



SYSTEMES INTELIGENT ET MULTIMÉDIAS (SIM)  
PROMOTION 23 (2018-2020)

---

**RECONNAISSANCE DES FORMES**  
**Application de contrôle de présence scolaire**

---

*Auteur :*

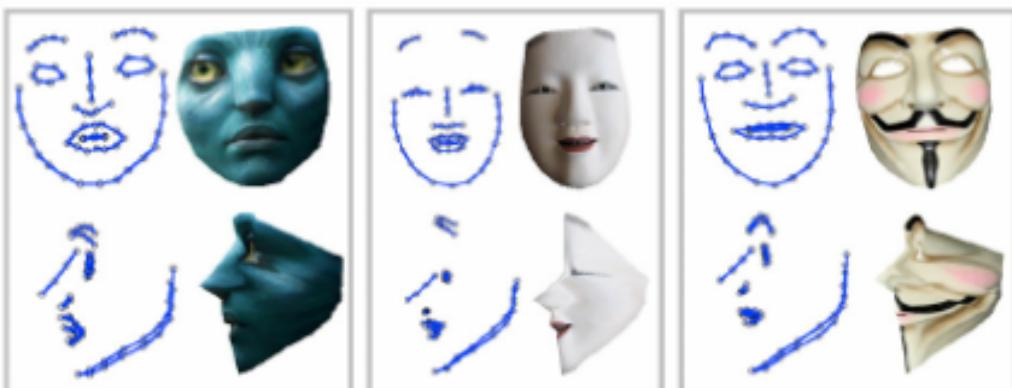
Kenley FAVARD  
Eric Papain MEZATIO

*Encadrer par :*

DR HO Tuong Vinh

*Adresse :*

ho.tuong.vinh@gmail.com



# Résumé

Dans la majeur partie des activités de l'Homme, intervient depuis une décennie la vision par ordinateur, la reconnaissance de forme qui lui permettent de prendre de façon plus pointue des décisions, ainsi donc, les chaînes de télé pour la recherche de certains personnages dans un contexte ou dans une vidéo donnée, les postes de contrôle dans l'armée, les services médicaux pour le contrôle et la reconnaissance des patients, les vidéos de surveillance pour le tracking des joueurs dans un stade de football, de basket, de volley.... pour n'en citer que ceux-ci, son contexte ou la nécessité d'utiliser des applications de reconnaissance sont très demandées et convoitées. Ainsi donc, nous nous sommes concentrés dans notre cas à la question de comment utiliser une application pour permettre et faciliter le contrôle de présence des étudiants dans une salle de cours afin de savoir à tout moment qui est présent et qui est absent et aussi de maintenir à jour l'information sur les différents étudiants de la salle afin de s'assurer à tout moment qu'il n'y est pas d'intuis. Dans ce contexte, la vidéo surveillance est une solution innovante qui peut permettre de résoudre de façon raisonnable ce problème et fournir un environnement sécurisé. L'idée serait de placer un réseau de caméras dans la salle de cours pour détecter automatiquement les personnes présentes et par la suite, notre application qui contient un modèle qui est formé pour reconnaître l'ensemble des étudiants de la salle détectera les différents visages et mentionnera sur la tête de chaque visage détecté le nom de l'individu et si l'individu n'a pas été reconnu par l'application, il sera signalé alors comme intrus. Quand des étudiants sont détectés, à la fin l'application devra dire le nombre de personnes détectées et leurs noms afin de permettre de voir les étudiants absents. Puisque notre système est à bas coût, nous avons utilisé deux caméras pour les tests une webcam externe et une webcam de notre ordinateur.

nous avons réalisé plusieurs expérimentations basées dans un premier temps sur les bibliothèques existantes dans opencv (Harr, Dnn), nous avons également expérimenté des techniques basées sur le deep learning en utilisant les réseaux de neurones convolutionnels qui nous ont permis d'obtenir des résultats un peu mieux mais faute de la quantité de données, nous n'avons pas poussé plus loin avec cette approche, mais les travaux de recherche n'étant pas limités, nous continuons à travailler pour agrandir notre base de données d'apprentissage car d'après les articles lus, pour une bonne prédiction avec les CNNs, il est préférable d'avoir au moins 1000 échantillons par classe, or nous avons une somme à près de 300 à 400 échantillons par classe, ce qui fausse un peu les meilleurs résultats mais avec les modèles Haar descriptor implémentés dans opencv, nous arrivons à faire une reconnaissance à temps réel des personnes préformée dans le modèle.

Pour améliorer ce travail, nous continuons à constituer nos différentes classes afin d'atteindre 1000 échantillons par classe (nous avons 6 classes pour 6 étudiants) afin de tester et de tirer des meilleures conclusions sur l'utilisation des CNNs pour la reconnaissance à temps réel des personnes qui s'annonce plutôt meilleures dans les prédictions des visages.

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Contexte . . . . .	1
1.2	Motivation . . . . .	2
1.3	Objectifs . . . . .	2
1.4	Contribution . . . . .	3
<b>2</b>	<b>Analyse du sujet</b>	<b>4</b>
2.1	Quel est le domaine d'application . . . . .	4
2.2	Quel est le problème visé ? . . . . .	4
2.3	Qu'est-ce qu'on veut résoudre ? . . . . .	4
2.4	Quelle est la situation actuelle ? . . . . .	4
2.5	Résultats attendus . . . . .	5
<b>3</b>	<b>État de l'art</b>	<b>6</b>
3.1	Introduction et généralités . . . . .	6
3.2	Quelques techniques de détection et de reconnaissance faciale . . . . .	7
3.2.1	Le traitement automatique du visage . . . . .	7
3.2.2	Eigenface . . . . .	7
3.2.3	L'analyse des points particuliers . . . . .	7
3.2.4	LDA (Linear Discriminant Analysis) fisher . . . . .	7
3.2.5	Descripteur de Haar . . . . .	7
3.2.6	Methode LBP (Local Binary Patterns) . . . . .	8
3.2.7	Utilisation des DNN(deep neural netwoork) . . . . .	8
3.3	Bilan des solutions existants . . . . .	9
<b>4</b>	<b>Solution</b>	<b>10</b>
4.1	Données . . . . .	10
4.1.1	Constitution du jeux de données . . . . .	10
4.2	Outils . . . . .	13
4.2.1	Présentation de la bibliothèque OpenCV . . . . .	13
4.2.1.1	Classifieur OpenCV . . . . .	13
4.2.2	Pré-traitement . . . . .	14
4.2.3	Utilisation des Réseaux de Neurone Convolutionels . . . . .	14
4.3	Modèles conceptuels / Pipeline de traitement . . . . .	14
4.3.1	Modèles conceptuels . . . . .	14
4.3.1.1	Diagramme de classe utilisation Haar . . . . .	15
4.3.1.2	Modèle de conception de l'architecture de notre CNN . . . . .	15
4.3.2	Pipeline de traitement . . . . .	16
<b>5</b>	<b>Implémentation et expérimentation</b>	<b>17</b>
5.1	Introduction . . . . .	17
5.2	Implémentation Expérimentation . . . . .	17
5.2.1	Implementation des Cnns . . . . .	17
5.2.1.1	Logiciels et librairies Utilisés dans l'implémentation . . . . .	17
5.2.1.2	Architecture de notre réseau . . . . .	18
5.2.2	Implémentation utilisant opencv . . . . .	19
<b>6</b>	<b>Conclusion et perspectifs</b>	<b>26</b>

# Chapitre 1

## Introduction

Nous vivons dans un monde numérique, où les informations sont stockées, traitées, indexées et recherchées par des systèmes informatiques, ce qui offre des temps d'accès rapide et pas cher. Au cours des dernières années, des progrès considérables ont été réalisés dans le domaine de la vision par ordinateur, poussant ses limites jusqu'à la reconnaissance des formes en passant par la classification d'images. Ce progrès est dû aux nombreux travaux dans ce domaine et à la disponibilité des bases d'images internationales qui ont permis aux chercheurs de signaler de manière crédible l'exécution de leurs approches dans ce domaine, avec la possibilité de les comparer à d'autres approches utilisant les mêmes bases. Dans la fin des années 80 Yan Le Cun a développé un type de réseau particulier qui s'appelle le réseau de neurone convolutionnel, ces réseaux sont une forme particulière de réseau neuronal multicouche dont l'architecture des connexions est inspirée de celle du cortex visuel des mammifères. Par exemple, chaque élément n'est connecté qu'à un petit nombre d'éléments voisins dans la couche précédente. En 1995, Yan Le Cun et deux autres ingénieurs ont développé un système automatique de lecture de chèques qui a été déployé largement dans le monde. À la fin des années 90, ce système lisait entre 10 et 20 % de tous les chèques émis aux États-Unis. Mais ces méthodes étaient plutôt difficiles à mettre en œuvre avec les ordinateurs de l'époque, et malgré ce succès, les réseaux convolutionnels et les réseaux neuronaux plus généralement ont été délaissés par la communauté de la recherche entre 1997 et 2012. En 2011 et 2012 trois événements ont soudainement changé la situation. Tout d'abord, les GPU (Graphical Processing Unit) capables de plus de mille milliards d'opérations par seconde sont devenus disponibles pour un prix moins cher. Ces puissants processeurs spécialisés, initialement conçus pour le rendu graphique des jeux vidéo, se sont avérés être très performants pour les calculs des réseaux neuronaux. Deuxièmement, des expériences menées simultanément à Microsoft, Google et IBM avec l'aide du laboratoire de Geoff Hinton ont montré que les réseaux profonds pouvaient diminuer de moitié les taux d'erreurs des systèmes de reconnaissance vocale. Troisièmement plusieurs records en reconnaissance d'image ont été battus par des réseaux de neurones convolutionnels. L'événement le plus marquant a été la victoire éclatante de l'équipe de Toronto dans la compétition de reconnaissance d'objets « ImageNet ». La diminution des taux d'erreurs était telle qu'une véritable révolution. Du jour au lendemain, la majorité des équipes de recherche en parole et en vision ont abandonné leurs méthodes préférées et sont passées aux réseaux de neurones convolutionnels et autres réseaux neuronaux. L'industrie d'Internet a immédiatement saisi l'opportunité et a commencé à investir massivement dans des équipes de recherche et développements en apprentissage profond. Dans notre projet on va utiliser les réseaux de neurones convolutionnels pour la reconnaissance cette fois en temps réel des personnes dans une salle de classe, ensuite nous ferons une autre expérimentation en utilisant le descripteur de HAAR et le DNN(deep neural network) pour la reconnaissance implémenter dans opencv.

### 1.1 Contexte

La reconnaissance de visages et de détection est une technologie biométrique en vogue ces dernières années, elle est très utilisée dans les applications d'authentification, de contrôle d'accès et de vidéo de surveillance, on trouve plusieurs méthodes globales, locales et hybrides de reconnaissance de visages et/ou de détection de présence. Ces derniers temps les chercheurs de différents domaines ont orienté leurs travaux sur des clés et des mots de passe impossible à falsifier, sûre et surtout efficace. Et c'est ainsi que fut inventée la biométrie qui est devenue à la mode dans les domaines qui requiert un niveau élevé de sécurité et de contrôle.

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 nous permet d'exploiter de nombreuses informations relatives à une personne. Malgré l'énorme progression en informatique surtout dans le domaine du traitement d'images on trouve toujours des difficultés concernant les systèmes appliquant les techniques de reconnaissance faciale, c'est pour cela qu'elle a engendré un grand nombre de travaux de recherche au cours des 50 dernières années

et ceci dans différents domaines tels que l'identification de modèles, réseaux de neurones, vision par ordinateur, infographie ... Dans notre travail, on s'est justement intéressé à cette thématique fort utile et très demandée dans le domaine de la sécurité des réseaux, la sécurité nationale et même internationale. On s'est proposé d'implémenter une application de reconnaissance faciale et/ou de détection de présence utilisant la technique la plus performante de l'état de l'art.

## 1.2 Motivation

De nos jours, beaucoup de particuliers et d'entreprises installent des caméras de vidéosurveillance. Pourtant auparavant, seuls les plus grands commerces avaient les moyens de s'en procurer. Aujourd'hui, c'est différent : les produits de vidéosurveillance sont à la portée de tous. Mais quels sont les avantages ?

- elle offre une sérénité aux usagers en leur absence,
- elle permet de rassurer des personnes qui ne se sentent pas en sécurité,
- elle permet de détecter les vols, infractions, etc.
- elle permet également de surveiller les seniors en cas de besoin.
- permet également de faire un contrôle de présence dans un lieu de travail, un chantier etc...

Il y a quelques années, il était impossible d'accéder à distance à ses caméras, de lancer des enregistrements via son smartphone, etc. Grâce à la technologie de vidéosurveillance IP, il est tout à fait possible de visualiser ses caméras à distance via Internet. L'accès peut se faire sur un smartphone, ordinateur et/ou tablette. Les caméras IP possèdent des caractéristiques intéressantes :

- Accès et contrôle à distance,
- Enregistrement en local, dans le Cloud ou dans un autre lieu,
- Détection de mouvement avec envoi de photos ou vidéos,
- Notifications par mail en cas de détection de mouvement,
- Vision nocturne,Etc.

de plus, la vidéosurveillance n'est plus aussi cher qu'avant. elle est devenue, beaucoup moins onéreuse qu'auparavant offrant de très meilleurs résultats que la capacité humaine donnant ainsi la possibilité à tous les particuliers maintenant de s'équiper de vidéosurveillance, afin de dissuader, prévenir et sécuriser votre domicile ou lieu de travail.

Ainsi va notre motivation de mettre en place un système de surveillance à temps réel de contrôle de présence dans une salle de classe qui faciliteras ainsi l'heure d'arrivée des étudiants, leurs heures de départ,et aussi mettre un acsent sur l'assiduité,et aussi se rassurer de la non présence des intrus dans la salle car pour un nouvel enseignant qui arrive il est impossible de connaitre si les étudiants suivant son cours sont tous de la sale alors avec ce système nous pourrons facilement détecter des intrus, des absent et nous pouvons non seulement étendre ce système dans la surveillance scolaire, mais aussi dans plusieurs système interactif ou intreviens forcement la vidéo surveillance.

## 1.3 Objectifs

L'idée serait de placer un réseau de caméras dans la salle de cours pour détecter automatiquement les personne présente et par la suite, notre application qui contient un modèle qui est formé pour reconnaître l'ensemble des étudiants de la salle détectera les différents visage et mentionneras sur la tête de chaque visage détecter le nom de l'individu et si l'individu n'ai pas reconnue par l'application on signalera alors comme intrus. Quand des étudiants son détecter, a la fin l'application devras dire le nombre de personne détecter et leurs noms afin de permettre de voir les étudiants absent. puisque notre système est a bas coût nous allons utilisé deux caméras pour les test une webcam externe et une webcam de notre ordinateur.

nous allons réaliser plusieurs expérimentation basé dans un premier temps sur les bibliothèques existance dans opencv( Haar desciptor, Dnn, deep neural network, yolo), nous allons également expérimenté des technique basé sur le deep learning en utilisant cette fois les réseaux de neurone convolutionels qui nous permettrons d'obtenir des résultats un peu mieux mais faute de la quantité de données, nou n'allons pas poussé plus loin avec cette approche, mais les travaux de recherche n'étant pas limité, nous continuons a travailler pour agrandir notre base de donnée d'apprentissage car d'après les articles lu, pour une bonne prédiction avec les cnns, ils est convenable d'avoir au moins 1000 échantillons d'images par classe, or nous somme a près de 300 a 400 échantillons par classe, se qui fausse un peu les meilleurs résultats mais avec les modèles Haar desciptor implémenter dans opencv, nous allons faire une reconnaissance a temps réel des personnes préformé dans le modèle et afficher nos résultats commentés.

Pour améliorer se travail, nous continuons a constituer nos différentes classe afin d'atteindre 1000 échantillons par classe (nous avons 6 classes pour 6 étudiants) afin de tester et de tirer des meilleurs conclusion sur l'utilisation des Cnns pour la reconnaissance a temps réels des personnes qui s'anonce plutot meilleurs dans la prédiction des visages.

## 1.4 Contribution

Dans la réalisation de ce travail, notre contribution à été celle de l'utilisation et de l'extention des Cnns pour la reconnaissance a temps réel car dans les application existante qui font usage des Cnns, ils sont beaucoup plus utiliser dans la classification des images, mais dans notre contexte nous avons réaliser une approche expérimentale en utilisant les différents algorithme de reconnaissance de visage et de reconnaissance de forme qui existe, et a cela nous avons ajouter l'extention de pouvoir utiliser ses algorithme dans le contexte de suivie et de surveillance a temps réel des objets qui dans notre cas se sont les étudiants d'une sale de classe( dans notre cas d'exemple nous avons utiliser 6 classes dans notre expérimentation composé de 6 étudiants de l'Institut Francophone International pour la phase d'expérimentation) a la fin de notre travail nous avons obtenue des résultats satisfaisant mais qui ne challenge pas avec les méthodes actuellement existante basé sur des API déjà préconçu. notre travail continue toujours de se performer en se moment nous travaillons dans la constitution de notre dataset pour terminer avec les Cnns qui nous permettrons d'obtenir de résultats meilleurs.

# Chapitre 2

## Analyse du sujet

### 2.1 Quel est le domaine d'application

La vidéosurveillance couvre plusieurs domaines d'application. La lutte contre le terrorisme est l'un des plus connus. Nous avons plusieurs enquêtes qui connaissent des succès non seulement sur le plan préventif, mais pour l'enquête criminelle, en permettant d'identifier des visages et de reconstituer le trajet des personnes suspecte et ou des zones à risque (grâce à l'analyse des images enregistrées durant une certaine période dans un espace bien déterminé).

C'est dans la lutte contre la petite et moyenne délinquance que la vidéosurveillance se révèle la plus efficace. La reconnaissance faciale permet, par exemple, de repérer et d'identifier d'éventuels des fauteurs de trouble aux abords ou sur les gradins des stades. De même, les statistiques de la criminalité montrent que les vols à l'étalage ou à l'arraché régressent sensiblement dans les centres commerciaux ou dans les rues des centres-villes équipés de caméras.

L'installation de caméras dans les transports en commun (trains, bus et métros) poursuit trois objectifs : la sécurité des voyageurs, la lutte contre la fraude et la protection contre les dégradations (non seulement dans les rames, mais aussi dans les gares, sur les quais et, depuis peu, le long de certaines voies). La surveillance des sites sensibles, publics et privés, liés à des intérêts vitaux et stratégiques sur un plan national (énergie, défense, transport).représente également l'un des domaines d'application les plus courants. Elle permet de contrôler l'accès à des zones réservées en associant identification biométrique (iris, empreintes digitales), reconnaissance faciale et vidéosurveillance. Plusieurs aéroports internationaux, dont celui de Londres-Heathrow, sont équipés de systèmes de ce type, destinés à fluidifier le passage aux frontières ou encore à automatiser l'accès du personnel aux différentes zones réservées (bagages, fret, pistes).[1]

Nous voyons dans cet extrait de nombreux domaines où peuvent s'appliquer notre système de vidéo surveillance que nous avons ramené dans notre cas dans le contexte de surveillance dans une salle de classe.

### 2.2 Quel est le problème visé ?

Dans la section précédente, nous avons présenter plusieurs domaine d'application du système de vidéo surveillance, ainsi donc, dans notre cas particulier, nous voulons ramené notre système de vidéo surveillance dans le contexte de contrôle de présence effective dans une salle de cours afin de pallier au problème d'absence et ou de présence d'intrus dans la salle.

### 2.3 Qu'est-ce qu'on veut résoudre ?

Ainsi donc, nous voulons résoudre le problème de contrôle effectif et l'intégrer des étudiants dans une salle de classe, ce qui apportera un temps optimal au enseignant pendant le suivie de leurs programme de cours non seulement, mais également assurera une certaine sécurité au niveau des institution accadémique, car grâce à ce type de système, nous pouvons avoir une traçabilité réelle sur un étudiant, ceci permettant de supprimer en même temps la paperasse, qui est un soucis majeur de l'informatique, celui de numériser l'ensemble de toutes les informations existante dans le monde.

### 2.4 Quelle est la situation actuelle ?

Parlant des travaux actuel qui existe, plusieurs applications avec Licence commercialisable existe et aussi des APIs permettant de faire de la reconnaissance faciale dans les entreprise, les bureau, et bien d'autre lieu

public, c'est l'exemple de l'API de reconnaissance faciale développer par les ingénieur de facebook en 2011 qui est accessible a tous mais sans code source, plusieurs systèmes également existe dans le domaine de sécurité militaire. mais dans notre cas, nous utilisons des outils et technologie open sources existant pour développer et adapter un système pour la surveillance académique de présence des étudiants dans une salle de cours.

## 2.5 Résultats attendus

A la fin de notre travaille, nous aimerions obtenir une application qui nous permettra de reconnaître a temps réel les individue dans une salle de classe en s'inspirant d'un modèle préformé sur l'ensemble des étudiants de la classe. ce modèle s'adapte uniquement pour des personnes bien défini. par exemple l'application devras reconnaître spécialement les étudiants de notre salle de classe, mais dans notre cas, puisque nous n'avons pas une ressources matériels lourdes pour lancer les entraînements, nous avons restreint les classe d'entraînement juste a 6 étudiants. a la fin de la réalisation, l'application devrais reconnaître des intrus dans le video en temps réel, elle permettras aussi de rechercher certain personnes spécifique dans une vidéo prise en entrée pour une inspection ou bien pour le tracking, dans un futur nous aimerons étendre notre application afin qu'il nous permettent de donner toutes les informations spécifique sur un étudiant par exemple son heure d'arrivé au cours, son heure de sortie de la salle, combien de temps il a suivie le cours, ceci pourrons aider pour connaitre le comportement relatif d'un étudiant et aider dans la prise de décision en cas de besoin sécuritaire.

# Chapitre 3

## État de l'art

### 3.1 Introduction et généralités

Dans le domaine de la reconnaissance de forme appliquée à la vidéo surveillance, plusieurs travaux existe nous l'avons précisé plus haut mais son beaucoup orienté vers la reconnaissance faciale dans un contexte un peu différents par exemple d'identification pour l'ouverture des portes ou bien plus récent encore pour l'identification sur facebook et sur les téléphones portable, mais aussi le système d'exploitation windows offre déjà des possibilité d'authentification en utilisant la reconnaissance faciale.

Ainsi donc, La reconnaissance faciale est un procédé biométrique au même titre que la reconnaissance d'empreintes digitales, d'iris, ou vocale qui consiste à déterminer l'identité d'une personne. Le système de reconnaissance faciale ou de présence est une application logicielle visant à reconnaître une personne grâce à son visage de manière automatique.

A l'aide d'algorithmes, cette application analyse toutes les caractéristiques faciales telles que l'écartement des yeux, des arêtes du nez, des lèvres, des oreilles, du menton, et bien d'autre caractéristiques corporelle encore, à partir d'une image de son visage qui peut provenir à la fois d'une photo ou d'une vidéo. Donc de nombreux formats d'image peuvent être utilisés qu'ils soient statiques (JPG, bmp, png, GIF ...) ou en mouvement (MPEG 4)[2]

Plusieurs méthodes de reconnaissance de visages ont été proposées durant ces 30 dernières années, suivant deux grands axes :

- La reconnaissance à partir d'images fixes
- La reconnaissance à partir de séquence d'images (vidéo).

Plusieurs articles sollicite la reconnaissance de visages basée sur la vidéo que celle basée sur des images fixes, car elles précisent que l'utilisation simultanée des informations temporelles et spatiales aide dans la reconnaissance et le suivi.

Dans l'article de [3], présente quelques domaine où on a été appliquée la reconnaissance faciale, La reconnaissance faciale ou de présence est aussi utilisée dans les Applications militaires. Un bon exemple de ce domaine est l'utilisation des lunettes de style « Robocop » munies d'une petite caméra d'une portée de 12 milles (19,3 km) par la marine américaine, la caméra peut aussi faire partie de l'optique d'un soldat sur son arme. Grâce à cet équipement, les soldats peuvent identifier des ennemis en quelques secondes sur le terrain, et cela sans réseau à large bande. [3]

En revanche, on distingue un autre domaine d'application de ces systèmes qui est l'assistance à l'utilisateur. Les systèmes de reconnaissance faciale ou de présence sont de plus en plus présents au quotidien. Ils sont par exemple utilisés sur les réseaux sociaux sur internet pour identifier quelqu'un sur une photo, sur les Smart phones pour les déverrouiller...

La nouveauté dans la reconnaissance faciale ou de présence arrive grâce au développement de nouvelles caméras de type 3D. Ces caméras obtiennent de meilleurs résultats que les caméras classiques, parce qu'elles acquièrent une image tridimensionnelle de chaque visage (perspectives) pour identifier une personne lorsqu'elle passe par le portail d'authentification.

Ce projet se concentre principalement sur deux tâches : la détection et la reconnaissance de visages. Il existe dans la littérature une multitude de méthodes de détection de visage et d'identification. Nous passerons en revue, dans ce chapitre, les techniques les plus utilisées.

## 3.2 Quelques techniques de détection et de reconnaissance faciale

Il existe une multitude de techniques consacrées à la détection du visage. Mais pour la plupart il est d'intérêt que ces techniques se basent sur des éléments du visage qui sont le moins susceptibles aux changements : Les méthodes les plus utilisées sont :

### 3.2.1 Le traitement automatique du visage

Le traitement automatique du visage apparaît comme une technologie rudimentaire, elle caractérise les visages par des distances et des proportions entre des points particuliers comme les deux yeux, le nez, les coins de la bouche. Aussi éprouvé que les autres technologies, le traitement automatique du visage est le plus efficace dans des situations de capture d'image avec peu d'éclairage. elle est bien détaillé dans l'article .[] <https://www.biometrie-online.net/technologies/visage>

### 3.2.2 Eigenface

Les « Eigenfaces » furent le premier type de caractérisation utilisé avec succès dans des traitements faciaux tels que la détection et la reconnaissance du visage. Elle utilise une représentation des éléments caractéristiques d'une image de visage à partir d'images modèles en niveau de gris. Des variantes de Eigenface sont fréquemment utilisées comme base pour d'autres méthodes de reconnaissance.[4]

Dans ce projet les auteurs présentent un travail d'une reconnaissance faciale utilisant la méthode Eigenface, en employant des photos de référence des personnes connues et en soumettant au programme la photo d'une personne à identifier. La reconnaissance est donc limitée à l'emploi d'images en deux dimensions. La méthode de reconnaissance faciale Eigenfaces emploie la technique de l'analyse en composante principale, qui marque une différence notable avec les méthodes plus classiques, appelées méthodes géométriques ou locales, qui se basent sur les particularités du visage analysé, et dont les défauts résident dans son manque de précision, ainsi que sa sensibilité aux informations qui ne sont pas pertinentes. La méthode utilisée ici est qualifiée de globale, puisque l'ensemble du visage est alors analysé.

Ils présentent les eigenfaces comme un ensemble de vecteurs propres utilisés dans le domaine de la vision artificielle afin de résoudre le problème de la reconnaissance du visage humain. Le recours à des eigenfaces pour la reconnaissance a été développé par Sirovich et Kirby (1987) et utilisé par Matthew Turk et Alex Pentland pour la classification de visages. Cette méthode est considérée comme le premier exemple réussi de technologie de reconnaissance faciale. Ces vecteurs propres sont dérivés de la matrice de covariance de la distribution de probabilité de l'espace vectoriel de grande dimension des possibles visages d'êtres humains.

### 3.2.3 L'analyse des points particuliers

Elle est la technique d'identification faciale la plus utilisée. Cette dernière se rapproche de Eigenface, mais elle est capable de s'adapter à des changements d'aspect facial (sourire, froncement des sourcils, ...). Les ingénieurs numériques l'utilisent souvent.

### 3.2.4 LDA (Linear Discriminant Analysis) fisher

Elle fait partie des techniques d'analyse discriminante prédictive. Il s'agit d'expliquer et de prédire l'appartenance d'un individu à une classe (groupe) prédéfinie à partir de ses caractéristiques mesurées à l'aide de variables prédictives.

### 3.2.5 Descripteur de Haar

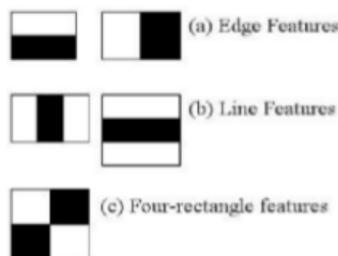


FIGURE 3.1 – Disposition des descripteur HAAR

La détection du visage dans ce filtre est réalisée par un filtre multi-échelles de Haar. Les propriétés d'un visage sont décrites dans un fichier XML. Elles ne sont pas choisies au hasard et reposent sur un échantillon de quelques centaines d'images tests.

L'extraction des yeux, du nez et de la bouche du visage humain sont des tâches largement étudiées dans le domaine de la reconnaissance de formes. Localiser ces régions anatomiques pertinentes du visage est souvent la première étape de nombreuses approches de la vision par ordinateur, comme la segmentation, la reconnaissance ou l'identification de personne, la reconnaissance de l'expression ou de l'émotion du visage, la localisation de points d'intérêts, l'estimation de pose ou encore le suivi du visage. Dans cet article, les auteurs proposent une méthode basée sur l'analyse des lignes horizontales. Elles sont extraites d'une carte d'énergie calculée sur des filtres de Haar adaptatifs. L'introduction de connaissances, notamment sur les positions des différentes régions anatomiques pertinentes, ainsi que sur leurs relations spatiales qui leurs permet de les séparer. Une des difficultés majeures de la détection des éléments anatomiques pertinents du visage réside dans la variabilité de l'illumination d'un visage à l'autre, mais aussi des conditions d'illumination inégale sur un visage donné. Afin de rendre la méthode robuste à ces variations d'illumination, ils proposent une analyse multi-seuils capable de choisir, pour chaque région anatomique, un seuil adéquat sur la carte d'énergie horizontale. Leurs approches sont testées sur les bases BioID, Color FERET et LFW et montrent des résultats prometteurs. Leurs résultats et leurs travaux sont mieux expliqués dans l'article.[5]

### 3.2.6 Méthode LBP (Local Binary Patterns)

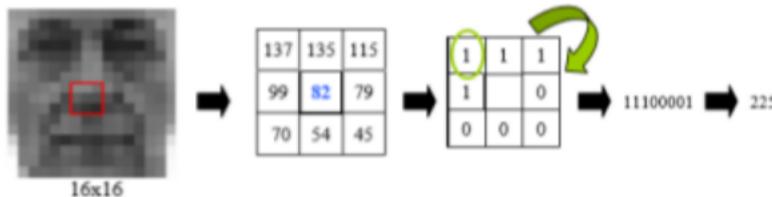


FIGURE 3.2 – location d'un pixel dans l'image

Les LBP (Modèles binaires locaux) ont également été utilisés pour la détection du visage où ce dernier est subdivisé en sous-régions carrées de taille égale sur lesquelles sont calculées les caractéristiques LBP. Les vecteurs obtenus sont ensuite concaténés pour obtenir le vecteur de caractéristiques final.

Ainsi donc, dans l'article [6], les auteurs présentent un travail, basé sur une nouvelle approche de la reconnaissance faciale qui considère à la fois les informations de forme et de texture pour représenter les images de visage. La zone du visage est d'abord divisée en petites régions à partir desquelles les histogrammes du modèle binaire local (LBP) sont extraits et concaténés en un seul histogramme de caractéristique amélioré spatialement représentant efficacement l'image du visage. La reconnaissance est effectuée en utilisant un classificateur de voisin le plus proche dans l'espace des caractéristiques calculé avec Chi Carré comme mesure de dissimilarité. Des expériences approfondies montrent clairement la supériorité du schéma proposé sur toutes les méthodes considérées (PCA, Bayesian Intra / extrapersonal Classifier et Elastic Bunch Graph Matching) sur les tests FERET qui incluent le test de la robustesse de la méthode contre différentes expressions faciales, l'éclairage et le vieillissement des sujets. En plus de son efficacité, cette méthode fournit une très bonne méthode pour l'extraction de caractéristiques dans les visages, nous pouvons voir l'intégralité de l'exploitation de cette méthode dans l'article ;[6]

[6] [https://link.springer.com/content/pdf/10.1007%2F978-3-540-24670-1\\_36.pdf](https://link.springer.com/content/pdf/10.1007%2F978-3-540-24670-1_36.pdf)

### 3.2.7 Utilisation des DNN(deep neural network)

Il y a 3 ans, OpenCV 3.3 a été officiellement publié, apportant avec lui un apprentissage en profondeur très amélioré (dnn) pour la reconnaissance de forme. Ce module prend désormais en charge un certain nombre de cadres d'apprentissage en profondeur, notamment Caffe, TensorFlow et Torch / PyTorch. De plus, cette API qui utilise des modèles d'apprentissage en profondeur pré-formé est compatible avec les API C ++ API et les liaisons Python, ce qui rend un modèle facile à charger à partir du disque.

les étapes suivies pour la réalisation et l'utilisation des modèles de DNN préformé est la suivante :

- Pré-traitez l'image d'entrée.
- Passez l'image à travers le réseau
- obtenir les classifications en sortie.

Cette approche d'utilisation de l'API Dnn est mieux expliquée et implémentée dans l'article [7].

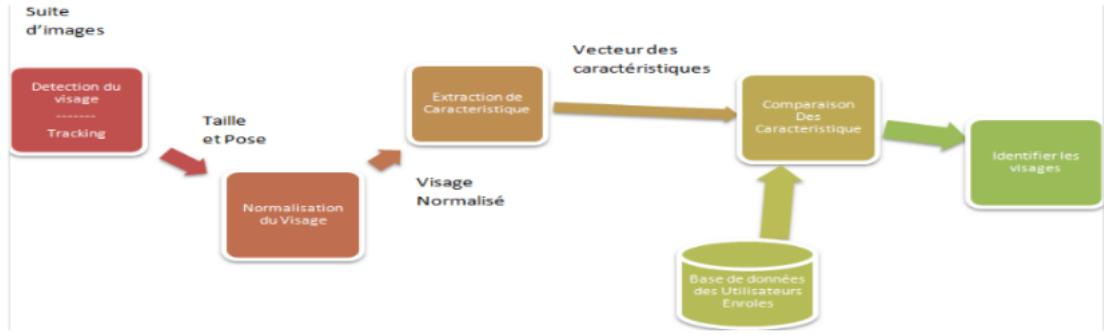


FIGURE 3.3 – schéma générale de parcours d'une image dans une application de reconnaissance

### 3.3 Bilan des solutions existants

Dans cette section nous présenterons une comparaison entre HAAR et LBP qui sont beaucoup plus utilisés.

Une cascade de LBP peut être entraînée de manière similaire (ou meilleure) que la cascade de Haar, mais la cascade de Haar est environ 3 fois plus lente, et selon nos données, environ 1 à 2% de plus pour détecter avec précision l'emplacement d'un visage. Cette augmentation de la précision est assez importante étant donné que la détection du visage peut fonctionner dans la plage de précision de +95%.

Voici quelques résultats lorsque nous utilisons le jeu de données MUCT. La base de données MUCT comprend 3635 visages avec 76 points de repère manuels. La base de données a été créée pour fournir une plus grande diversité d'éclairage, d'âge et d'origine ethnique que les bases de données de visages 2D.

Une détection correcte est notée lorsqu'il y a un chevauchement d'au moins 50% .

Dans le tableau ci-dessous nous pouvons voir une comparaison entre Haar et Lbp :

Frappe	Non détecté	Faux détecté	Double détecté
3635	55	63	5
	Temps: 4m2.060s		

Tableau 1: Les résultats par détection Haar

Frappe	Non détecté	Faux détecté	Double détecté
3635	106	77	3
	Temps: 1m12.511s		

Tableau 2: Les résultats par détection LBP

On remarque que la méthode cascade Haar est beaucoup plus précise que le LBP. Cependant, LBP est beaucoup plus rapide.

# Chapitre 4

## Solution

### Généralités

Ce chapitre comportera deux grandes parties donc la première consistera à présenter les procédés d'acquisition de données, traitements, et constitution de notre dataset qui servira enfin dans la seconde partie à la reconnaissance de forme de visage et à la reconnaissance faciale.

### 4.1 Données

#### 4.1.1 Constitution du jeu de données

Dans cette sous section, nous présentons le processus d'acquisition des données qui nous permettront de mettre en place notre modèle de reconnaissance de visage et d'identification de personne.

Nous pouvons décrire de façon générale le processus d'acquisition des données comme dans le schéma ci-dessous :

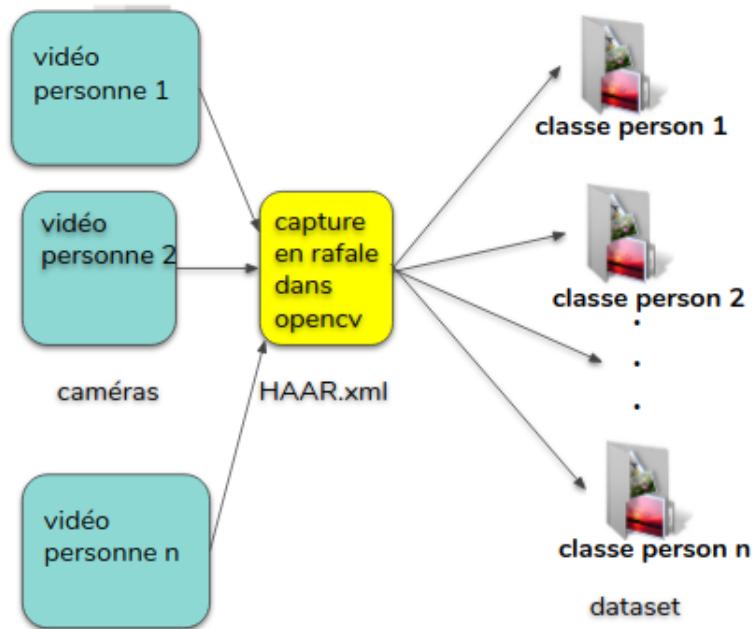


FIGURE 4.1 – acquisition des données d'entraînement

Dans la figure ci-dessus, nous essayons de récupérer les informations pour chaque étudiants car puisque notre application est orienté pour la surveillance dans une salle de classe, alors nous avons choisi dans notre cas 6 étudiants car nos ressources matériels ne nous permettent pas de prendre en charge beaucoup de classe pour entraîner notre modèle de reconnaissance faciale.

Les étapes que nous avons suivies sont les suivantes :

- Enregistrement d'une vidéo de 1 minutes de chaque étudiants
- Passer les vidéo dans opencv en utilisant les descripteur de HAAR pour faire la reconnaissance faciale
- sauvegarder les captures des images correspondant à la face de chaque personne dans la vidéo dans un dossier unique.

Dans la première étape où nous avons enregistrer les vidéo pour chaque étudiants, le but était de récupérer dans chaque vidéo les différents formes correspondant au visage et tous ses visages seront utilisés pour la construction d'un modèle de reconnaissance de visage. Nous avons réalisé l'enregistrement pour 6 étudiants pour faciliter mes traitements.

Après avoir enregistrer les vidéo pour chaque étudiant, nous utilisons la fonction **CascadeClassifier()** disponible dans opencv qui prend en paramètre **haarcascade\_frontalface\_alt2.xml** qui est un fichier **xml** contenant les descripteurs faciale d'une personne et nous faisons la même opération avec le **haarcascade\_profileface.xml** qui contient à son tour les descripteurs en vue de profil d'une face humaine.

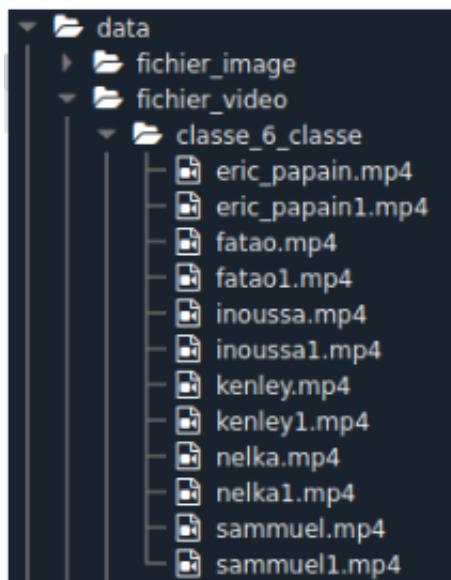
Une fois ces captures effectuer, elles seront stockées par la suite dans le dossier qui porte le nom de la vidéo de la personne donc les images faciale sont en cours d'extraction.

Par exemple, dans le schéma que nous avons présenté plus haut, la **vidéo de "personne 1"** sera pris en paramètre dans la fonction **capture\_photos\_de\_classe\_2.py** et en retour toutes les images de captures de cette personne seront stockées dans son dossier **classe person 1** de cette façon nous effectuerons plusieurs captures de la face dans chacune des vidéos et cela nous permettra d'obtenir une base de données de connaissance solide pour lancer dans un premier temps nos expérimentations.

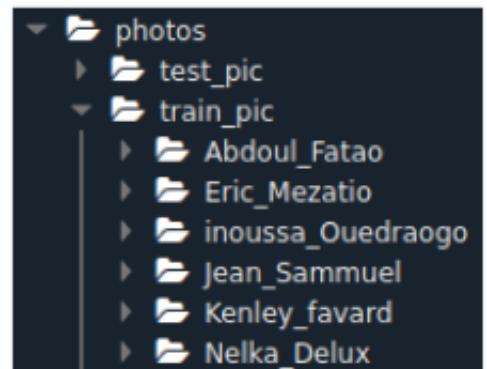
Les descripteur Haar permettent d'extraire dans les images les formes des objets, elle existe pour plusieurs types comme par exemple la figure, le corps, la main, les yeux et bien d'autres encore, nous l'avons présenté un peu plus haut en détails dans l'état de l'art.

Pleins d'autres paramètres seront mis en valeurs dans cette fonction que vous trouverez dans notre code sources.

Nous pouvons voir dans la capture ci-dessous l'arborescence de notre dossier des vidéos et les étiquetages des dossiers correspondant à la capture en format (.png) des différents visages détectés grâce au descripteur XML de HAAR pour front\_face.xml et profile\_face.xml donc nous allons discuter plus haut.



**fig 4.3: video mp4 personnes**

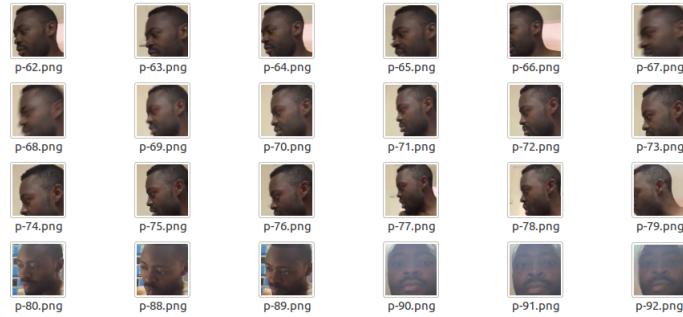


**fig 4.4: photos extraites de chaque vidéo dans en format .png**

1. Abdoul\_Fatao : contient 107 captures frontale et profile



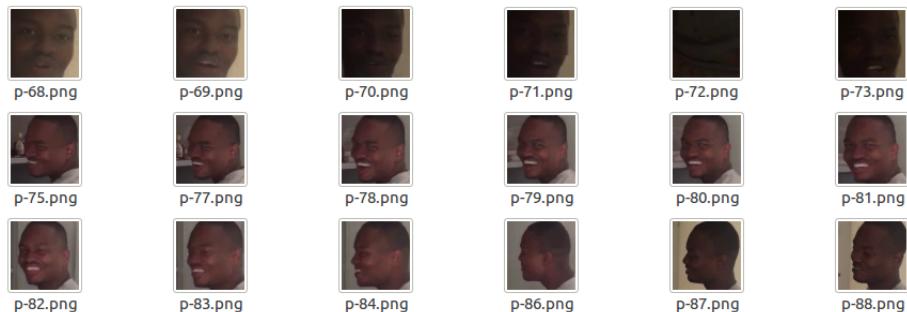
2. Eric\_Mezatio : contient 252 captures frontale et profile



3. inoussa\_Ouedraogo : contient 502 captures frontale et profile



4. jean\_Sammuel : contient 243 captures frontale et profile



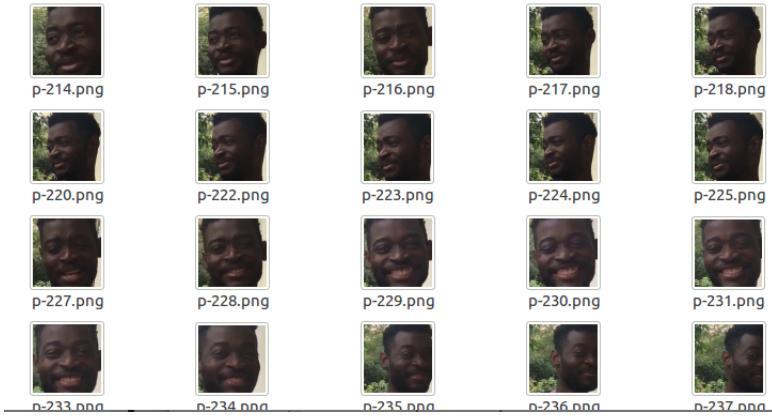
5. Kenley\_favard : contient 1211 captures frontale et profile



6. Nelka\_Delux : contient 300 captures frontale et profile

une fois le data-set obtenu, nous pouvons commencer à appliquer les algorithmes de reconnaissance faciale en temps réel dans la suite nous présenterons les outils que nous utiliserons .

**NB :** Il faut savoir que, le jeu de données que nous constituons nous serviront également pour la phase d'expérimentation en utilisant les CNNs(Convolutional Neural Networks). mais le seul souci est que les données sont insuffisantes pour obtenir de meilleurs résultats avec les CNNs



## 4.2 Outils

Pour la détection du visage nous avons utilisé des outils existants dans la bibliothèque OpenCV car elle possède plusieurs outils pour le traitement d'image déjà implémentés, dont en particulier la méthode de « Viola-Jones » que nous avons choisie pour la détection de visage.

### 4.2.1 Présentation de la bibliothèque OpenCV

**OpenCV [OPENCV, 2007]** (Open source Computer Vision library), est une bibliothèque de traitement d'images et de vision par ordinateur en langage C/C++, et plus récemment en python, optimisée, proposée par Intel pour Windows et Linux. Elle est « Open Source », Elle comprend plusieurs solutions pour le traitement d'image et de l'analyse du mouvement.

ainsi nous utiliserons les environnements suivants :

1. Framework : Anaconda
2. IDE : Spyder
3. Langage : Python
4. Librairies : Keras, Scikit learn, numpy, pickle...

#### 4.2.1.1 Classifieur OpenCV

Nous nous servons, dans nos travaux de détection du visage, du classifieur OpenCV appelé. Il s'applique sur des régions de l'image tout en faisant subir des transformations d'échelle, afin de reconnaître si un objet d'une région rassemble à un visage, une fois qu'une zone image est détecter, un bounding box est créer autour de l'image dans la photo et ou de l'image dans une vidéo.

Le module CV contient les fonctions nécessaires pour notre application, dont les plus importantes sont :

1. `detectMultiScale()`  
C'est une fonction pratique pour détecter des objets dans une images.  
Les paramètres d'entrée sont :
  - Une image
  - scaleFactor(Facteur d'échelle pour chaque image)
  - minNeighbors(qui définit le voisinage entre deux captures)
  - minSize : définie la taille minimale des photos à prendre en considération
2. `CascadeClassifier()`  
Cette fonction permet de prendre en paramètre le type de descripteurs HAAR qui sera utiliser pour détecter un objet dans la vidéo ou la photo. Dans notre cas ils contiendront les descripteurs front-face et profile-face pour la détection de visage.
3. `VideoCapture()`  
Elle permet de prendre en paramètre l'image ou la vidéo et ou la caméra et les images seront capturées en rafale et identifier en temps réel.
4. `haarcascade_frontalface_default.xml`  
fichier XML contenant les descripteurs frontaux d'une photo d'une personne
5. `haarcascade_profileface.xml`  
fichier XML contenant les descripteurs de profil d'une photo d'une personne

#### 4.2.2 Pré-traitement

Pour les opération de pré-traitement sur nos images, nous utilisons les fonctions :

— cvtColor()

qui prend en paramètre une image ou une capture et puis la transforme en niveau de gris pour la manipulation juste d'un paramètre de couleurs car en utilisant les images en couleurs, ils contiendront beaucoup plus de caractéristique qui peut fausser et causer un long temps dans le traitement des captures.

#### 4.2.3 Utilisation des Réseaux de Neurone Convolutionnels

L'architecture générale de fonctionnement des réseaux de neurones convolutionnels est la suivante : Nous

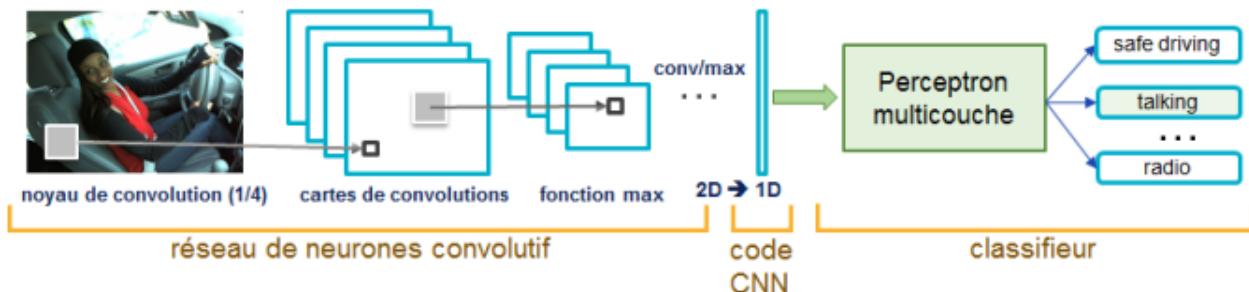


FIGURE 4.2 – Fonctionnement générale d'un CNN

utiliserons dans ce contexte les fonctions :

- convolution2d() pour ajouter des couches de convolution
- maxpooling2d() pour réduire la taille des différentes images reçues des couches de convolutions
- flatten() permet d'aplatir les images sous forme de tableau array qui pourra être pris en paramètre pendant l'entraînement du modèle
- dropout() fonction permettant de diminuer le taux d'apprentissage du modèle afin d'éviter un sur-apprentissage
- sequential() permet d'initialiser notre réseau de neurone.

### 4.3 Modèles conceptuels / Pipeline de traitement

Dans cette sous section, nous allons présenter le modèle conceptuel de donnée et quelques différents diagrammes de classe de notre application, des algorithmes utilisés et notre pipeline de traitement.

#### 4.3.1 Modèles conceptuels

ici, nous allons présenter deux diagrammes de classe qui représenteront pour l'un l'utilisation des descripteurs HAAR pour la reconnaissance et l'autre l'exploitation des CNN.

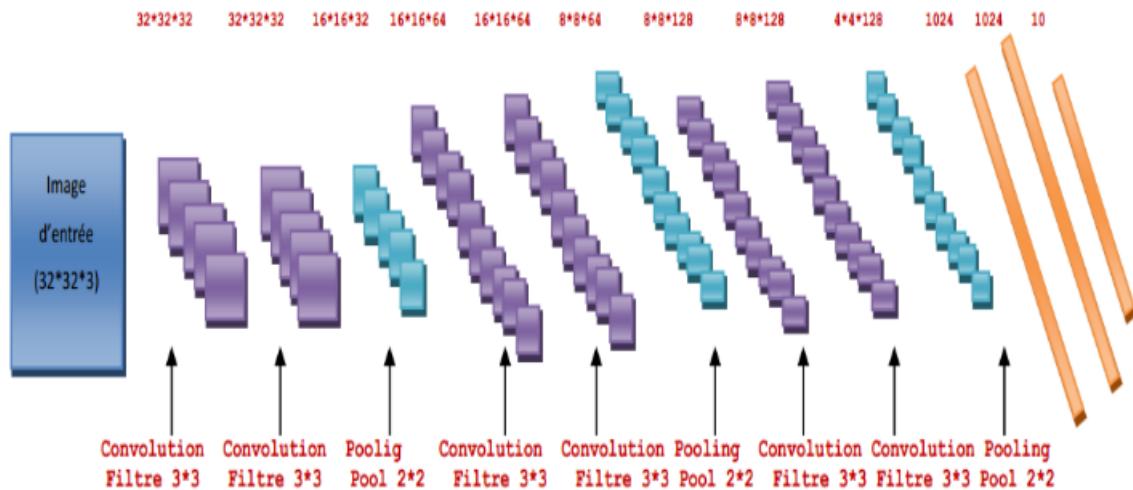
#### 4.3.1.1 Diagramme de classe utilisation Haar



Dans ce cas d'utilisation, l'utilisateur active sa caméra, retourné vers la salle où il y'a les étudiants à identifier, OpenCV récupère l'ensemble des captures dans images de visage détecté qui se trouvent dans la vidéo en direct, ensuite le système effectue la reconnaissance de chaque images de figure détecté. Deux résultats existent : soit les personnes ne sont pas tous reconnues(n'existe pas dans le modèle ou soit le modèle ne fait des erreurs de reconnaissance) soit les personnes identifiées dans la vidéo sont tous reconnues (le modèle les reconnaît tous) et la dernière possibilité est que le modèle détecte inconnu et que la personne ne soit pas réellement une personne de la classe( c'est un bon résultat) car n'oublions pas dans nos spécifications, nous avons dit que notre modèle devait être capable de détecter les inconnus dans la classe.

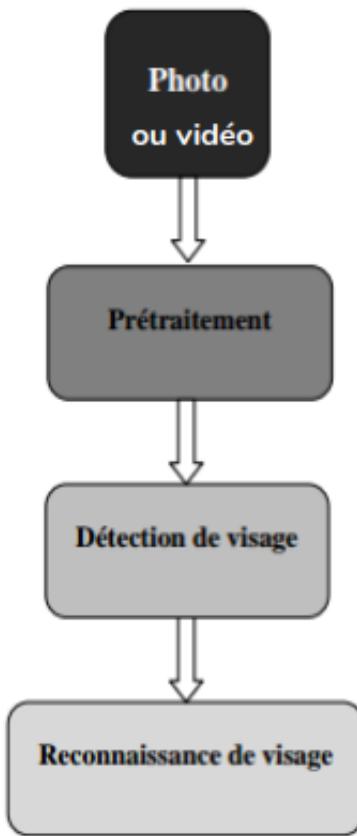
#### 4.3.1.2 Modèle de conception de l'architecture de notre CNN

Notre modèle utilisant les CNNs.



### 4.3.2 Pipeline de traitement

Nous présentons ici le pipeline de traitement jusqu'à la reconnaissance de la photos dans les vidéos.



Nous avons présenté plus haut le modèle conceptuel de notre application de détection de visage basé sur les descripteur HAAR et les reseaux de neurones.

Nous pouvons voir dans ce schéma plus détaillé plus bas le pipeline générale de fonctionnement de notre application.

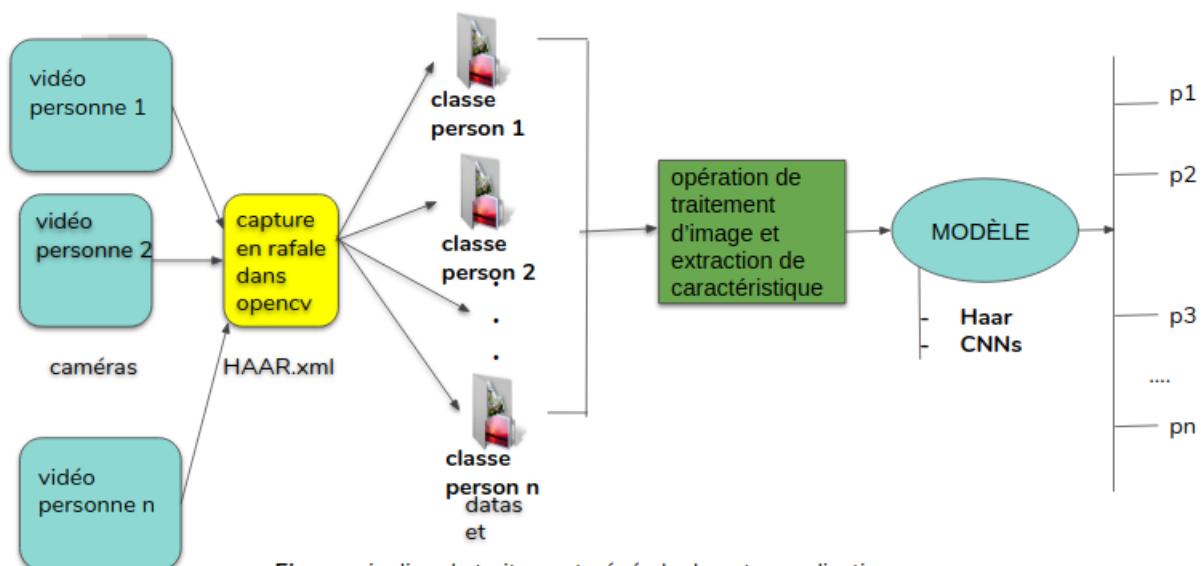


Figure: pipeline de traitement générale de notre application

# Chapitre 5

# Implémentation et expérimentation

## 5.1 Introduction

Dans ce chapitre, on va définir l'architecture des deux modèles que nous avons utilisé, ces modèles sur la base des images que nous avons capturé plus haut. Pour cela, on va travailler avec les bibliothèques Tensorflow et Keras pour l'apprentissage et la classification avec les cnns et aussi quelques fonctions de la bibliothèque de opencv afin d'améliorer les performances des modèles on va utiliser quelques techniques simple et efficaces comme data augmentation et dropout pour les Cnns.

## 5.2 Implémentation Expérimentation

cette partie sera divisé en deux partie :

- Implementation des Cnns
- Implementation avec les bibliothèques opencv

### 5.2.1 Implementation des Cnns

#### 5.2.1.1 Logiciels et librairies Utilisés dans l'implémentation

##### 1. TensorFlow

TensorFlow est un framework de programmation pour le calcul numérique qui a été rendu Open Source par Google en Novembre 2015. Depuis son release, TensorFlow n'a cessé de gagner en popularité, pour devenir très rapidement l'un des frameworks les plus utilisés pour le Deep Learning et donc les réseaux de neurones. Son nom est notamment inspiré du fait que les opérations courantes sur des réseaux de neurones sont principalement faites via des tables de données multi-dimensionnelles, appelées Tenseurs (Tensor). Un Tensor à deux dimensions est l'équivalent d'une matrice. Aujourd'hui, les principaux produits de Google sont basés sur TensorFlow : Gmail, Google Photos, Reconnaissance de voix.

##### 2. Keras

Keras est une API de réseaux de neurones de haut niveau, écrite en Python et capable de fonctionner sur TensorFlow ou Theano. Il a été développé en mettant l'accent sur l'expérimentation rapide. Être capable d'aller de l'idée à un résultat avec le moins de délai possible est la clé pour faire de bonnes recherches. Il a été développé dans le cadre de l'effort de recherche du projet ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), et son principal auteur et mainteneur est François Chollet, un ingénieur Google. En 2017, l'équipe TensorFlow de Google a décidé de soutenir Keras dans la bibliothèque principale de TensorFlow. Chollet a expliqué que Keras a été conçue comme une interface plutôt que comme un cadre d'apprentissage end to end. Il présente un ensemble d'abstractions de niveau supérieur et plus intuitif qui facilitent la configuration des réseaux neuronaux indépendamment de la bibliothèque informatique de backend. Microsoft travaille également à ajouter un backend CNTK à Keras aussi.[8]

##### 3. Python

Python est un langage de programmation de haut niveau interprété (il n'y a pas d'étape de compilation) et orienté objet avec une sémantique dynamique. Il est très sollicité par une large communauté de développeurs et de programmeurs. Python est un langage simple, facile à apprendre et permet une bonne réduction du cout de la maintenance des codes. Les bibliothèques (packages) python encouragent la modularité et la réutilisabilité des codes. Python et ses bibliothèques sont disponibles (en source ou

en binaires) sans charges pour la majorité des plateformes et peuvent être redistribués gratuitement.

#### 4. Scikit-learn

Scikit-learn est une bibliothèque libre Python dédiée à l'apprentissage automatique. Elle est développée par de nombreux contributeurs notamment dans le monde académique par des instituts français d'enseignement supérieur et de recherche comme Inria et Télécom ParisTech. Elle comprend notamment des fonctions pour estimer des forêts aléatoires, des régressions logistiques, des algorithmes de classification, et les machines à vecteurs de support. Elle est conçue pour s'harmoniser avec des autres bibliothèques libre Python, notamment NumPy et SciPy.

#### 5. Configuration matérielle

- Un PC portable Dell i7 CPU 2.40\*8 GHZ
- Carte graphique Nvidia GeForce GT2G
- RAM de taille 8 GO
- Disque dur de taille 500 GO
- Système d'exploitation Linux Ubuntu version 16.04

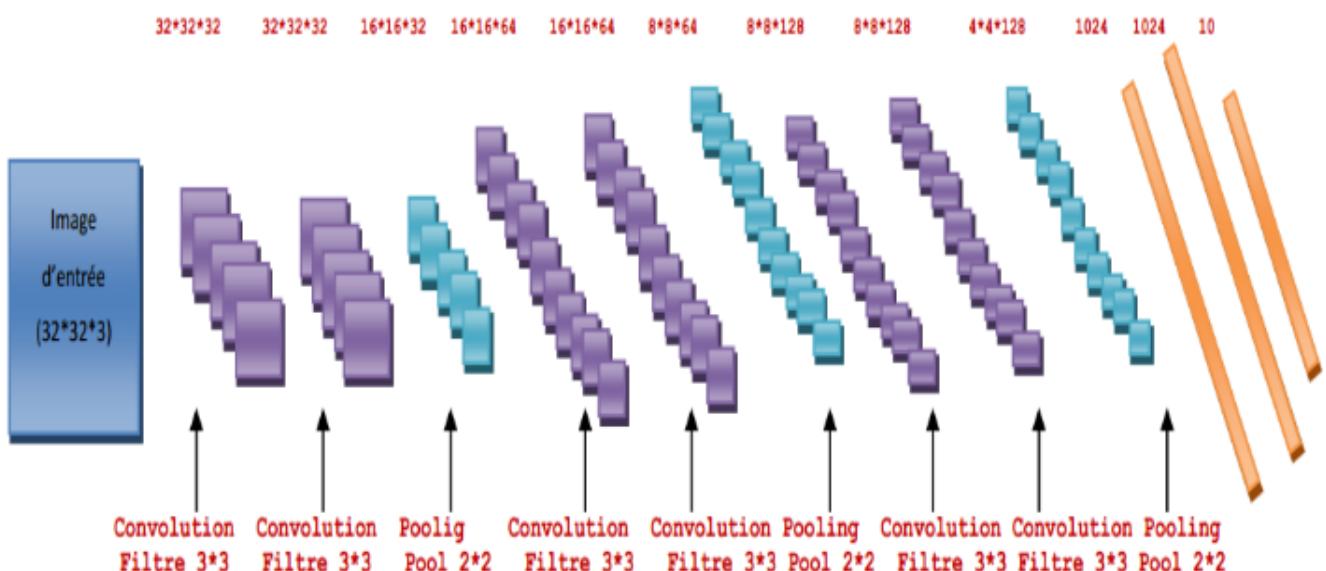
##### 5.2.1.2 Architecture de notre réseau

Au cours de nos expérimentations, nous avons créé un modèle (améliorable) avec différents architectures, sur la base d'image capturé plus haut. Dans ce qui suit on présente l'architecture de notre réseau : notre modèle est composé de cinq couches de convolution et deux couches de maxpooling et trois couches de fully connected. L'image en entrée est de taille 150\*150, l'image passe d'abord à la première couche de convolution. Cette couche est composée de 32 filtres de taille 3\*3, Chacune de nos couches de convolution est suivie d'une fonction d'activation ReLU cette fonction force les neurones à retourner des valeurs positives, après cette convolution 32 features maps de taille 32\*32 seront créés.

Les 32 feature maps qui sont obtenus auparavant ils sont donnés en entrée de la deuxième couche de convolution qui est composée aussi de 32 filtres, une fonction d'activation RELU est appliquée sur la couche de convolution, ensuite on applique Maxpooling pour réduire la taille de l'image ainsi la quantité de paramètres et de calcul. À la sortie de cette couche, nous aurons 32 feature maps de taille 16\*16.

On répète la même chose avec les couches de convolutions trois, quatre et cinq, ces couches sont composées de 64 filtres, la fonction d'activation ReLU est appliquée toujours sur chaque convolution. Une couche de Maxpooling est appliquée après la couche de convolution cinq. À la sortie de cette couche, nous aurons 64 feature maps de taille 8\*8. Le vecteur de caractéristiques issu des convolutions a une dimension de 4096.

Après ces cinq couches de convolution, nous utilisons un réseau de neurones composé de trois couches fully connected. Les deux premières couches ont chacune 1 024 neurones où la fonction d'activation utilisée est le ReLU, et la troisième couche est un softmax qui permet de calculer la distribution de probabilité des 6 classes (nombre de classe dans la base d'image plus haut). dans la section précédente, nous avons présenté l'architecture



de notre réseau, dans la partie plus bas, nous présentons la configuration de notre réseau. remarquons que notre architecture avec les CNNs donne la possibilité d'utiliser les trois propriétés couleurs des l'images.

Layer (type)		Output Shape	Param #	Connected to
input_1 (InputLayer)		(None, 3, 32, 32)	0	
convolution2d_1 (Convolution2D)		(None, 32, 32, 32)	896	input_1[0][0]
convolution2d_2 (Convolution2D)		(None, 32, 32, 32)	9248	convolution2d_1[0][0]
maxpooling2d_1 (MaxPooling2D)		(None, 32, 16, 16)	0	convolution2d_2[0][0]
convolution2d_3 (Convolution2D)		(None, 64, 16, 16)	18496	maxpooling2d_1[0][0]
convolution2d_4 (Convolution2D)		(None, 64, 16, 16)	36928	convolution2d_3[0][0]
convolution2d_5 (Convolution2D)		(None, 64, 16, 16)	36928	convolution2d_4[0][0]
maxpooling2d_2 (MaxPooling2D)		(None, 64, 8, 8)	0	convolution2d_5[0][0]
flatten_1 (Flatten)		(None, 4096)	0	maxpooling2d_2[0][0]
dense_1 (Dense)		(None, 1024)	4195328	flatten_1[0][0]
dropout_1 (Dropout)		(None, 1024)	0	dense_1[0][0]
dense_2 (Dense)		(None, 1024)	1049600	dropout_1[0][0]
dropout_2 (Dropout)		(None, 1024)	0	dense_2[0][0]
dense_3 (Dense)		(None, 100)	102500	dropout_2[0][0]
<hr/>				
Total params: 5,449,924				

### 5.2.2 Implémentation utilisant opencv

Nous nous servons, dans nos travaux de détection du visage, du classifieur OpenCV appelé. Il s'applique sur des régions de l'image tout en faisant subir des transformations d'échelle, afin de reconnaître si un objet d'une région rassemble à un visage.

dans cette section nous avons utiliser :

1. face.LBPHFaceRecognizer\_create()

cette fonction nous permet de lancer notre entrainement avec les capture réalisé plus haut, cette methode a déjà été beaucoup plus détailler plus haut dans l'analyse de la solution, ici nous passerons très vite sur les solutions et test.

2. Détection à l'aide des «Classieurs de Haar»

Pour détecter des objets dans une image, OpenCV nous fournis la fonction « cvHaarDetectObjects » qui est basé sur la méthode de Viola et Jones, la nature des objets détectées dépend de classifieur utilisé. OpenCV nous propose des classifieurs prédéfinis pour la détection d'un visage dans une image sous forme des fichiers XML. Ces classifieurs différent par le type d'algorithme de boosting qu'ils utilisent, tels que Discrete Adaboost, Real Adaboost ou encore Gentle Adaboost et position de visage dans l'image. Ci-dessous les fichiers :

- haarcascade\_frontalface\_alt.xml
- haarcascade\_frontalface\_alt\_tree.xml
- haarcascade\_frontalface\_alt2.xml
- haarcascade\_frontalface\_default.xml
- haarcascade\_profileface.xml

Voici les résultats obtenus par les cascades de type frontalface (vue de face) : Il s'avère que les résultats obtenus par la cascade frontalface\_default est le meilleur avec un taux de bonne détection égale à 92.78% et un taux de mauvaise de détection égale à 7.22%. Nous retiendrons donc uniquement le dernier fichier car c'est celui qui offre les meilleures performances pour la détection de visages selon notre expérience. la figure ci-dessous représente les résultats de test.

	Bonnes détections		Mauvaises détections	
	Nombre	Pourcentage	Nombre	Pourcentage
frontalface_alt	710	82.75 %	148	17.25 %
frontalface_alt2	620	72.30%	238	27.70%
frontalface_alt_tree	595	69.35%	263	30.65%
frontalface_default	796	92.78%	62	7.22%

Tableau 5.1: Résultat des détections de visage



FIGURE 5.1 – Bonne détection

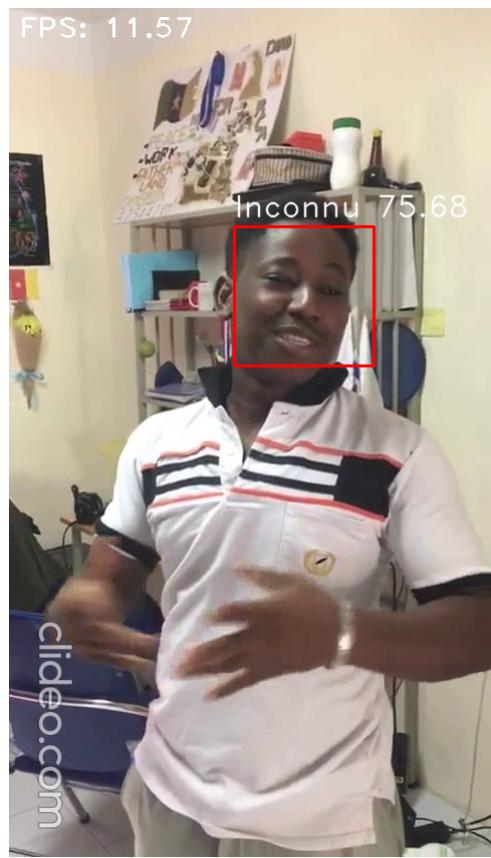


FIGURE 5.2 – Mauvaise détection

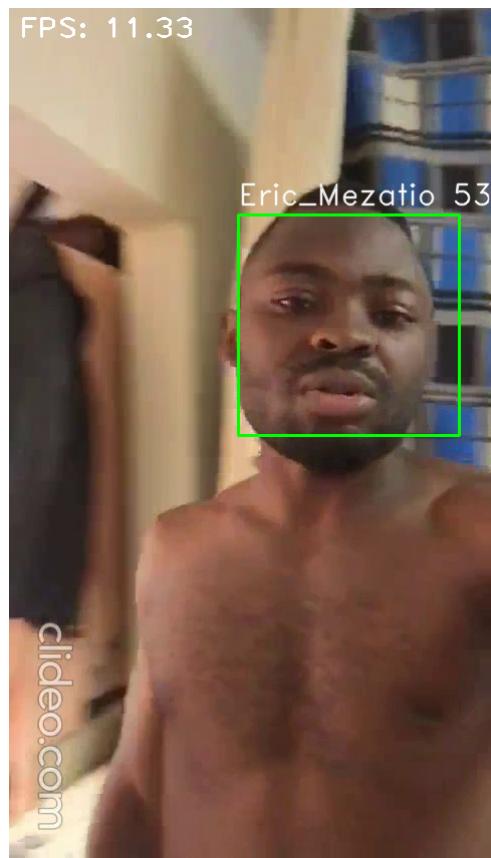


FIGURE 5.3 – Bonne détection

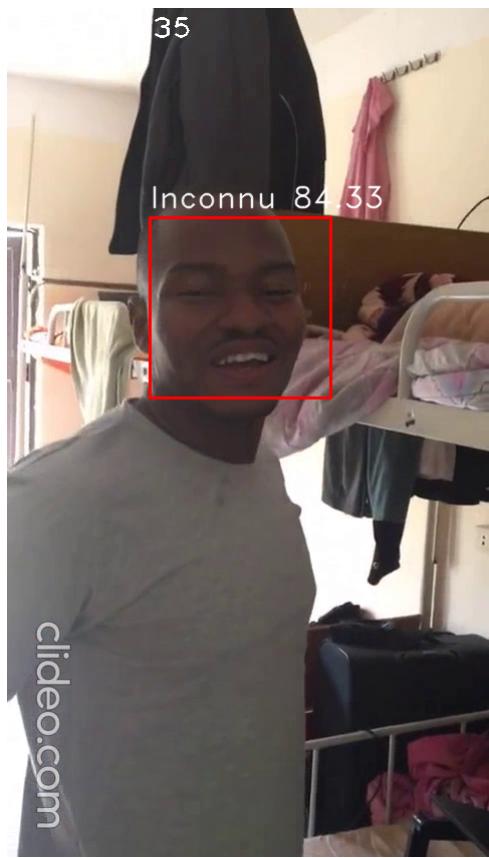


FIGURE 5.4 – Mauvaise détection

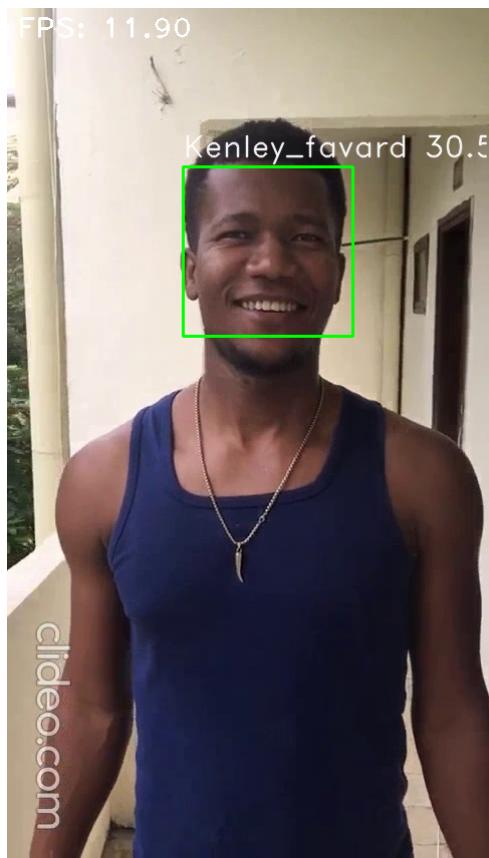


FIGURE 5.5 – Bonne détection

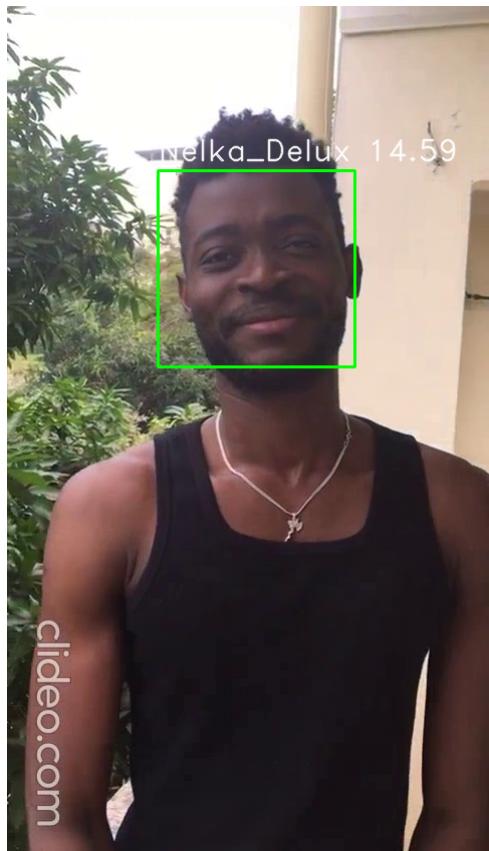


FIGURE 5.6 – Bonne détection

Dans la mesure où nous n'avons pas pu trouver une vidéo et où faire une simulation en temps réel de notre application faute de problème de sécurité et donc il a été impossible pour nous de prendre une vidéo devant contenir tous les étudiants et où tous les 6 classes d'élèves qui ont été entraînées dans notre modèle, nous avons pris une vidéo open source sur internet que nous avons essayé de faire montrer comment fonctionnerait notre application que nous pouvons voir dans la figure plus bas :



FIGURE 5.7 – détection des visages

dans cette image nous voyons notre application qui arrive à détecter et à faire des bounding box quand il détecte un visage mais il n'a pas effectif car comme nous l'avons précisé plus haut, les descripteur HAAR ne prennent pas en considération des visage baisser ou incliné d'où l'impossibilité de détecter tous les visage et aussi le plus important est aussi de remarque que l'application amis inconnu sur la tête de tous les élèves se qui relève du fait que tous ces gens ne sont pas reconnu par notre modèle se qui est un succès.

```

57/57 [=====] - 82s 1s/step - loss: 0.1213 - acc: 0.9593 - val_loss: 0.9203 - val_acc: 0.7645
Epoch 98/100
57/57 [=====] - 84s 1s/step - loss: 0.1183 - acc: 0.9622 - val_loss: 1.3506 - val_acc: 0.6911
Epoch 99/100
57/57 [=====] - 84s 1s/step - loss: 0.0751 - acc: 0.9709 - val_loss: 1.3502 - val_acc: 0.6881
Epoch 100/100
57/57 [=====] - 82s 1s/step - loss: 0.0735 - acc: 0.9726 - val_loss: 1.1567 - val_acc: 0.7339
Out[1]: <keras.callbacks.History at 0x7fd8e0ed5fd0>
In [2]:

```

FIGURE 5.8 – précision obtenue après entraînement avec les Cnns



FIGURE 5.9 – images de classification obtenue par les cnns

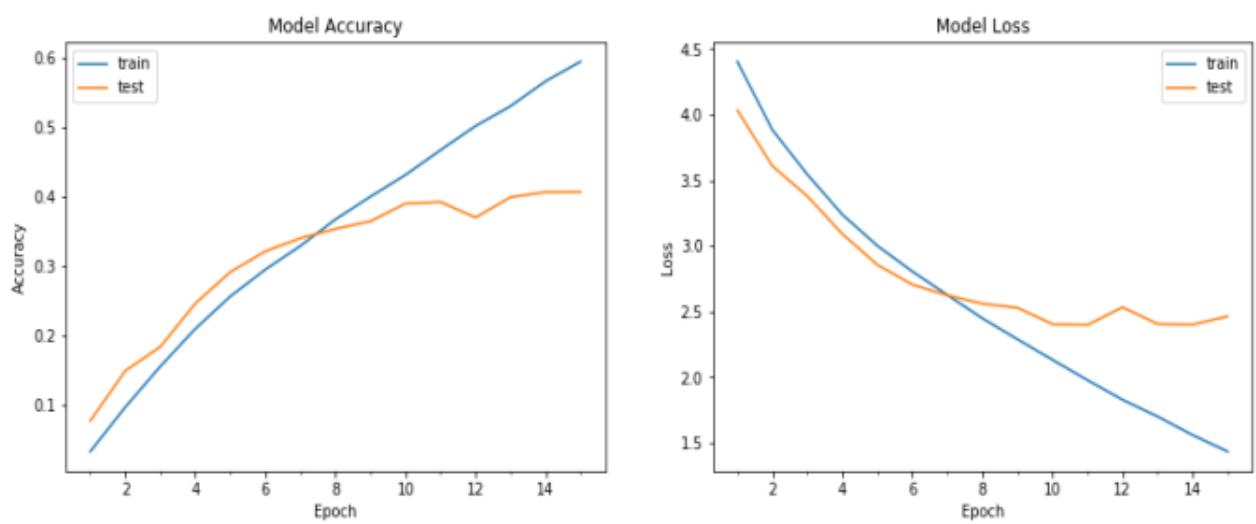


FIGURE 5.10 – précision obtenue a partir du nombre d'epoques de formations

# Chapitre 6

## Conclusion et perspectifs

La reconnaissance d'individus demeure un problème complexe et non parfaitement résolu, malgré tous les travaux réalisés au cours des dernières années. Plusieurs problèmes incombent à cette tâche d'identification et chacun d'eux est non trivial. De nombreuses conditions réelles affectent la performance d'un système comme par exemple comment réussir à identifier une personne cagouler, comment continuer à reconnaître une personne en cas de déformation ou en cas d'accident, et si la personne changeait de style par exemple garder de la barbe, mettre des lunettes, mettre un chapeau et bien d'autre questions encore son en suspend, cependant la détection automatique des visages influence énormément la performance du module d'identification. A travers ce projet nous avons mis en œuvre une approche d'identification du visage, et pour aboutir à ce but, il fallait au préalable constituer une base de données d'apprentissage pour notre futur système destiné à reconnaître les différents visages, ensuite aborder un travail de détection du visage. Pour cela nous avons détaillé dans les chapitres précédents, l'état de l'art, nos travaux et ensuite les solutions à nos travaux de détection et notre approche pour améliorer le résultat obtenu en ajoutant un module de prétraitement. Bien que notre méthode d'amélioration ait annoncé de bons résultats avec l'augmentation de données pour l'entraînement d'un CNN, au niveau de l'interpolation de visages non détectés par la bibliothèque OpenCV, elle élimine parfois de vrais visages. Après la phase de détection, nous avons pu aborder la tâche de reconnaissance. Notre apport dans cette tâche délicate, est d'utiliser la notion des points d'intérêt pour reconstruire un modèle de visage et d'utiliser les CNNs qui utilisent les réseaux de neurones plus performants dans la classification d'images. Ce projet ne manque pas de perspectives : pour la tâche de détection, et à partir des visages détectés par la bibliothèque OpenCV, il est intéressant de trouver d'autres méthodes d'élimination des fausses alertes et de détecter en contre-partie les visages oubliés .

---

### Compléments

les fonctions écrites qui ont été utilisées dans le cadre de notre travail sont disponibles sur le lien GitHub :  
[https://github.com/ericpapain/Computer\\_Vision\\_Form\\_Recognition](https://github.com/ericpapain/Computer_Vision_Form_Recognition)

le dataset étant trop lourd nous l'avons mis dans notre drive et il est accessible à partir de l'adresse :

<https://drive.google.com/file/d/1ZgAmtn5bkEj9OTRFZVHp-D60hpgh3PAz/view?usp=sharing>

1. **capture\_photos\_de\_classe\_2.py**  
Pour la capture de photos dans les vidéos
2. **apprentissage\_reconnissance.py** Formation du modèle après collecte des données images
3. **identiter\_visage\_after\_train.py** Identification des personnes à temps réel dans la vidéo
4. **Cnn\_faciale.py** Identification des personnes en utilisant les CNNs

# Annexes

- [1] <https://www.larousse.fr/encyclopedie/divers/vid%C3%A9osurveillance/101594>
- [2] <https://labiometrie.wordpress.com/2017/02/12/reconnaissance-faciale/>
- [3] <http://www.speroforum.com/a/HCESHCICAJ39/73073-Military-facial-recognition-technology-advances-in-Maryland.WuMCn4huY2x>
- [4] <https://silanus.fr/sin/?p=785>
- [5] <https://projet.liris.cnrs.fr/coresa/articles/2014/2014-ReimsImage-AFRV-AFIG-CORESA-GTGEODIS/reimsimage2014.univ-reims.fr/articles/4.pdf>
- [6] [https://link.springer.com/content/pdf/10.1007%2F978-3-540-24670-1\\_36.pdf](https://link.springer.com/content/pdf/10.1007%2F978-3-540-24670-1_36.pdf)
- [7] <https://www.pyimagesearch.com/2017/08/21/deep-learning-with-opencv/>
- [8] Mémoire de fin d'études/Classification des images avec les réseaux de neurones convolutionnels/Mokri Mohammed Zakaria/03/07/2017
- [9] PROJET DE FIN D'ETUDES/DETECTION ET RECONNAISSANCE DE VISAGE/Mohamed Aymen FODDA 19 Octobre 2010