inception, MobileNet).
- L'apprentissage par transfert est une méthode populaire en vision par ordinateur, car elle nous permet de construire des modèles précis de manière à gagner du temps.

Traditional ML vs Transfer Learning

- | | | |
|---|-----------|---|
| <ul style="list-style-type: none"> • Isolated, single task learning: <ul style="list-style-type: none"> ◦ Knowledge is not retained or accumulated. Learning is performed w.o. considering past learned knowledge in other tasks | vs | <ul style="list-style-type: none"> • Learning of a new tasks relies on the previous learned tasks: <ul style="list-style-type: none"> ◦ Learning process can be faster, more accurate and/or need less training data |
|---|-----------|---|

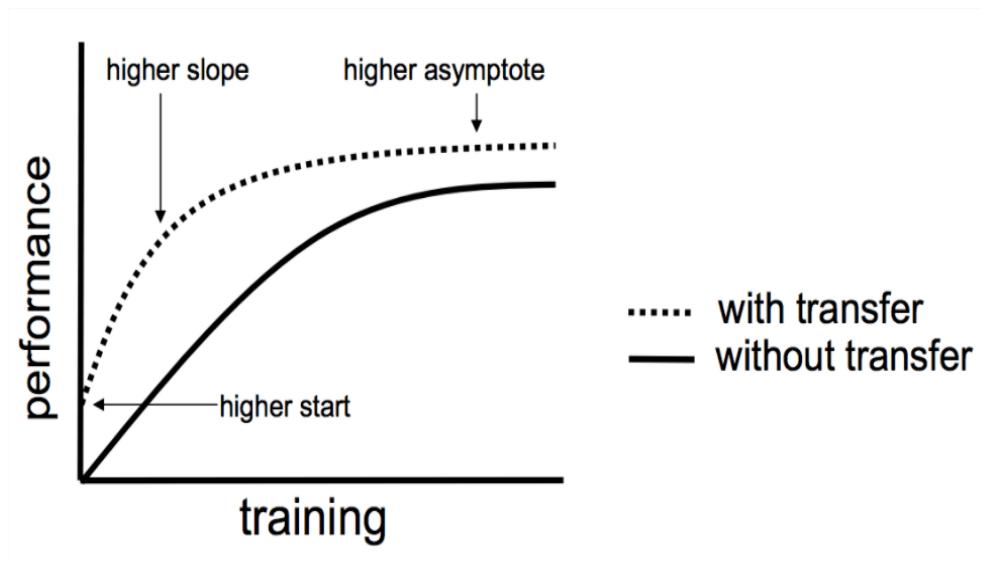


Traditional Learning vs Transfer Learning

- Le dernier figure prouve que dans l'apprentissage par transfert, vous pouvez tirer parti des connaissances (caractéristiques, poids, etc.) de modèles précédemment formés pour former de nouveaux modèles et même résoudre des problèmes tels que le fait d'avoir moins de données pour la nouvelle tâche !

Rapport de stage d'ingénieur

- Pourquoi utilisons-nous l'apprentissage par transfert ? la réponse est simplement dans la figure suivante :



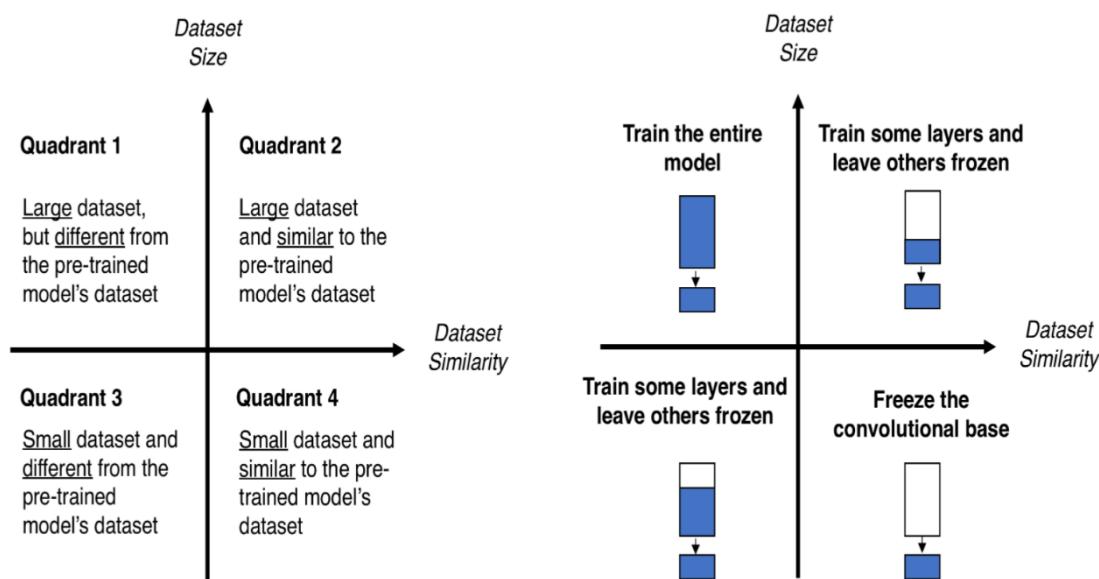
- Donc il y a trois avantages possibles à rechercher lors de l'utilisation de l'apprentissage par transfert : Début plus élevé, Pente plus élevée et Asymptote plus élevée.

3-2 Processus de l'apprentissage par transfert

- 1- Sélectionnez un modèle pré-entraîné : les modèles sont chargées du module python “tensorflow.keras.applications”
- 2- Classez votre problème selon la matrice de similitude de taille : Selon les données, vous devez choisir le modèle le plus similaire pour obtenir le plus de précision possible.

Rapport de stage d'ingénieur

3- Affiner votre modèle : En fonction de vos données et de leur similarité avec les données sur lesquelles le modèle a déjà été entraîné, vous devez décider comment vous allez affiner votre modèle.



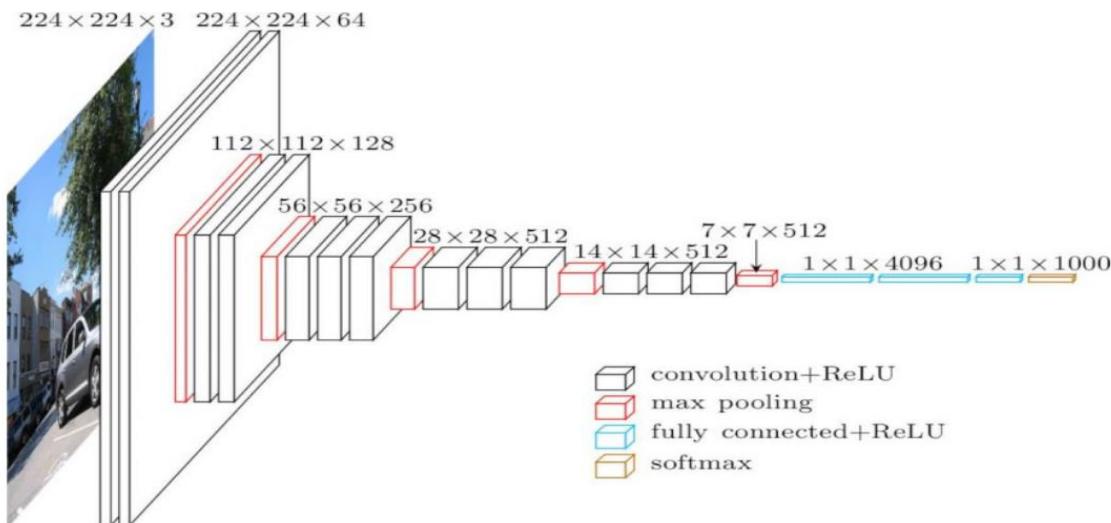
4-VGG16

4-1Introduction

- VGG16 est un modèle de réseau neuronal convolutif proposé par K. Simonyan et A. Zisserman de l'Université d'Oxford dans l'article « Very Deep Convolutional Networks for Large-Scale Image Recognition ».
- Le modèle atteint une précision de test de 92,7% dans le top 5 dans ImageNet, qui est un ensemble de données de plus de 14 millions d'images appartenant à 1000 classes.
- VGG16 a été formé pendant des semaines et utilisait les GPU NVIDIA Titan Black.

Rapport de stage d'ingénieur

4-2 Architecture



- L'entrée de la couche cov1 est d'une image RVB de taille fixe 224 x 224. L'image est passée à travers un empilement de couches convolutives (conv), où les filtres ont été utilisés avec un très petit champ récepteur : 3x3. Dans l'une des configurations, il utilise également des filtres de convolution 1 x 1, qui peuvent être considérés comme une transformation linéaire des canaux d'entrée.
- La foulée de convolution (convolution stride) est fixée à 1 pixel ; le remplissage spatial de conv. L'entrée de la couche est telle que la résolution spatiale est préservée après convolution, c'est-à-dire que le rembourrage est de 1 pixel pour 3x3 conv couches. La mise en commun spatiale (Spatial Pooling) est réalisée par cinq couches max-pooling, qui suivent une partie de la conv couches.
- La mise en commun maximale (Max-pooling) est effectuée sur une fenêtre de 2x2 pixels, avec foulée 2.
- Trois couches entièrement connectées (FC : Fully-Connected) suivent un empilement de couches convolutives : les deux premières ont 4096 canaux chacune, la troisième effectue une classification ILSVRC à 1000 voies et contient donc 1000 canaux (un pour chaque classe). La dernière couche est la couche soft-max. La configuration des couches entièrement connectées est la même dans tous les réseaux.
- Toutes les couches cachées sont équipées de la non-linéarité de rectification (ReLU).
- Il est également noté qu'aucun des réseaux (sauf un) ne contient de normalisation de réponse locale (LRN), une telle normalisation n'améliore pas les performances sur l'ensemble de données ILSVRC, mais

Rapport de stage d'ingénieur

entraîne une augmentation de la consommation de mémoire et du temps de calcul.

4-3Création de modèle

- J'ai travaillé sur une base de données de 2 classes (moi et personnes inconnues), 321 images d'entraînement et 101 images de test soit 76% des données pour l'entraînement et 24% pour la validation. Ceci est déjà expliqué précédemment le modèle VGG-16 est composé de 16 couches et voici la présentation keras du modèle :

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808

block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
dense_4 (Dense)	(None, 1024)	25691136
dropout (Dropout)	(None, 1024)	0
dense_5 (Dense)	(None, 1024)	1049600
dropout_1 (Dropout)	(None, 1024)	0
dense_6 (Dense)	(None, 512)	524800
dense_7 (Dense)	(None, 1)	513
Total params:	41,980,737	
Trainable params:	27,266,049	
Non-trainable params:	14,714,688	

Rapport de stage d'ingénieur

- Nous avons utilisé « Augmentation des données » pour augmenter notre ensemble de données afin que la précision du modèle s'améliore.

```
train_datagen = ImageDataGenerator(  
    rescale=1./255,  
    zoom_range=0.1,  
    #shear_range=0.1,  
    rotation_range=10,  
    width_shift_range=0.1,  
    height_shift_range=0.1,  
    vertical_flip=False,  
    horizontal_flip=True,  
    fill_mode='nearest')  
  
validation_datagen = ImageDataGenerator(rescale=1./255)
```

- De plus, nous avons utilisé des fonctions de rappels (Callback function) pour obtenir le meilleur résultat du modèle et éviter le surapprentissage.

```
from keras.callbacks import ModelCheckpoint, EarlyStopping  
  
checkpoint = ModelCheckpoint("best_model.h5",  
                            monitor="val_loss",  
                            mode="min",  
                            save_best_only = True,  
                            verbose=1)  
  
earlystop = EarlyStopping(monitor = 'val_loss',  
                         min_delta = 0,  
                         patience = 3,  
                         verbose = 1,  
                         restore_best_weights = True)
```

Rapport de stage d'ingénieur

- Pour entraîner le modèle, j'ai effectué 30 époques mais l'entraînement s'est arrêté à la 13ème époque à cause des 'Fonctions de rappels' (Callbacks functions) comme le montre la figure ci-dessous :

```

Epoch 00007: val_loss did not improve from 0.00000
Epoch 8/30
27/27 [=====] - 2s 74ms/step - loss: 0.5912 - accuracy: 0.8366 - val_loss: 3.1274e-12 - val_accuracy: 0.8333

Epoch 00008: val_loss improved from 0.00000 to 0.00000, saving model to best_model.h5
Epoch 9/30
27/27 [=====] - 2s 76ms/step - loss: 0.0256 - accuracy: 0.7084 - val_loss: 1.5941e-08 - val_accuracy: 0.7708

Epoch 00009: val_loss did not improve from 0.00000
Epoch 10/30
27/27 [=====] - 2s 75ms/step - loss: 0.0102 - accuracy: 0.8416 - val_loss: 1.8706e-19 - val_accuracy: 0.8958

Epoch 00010: val_loss improved from 0.00000 to 0.00000, saving model to best_model.h5
Epoch 11/30
27/27 [=====] - 2s 74ms/step - loss: 0.5554 - accuracy: 0.8500 - val_loss: 0.1317 - val_accuracy: 0.8125

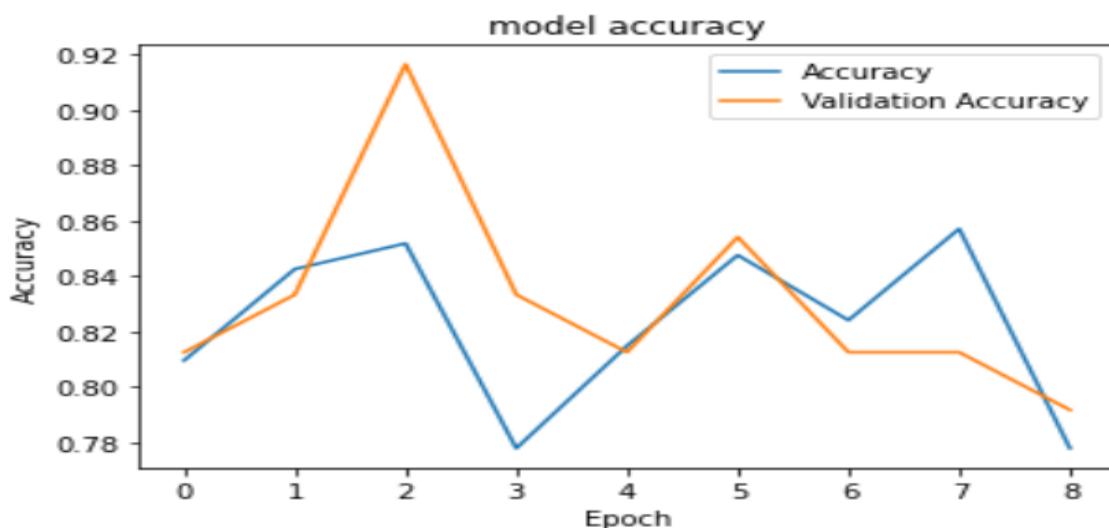
Epoch 00011: val_loss did not improve from 0.00000
Epoch 12/30
27/27 [=====] - 2s 73ms/step - loss: 0.1609 - accuracy: 0.8839 - val_loss: 6.2148e-08 - val_accuracy: 0.8125

Epoch 00012: val_loss did not improve from 0.00000
Epoch 13/30
27/27 [=====] - 2s 74ms/step - loss: 1.3678e-06 - accuracy: 0.7840 - val_loss: 1.3692e-07 - val_accuracy: 0.7917
Restoring model weights from the end of the best epoch.

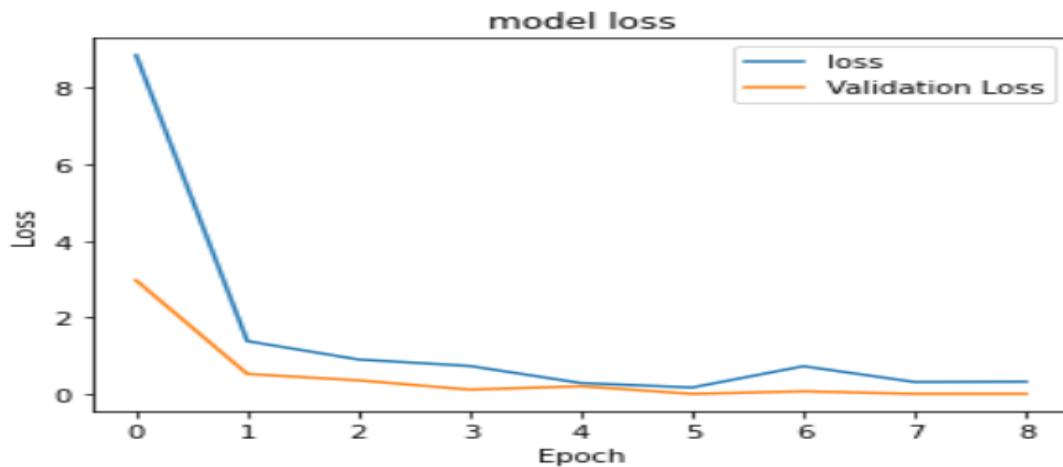
Epoch 00013: val_loss did not improve from 0.00000
Epoch 00013: early stopping
    
```

4-4Précision et erreur

- Du point de vue des résultats, après apprentissage du modèle, on obtient une précision de validation de 91,00% (stockée par la fonction callback).

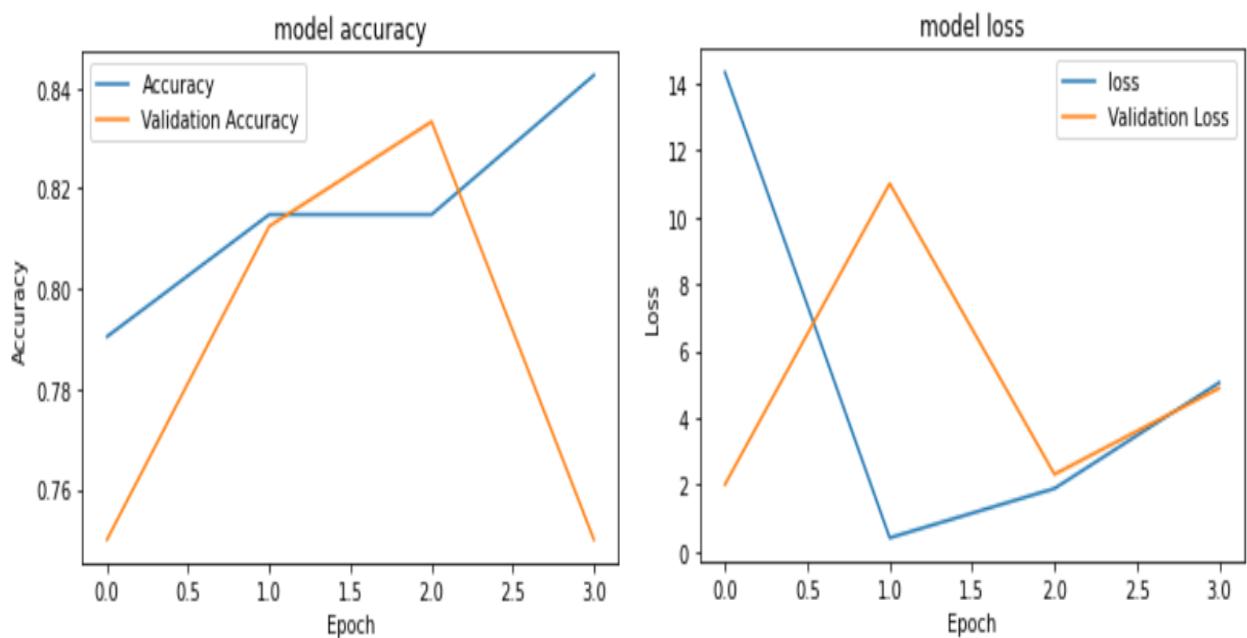


Rapport de stage d'ingénieur



4-5Autres modèles

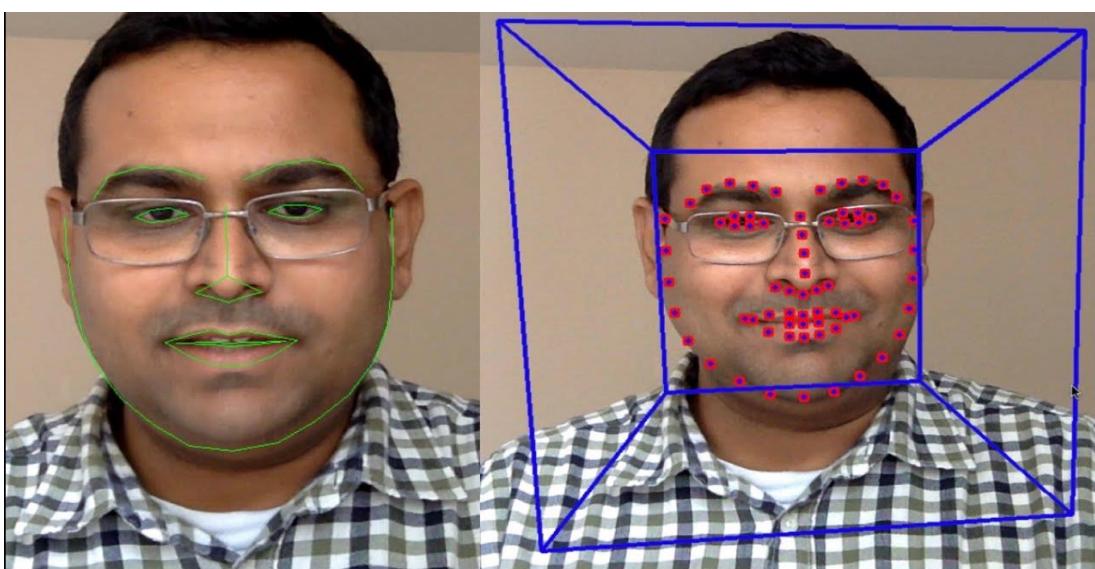
- Avec VGG16, nous avons également essayé avec les modèles pré-entraînés suivants : Resnet50, EfficientNetB0, InceptionV3.
- En comparant les résultats obtenus sur nos données, VGG16 donne la plus grande précision et la plus faible erreur.
- Résultats du modèle EfficientNetB0 :



La Bibliothèque Face_recognition

1-Introduction

- Le logiciel de reconnaissance faciale est génial. Le fait que nous soyons capables d'écrire un logiciel qui détermine avec précision où se trouvent les yeux et le nez de quelqu'un dans une image m'étonne toujours et le fait qu'il existe des bibliothèques pour ce genre de choses est génial. Ces bibliothèques aident à réduire la barrière à l'entrée pour les débutants qui cherchent à écrire leurs propres systèmes de reconnaissance faciale et permettent aux gens de faire des choses vraiment cool.
- Supposons que nous souhaitions identifier quel visage est présent dans une image donnée, il y a plusieurs choses que nous pouvons considérer comme un motif :
 - Le rapport entre la hauteur du visage et la largeur du visage (car il ne changera pas quelque soit la dimension de visage dans l'image)
 - Couleur du visage.
 - Largeur d'autres parties du visage comme la bouche, les lèvres, les yeux, le nez, etc...
- Ces repères de visage représentent les principales caractéristiques de chaque visage et font la différence entre eux. La figure suivante montre les repères détectés par l'algorithme :



Rapport de stage d'ingénieur

- Pour la bibliothèque face_recognition, il fonctionne en quelques étapes :
 - Identifier un visage dans une image donnée
 - Identifier les traits spécifiques du visage
 - Générer un vecteur d'encodage de visage de 128 valeurs
- Une fois que nous avons codé chaque image dans un vecteur de caractéristiques, le problème devient beaucoup plus simple. De toute évidence, lorsque nous avons 2 visages (images) qui représentent la même personne, les vecteurs de caractéristiques dérivés seront assez similaires. En d'autres termes, la "distance" entre les 2 vecteurs de caractéristiques sera assez petite.

2-Code

- J'ai commencé par l'importation des bibliothèques

```
import dlib
import face_recognition
from PIL import Image, ImageDraw
import numpy as np
import cv2
import time
from keras.preprocessing import image
```

- Ensuite, j'ai encodé mon visage et j'ai l'enregistré avec mon nom et prénom

```
wacef_image = face_recognition.load_image_file("/content/F16.jpg")
wacef_face_encoding = face_recognition.face_encodings(wacef_image)[0]

# Create arrays of known face encodings and their names
known_face_encodings = [
    wacef_face_encoding
]
known_face_names = [
    "Wacef chachia"
]
```

Rapport de stage d'ingénieur

- Puis, je commence à lire les images à partir du caméra, détecter les visages dans chaque image et les encoder

```
cam = cv2.VideoCapture(0)

pTime = 0
while True:
    ret, frame = cam.read()

    cols = frame.shape[1]
    rows = frame.shape[0]
    img = cv2.resize(frame, target_size)

    face_locations = face_recognition.face_locations(img)
    face_encodings = face_recognition.face_encodings(img, face_locations)

    pil_image = Image.fromarray(img)
    draw = ImageDraw.Draw(pil_image)
```

- Enfin, on compare les visages qui sont déjà enregistré (dans cette cas seul mon visage) avec les nouveaux visages encodés. Il y a deux cas possibles :

- Le nouveau visage encodé a une similarité à une des visages déjà enregistrés ≥ 0.6 (valeur par défaut qu'on peut modifier)
 ➔ L'algorithme va afficher le nom le plus similaire.
- Aucun visage enregistré n'a une similarité \geq avec le nouveau visage détecté
 ➔ L'algorithme va afficher 'Unknown'.

```
# Loop through each face found in the unknown image
for (top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):
    # See if the face is a match for the known face(s)
    matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
    name = "Unknown"

    # Or instead, use the known face with the smallest distance to the new face
    face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
    best_match_index = np.argmin(face_distances)
    if matches[best_match_index]:
        name = known_face_names[best_match_index]

    # Draw a box around the face using the Pillow module
    draw.rectangle(((left, top), (right, bottom)), outline=(0, 0, 255))

    # Draw a label with a name below the face
    text_width, text_height = draw.textsize(name)
    draw.rectangle(((left, bottom - text_height - 10), (right, bottom)), fill=(0, 0, 255), outline=(0, 0, 255))
    draw.text((left + 6, bottom - text_height - 5), name, fill=(255, 255, 255, 255))

    cTime = time.time()
    fps = 1 / (cTime - pTime)
    pTime = cTime
    draw.text((left + 100, bottom - text_height - 5), f'FPS: {int(fps)}', fill=(255, 255, 255, 255))
```

Rapport de stage d'ingénieur

3-Conclusion

- L'algorithme basé sur la bibliothèque de reconnaissance faciale nous donne une bonne précision avec une faible latence, ce qui nous incite à l'utiliser dans notre solution finale.

LBPH (Local Binary Patterns Histogram)

1-Introduction

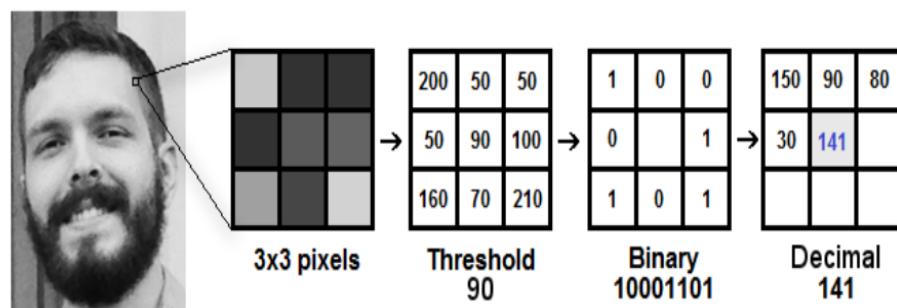
- Le motif binaire local (LBP : Local Binary Pattern en anglais) est un opérateur de texture simple mais très efficace qui étiquette les pixels d'une image en seuillant le voisinage de chaque pixel et considère le résultat comme un nombre binaire.
- Lorsque LBP est combiné avec le descripteur Histogramme de gradients orientés (HOG), il améliore considérablement les performances de détection et il est nommé algorithme LBPH.
- Ce dernier (LBPH) est fourni par la bibliothèque Python OpenCV.

2-Etapes

- L'algorithme LBPH a 4 étapes :
 - **Paramètres** : Le LBPH utilise 4 paramètres
 - Radius : le rayon est utilisé pour construire le motif binaire local circulaire. Il est généralement défini sur 1.
 - Neighbors : le nombre de points d'échantillonnage pour construire le motif binaire local circulaire. Il est généralement fixé à 8.
 - Grid X : le nombre de cellules dans le sens horizontal. Il est généralement fixé à 8.
 - Grid Y : le nombre de cellules dans le sens vertical. Il est généralement fixé à 8.
 - **Entraînement de l'algorithme** : Nous devons utiliser un ensemble de données avec les images faciales des personnes que nous voulons reconnaître. Nous devons également définir un ID pour chaque image, de sorte que l'algorithme utilisera ces informations pour reconnaître une image d'entrée et vous donner une sortie.

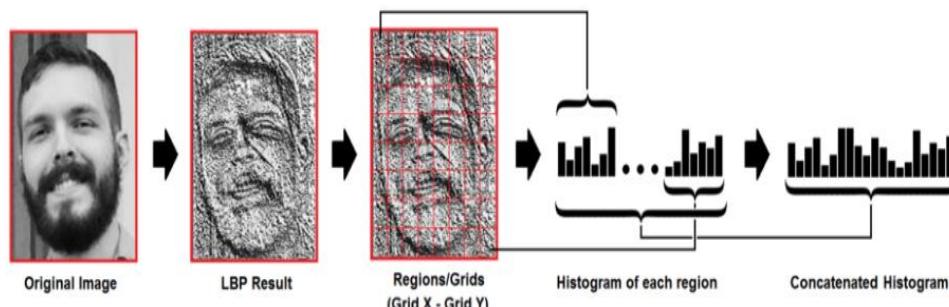
Rapport de stage d'ingénieur

- **Application de l'opération LBP** : Créez une image intermédiaire qui décrit mieux l'image originale, en mettant en évidence les caractéristiques du visage. Pour ce faire, l'algorithme utilise un concept de fenêtre glissante, basé sur les paramètres 'Radius' et 'Neighbors'.



A la fin de cette procédure (procédure LBP), nous avons une nouvelle image qui représente mieux les caractéristiques de l'image originale.

- **Extraction des histogrammes** : En utilisant l'image générée à la dernière étape, nous pouvons utiliser les paramètres Grille X et Grille Y pour diviser l'image en plusieurs grilles.



L'histogramme final représente les caractéristiques de l'image d'origine.

- **Exécution de la reconnaissance faciale** : Dans cette étape, l'algorithme est déjà formé. Étant donné une image d'entrée, nous effectuons à nouveau les étapes pour cette nouvelle image et créons un histogramme qui représente l'image. Ainsi, pour trouver l'image qui correspond à l'image d'entrée, il suffit de comparer deux histogrammes et de renvoyer

Rapport de stage d'ingénieur

l'image avec l'histogramme le plus proche (il existe diverses approches pour comparer les histogrammes : distance euclidienne, khi carré, valeur absolue, etc...).

3-Code

- Créer le modèle et charger les histogrammes enregistrés

```
recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read('trainer.yml')
```

- Reconnaître les visages détectés

```
for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
    id, confidence = recognizer.predict(gray[y:y + h, x:x + w])
    if (confidence < 100):
        id = names[id]
        confidence = " {0}%".format(round(100 - confidence))
    else:
        # Unknown Face
        id = "Who are you ?"
        confidence = " {0}%".format(round(100 - confidence))

    cv2.putText(img, str(id), (x + 5, y - 5), font, 1, (255, 255, 255), 2)
    cv2.putText(img, str(confidence), (x + 5, y + h - 5), font, 1, (255, 255, 0), 1)
```

4-Conclusion

- LBPH est l'un des algorithmes de reconnaissance faciale les plus simples et il peut donner d'excellents résultats.

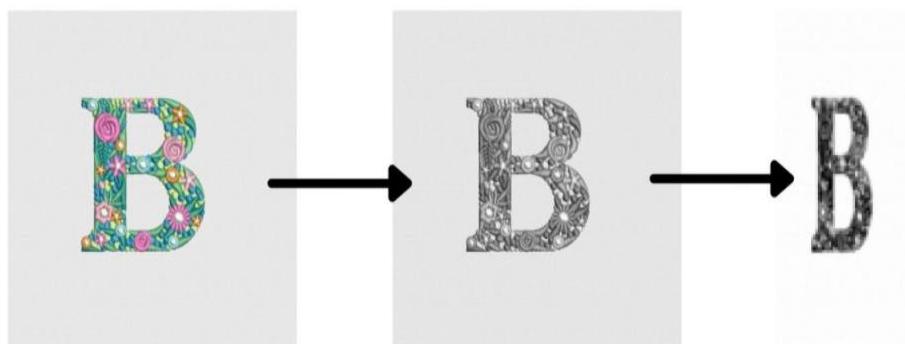
HOG & SVM

1-Introduction

- Histogram of Oriented Gradients, également connu sous le nom de HOG, est un descripteur de caractéristiques utilisé dans la vision par ordinateur et le traitement d'images à des fins de détection d'objets. La technique compte les occurrences d'orientation de gradient dans la partie localisée d'une image.
- HOG est meilleur que n'importe quel descripteur de bord car il utilise la magnitude ainsi que l'angle du gradient pour calculer les caractéristiques. Pour les régions de l'image, il génère des histogrammes en utilisant la magnitude et les orientations du gradient.

2-Etapes

- **Redimensionner l'image** : Redimensionner l'image en une image de 128x64 pixels et convertir l'image en niveaux de gris.



- **Calcul du gradient** : Calculer le gradient de l'image, dont on obtient en combinant la magnitude et l'angle de l'image. On commence par calculer G_x et G_y de chaque pixel avec r : n° ligne et c : n° colonne.

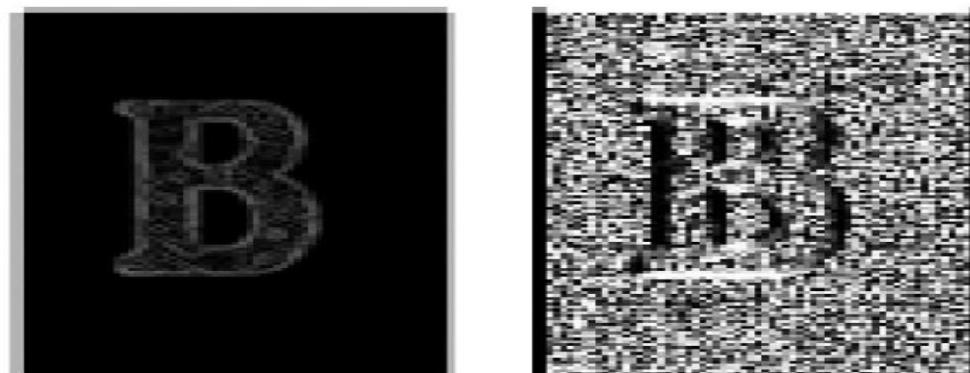
$$G_x(r, c) = I(r, c + 1) - I(r, c - 1) \quad G_y(r, c) = I(r - 1, c) - I(r + 1, c)$$

Rapport de stage d'ingénieur

Puis on calcule la magnitude et l'angle de chaque pixel

$$\text{Magnitude}(\mu) = \sqrt{G_x^2 + G_y^2} \quad \text{Angle}(\theta) = |\tan^{-1}(G_y/G_x)|$$

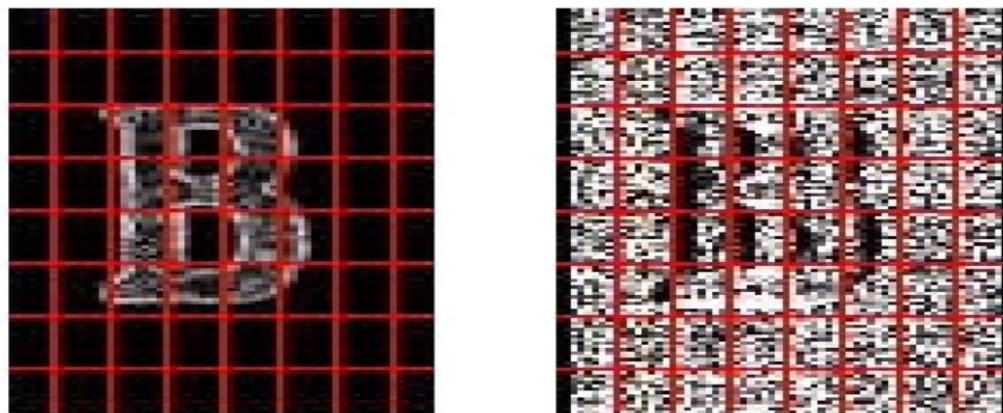
Voici le résultat de cette étape :



- **Calcul des histogrammes des gradients** : Les matrices de gradient (matrice de magnitude et d'angle) sont divisées en 8x8 cellules pour former un bloc. Pour chaque bloc, un histogramme de 9 points est calculé. Le calcul se fait sur chaque bloc par la formule suivante :

$$\begin{aligned} \text{Number of bins} &= 9 (\text{ranging from } 0^\circ \text{ to } 180^\circ) \\ \text{Step size}(\Delta\theta) &= 180^\circ / \text{Number of bins} = 20^\circ \end{aligned}$$

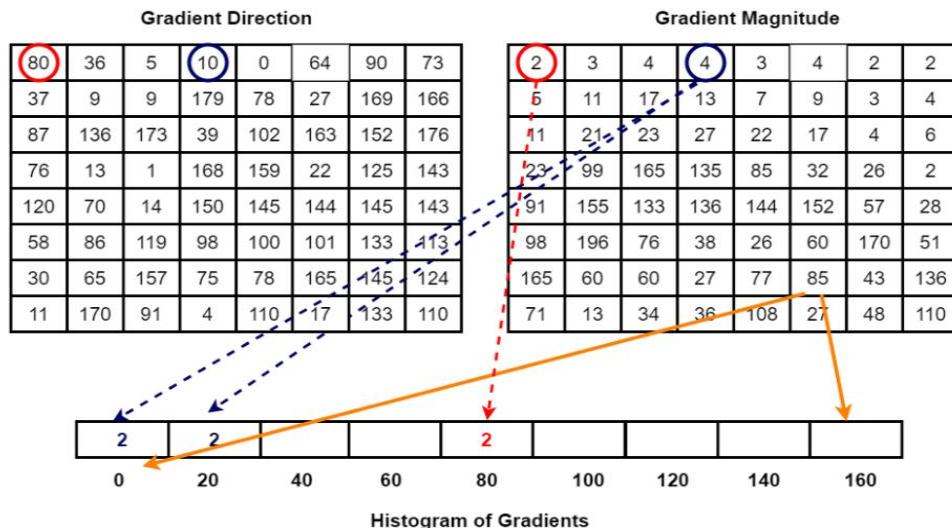
Voici le résultat de matrice d'angle :



Rapport de stage d'ingénieur

Une fois les deux matrices (magnitude et angle) calculées, on calcule l'histogramme du gradient.

Voici un exemple de calcul d'histogramme 9 bin :



- **Normalisation** : Cette normalisation est faite pour réduire l'effet des changements de contraste entre les images d'un même objet. Chaque bloc (un vecteur de 36 points) est normalisé par la norme L2.

$$k = \sqrt{b_1^2 + b_2^2 + b_3^2 + \dots + b_{36}^2}$$

$$f_{bi} = \left[\left(\frac{b_1}{k} \right), \left(\frac{b_2}{k} \right), \left(\frac{b_3}{k} \right), \dots, \left(\frac{b_{36}}{k} \right) \right]$$

Voici le résultat final :



Rapport de stage d'ingénieur

3-Code

```
from skimage.feature import hog
from sklearn.svm import SVC
```

```
ppc =8
cb=4

#tr-data
tr_hog_features=[]
tr_hog_image=[]
for image in tqdm(tr_data_gray):
    fd , hogim = hog(image , orientations=9 , pixels_per_cell=(ppc , ppc) , block_norm='L2' , cells_per_block=(cb,cb) , visualize=True )
    tr_hog_image.append(hogim)
    tr_hog_features.append(fd)
```

```
svm = SVC(kernel='rbf' , class_weight='balanced' , C=1000 , gamma=0.0082)
svm.fit(tr_hog_features,tr_labels)
```

4-Conclusion

- Connu que HOG est meilleur que n'importe quel descripteur de bord, les expériences pratiques montrent l'efficacité de celui-ci avec SVM dans la création d'un algorithme de reconnaissance faciale rapide et précis.

Rapport de stage d'ingénieur

Partie matérielle

LsVision IpCamera

1-Introduction

- La caméra LsVision ls-zd7310e-p est l'un des modèles caméra IP les plus efficaces car elle prend en charge deux protocoles importants :
 - **RTSP** : Real Time Streaming Protocol est un protocole de communication au niveau de l'application (niveau 7 du modèle OSI) destiné aux systèmes de streaming multimédia qui réduisent la latence du flux vidéo et facilitent le traitement vidéo.
 - **ONVIF** : est l'acronyme de l'Open Network Video Interface Forum qui permet aux produits de vidéosurveillance IP d'échanger des informations entre eux. En d'autres termes Avec le protocole ONVIF, les caméras réseau, les enregistreurs vidéo en réseau (NVR ou DVR) et les logiciels de gestion des caméras de surveillance, même de fabricants différents, peuvent tous communiquer entre eux.



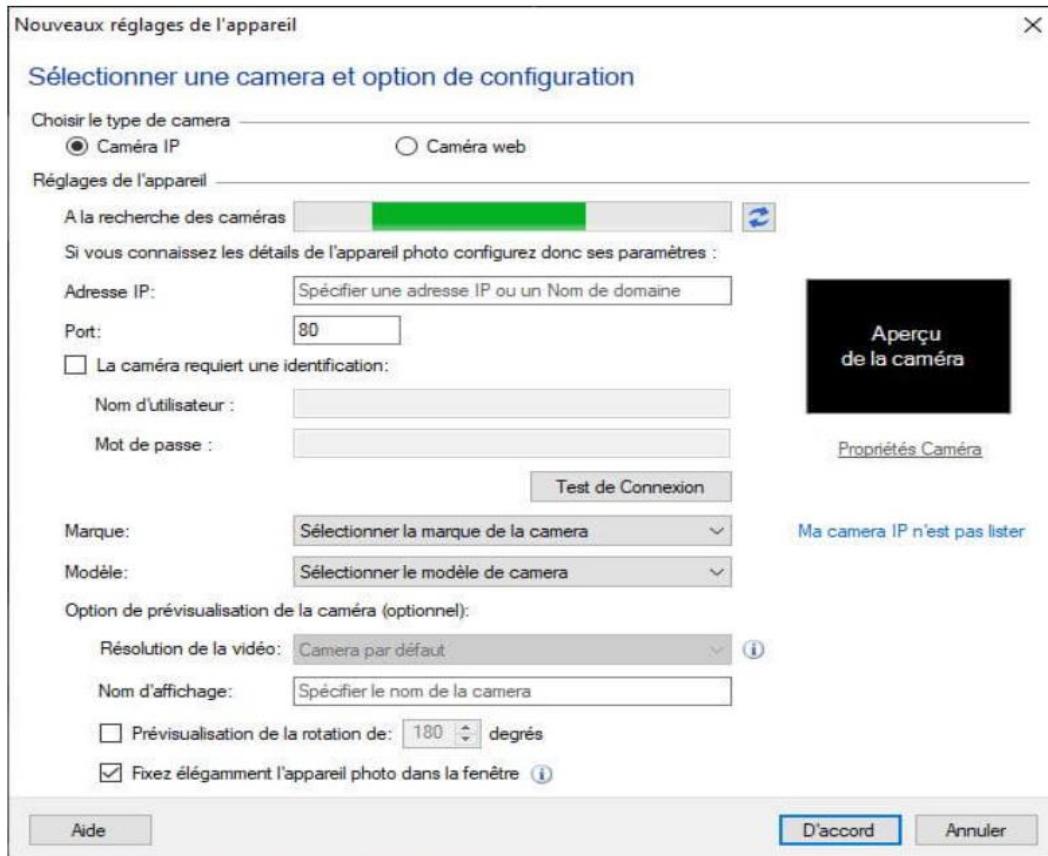
- Ses caractéristiques techniques :
 - **Résolution Capteur : 2 MP**
 - **Capteur d'image : CMOS**
 - **Objectif fixe standard de 3.6 mm**
 - **Type de Caméra : IP**
 - **20fps @ 1920 x 1080**
 - **20 mètres IR**

Rapport de stage d'ingénieur

- Étanche IP66

2-Caméra IP avec Viewer4

- Téléchargez et installez le logiciel [IP Camera Viewer](#)
- Ouvrez le logiciel puis cliquez sur « Ajouter une caméra »



- Entrez ensuite les informations que vous avez récupérées pour récupérer le flux vidéo.

3-Caméra IP avec Python

- L'url du flux RTSP est de cette forme :

```
rtsp://user:password@ipaddress:rtspPort
```

Rapport de stage d'ingénieur

- Pour lire le flux vidéo avec python, on utilise la fonction VideoCapture de OpenCV

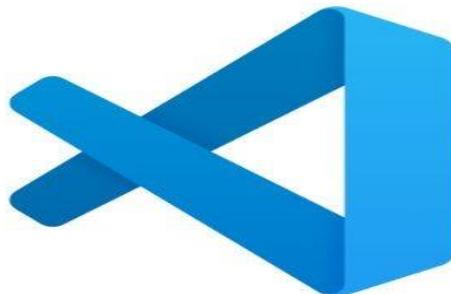
```
cap = cv2.VideoCapture('rtsp://user:password@ipaddress:rtspPort')
```

- A partir du flux vidéo obtenu, on peut exécuter notre algorithme de reconnaissance faciale.

Les Outils informatiques

1-Éditeurs de code

- **Visual Studio Code** est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS. Les fonctionnalités incluent la prise en charge du débogage, la mise en évidence de la syntaxe, la complétion intelligente du code, les snippets, la refactorisation du code et Git intégré.



- **Colab** est un produit de Google Research. Il permet à n'importe qui d'écrire et d'exécuter le code Python de son choix par le biais du navigateur. C'est un environnement particulièrement adapté au machine learning, à l'analyse de données et à l'éducation. En termes plus techniques, Colab est un service hébergé de notebooks Jupyter qui ne nécessite aucune configuration et permet d'accéder gratuitement à des ressources informatiques, dont des GPU.



Rapport de stage d'ingénieur

- **IP Camera Viewer** est un système de caméra de sécurité qui vous permet de visionner des vidéos en direct à partir de caméras IP ou USB. Il peut être utilisé dans n'importe quel endroit où vous avez besoin de sécurité. Il prend en charge plus de 2000 modèles de caméras IP différents. Caractéristiques : Il télécharge automatiquement les vidéos enregistrées sur un serveur



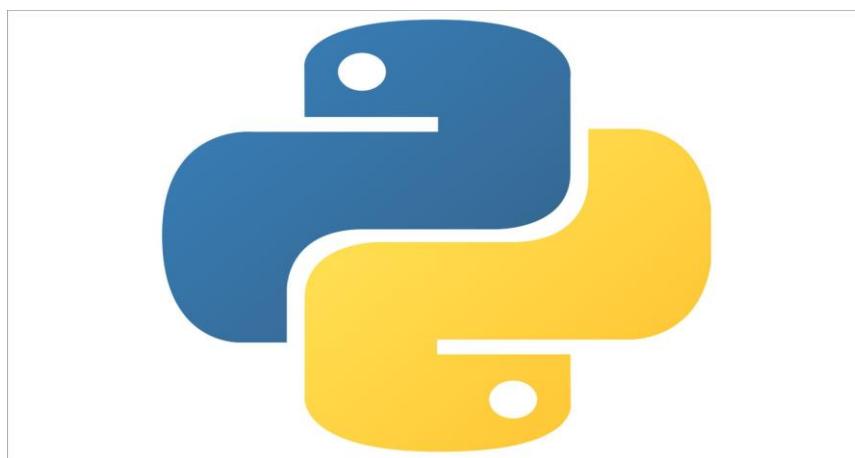
- ONVIF concerne le matériel fonctionnant avec le Protocole Internet ou IP. Cette technologie est une évolution de la technologie analogique, conçue pour faciliter l'accès aux caméras et à leurs paramètres sans plus avoir à passer par un enregistreur.



Rapport de stage d'ingénieur

Langage de programmation

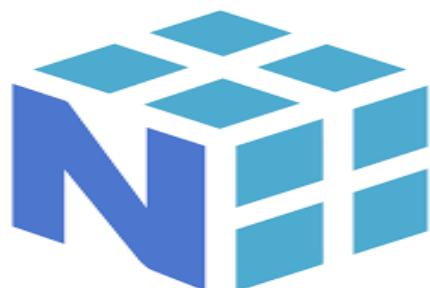
- Python est un langage de programmation interprété, orienté objet, de haut niveau avec une sémantique dynamique. ... La syntaxe simple et facile à apprendre de Python met l'accent sur la lisibilité et réduit donc le coût de maintenance du programme. Python prend en charge les modules et les packages, ce qui encourage la modularité du programme et la réutilisation du code.



Les bibliothèques

1-Numpy

- Numpy est une bibliothèque pour langage de programmation Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux.



Rapport de stage d'ingénieur

2-Matplotlib

- Matplotlib est une bibliothèque du langage de programmation Python destinée à tracer et visualiser des données sous formes de graphiques. Elle peut être combinée avec les bibliothèques python de calcul scientifique NumPy et SciPy. Matplotlib est distribuée librement et gratuitement sous une licence de style BSD.



3-OpenCV

- OpenCV est une bibliothèque graphique libre, initialement développée par Intel, spécialisée dans le traitement d'images en temps réel. La société de robotique Willow Garage et la société ItSeez se sont succédé au support de cette bibliothèque.



Rapport de stage d'ingénieur

4-TensorFlow

- TensorFlow est une bibliothèque logicielle gratuite et open source pour le flux de données et la programmation différentiable pour une gamme de tâches. Il s'agit d'une bibliothèque mathématique symbolique, également utilisée pour les applications d'apprentissage automatique telles que les réseaux de neurones. Il est utilisé à la fois pour la recherche et la production chez Google.



5-Keras

- Keras est une bibliothèque de réseaux de neurones open source écrite en Python. Conçu pour permettre une expérimentation rapide avec les réseaux de neurones profonds, il se concentre sur la convivialité et l'extensibilité. Keras contient de nombreuses implémentations de blocs de construction de réseaux de neurones couramment utilisés, tels que des couches, des objectifs, des fonctions d'activation, des optimiseurs et une multitude d'outils pour faciliter le travail avec les données d'image et de texte.



Rapport de stage d'ingénieur

6-OS

- Le module OS en python fournit des fonctions pour interagir avec le système d'exploitation. OS, relève des modules utilitaires standard de Python. Ce module fournit un moyen portable d'utiliser les fonctionnalités dépendantes du système d'exploitation. Les modules *os* et *os.path* incluent de nombreuses fonctions pour interagir avec le système de fichiers.



7-Pillow

- Python Imaging Library est une bibliothèque de traitement d'images pour le langage de programmation Python. Elle permet d'ouvrir, de manipuler, et de sauvegarder différents formats de fichiers graphiques. La bibliothèque est disponible librement selon les termes de la Python Imaging Library license



8-Dlib

- Dlib est une boîte à outils C++ moderne contenant des algorithmes d'apprentissage automatique et des outils permettant de créer des logiciels complexes en C++ pour résoudre des problèmes du monde réel. Il est utilisé à la fois dans l'industrie et dans les universités dans un large éventail de domaines, notamment la robotique, les appareils embarqués, les téléphones mobiles et les grands environnements informatiques haute performance.

Rapport de stage d'ingénieur



9-Pickle

- Pickle est utilisé pour sérialiser et désérialiser les structures d'objets Python. La sérialisation fait référence au processus de conversion d'un objet en mémoire en un flux d'octets pouvant être stocké sur disque ou envoyé sur un réseau. Plus tard, ce flux de caractères peut ensuite être récupéré et désérialisé en un objet Python.



10-Face-recognition

- Reconnaissez et manipulez les visages depuis Python ou depuis la ligne de commande avec la bibliothèque de reconnaissance faciale la plus simple au monde. Construit à l'aide de la reconnaissance faciale de pointe de Dlib, construit avec un apprentissage en profondeur. Le modèle a une précision de 99,38% sur l'indice de référence Labeled Faces in the Wild. Cela fournit également un outil de ligne de commande simple qui vous permet d'effectuer une reconnaissance faciale sur un dossier d'images à partir de la ligne de commande.

11-Scikit-learn

- Scikit-learn est une bibliothèque d'apprentissage automatique gratuite pour Python. Il comporte divers algorithmes tels que la machine à vecteurs de support, les forêts aléatoires et les k-voisins, et il prend également en charge les bibliothèques numériques et scientifiques Python telles que NumPy et SciPy.

Rapport de stage d'ingénieur



12-MTCNN

- MTCNN est une bibliothèque python (pip) écrite par l'utilisateur de Github ipacz, qui implémente l'article Zhang, Kaipeng et al. « Détection et alignement conjoints des visages à l'aide de réseaux convolutifs multitâches en cascade. »

13-Skimage

- Scikit-image, ou skimage, est un package Python open source conçu pour le prétraitement d'images.

14-Joblib

- Joblib offre un meilleur moyen d'éviter de recalculer la même fonction de manière répétitive, ce qui permet d'économiser beaucoup de temps et de coûts de calcul.



Rapport de stage d'ingénieur

Conclusion

- L'objectif de ce stage est de concevoir et d'implémenter une application de reconnaissance faciale capable, en temps réel, de reconnaître les visages. Vu la quantité de logiciels potentiels (sécurité, réseaux sociaux,) pouvant se baser sur cette application, celle- ci doit répondre à des exigences de rapidité et de robustesse des résultats.
- Les systèmes de suivi de visage se sont beaucoup développés ces dernières années grâce à l'amélioration du matériel et à la forte demande industrielle, par exemple pour la recherche d'individu par le biais d'une caméra de surveillance.
- Ce fut une expérience formidable en tant que stagiaire en vision par ordinateur à SFM Telecom, cela m'a aidé à approfondir mes connaissances en intelligence artificielle qui était un nouveau domaine pour moi, était un défi dès le début. Ensuite, après avoir fait quelques recherches en ligne à ce sujet et avec de la pratique, je réussis à faire le travail comme il est censé être.