الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université de 8 Mai 1945 – Guelma -

Faculté des Mathématiques, d'Informatique et des Sciences de la matière

Département d'Informatique

Mémoire de Fin d'études Master

Filière : Informatique

Option : Systèmes Informatiques

Thème :

Reconnaissance d'expression faciale à partir d'un visage réel

Encadré Par :

Mme BORDJIBA YAMINA

Présenté par :

NACER Foued

Juillet 2019

DEDICACE

Je dédie ce mémoire à :

Ma mère,

Qu'elle puisse recevoir à travers ce travail aussi modeste soit-il, l'expression de mes sentiments et de mon éternelle gratitude.

REMERCIEMENTS

Je remercier énormément mon encadreur :

Madame BORDJIBA Yamina *pour l'appui académique et sa rigueur scientifique et également pour la pertinence de ses conseils,*

Je tiens à remercier tout particulièrement ma femme pour son soutien et ses encouragements

.

Résumé

L’efficacité des systèmes de reconnaissance d’expressions faciles est importante pour une bonne interaction homme-machine. Mais la tâche de la reconnaissance de l’expression faciale est liée à plusieurs méthodes qui fournissent des facteurs influençant sur la performance des Systèmes FER. Ce mémoire fournit l’étude de deux architectures du réseau CNN : VGG16 et Xception, afin d’examiner leur performances et de proposer un modèle hybride qui permet de fusionner les deux et on obtenir un nouveau modèle qui optimise les performances. Notre choix de ces deux architectures est motivé par leur succès dans la reconnaissance d’image. Contrairement à d'autres travaux, nous nous concentrons sur la structure de CNN tout en analysant les convolutions internes dans les réseaux qui influencent directement sur les résultats.

Les deux approches proposées sont testée sur la base de données FER-2013. Les résultats démontrent la supériorité du CNN Xception avec sa spécificité conventionnelle qui inclut des couches de Convolution séparable**.** La précision augmente même lorsque le modèle ne traite pas la totalité de paramètres, ce qui génère des résultats remarquables sur la base de données FER-2013 où il atteint 73% sur FER-2013 par rapport à 62% pour VGG16.

Table des matières

Liste des Figures :

Les Tables

Introduction générale

Introduction Générale :

La reconnaissance de l'expression faciale est un problème intéressant et qui a d'importantes applications dans de nombreux domaines, tels que les interactions homme-machine. Cela aide à construire des machines plus intelligentes capables de comprendre les émotions humaines. De nombreuses autres applications du monde réel, telles que :

- Marketing : des applications pour mesurer la satisfaction des clients, prévoir les produits qui les intéressent.
- Médecine : aide à la détection de certaines maladies psychologiques, aide à l'apprentissage des émotions pour les enfants autistes.
- Sécurité : détection du stress.
- Interaction Homme-Machine : robot d'accompagnement, voiture intelligente.
- Éducation : apprentissage à distance et le développement de jeux interactifs.

Ekman au début des années 1970 a montré qu'il existe six expressions émotionnelles universelles: dégoût, colère, bonheur, tristesse, surprise et peur. Ces expressions sont identifiables en observant les signes du visage. En raison de l'importance de la reconnaissance de l'expression faciale dans la conception de systèmes d'interaction homme-machines, divers algorithmes d'extraction de caractéristiques et d'apprentissage automatique ont été développés. La plupart de ces méthodes ont déployé des fonctionnalités manuelles suivies d'un classifieur, tel que la méthode SVM **[MNT16]**, Adaboost **[YXC05],** ou encore la méthode Forêts aléatoires **[SBE09].** Le succès actuel des réseaux de neurones convolutif (CNN) dans la classification d'images s'est étendu au problème de la reconnaissance de l'expression faciale. Contrairement aux méthodes traditionnelles d'apprentissage par la machine et de vision par ordinateur où les caractéristiques sont définies par des méthodes traditionels , CNN apprend à les extraire directement de la base de données d'apprentissage. le reseau CNN est généralement associé à un classifieur qui permet de former le modèle de bout en bout sur l'ensemble de données. Dans la plupart des cas, il nécessite beaucoup de données d'entraînement pour bien généraliser. La disponibilité des bases de données et la puissance de calcul des machines actuelles présentent des valeurs à ajouter pour les CNNs. Cependant, trouver une architecture idéale pour optimisé les perfermances des système FER (*facial-expression-recognition*) n'est pas une tâche facile, la plupart des CNN fournissent des résultats moins précis quand on déminue le volume du réseaux et les paramétres à traiter . L'objectif de ce projet de fin d'étude est de concevoir et de réaliser une application qui

permet de reconnaitre les expressions faciales d'un visage réel en utilisant la méthode d'apprentissage profond. Dans ce mémoire, nous comparons entre deux architectures CNN différentes VGG16 **[KSI15]** et Xception **[OAR17]** qui sont élaboré pour la classifications des images.

Les contributions de ce projet sont doubles : d'abord, nous étudions l'impact de la modification de couches convolutifs sur l'amélioration des performances de la reconnaissance de l'expression faciale. et ensuite, nous proposons une conception d'une architecture d'un CNN pour la reconnaissance de l'expression faciale, en s'inspirant des deux modèles VGG16 et Xception dans l'objectif d'obtenir des résultats meilleurs. les tests sont réalisés sur la base de données FER-2013 .

Ce mémoire est constitué en quatre chapitres, et organisé comme suit :

Dans le premier chapitre, nous présentons l'expression faciale, les systèmes de codifications et les systèmes de reconnaissances.

Ensuite, dans le second chapitre, nous le consacrons à la présentation de l'apprentissage profond, où nous détaillons plus les réseaux de neurone convolutifs.

La conception de notre approche de reconnaissance des expressions faciales basée sur les CNNs est présentée dans le troisième chapitre,

Le dernier chapitre est consacré à la description des différents outils utilisés dans le développement de notre application, ainsi que les différents résultats obtenus

Chapitre 01 :
Analyse des expressions faciales : Etat de l'art

"What you feel inside reflects on your face.

So be happy and positive all the time"

Anonymous

Chapitre 01 :Analyse des expressions faciales : Etat de l'art

1. Introduction

Dans ce chapitre, nous allons présenter quelques notions sur les émotions et les différentes théories de codification, ainsi que l'architecture globale des systèmes de reconnaissance faciale. En se basant sur ces dernières, nous présentons les approches qui aident à la reconnaissance d'expressions faciale, et nous terminons le chapitre par les différentes techniques d'apprentissage automatique.

2. Expressions faciales et emotions

2.1.Définitions

a- L'Emotion

Charles Darwin, dans son ouvrage intitulé The expression of the Emotions in Man and Animals (1872), est le premier à poser des postulats qui influenceront de façon déterminante les recherches sur les émotions.

Selon Le dictionnaire français LAROUSSE

Emotion : *de émouvoir, d'après l'ancien français motion, mouvement)*

Trouble subit, agitation passagère causés par un sentiment vif de peur, de surprise, de joie, etc. : Parler avec émotion de quelqu'un.

Réaction affective transitoire d'assez grande intensité, habituellement provoquée par une stimulation venue de l'environnement.

En 2005, Scherer abonde dans le même sens avec la théorie des composantes et propose la définition suivante : « L'émotion est un ensemble de variations épisodiques dans plusieurs composantes de l'organisme en réponse à des évènements évalués comme importants par l'organisme. L'émotion est un processus dynamique d'une durée relativement brève. » [SNO18].

L'émotion se traduit via de nombreux canaux comme la position du corps, la voix et les expressions faciales. Une émotion implique généralement une expression faciale correspondante, mais l'inverse n'est pas vrai : il est possible de mimer une expression représentant une émotion sans pour autant ressentir cette émotion. Alors que les expressions dépendent des individus et des cultures, on distingue généralement un nombre limité d'émotions universellement reconnues comme primaires **[**FKH10**]**

b- L'expression Faciale

L'expression faciale est une mimique faciale chargée de sens. Le sens peut être l'expression d'une émotion, un indice sémantique ou une intonation dans la Langue des Signes.

L'interprétation d'un ensemble de mouvements musculaires en expression est dépendante du contexte d'application.

Par exemple dans le cas d'une application en interaction Homme-Machine où l'on désire connaître une indication sur l'état émotionnel d'un individu, on cherchera à classifier les mesures en termes d'émotions. [**KGH** 10]

2.3.Les six expressions faciales universelles

Dès le vingtième siècle, Ekman et Friesen [EKM71] définissent six émotions de base sur la base d'une étude culturelle et psychologique, et qui ont expliquait que les humains perçoivent certaines émotions de base par la même manière indépendamment de la culture de l'individu. Les tentatives pour décrire l'émotion humaine repose sur l'hypothèse que les humains expriment universellement un ensemble d'émotions primaires discrètes qui inclus **le bonheur, la tristesse, la peur, la colère, le dégoût et la surprise**(voir figure 1.1). Principalement à cause de sa simplicité et derevendication de l'universalité, cette hypothèse des émotions primaires universelle sa été largement exploité dans l'informatique affective. [CMS16]

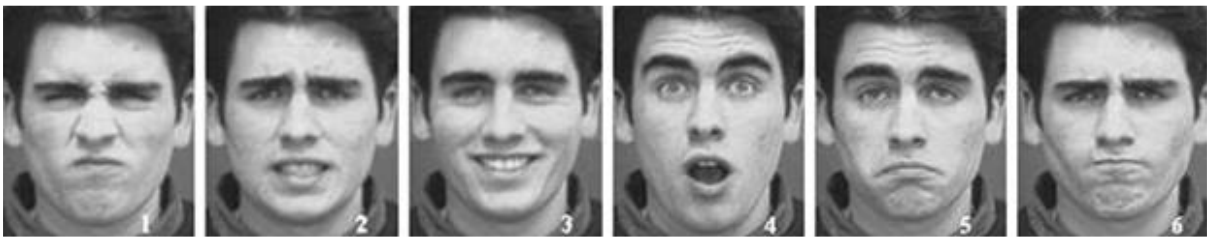

Figure 1.1: Émotions primaires exprimées sur le visage. De gauche à droite: dégoût, peur, joie, surprise, tristesse, colère. [CMS16]

2.4.Les Systèmes de codification

Du point de vue physiologique, l'expression faciale est une conséquence de l'activité musculaire faciale. Ces muscles sont également appelés muscles mimétiques ou muscles des expressions faciales. L'étude de l'expression faciale ne peut se faire sans l'étude de l'anatomie du visage et de la structure sous-jacente des muscles faciaux.

Les chercheurs ont concentré leur attention sur un système de codage pour les expressions faciales. Plusieurs systèmes ont été proposés parmi eux l'outil d'Ekman.

En 1978 Ekman a développé un outil de codification des expressions du visage largement utilisé aujourd'hui. Il s'intéresse désormais à l'analyse des expressions de manière informatique.[KGH 10]

Ce qui suit nous définissant les principes de quelque système :

a- Facial Action Coding System (FACS) :

Le système a été spécialisé pour l'analyse des séquences vidéo d'une gamme d'individus et en associant les changements d'apparence faciale avec les contractions des muscles sous-jacents. Cette étude a permis le codage de 44 unités d'action distinctes (AUs) c'est-à-dire anatomiquement liées à la contraction de muscles faciaux spécifiques, chacune étant intrinsèquement liée à un petit ensemble d'activations musculaires localisées

Bien que FACS soit un système de description bénéficiant d'un grand succès, il souffre cependant de quelques inconvénients [**DLS16**], :

Complexité : Il faut 100 heures d'apprentissage pour maîtriser les principaux concepts.

[KGH 10]

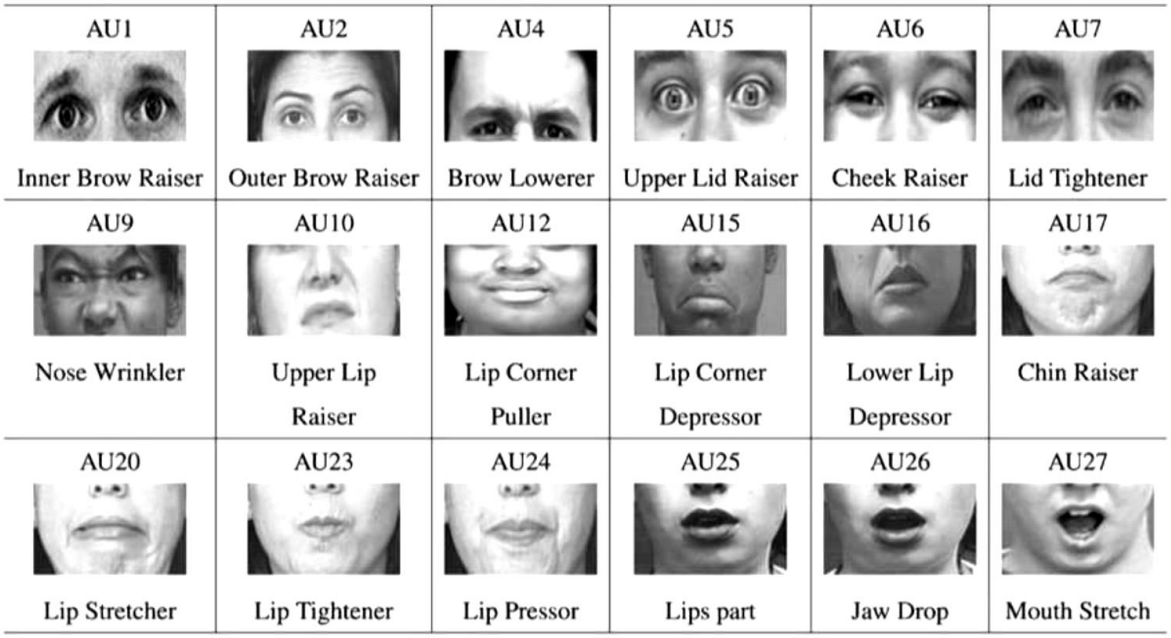

Figure 1.2 : Liste des Action Units relatives aux 6 expressions faciales [**DLS16**]

Difficulté de manipulation par une machine : FACS a d'abord été créé pour des psychologues, Certaines mesures restent floues et difficilement évaluables par une machine.

Manque de précision : les transitions entre deux états d'un muscle sont représentées de manière linéaire, ce qui est une approximation de la réalité. En particulier les mesures temporelles de l'activation des muscles faciaux (onset, apex et offset) ne sont pas mises en Evidence **[KGH 10]**

b. MPEG4 : La norme de codage vidéo MPEG-4 dispose d'un modèle du visage humain développé par le groupe d'intérêt Face and Body AdHocGroup . C'est un modèle 3D articulé. Ce modèle est construit sur un ensemble d'attributs faciaux, appelés Facial Feature Points(FFP). Des mesures sur ces points sont effectuées pour former des unités de mesure qui servent à la description des mouvements musculaires (FacialAnimation Paramètres - équivalents des Actions Unitaires d'Ekman).**[KGH 10]**

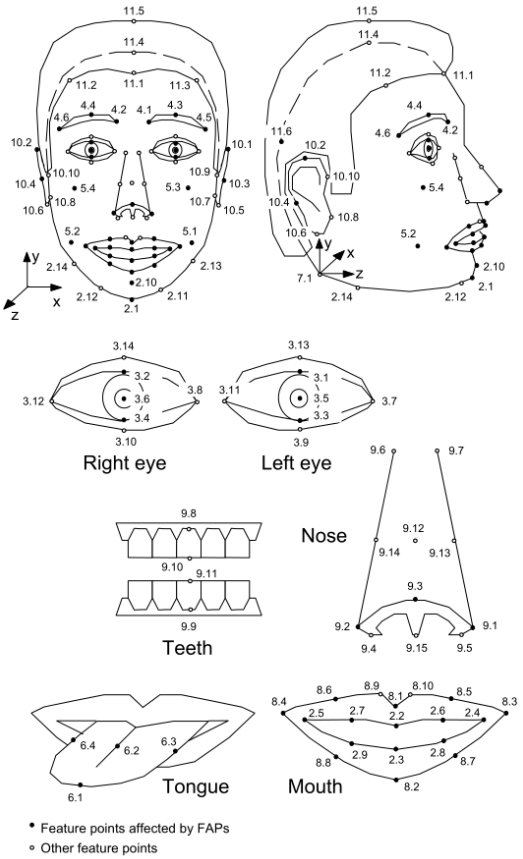

Figure 1.3 : Model MPEG4 l', ensemble d'attributs faciaux [Web01]

c.Candide : Candide est un modèle du visage, contenait 75 sommets et 100 triangles. Il est composé d'un modèle en fil de fer présentant un visage générique et d'un ensemble de paramètres(*Shape Units*) ces paramètres permettent d'adapter le modèle générique à un

individu particulier. Ils représentent les différences inter-individus et sont au nombre de 12 **[JAH01]**:

1. Hauteur de la tête,
2. Position verticale des sourcils,
3. Position verticale des yeux,
4. Largeur des yeux,
5. Hauteur des yeux,
6. Distance de séparation des yeux,
7. Profondeur des joues,
8. Profondeur du nez,
9. Position verticale du nez,
10. Degré de courbure du nez (s'il pointe vers le haut ou non),
11. Position verticale de la bouche,
12. Largeur de la bouche.

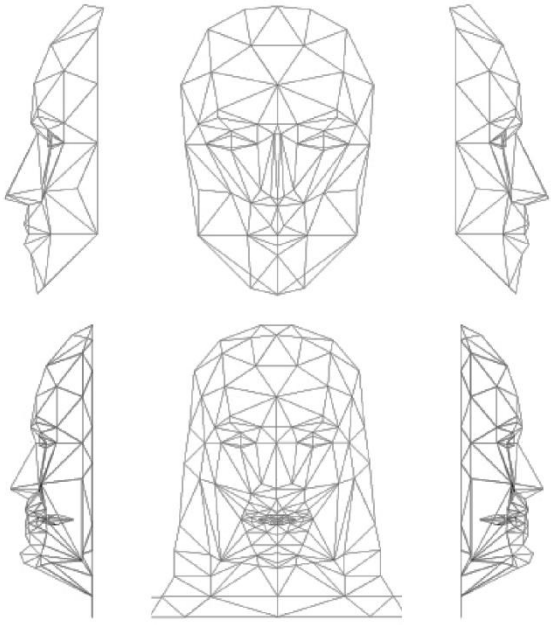

Figure 1.4 : En haut: CANDIDE-1 avec 79 sommets et 108 surfaces.En bas: CANDIDE-2 avec 160 sommets et 238 surfaces. [JAH01]

Le modèle standard CANDIDE est légèrement modifié .Modèle avec 79 sommets, 108 surfaces et 11Unités d'action; voir la figure 1.4 (en haut). Ce modèle a été créé par Marten Strömberg lors de la mise en œuvre du premier progiciel CANDIDE, et est appelé ici CANDIDE1 [JAH01] .Plus tard, Bill Welsh de British Telecom a créé une autre version avec

160 sommets et 238triangles couvrant toute la tête frontale (y compris les cheveux et les dents) et les épaules; voir Figure 1.4 (en bas). Cette version, connue sous le nom de CANDIDE-2, est également incluse dans le progiciel CANDIDE, mais n'est livrée qu'avec six Unités d'action. **[JAH01]**

2.5.Domaines d'applications

De nombreuses études ont été menées sur l'analyse automatique de l'expression faciale en raison de son importance pratique en robotique sociable, traitement médical, surveillance de la fatigue du conducteur et de nombreuses autres interactions homme-machine systèmes. Ainsi que Dans le domaine de la vision par ordinateur et de l'apprentissage automatique. Divers systèmes de reconnaissance de l'expression faciale (FER) ont été explorés pour coder les informations d'expression à partir de représentations faciales.

3. Architecture de systeme de reconnaissance des expressions faciales(FER):

Un système qui effectue une reconnaissance automatique des expressions faciales est généralement compose de trois modules principaux, comme illustre dans la figure 1.6. Le premier module consiste à détecter et enregistrer la région du visage dans les images ou les séquences d'images d'entrée. Il peut s'agir d'un détecteur pour détecter le visage dans chaque image ou simplement détecter le visage dans la première image, puis suivre le visage dans le reste de la séquence vidéo. Le deuxième module consiste à extraire et représenter les changements faciaux causes par les expressions faciales. Le dernier module détermine une similarité entre l'ensemble des caractéristiques extraites et un ensemble de caractéristiques de

référence. D'autres filtres ou modules de prétraitement de données peuvent être utilises entre ces modules principaux pour améliorer les résultats de détection, d'extraction de caractéristiques ou de classification.

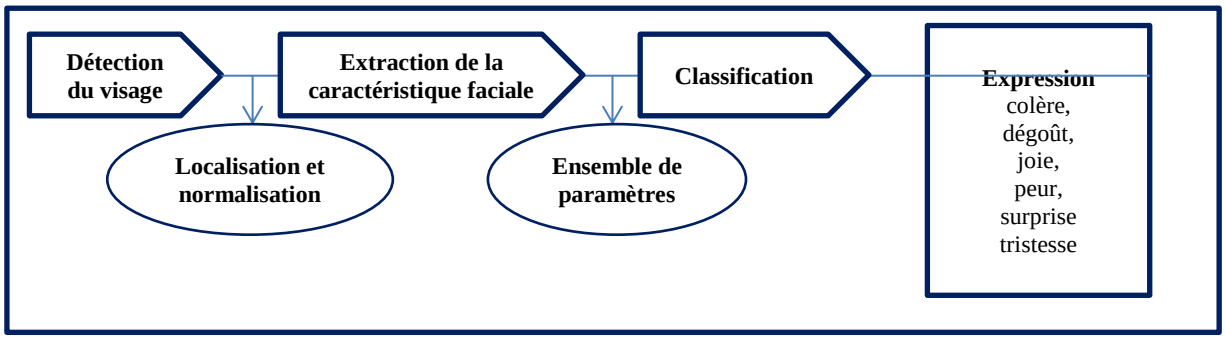

***Figure 1.6* :** Architecture d'un système de reconnaissance des expressions faciales

3.1.Detection du visage

Le problème de détection des visages implique l'identification de la présence de visages dans une image et la détermination des emplacements et des échelles des visages. La précision de la détection du visage est particulièrement importante dans des conditions réalistes, ou la présence du visage dans une scène et sa localisation globale ne sont pas connues a priori.

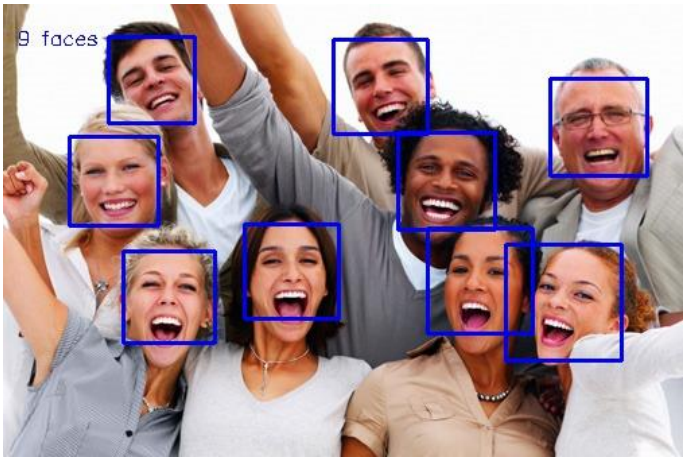

Figure 1.7 : *Détection des visages*

Conditions d'acquisition de l'image : L'éclairage et les changements dans les caractéristiques de la camera affectent de manière significative la chrominance des régions du

visage. Certaines caractéristiques peuvent être masquées ou combinées avec des ombres ou des brillances sur le visage provoquant ainsi une perte d'informations sur les couleurs.

La détection de visage consiste à déterminer la présence ou l'absence de visages dans une image. C'est une tâche préliminaire nécessaire à la plupart des techniques d'analyse du visage. Les techniques utilisées sont généralement issues du domaine de la reconnaissance des formes. En effet, le problème peut être vu comme la détection de caractéristiques communes à l'ensemble des visages humains : il s'agit de comparer une image a un modèle générique de visage et d'indiquer s'il y a ou non ressemblance Les principales difficultés sont la robustesse aux différentes identités, poses du visage, expressions faciales et aux variations d'illumination.

La sortie d'un détecteur de visage indique le nombre de visages présents dans l'image. De plus, la plupart des détecteurs de visage actuels sont aussi des localisateurs de visages : ils renvoient une localisation des visages détectes.

Il existe plusieurs techniques de détection du visage, nous citons les plus utilisées :

• Le traitement automatique du visage

C'est une méthode qui spécifié les visages par des distances et des proportions entre des points particuliers comme les points autour des yeux, le nez, les coins de la bouche. Mais cette technique n'est pas efficace dans des situations de capture d'image avec peu d'éclairage **[PHM99]**.

• Eigenface

Les « Eigenfaces » est une méthode de caractérisation efficace dans des traitements faciaux tels que la détection et la reconnaissance du visage est basé sur la représentation des caractéristiques de visage à partir d'images modèles en niveau de gris. **[MTA91]**.

• L'analyse des points particuliers

C'est une technique d'identification faciale la plus utilisée, elle est capable de s'adapter à des changements d'aspect facial comme le sourire, froncement des sourcils, et les grimasse les plus compliquées. . **[MTA91]**.

• LDA (Linear discriminant analysis) fisher

Elle basée sur l'analyse discriminante prédictive. Il s'agit d'expliquer et de prédire l'appartenance d'un individu à une classe prédéfinie à partir de ses caractéristiques mesurées à l'aide de variables prédictives **[PJP96].**

• Méthode LBP (Local Binary Patterns)

La technique des Modèles binaires locaux divisent le visage en sous–régions carrées de taille égale là où on calcule les caractéristiques LBP. Les vecteurs obtenus sont ensuite concaténés pour obtenir le vecteur de caractéristiques final. . **[MTA91]**.

• Filtre de Haar

Cette méthode de détection du visage utilise un filtre multi-échelles de Haar. Les caractéristiques d'un visage sont décrites dans un fichier XML.

3.2.Extraction des caracterestiques :

Les points caractéristiques du visage sont principalement situés autour des composants faciaux tels que les yeux, la bouche, les sourcils, le nez et le menton. La détection des points caractéristiques du visage commence habituellement à partir d'une boite englobante rectangulaire renvoyée par un détecteur de visage qui localise ce dernier. L'extraction de caractéristiques géométriques telles que les contours des composants faciaux, les distances faciales, etc. fournit les emplacements ou les caractéristiques d'apparence peuvent être calculées**.** De ce fait, les méthodes d'extraction des caractéristiques pour l'analyse d'expression peuvent êtres éparées en deux types d'approches : les méthodes basées sur **les caractéristiques géométriques** et les méthodes basées sur **l'apparence [MNT16] :**

a- **Les caractéristiques géométriques** représentent la forme et l'emplacement des composants du visage (y compris la bouche, les yeux, les sourcils et le nez). Les composants faciaux ou les traits faciaux sont extraits pour former un vecteur de caractéristiques représentant la géométrie du visage.

b- **Les caractéristiques d'apparence** représentent les changements d'apparence (texture de la peau) du visage, tels que les rides et les sillons. Ces caractéristiques d'apparence peuvent être extraites sur tout le visage ou sur des régions spécifiques du visage.

Selon les différentes méthodes d'extraction des caractéristiques, les effets de la rotation de la tête dans le plan et les différentes échelles de prise de vue du visage peuvent être éliminés par **une normalisation** de ce dernier avant l'extraction des caractéristiques ou par une représentation des caractéristiques avant l'étape de reconnaissance d'expression.

3.3. La reconnaissance de l'expression faciale :

C'est la dernière étape d'un système de reconnaissance des expressions faciales. Cette étape consiste à reconnaitre l'ensemble de six expressions prototypes .Les travaux de recherche concernés par ce créneau sont divisés en trois parties, à savoir, les approches globales, les approches locales, et enfin les approches hybrides. Chaque famille d'approches présente ses avantages et ses inconvénients vis-à-vis des problèmes liés aux conditions environnementales, le changement de l'échelle, les orientations des images, les positions de la tête…etc.

a- Les approches globales : sont indépendantes des positions (haut ou bas) de la tête et même des orientations de l'image de visage. Ces méthodes sont très efficaces mais nécessitent, généralement, une lourde phase primaire d'apprentissage, et le résultat dépend quelques fois du nombre d'échantillons utilisé; (même la sensibilité aux bruits est à signaler).

b- Les approches locales : sont basées sur la détection des objets du visage, sont très robustes aux changements de luminances. Néanmoins, l'imprédictibilité de la position de la tête ainsi que son orientation peuvent provoquer des lacunes dans le système.

c- les approches hybrides C'est l'alternative consiste à combiner les deux approches (locales et globales) afin de profiter des avantages de l'une pour combler les inconvénients de l'autre.

La phase de reconnaissance dans le système FER est basée sur la théorie de l'apprentissage automatique.

Le vecteur de caractéristique est formé pour décrire l'expression du visage, donc la première partie du classifieur c'est l'apprentissage. La formation du classifieur consiste à étiquetées les images après détection. Une fois que le classificateur est formé, il peut reconnaître les images d'entrée en leur attribuant une étiquette particulière de la classe d'expression.

Les méthodes de la classification peuvent être divisées en deux groupes:

* Reconnaissance basée sur des données statiques qui ne concerne que les images

* Et la reconnaissance basée sur des données dynamiques qui concerne les séquences d'images ou vidéos.

Dans les différents recherches, divers classifieurs ont été appliquées telles que, réseau neuronal, réseau bayésien(BN), Support Vector Machine (SVM) **[MNT16]**, basé sur des règles de classification de Markov cachés **[MNT16].**

3.4 Bases des données d'expressions faciales :

L'un des aspects les plus importants du développement de tout nouveau système de reconnaissance ou de détection d'expression faciale est le choix de la base de données qui sera utilisée pour tester ce nouveau système. De plus, des bases de données communes sont nécessaires pour évaluer les algorithmes de manière comparative. Les bases de données disponibles peuvent être classées en deux catégories : les bases d'expressions faciales spontanées et les bases d'expressions faciales posées. **[BCK18]**

3.4.1 Exemple de base des données

La base de données	Description	Lien
Ck+(Cohn-Kanade)	593 séquences vidéo à la fois posées et non posées émotions (spontanées) 123 sujets âgés de 18 à 30 ans Fournit des protocoles et des résultats de base pour le visage suivi des fonctionnalités, unités d'action et reconnaissance des émotions Résolutions d'image de 640 X480 et 640 X490	http://www.consortium.ri.cmu.edu/ckagree/
CE(Compound Emotion)	5060 images correspondant à 22 catégories de base et émotions composées 230 sujets humains (130 femmes et 100 hommes, l'âge moyenne 23 ans) Comprend la plupart des races Résolution d'image de 3000X 4000	http://cbcsl.ece.ohio-state.edu/ dbform_compound.html
DISFA	130 000 images vidéo stéréo en haute résolution 27 sujets adultes (12 femmes et 15 hommes) 66 points de repère du visage pour chaque image Résolution d'image de 1024X 768	http://www.engr.du.edu/mmahoor/DISFA.htm
BU-3DFE	Visages humains 3D et émotions faciales 100 sujets dans la base de données, 56 femmes et 44 hommes, avec environ six émotions 25 modèles d'émotion faciale 3D par sujet Résolution d'image de 1040 1329	http://www.cs.binghamton.edu/ ~lijun/Research/3DFE/3DFE_ Analysis.html
JAFFE	213 images de sept émotions faciales Dix modèles féminins japonais Six adjectifs d'émotion de 60 sujets japonais Résolution d'image 256 256	http://www.kasrl.org/jaffe_info.html
B+	16 128 images faciales 28 sujets distincts pour 576 conditions	http://vision.ucsd.edu/content/ extended-yale-face-database-b-b

	d'observation Résolution d'image de 320 243	
MMI	Plus de 2900 séquences vidéo et images haute résolution images de 75 sujets _ 238 séquences vidéo sur 28 sujets, hommes et femmes _ Résolution d'image de 720 _ 576	https://mmifacedb.eu/
BP4D-Spontanous	La base de données vidéo 3D comprend 41 participantes (23 femmes, 18 hommes), avec des émotions faciales spontanées 11 Asiatiques, six Afro-Américains, quatre Hispaniques, et 20 euro-américains Résolution d'image de 1040 1329	http://www.cs.binghamton.edu/ ~lijun/Research/3DFE/3DFE_ Analysis.html
KDEF	4900 images d'expressions faciales humaines d'émotion 70 individus, sept expressions émotionnelles différentes avec 5 angles différents Résolution de l'image 562 762	http://www.emotionlab.se/ resources/kdef
FER2013	compte environ 37 000 personnes bien structurées En Images de gris à 48X48 pixels les visages regroupés automatiquement par l'API Google de recherche d'image	https://www.kaggle.com/datasets

Table 1.1: Exemple de bases des données [BCK18]

Dans ce projet, nous avons utilisé une base des données FER2013fourni par Kaggle

3.5. Apprentissage automatique (Machine learning)

Dans un problème donné on peut noter deux aspects qui peuvent nécessiter l'utilisation de programmes qui apprennent et s'améliorent sur la base de leur expérience.

Tâches trop complexes à programmer : exemples de telles tâches comprennent la conduite, la reconnaissance de la parole et la compréhension de l'image. Dans toutes ces tâches, des programmes d'apprentissage automatique à la fine pointe de la technologie, des programmes qui tirent des leçons de leur expérience, «obtiennent des résultats tout à fait satisfaisants, une fois exposés à suffisamment d'exemples de formation. **[MMZ17]**

Tâches au-delà des capacités humaines: une autre grande famille de tâches bénéficiant des techniques d'apprentissage automatique est liée à l'analyse d'ensembles de données très volumineux et complexes: données astronomiques, transformation des archives médicales en connaissances médicales, prédiction météorologique, analyse de données génomiques, moteurs de recherche Web Avec de plus en plus de données enregistrées numériquement disponibles, apprendre à détecter des modèles significatifs dans des ensembles de données volumineux et complexes est un domaine prometteur dans lequel la combinaison de programmes qui apprennent avec la capacité de mémoire presque illimitée et la vitesse de traitement croissante des ordinateurs ouvre de nouveaux horizons. **[MMZ17]**

L'apprentissage en profondeur (Deep Learning)

L'apprentissage en profondeur est une option de l'apprentissage automatique qui consiste à faire former l'ordinateur à faire ce qui vient naturellement aux humains ou apprendre de l'expérience.

Les algorithmes d'apprentissage automatique utilisent des méthodes de calcul on utilisant des informations directement à partir de données sans s'appuyer sur une équation prédéterminée comme modèle.

Le principe de l'apprentissage passe par l'analyse de données pour conclure des règles qui permettront de tirer des conclusions sur de nouvelles données.

L'apprentissage profond s'appuyer sur les réseaux de neurones artificiels» Les résultats d'une couche de « neurones » servent d'entrée aux calculs d'une autre couche et ainsi de suite. Les progrès de l'apprentissage profond dépend de la puissance des ordinateurs et au développement de grandes bases de données « big data ». **[BMB17]**

L'apprentissage profond est particulièrement adapté à la reconnaissance d'image, (qui est importante pour résoudre des problèmes tels que la reconnaissance faciale, la détection de mouvement)

Et de nombreuses technologies avancées (d'assistance au conducteur telles que conduite autonome, détection de voie, détection de piétons et stationnement autonome).

Pour choisir d'utiliser un réseau pré-entraîné ou créer un nouveau réseau profond,

2. Conclusion :

Dans ce chapitre nous avons présenté les théories et les représentations les plus connues dans la reconnaissance d'expressions faciales : les techniques de codifications, les approches de détections de visages dans des images ainsi que l'extraction des caractéristiques, Finalement, les bases de données fréquemment utilisées ont été décrites. A l'issue de cette étude, nous avons choisi le deep learning comme approche pour la reconnaissance ça sera la base pour la suite de notre travail.

Dans le chapitre suivant nous présenterons l'apprentissage profond en citant les différents types de réseaux de neurones et en détaillant les réseaux de neurones convolutifs .

Chapitre 02 :

Apprentissage profond

“Develop a passion for learning.
If you do, you will never cease to grow”
Anthony J D’Angelo

Chapitre 02 : Apprentissage profond

1. Introduction :

Le terme "Deep Learning" ou Apprentissage profond, a été introduit pour la première fois au ML par Dechter (1986)**,** et aux réseaux neuronaux artificiels par Aizenberg et al (2000) **[IAZ13].**

Le Deep Learning est un nouveau domaine de recherche du ML, qui a été introduit dans le but de rapprocher le MLde son objectif principal : l'intelligence artificielle. Il concerne les algorithmes inspirés par la structure et le fonctionnement du cerveau. Ils peuvent apprendre plusieurs niveaux de représentation dans le but de modéliser des relations complexes entre les données.

Figure 2.1 : Un processus de Deep Learning : les images sont transmises à un réseau, qui apprend automatiquement les caractéristiques et classe les objets. **[Web07]**

Le Deep Learning est basé sur l'idée des réseaux de neurones artificiels et il est taillé pour gérer de larges quantités de données en ajoutant des couches au réseau. Un modèle de deep learning a la capacité d'extraire des caractéristiques à partir des données brutes grâce aux multiples couches de traitement composé de multiples transformations linéaires et non linéaires et apprendre sur ces caractéristiques petit à petit à travers chaque couche avec une intervention humaine minime. **[MDY17]**

2. Pourquoi le deep learning

Les algorithmes de ML fonctionnent bien pour une grande variété de problèmes. Cependant ils ont échoues à résoudre quelques problèmes majeurs de l'IA telle que la reconnaissance vocale et la reconnaissance d'objets. **[MDY17]**

L'IA émotionnelle regroupe un ensemble de cas d'utilisation de la machine learning pour lesquelles les techniques d'apprentissage profond sont particulièrement bien adaptées. **[MDY17]**

Le principe fondamental de ML est de chercher à extraire des caractéristiques « features » qui aide à la classification et pour les extraire il y a deux possibilités :

- Soit à l'aide d'un expertise métier, celle d'un physionomiste par exemple, pour construire manuellement et directement ces variables prédictives à partir d'un visage . L'inconvénient de ce processus d'extraction manuel est qu'il comporte une part d'arbitraire et qu'il n'exploite pas nécessairement l'intégralité de l'information disponible dans une photo ou dans une vidéo.

- Soit on utilisant un réseau de neurones profond capables d'extraire les caractéristiques par collection de toute l'information utile disponible dans les données brutes et pour bénéficier de cet atout il faudra cependant renoncer à interpréter les prédictions d'un tel modèle et disposer d'un grand nombre de d'exemples étiquetés. Comme dans beaucoup de problèmes d'IA c'est là que réside l'une des principales difficultés : construire des bases de données assez riches pour entraîner les algorithmes de deep learning à reconnaître des émotions.

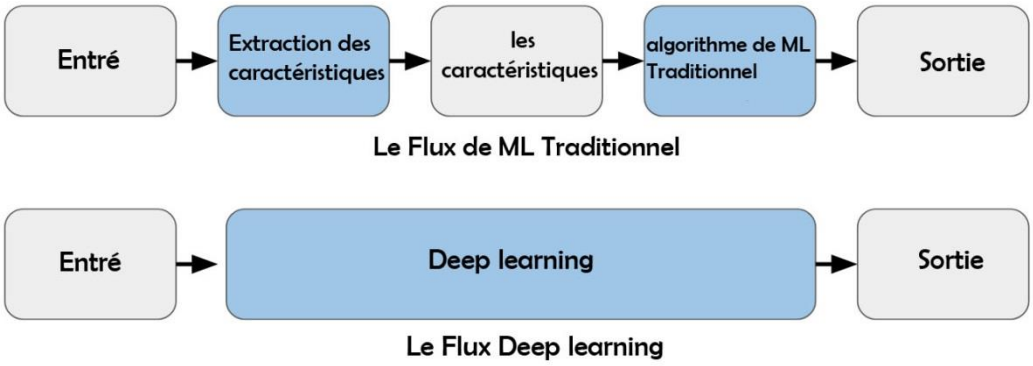

Figure 2.2 : Le procède du ML classique comparé à celui du Deep Learning [Web11]

Donc l'apprentissage en profondeur utilise des réseaux de neurones pour apprendre des représentations utiles de caractéristiques directement à partir de données

3. Les Réseaux de neurones artificiels

Les techniques et les calculs numériques le plus complexe permettent aux ordinateurs de faire le traitement durant un temps très court et on les comparant avec le cerveau humain, on notera que, théoriquement, l'ordinateur devrait être plus puissant que notre cerveau:
S'il comprend 10^9 transistors alors le temps de commutation est de 10^{-9} secondes.
Le cerveau humain contient 10^{11} neurones, mais ceux-ci ont seulement un temps de commutation d'environ 10^{-3} secondes. La plus grande partie du cerveau fonctionne en permanence, par contre la plus grande partie de l'ordinateur est un espace de sauvegarde de données passives. Ainsi, l'ordinateur est statique, le cerveau comme un réseau neuronal biologique peut se réorganiser au cours de sa durée de vie, et est donc en mesure d'apprendre, pour compenser les erreurs et ainsi de suite et c'est la capacité d'apprentissage qu'on ne trouve pas dans les ordinateurs. **[DKR07]**
Le but de l'utilisation des réseaux de neurones est de s'inspirer de la biologie pour établir un système artificiel capable d'apprendre et de compenser ses erreurs donc c'est un modèle de calcul dont la conception est inspirée du fonctionnement des neurones biologiques.
On trouve beaucoup de types de réseaux neurones, chaque type a pour un objectif particulier.

Rappel sur les réseaux de neurones :

Les points essentiels qu'on peut retenir sur les réseaux de neurones sont les suivants :

- Ce sont des systèmes composés de neurones répartis en plusieurs couches connectées entre elles
- ces systèmes peuvent résoudre de divers problèmes statistiques en général, et spécialement des problèmes de classification, en calculant à partir de l'entrée du réseau le score (ou la probabilité) de chaque classe. La classe attribuée à l'objet c'est celle de la probabilité la plus élevé
- à l'entrée de chaque couche les données sont traitées et transformée en calculant une combinaison linéaire puis en appliquant une fonction non-linéaire, appelée fonction d'activation. Les coefficients de la combinaison linéaire définissent les paramètres (ou poids) de la couche
- La dernière couche calcule les probabilités finales à partir d'une fonction d'activation (classification binaire) ou la fonction *softmax*(classification multi-classes)

- On associée aussi une fonction de perte *lossfunction* à la couche finale pour calculer l'erreur de classification. Il s'agit en général de l'entropie croisée **(accurancy)**
- On calcule les poids des couches par rétropropagation du gradient : calcule les paramètres qui minimisent la fonction de perte régularisée progressivement on partant de la dernière couche à la première couche, L'optimisation se fait avec une descente du gradient stochastique. **[Web02]**

3.1 Exemple réseaux neurones artificiels *:* [BMB17]

Neurone Formel : c'est une représentation mathématique et un model informatique de neurone biologique sa structure possède généralement plusieurs entrées et une sortie, dans un simple neurone formel on calcule la somme pondérée des données en entrées, puis on applique des fonctions d'activations, généralement non linéaire et à la sortie on obtient la valeur finale ce type de neurone est généralement l'unité élémentaire des réseaux de neurones artificiels utilisé dans une ou plusieurs couches d'autres types de réseaux .

Neurones multicouche : Ou le perceptron multicouche (multi layer perceptron MLP) est utilisé comme classifieur linéaire, il est de type réseau neuronal formel structuré en plusieurs couches où circule l'information d'une couche à une autre jusqu' à la sortie sans cycle (feed-forward) dans un seul sens ; les neurones de la couche de sortie correspondant toujours aux sorties du système.

Neurones récurrents : Les réseaux de Neurones récurrents (RNNs) permettent d'analyser les séquences de vecteurs tout comme les modèles de Markov cachés. C'est un réseau qui est structuré de tel façon qu'il prend en charge le recyclage de l'information entres les couches au cours du temps car on peut réinjecter les vecteurs qui sort d'une couche (même à la couche du sortie de système) a une autre (même au couches précédentes). Ce qui nous permet de conserver dans le réseau la mémoire de ce qui s'y est passé depuis le début.

Réseaux de Hopfield : Il a été découvert par le physicien John Hopfield en 1982 est un réseau particulier RNNs à temps discret dont la matrice des connexions est symétrique et nulle sur la diagonale et où la dynamique est asynchrone (un seul neurone est mis à jour à chaque unité de temps).

4.Réseaux de neurones convolutifs CNN

4.1 Présentation :

Les réseaux de neurones convolutifs CNN (Convolutional Neural Network) , sont les structures les plus performantes pour faire la classification des images,

Le nom « réseau de neurones convolutif » indique que le réseau emploie une opération mathématique appelée convolution. La convolution est une opération linéaire spéciale. Les réseaux convolutifs sont simplement des réseaux de neurones qui utilisent la convolution à la place de la multiplication matricielle dans au moins une de leurs couches.

Ils comportent deux parties principales. S'il y on a en entrée, une image qu'elle doit être sous la forme d'une matrice de pixels, de 2 dimensions pour une image en niveaux de gris et en 3 dimensions si elle est en couleur, pour représenter les couleurs fondamentales [Rouge, Vert, Bleu]. Cette image passe par **la première partie** d'un CNN qui est la partie convolutive Elle fonctionne comme un extracteur de caractéristiques des images. L'image passe à travers une succession de filtres, ou noyaux de convolution, pour la transformé en nouvelles images appelées cartes de convolutions « feature maps » . Certains filtres intermédiaires réduisent la résolution de l'image par une opération de maximum local. Au final, les cartes de convolutions sont mises à plat et concaténées en un vecteur de caractéristiques, appelé code CNN . Le résultat en sortie de la partie convolutive est branché en entrée d'une deuxième partie, constituée de couches entièrement connectées (perceptron multicouche) qui consiste à combiner les caractéristiques de tous le réseau pour classer l'image et à la sortie qui est une couche comportant un neurone par classe, on obtient des valeurs numériques généralement normalisées entre 0 et 1, pour présenter la distribution de probabilité sur les classes . **[DKR07]**

4.2 Architecture de réseaux de neurone convolutifs

Comme nous l'avons mentionnée précédemment, les réseaux de neurones convolutifs sont basés sur le perceptron multicouche(MLP), et il sont inspirés du comportement du cortex visuel des vertébrés. Bien qu'efficaces pour le traitement d'images, les MLP ont beaucoup de mal à gérer des images de grande taille, ce qui est dû à la croissance exponentielle du nombre de connexions avec la taille de l'image. Par exemple, si on prend une image de taille 32x32x3 (32 de large, 32 de haut, 3 canaux de couleur), un seul neurone entièrement connecté dans la première couche cachée du MLP aurait 3072 entrées (32*32*3). Une image 200x200 conduirait ainsi à traiter 120 000 entrées par neurone ce qui, multiplié par le nombre de neurones, devient énorme. **[MMZ17]**

L'architecture CNN est formée par un empilement de couches de traitement indépendantes :

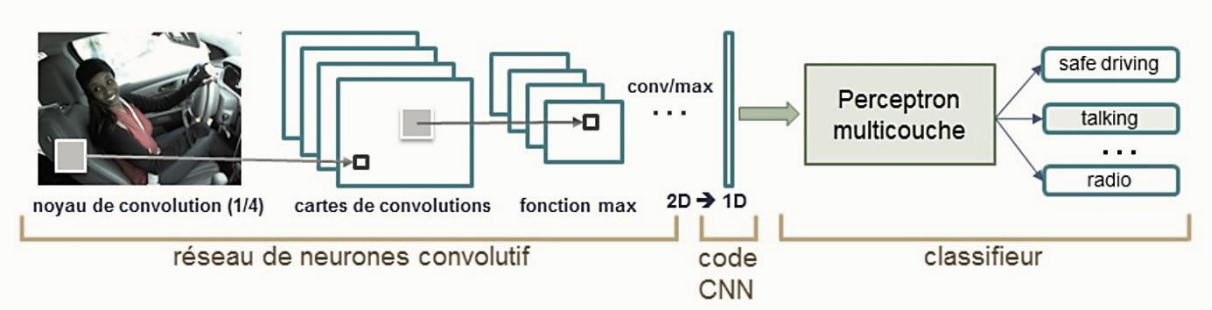

Figure 2.3 Architecture standard d'un réseau de neurone convolutionnel [MMZ17]

4.2.1 La couche de convolution (CONV) :

Trois hyper paramètres permettent de dimensionner le volume de la couche de convolution (aussi appelé volume de sortie) : la profondeur, le pas et la marge.

1. Profondeur de la couche : nombre de noyaux de convolution (ou nombre de neurones associés à un même champ récepteur).

2. Le pas : contrôle le chevauchement des champs récepteurs. Plus le pas est petit, plus les champs récepteurs se chevauchent et plus le volume de sortie sera grand.

3. La marge (à 0)' ou 'zero padding ' : parfois, il est commode de mettre des zéros à la frontière du volume d'entrée. La taille de ce 'zero-padding' est le troisième hyperparamètre. Cette marge permet de contrôler la dimension spatiale du volume de sortie. En particulier, il est parfois souhaitable de conserver la même surface que celle du volume d'entrée **[MMZ17]**

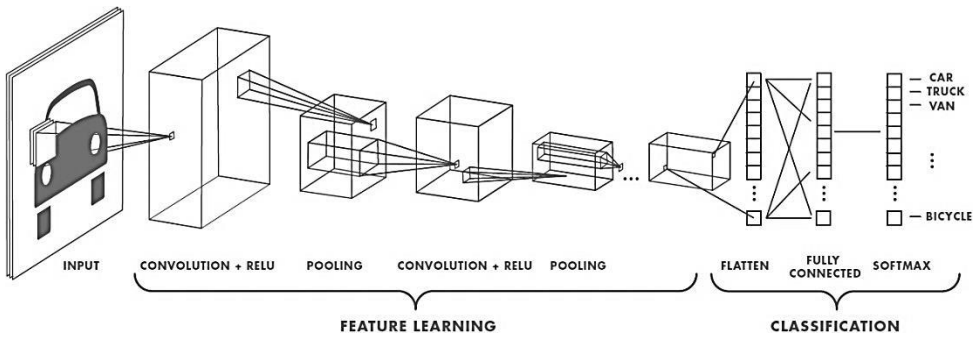

Figure2.4 *:*Exemple de réseau composé de nombreuses couches à convolution. Des filtres sont appliqués à chaque image utilisée pour l'apprentissage à différentes résolutions, et la sortie de chaque image convoluée est utilisée comme entrée de la couche suivante.**[Web07]**

Dans la terminologie du réseau convolutif, le premier argument de la convolution est souvent appelé l'entrée (input) et le second argument comme noyau (kernel). La sortie est parfois appelée la carte des caractéristiques (feature map).

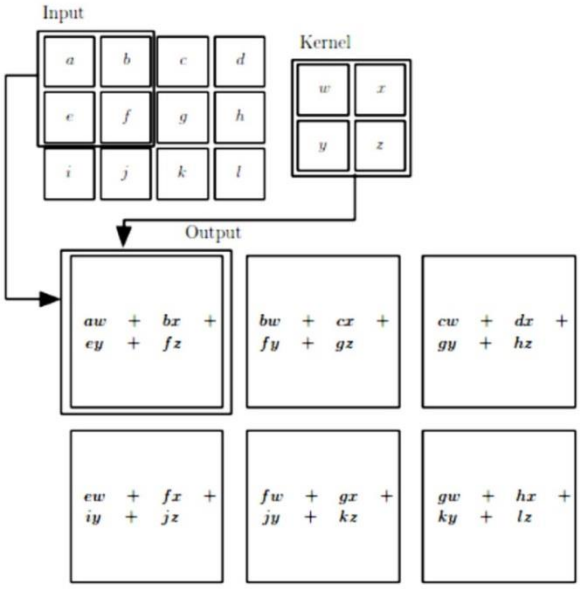

Figure 2.5 : Exemple d'une convolution 2D. **[GBC16]**

4.2.2 Couche de pooling

Le pooling est une forme de sous-échantillonnage de l'image, il permet de réduire progressivement la taille des représentations afin de réduire la quantité de paramètres et de calcul dans le réseau ainsi que l'invariance aux petites translations ,il est donc fréquent d'insérer périodiquement une couche de pooling entre deux couches convolutives successives d'une architecture CNN pour contrôler le sur-apprentissage. L'opération de pooling créait aussi une forme d'invariance par translation. La couche de pooling fonctionne indépendamment sur chaque tranche de profondeur de l'entrée et la redimensionne uniquement au niveau de la surface.

La forme la plus courante est une couche de mise en commun avec des filtres de taille 2x2 (largeur/hauteur) et comme valeur de sortie la valeur maximale en entrée. On parle dans ce cas de « Max-Pool 2x2 ».

Il est possible d'utiliser d'autres fonctions de pooling que le maximum. On peut utiliser un « average pooling » la sortie est la moyenne des valeurs du patch d'entrée.

Le pooling permet de gros gains en puissance de calcul. Cependant, en raison de la réduction agressive de la taille de la représentation et donc de la perte d'information associée,

la tendance actuelle est d'utiliser de petits filtres (type 2x2). Il est aussi possible d'éviter la couche de pooling mais cela implique un risque sur-apprentissage plus important

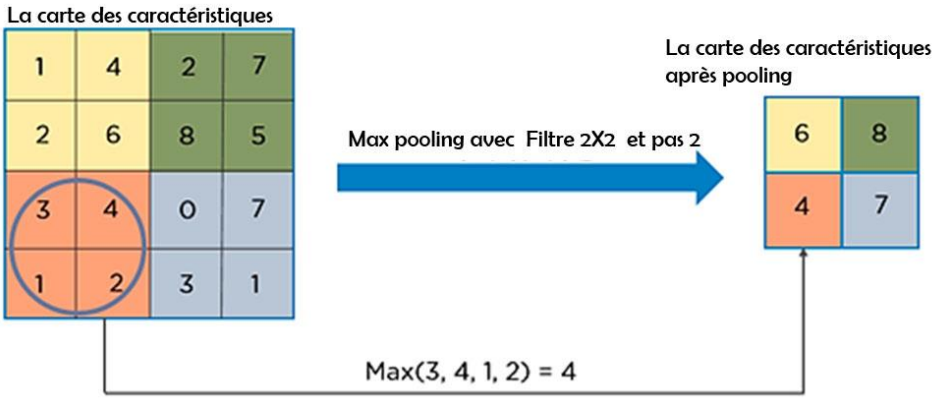

Figure 2.6 : Pooling avec un filtre 2x2 et un pas de 2

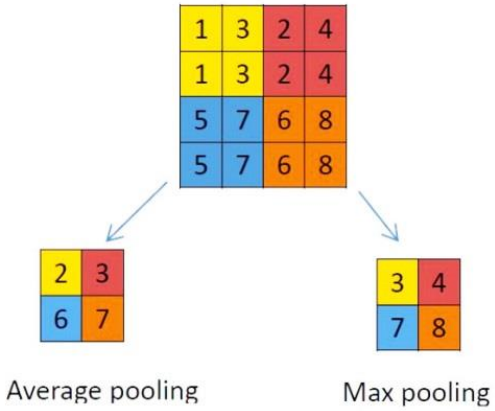

Figure2.7 (à gauche) Average pooling : chaque case correspond à la moyenne du carré d'entrée de la même couleur, ex de la case jaune : (1+ 3+ 1+ 3)/4 = 2. (à droite) Max pooling: chaque case correspond à la valeur maximum du carré d'entrée de la même couleur, ex de la case bleu : max(5, 7, 5, 7) = 7 **[MDY17]**

4.2.3 Couches entièrement connectées

Après l'extraction des caractéristiques des entrées, on attache à la fin du réseau un perceptron ou bien un MLP (multi layer perceptron). Le perceptron prend comme entrée les caractéristiques extraites et produit un vecteur de N dimensions ou N est le nombre de classe ou chaque élément est la probabilité d'appartenance à une classe. Chaque probabilité est calculée à l'aide de la fonction ***softmax*** **[MDY17]** dans le cas où les classes sont exclusivement mutuelles

4.3 Les fonctions d'activation

la fonction d'activation est une fonction mathématique appliquée à un signal en sortie d'un neurone artificiel. Le terme de "fonction d'activation" vient de l'équivalent biologique "potentiel d'activation", seuil de stimulation qui, une fois atteint entraîne une réponse du neurone. La fonction d'activation est souvent une fonction non-linéaire. Leur but est de permettre aux réseaux de neurones d'apprendre des fonctions plus complexes qu'une simple régression linéaire car le fait de multiplier les poids d'une couche cachée est juste une transformation linéaire.

Exemple de fonction d'activation Le **ReLu** (Rectified Linear Units) : c'est une fonction qui permet d'éliminer tous les valeurs négatifs.

5. Quelques réseaux convolutifs célèbres :

• **LeNet [YLC98]**: Les premières applications réussies des réseaux convolutifs ont été développées par Yann LeCun dans les années 1990. Parmi ceux-ci, le plus connu est l'architecture **LeNet** utilisée pour lire les codes postaux, les chiffres, etc.

• AlexNet **[AKR12]** : Le premier travail qui a popularisé les réseaux convolutifs dans la vision par ordinateur était AlexNet, développé par Alex Krizhevsky, Ilya Sutskever et Geoff Hinton. Ce CNN été soumis au défi de la base ImageNet en 2012 et a nettement surpassé ses concurrents. Le réseau avait une architecture très similaire à LeNet, mais était plus profond, plus grand et comportait des couches convolutives empilées les unes sur les autres (auparavant, il était commun de ne disposer que d'une seule couche convolutifs toujours immédiatement suivie d'une couche de pooling).

• ZFnet **[MDZ14]** : C'était une amélioration de AlexNet en ajustant les hyper-paramètres de l'architecture, en particulier en élargissant la taille des couches convolutifs et en réduisant la taille du noyau sur la première couche.

• GoogLeNet**[CSZ15] :** C'est un modèle de Google. Sa principale contribution a été le développement d'un module inception qui a considérablement réduit le nombre de paramètres dans le réseau (4M, par rapport à AlexNet avec 60M). En outre, ce module utilise le global Average pooling ce qui élimine une grande quantité de paramètres. Il existe également plusieurs versions de GoogLeNet, parmi elles, **Inception-v4 [FCH 17]** et

Xception [OAR17] ce dernier est l'un des modèles lesquels notre système s'inspire, plus de détails dans le chapitre de conception.

• **ResNet** [KHZ16] **:** Residual network développé par Kaiming He et al. Été le vainqueur de ***ILSVRC 2015***[1]. Il présente des sauts de connexion et une forte utilisation de la batch normalisation. Il utilise aussi le global AVG pooling au lieu du PMC à la fin. [MDY17]

VGG Net : [KSI15] Il s'agit d'une structure du Visual Geometry Group d'Oxford réalisée par Andrea Vedaldi et Andrew Zisserman (en 2017).

6. Conclusion

Dans ce chapitre, nous avons éclairci les réseaux de neurones et les différentes types ensuite on a focalisé notre attention sur les réseaux de neurones convolutifs CNN et leur structures, et ses différentes couches, nous avons présenté après quelques exemple d'architectures, parmi eux Xception et VGGnet qui sera implémenté dans notre système.

Le chapitre suivant explique l'idée d'utilisé ces deux architectures pour un système de reconnaissance d'expression faciale.

[1]*ILSVRC :* **Large Scale Visual Recognition Challenge** ou "Compétition ImageNet de Reconnaissance Visuelle à Grande Échelle": *Elle consiste en une compétition logicielle dont le but est de détecter et classifier précisément des objets et des scènes dans les images naturelles*.

Chapitre 3 :

Conception d'un Système FER basé sur CNNs (FACECNN)

"Create with the heart;
build with the mind". Criss Jami

Chapitre 3 : Conception d'un Système FER basé sur CNNs (FACECNN)

1. Introduction :

Avec l'évolution actuelle des technologies et les quantités énormes d'information , l'apprentissage profond qui vise à donner aux ordinateurs la capacité d'apprendre à faire les taches humaines beaucoup mieux que l'humain lui-même, est une solution souvent idéal pour résoudre beaucoup de problèmes, et principalement les problèmes de reconnaissance .

La reconnaissance des expressions faciales est un problème important, qui trouve des applications dans différents domaines. Plusieurs méthodes traditionnelles ont été utilisées dans la reconnaissance d'expression telle que les SVM **[MNT16]** ,Adaboost **[YXC05],**Forêts aléatoires**[SBE09],** l'apprentissage profond (et principalement les CNN**)** permet de supprimer ou réduire fortement la dépendance des modelés physiques .

L'objectif de notre projet est d'utilisé un CNN pour reconnaitre les expressions faciales ; nous utilisant deux architectures différentes, afin nous comparons les résultats.

2. Présentation du système FACECNN

Nous proposons une conception de FACECNN un système de reconnaissance d'expression faciale à partir d'un visage en temps réel à base de réseaux CNNs.

FACECNN consiste à détecter un visage d'une personne à partir d'une image, séquence vidéo ou via une caméra pour connaître l'expression avec un taux de précision associé au six expression universel à savoir la joie, Le dégoût la peur, la colère, La tristesse, La surprise plus l'état neutre

2.1. Architecture générale de FACECNN :

Nos recherches concernant la reconnaissance d'expression faciale, nous ont permis de constater que toutes les solutions apportées à la reconnaissance d'émotions sont structurées selon la même architecture globale, en trois modules principaux fonctionnant indépendamment :

- **la détection de visage**,
- **l'extraction de caractéristiques**
- et **la classification.**

Et ce sont les étapes qui constituent FACECNN :

1. Détection de visage
2. Extraction de caractéristiques faciales
3. Classification des expressions

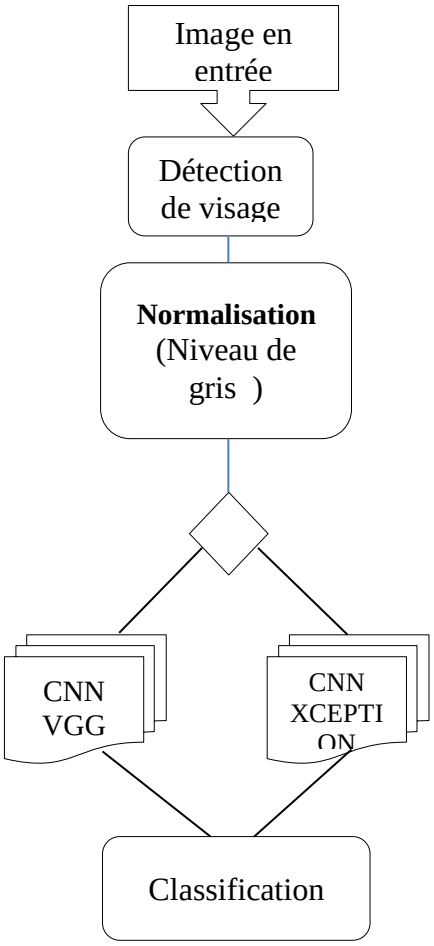

Figure3.2 : Schéma générale de FACECNN

.

2.1.1 La Détection de visage :

L'efficacité des systèmes FER dépend essentiellement de la méthode utilisée pour localiser le visage dans l'image. Dans notre méthode, nous utilisons l'algorithme Viola-Jones **[MBE19]** pour détecter diverses parties du visage humain telles que la bouche, les yeux, le nez, les narines, les sourcils, la bouche, les lèvres et oreilles.

Cet Algorithme explore les caractéristiques de type Haar via le classificateur Cascade, qui peut efficacement combiner de nombreuses fonctionnalités et déterminer les différents filtres sur un classificateur résultant.

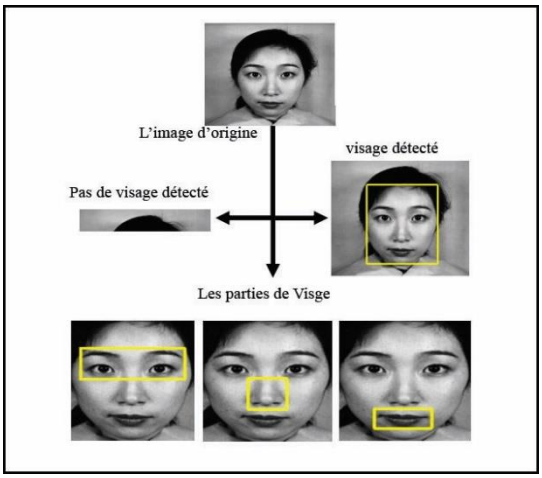

Figure 3.3 : Un exemple de l'algorithme Viola-Jones **[MBE19]**

2.1.2 Extraction de caractéristiques faciales.

Une fois le visage détecté dans l'image, le système lance le processus d'extraction des caractéristiques qui va convertir les données des pixels à des représentations et configuration plus réduite et optimal pour que la représentation extraite soit utilisé dans le processus de la classification , cette étape réduit les dimensions de l'image en entrée en gardant les données les plus utiles pour la classification.

Ce processus est considéré comme le cœur de notre système FACECNN, il est basé sur la technologie de réseau de neurones convolutionel (CNN), Nous avons utilisé deux architectures de CNN, VGG16 **[KSI15]** et Xception **[OAR17],** ces deux architectures ont été proposé pour la classification d'image, dans notre cas nous les testons pour la reconnaissance d'expression faciale, dans ce qui suit nous représentant chaque architecture et son adaptation à notre problème

3. Présentation l'architecture VGG16

VGG16 est un modèle de réseau neuronal convolutionnel proposé par K. Simonyan et A. Zisserman de l'Université d'Oxford dans l'article intitulé «*Very deep convolutional networks for large-scale image recognition* » **[KSI15]**.

Le modèle atteint une précision de 92,7% sur la base de données ImageNet **[AKR12]** qui contient plus de 14 millions d'images appartenant à 1000 classes.

C'est un modèle de 16 couches qui reçoit a l'entré une taille d'image RVB de taille 224 x 224 VGG16 utilise les filtres de noyaux 3X3 dans les couche de convolution (qui est la plus petite taille pour capturer la notion de gauche / droite, haut / bas, centre) et un filtre de taille 2X2 en

max pooling, toutes les couches cachées sont équipées de la non-linéarité de rectification (ReLU).

Et à la fin du réseau, trois couches entièrement connectées qui calcule le score de chaque classe des caractéristiques extraites de la convolution dans les étapes précédentes. Les cartes de caractéristiques de la couche finale sont représentées sous forme de vecteurs avec des valeurs scalaires, la fonction de classification soft-max. qui formule le score final entre 0 et 1 pour présenter la classification sou forme un pourcentage partagé entre les différentes classes, Les couches entièrement connectées sont coûteuses en termes de calcul, des approches alternatives ont été proposées au cours des dernières années. Par exemple la mise en œuvre d'une couche Globale de pooling moyenne qui aide à réduire le nombre de paramètres dans le réseau de manière significative.

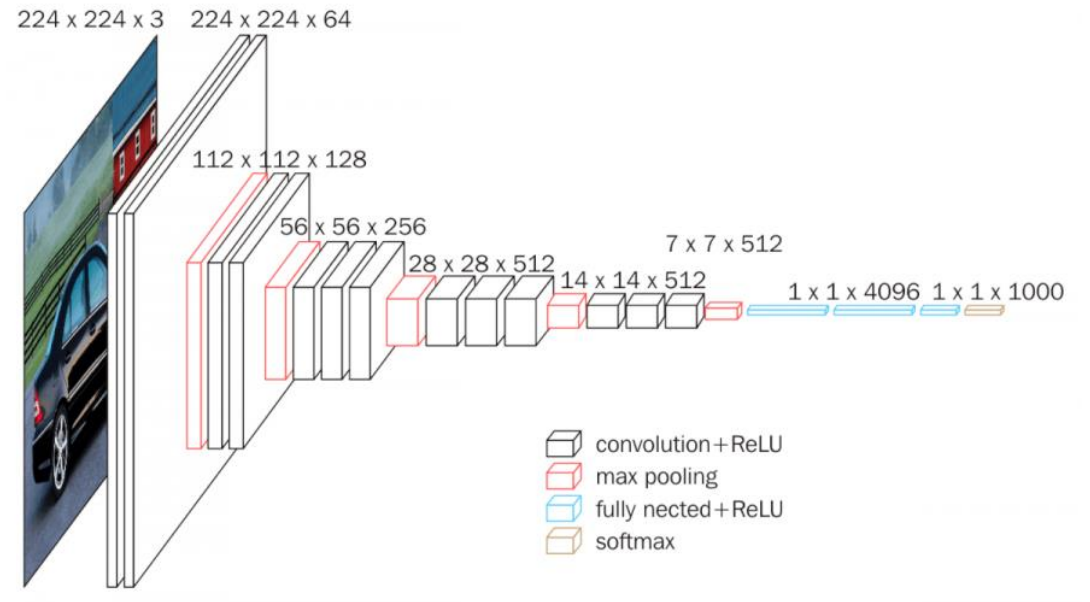

Figure 3.4 : Architecture de VGG **[Web04]**

La phase d'apprentissage du modèle prend des semaines sur cette base des données gigantesque [**KSI15].** Malheureusement, VGGNet présente deux inconvénients majeurs :

1. Il est péniblement lent à s'entraîner.
2. Les poids de l'architecture de réseau sont eux-mêmes assez importants (en ce qui concerne la bande passante / disque).

En raison de sa profondeur et de son nombre de nœuds entièrement connectés, VGG16 dépasse la taille de 533 Mo. Cela rend le déploiement de VGG une tâche fastidieuse. Le format VGG16 est utilisé dans de nombreux problèmes de classification des images à apprentissage approfondi ; cependant, les architectures de réseau plus petites sont souvent plus souhaitables**.**

4. Présentation l'architecture Xception

Xception est une structure moderne proposé par *François Chollet* lui-même, créateur et responsable de la maintenance de la bibliothèque Keras de Google. Xception est une extension de l'architecture Inception **[FCH 17]** qui remplace les modules Inception standard par des **convolutions séparables en profondeur** ce qui permet de diminuer la taille de l'architecture jusqu' au 91Mo.(voir figure 3.5)

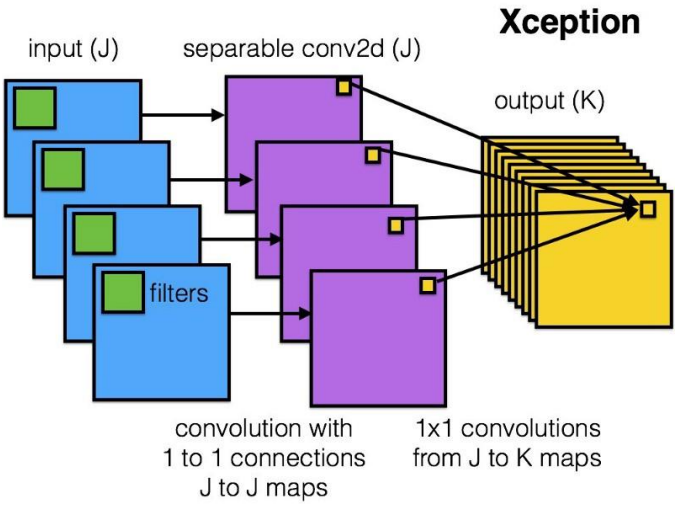

Figure 3.5 : Architecture de Xception **[Web05]**

La spécification majeure de la structure Xception c'est le concept de **Convolution spatiale en profondeur [FCH 17].**

Mais qu'est-ce que c'est et en quoi est-ce différent d'une convolution normale?

4.1 Définition de la Convolution séparable :

C'est une solution au problème de la complexité des calculs au niveau de convolution qui rend le fonctionnement de réseau un peu long, avec moins de calculs, le réseau est en mesure de traiter davantage de données en un temps réduit. Il existe deux types de convolutions séparables : **[FCH 17]**

- **la convolution séparable spatiale :**

C'est la solution la plus facile parce qu'elle consiste à diviser la couche de convolution en deux, elle traite principalement des dimensions spatiales d'une image et du noyau : la largeur et la hauteur,

Malheureusement elle est très limité par conséquent elle n'est pas très utilisé dans l'apprentissage en profondeur

- **la convolution séparable en profondeur :**

C'est la solution qui nous intéresse dans notre étude car elle traite non seulement des dimensions spatiales, mais également de la dimension en profondeur c'est-à-dire le nombre de canaux.

Une image d'entrée peut avoir 3 canaux: RVB. Après quelques convolutions, une image peut avoir plusieurs canaux. Semblable à la convolution spatiale séparable, une convolution séparable en profondeur divise un noyau en 2 noyaux distincts qui effectuent deux convolutions: la convolution en profondeur et la convolution ponctuelle (taille de 1x1).

4.2 La convolution séparable en profondeur dans Xception

La convolution modifiée dans la structure Xception est la **convolution ponctuelle suivie d'une convolution en profondeur**. Selon lequel, une convolution 1 × 1 est effectuée avant toutes les convolutions spatiales de taille n × n. Ainsi, c'est un peu différent de l'original. Le modèle Xception atteint une précision de 94,5% sur la base de données ImageNe **[FCH 17]** .

5. Architecture de FACECNN

La création de FACECNN a été inspirée par les deux précédentes structures VGG16 et Xception. Après plusieurs tests des différents modèles, nous proposons deux modèles différents de CNNs, qui ont été conçus avec l'idée d'obtenir une meilleure précision en rapport avec le nombre de paramètres de réseau et en fonction de la base des données choisi. Réduire le nombre des paramètres nous aide à surmonter deux problèmes importants.

- D'abord, l'utilisation de petite CNN nous soulage des performances lentes dans des systèmes à contraintes matérielles.

- Et Deuxièmement, la réduction des paramètres permet une meilleure généralisation dans un cadre *de rasoir Occam*[1] «principe de parcimonie » ou encore « principe d'économie ».

Notre premier modèle repose sur l'idée d'éliminer complètement les couches entièrement connectés. La seconde architecture combine la suppression du couche entièrement connectée et l'inclusion de la combinaison profondeur séparée convolutions et modules résiduels qui filtre et élimine les éléments inutiles de l'image. Le système FACECNN a été formé avec l'optimiseur ADAM **[DPK17]**.

A partir de notre recherche au sujet de CNN on a constaté que la majorité des modelés incluent un ensemble de couches entièrement connectées à la sortie de réseau comme VGG16.

Quelques modèles ont réduit l'ensemble des paramètres dans leurs dernières couches en incluant une fonction Globale de pooling moyenne « Opération Global de pooling » comme dans **Xception** , ce qui réduit l'ensemble des cartes de caractéristiques en une valeur scalaire en prenant la moyenne de tous les éléments de la carte.

Nous proposons d'éliminer les couches entièrement connectées dans tous les CNN de notre système et nous incluons la fonction Globale de pooling moyen de tel sorte, nous obtenons à la sortie des réseaux un nombre des cartes de caractéristiques égale au nombre de classe (07 ; les six expressions et l'état neutre). Ensuite, nous appliquons la fonction d'activation softmax, qui est le même principe du modèle Xception.

[1] Le rasoir d'Ockham ou rasoir d'Occam est un principe de raisonnement philosophique entrant dans les concepts de rationalisme et de nominalisme. Également appelé « principe de simplicité », « principe d'économie » ou « principe de parcimonie » (en latin *lex parsimoniae*), il peut se formuler comme suit :

Pluralitas non est ponenda sine necessitate (les multiples ne doivent pas être utilisés sans nécessité)[wikipedia]

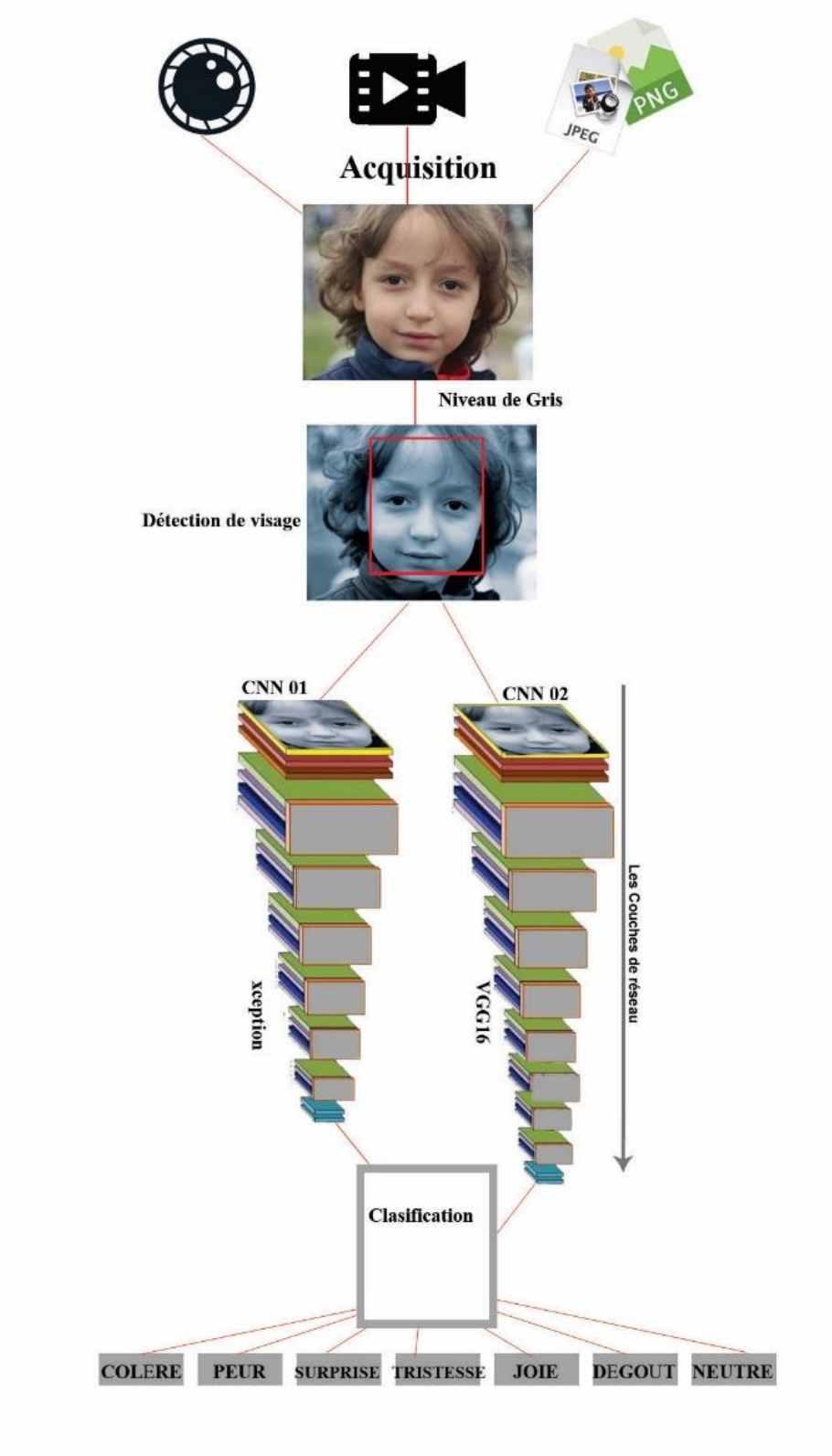

Figure 3.6 **: Architecture de FACECNN**

Conclusion

Dans ce chapitre, nous avons introduit les deux modèles VGG16 et Xception utilisé pour la classification des images, et nous avons remarqué que chaque architecture a une caractéristique spécifique qui la différencié totalement de l'autre.

Dans le prochain chapitre, nous présentons l'utilisation de ces deux modèles pour les tester sur la reconnaissance d'expression faciale d'un visage réel, afin de révéler les performances d'un système FER par apprentissage profond, et de comparer les différents résultats des deux modèles.

Chapitre 4 :

Implémentation et Réalisation de FACECNN

"Becoming is better than being" Carol Dweck

Chapitre 4 : Implémentation et Réalisation de FACECNN

1. Introduction :

L'objectif de ce projet de fin d'étude est de concevoir et de réaliser une application qui permet de reconnaitre les expressions faciales d'un visage réel. Nous nous sommes intéressés à l'utilisation de réseau neuronal convolutif, nous avons testé deux architectures différentes l'architecture Xception, et une version améliorée de l'architecture VGG16.
Dans ce chapitre nous présentons l'implémentation de notre système et les différentes étapes de réalisation, l'environnement de développement, ainsi que les différents outils utilisés pour effectuer ce travail. Nous poursuivons ce chapitre par la présentation des différents résultats expérimentaux obtenus, quelques captures d'écrans de notre application

2. Logiciels et outils :

Présentation des outils de programmation ;

En utilisant le langage généraliste python, la bibliothèque de vision par ordinateur Onencv, Tensarflow de google et le Framework Keras pour la mise en œuvre de nos réseaux de neurones, ainsi que d'autre modules et bibliothèques (numpy, marplotlib, etc...).

1.1 Python :

Le langage de programmation scientifique multi-paradigme Python a été créé en 1989 par Guido van Rossum, aux Pays-Bas. Le nom Python vient d'un hommage à la série télévisée Monty Python's Flying Circus dont G. van Rossum est fan. La première version publique de ce langage a été publiée en 1991. La dernière version de Python est la version 3.7 a été publiée en juin 2018. La version 2 de Python est désormais obsolète et cessera d'être maintenue après le 1er janvier 2020. Ce langage de programmation présente de nombreuses caractéristiques intéressantes :

- Il est gratuit et multiplateforme. : Windows, Mac OS X, Linux, Android, iOS, depuis les mini-ordinateurs Raspberry Pi jusqu'aux supercalculateurs.
- C'est un langage de haut niveau. Il demande relativement peu de connaissance sur le fonctionnement d'un ordinateur pour être utilisé.
- C'est un langage interprété. Un script Python n'a pas besoin d'être compilé pour être exécuté, contrairement à des langages comme le C ou le C++.
    - C'est un langage dynamique ,extensible Il favorise la programmation structurée fonctionnelle et orientée objet. C'est-à-dire qu'il est possible de concevoir en Python

des entités qui miment celles du monde réel (une cellule, une protéine, un atome, etc.) avec un certain nombre de règles de fonctionnement et d'interactions.

- Il est relativement simple à prendre en main. **[PFP19].**
- La syntaxe de Python est très simple et, combinée à des types de données évolués (listes, dictionnaires,...), conduit à des programmes à la fois très compacts et très lisibles.

- Il gère ses ressources (mémoire, descripteurs de fichiers...) sans intervention du programmeur. **[Web08]**

- Enfin, il est très utilisé en bio-informatique et l'intelligence artificielle et plus généralement en analyse de données. Toutes ces caractéristiques font que Python est otils idéal pour implémenter notre application

1.2 Opencv

OpenCV (Open Source Computer Vision Library) est une bibliothèque qui aide à la vision par ordinateur Depuis son lancement officiel en 1999 par l'équipe d'Intel, un certain nombre de programmeurs ont contribué aux derniers développements de la bibliothèque. Le dernier changement majeur intervenu en 2009 (OpenCV 2) qui inclut les principales modifications apportées à l'interface C. la version de la bibliothèque est disponible sur le site officiel d'OpenCV. De nos jours, la bibliothèque à 2500 algorithmes optimisés.

OpenCv est spécialisé en :

- Manipulation des données d'images et vidéo, les matrices et les vecteurs .
- Différentes structures de données dynamiques (listes, files d'attente, ensembles, arbres, graphiques).
- Analyse du mouvement (flux optique, segmentation du mouvement, suivi), Reconnaissance d'objets.
- Interface graphique de base (image / vidéo à afficher, gestion du clavier et de la souris, barres de défilement…) **[GAG06]**

1.3 Tensorflow :

TensorFlow est un framework de programmation pour le calcul numérique de Google initié et développé par l'équipe Google Brain spécialisé dans l'intelligence artificielle, et rendu Open Source en Novembre 2015, et devenir très rapidement l'un des frameworks les plus utilisés pour le Deep Learning,

Les caractérestiques principale de tensorflow sont :

- Multi-plateformes (Windows , Linux, Mac OS, et même Android et iOS !)
- APIs en Python, C++, Java et Go (l'API Python est plus complète cependant, c'est sur celle-ci que nous allons travailler)
- Temps de compilation très courts dû au backend en C/C++
- Supporte les calculs sur CPU, GPU et même le calcul distribué sur cluster
- Une documentation extrêmement bien fournie avec de nombreux exemples et tutoriels
- Last but not least: Le fait que le framework vienne de Google et que ce dernier ait annoncé avoir migré la quasi-totalité de ses projets liés au Deep Learning en TensorFlow est quelque peu rassurant .Cependant dans FACECNN nous utilisant Cette plateforme via la bibliothèque Keras ce qui est sur-couche à TensorFlow donc un niveau plus haut que Tensorflow

1.4 Keras :

C'est une API de haut niveau permettant de créer et de former des modèles d'apprentissage en profondeur, capable de s'exécuter sur **Tensorflow,** il est utilisé pour le prototypage rapide, la recherche avancée et la production. Les modèles Keras sont fabriqués en reliant des blocs de construction configurables ensemble, donc Keras nous permet de créer de nouvelles couches, des fonctions et développer des modèles à la pointe de la technologie avec peu de restrictions en quelques lignes de code.

1.5 Nmpy :

NumPy est le paquet fondamental du calcul scientifique avec Python. Il contient entre autres des choses:

- un puissant objet tableau à N dimensions
- fonctions sophistiquées (diffusion)
- des outils pour l'intégration du code C / C ++ et Fortran
- capacités utiles d'algèbre linéaire, de transformée de Fourier et de nombres aléatoires

Outre ses utilisations scientifiques évidentes, NumPy peut également être utilisé comme conteneur efficace multidimensionnel de données génériques. Des types de données arbitraires peuvent être définis. Cela permet à NumPy d'intégrer de manière transparente et rapide avec une grande variété de bases de données.

Numpy est sous licence BSD, ce qui permet sa réutilisation avec peu de restrictions**.[Web09]**

3. Base de données utilisé :

Comme décrit dans l'état de l'art, pour des meilleures résultats, il est préférable d'entraînés le réseau avec beaucoup d'échantillons d'image. Cela augmenterait la précision et améliorerait la performance des modèles.

Malheureusement, les bases de données de grande taille n'existent pas publiquement, mais nous avons accès à deux bases de données publics ; la base de visages expressifs FER2013 et la base de données CK+. Pour notre système FACECNN, nous utilisons la base FER2013 pour faire l'apprentissage.

Figure 4.1 : Cinq échantillons d'expression de chacune des 7 émotions de l'ensemble de données de la base FER2013.

4. Implémentation et réalisation :

Notre système FACECNN est constitué de deux modules

- le premier pour la détection de visage
- et le deuxième pour faire la reconnaissance d'expression.

Dans ce qui suit, nous détaillons chacun d'eux

4.1 Module de détection :

Pour détecter des visages nous avons utilisé une méthode populaire, proposée par Paul Viola et Michael Jones dans leur article "Détection rapide d'objets utilisant une cascade de fonctions simples" en 2001**. [WEB03]**

Dans notre système nous avons utilisé haarcascade_eye.xml de la bibliothèque **OpenCV** qui fournit la méthode Haar Cascade.

4.2 Module de reconnaissance :

Pour reconnaitre l'expression faciale du visage détecté par le module précédent, nous utilisons des réseaux de neurones convolutionel.

Le premier élément à prendre en compte est le choix d'architecture du CNN. Les architectures de réseaux neuronaux sont des combinaisons de couches qui se transmettent les sorties les unes après les autres.

Nous avons choisi une architecture initiale en fonction des approches rencontrées lors de notre étude de l'état de l'art sur les implémentations existantes de FER. Des résultats de performance très différents sont obtenus en fonction du choix et de la séquence des couches constituant les architectures de réseau neuronal. Pour améliorer le modèle et trouver une bonne architecture, nous avons entamé un processus d'expérimentation et de test avec diverses combinaisons.

Les figures 4.2 et 4.3 Présentent les deux architectures utilisées :

- **L'architecture VGG16 De FACECNN**

Le premier CNN est inspiré du modèle VGG, il est constitué de plusieurs couches, onze (11) en tous (après l'élimination des deux premières couches à cause de la taille de l'image déjà réduite à l'entrée, et les trois dernières couches entièrement connectées), dont 10 couches de convolution utilisant des filtres de tailles différentes et on applique une

normalisation par lot (BatchNormalization), ensuite des corrections ReLU sont appliquées pour éliminer les valeurs négatifs

A la fin, on a ajouté une couche de Pooling Global moyen, et on a appliqué la fonction Softmax pour calculer le taux des 7 Classes d'expressions (les six expressions universels et l'état neutre), elle renvoie donc un vecteur de taille 7, qui contient les probabilités d'appartenance à chacune des classes.

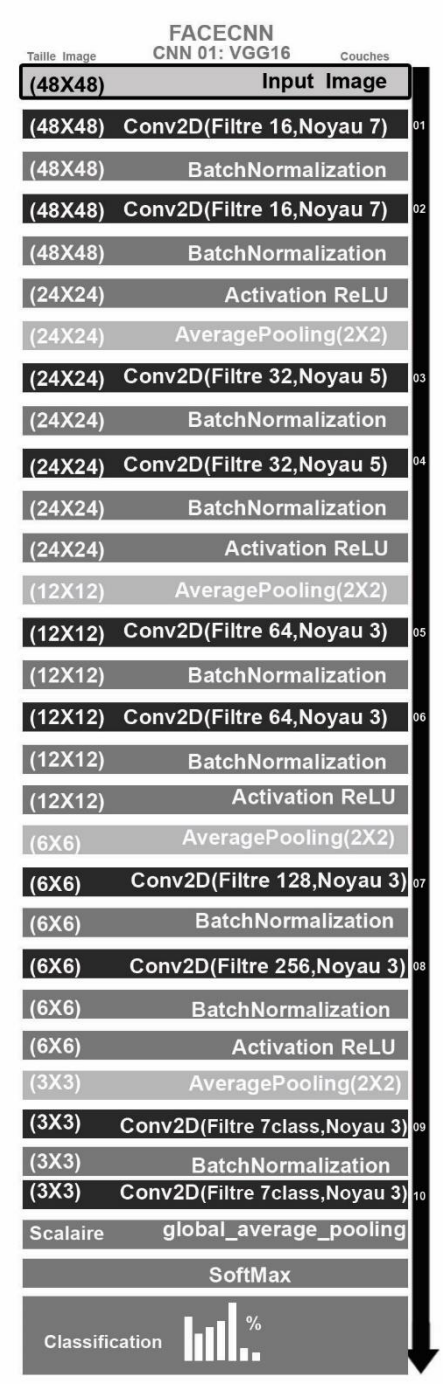

Figure 4.2 CNN 01 De FACECNN

- **L'architecture Xception De FACECNN**

Le deuxième CNN est inspiré du modèle Xception, il est constitué de deux couches de convolutions normale en premier lieu, ensuite il y a deux couches de convolutions, chacune d'elle est suivie par une convolution séparable en profondeur, Normalisation par lot (BatchNormalization) et Activation ReLu et une autre convolution séparable suivis par la normalisation et fonction Maxpooling

Et à la fin de notre réseau, on a ajouté une couche de convolution suivie par un pooling moyen et la fonction Softmax qui nous renvoi la classification. La figures 4.3 illustre les différentes couches de cette architecture.

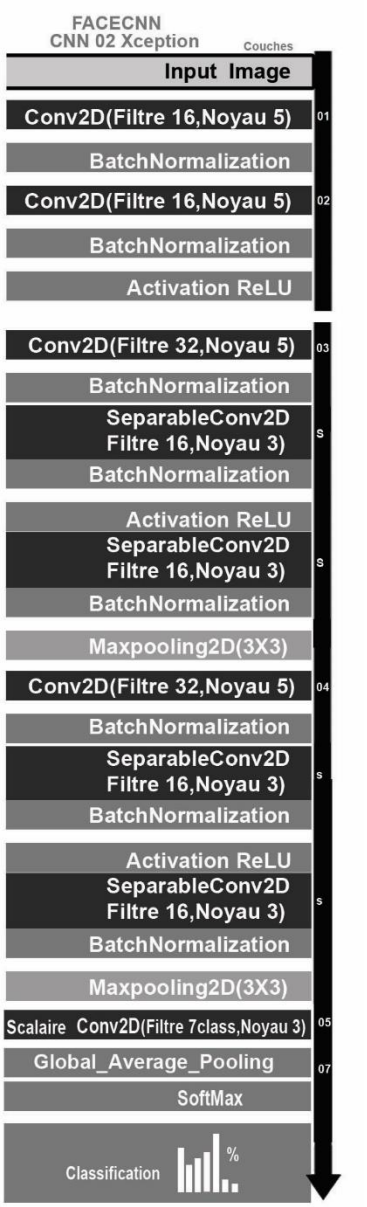

Figure 4.3 CNN 02 De FACECNN

Le système FACECNN au niveau de l'implémentation contient plusieurs modules, dans ce qui suit, nous allons spécifier chacun avec son rôle et son objectif.

4.3. Module qui charge la base des données

Dans la base des données utilisées (FER2013), les images sont traitées de manière à ce que les visages soient presque centrés et chaque visage occupe environ la même quantité d'espace dans chaque image. Chaque image a été classé dans l'une des sept classes exprimer différentes émotions du visage. Ces émotions faciale sont été classés comme suit : 0 = en colère, 1 = dégoûté,2 = Peur, 3 = Heureux, 4 = Triste, 5 = Surprise et 6 = Neutre.

Les images données sont divisées en trois ensembles différents qui sont **l'apprentissage, la validation, et ensembles de test**. Il y a environ 28 709 images de formation, 3589 images de validation et 3589 images à tester.

Chaque image est donnée sous forme de chaîne images de taille (48 × 48) stockée en tant que vecteur de ligne sous le format (.csv).

Le code ci-dessous charge la base de données et faire un traitement des images et les préparé pour les servir au modèle CNN choisi par la suite.

Le code se compose de deux fonctions :

La fonction charger data (*def charger_data*):

Cette fonction lit le fichier csv(le type de fichier de la base FER2013) et convertit la séquence de pixels de chaque ligne en image de dimension 48 * 48 et retourne des image contiens des visages et des étiquettes d'émotion Return*(faces, emotion).*

```
def charger_data():
    data = pd.read_csv(lien_data)
    pixels = data['pixels'].tolist()
    width, height = 48, 48
    faces = []
    for pixel_sequence in pixels:
        face = [int(pixel) for pixel in pixel_sequence.split(' ')]
        face = np.asarray(face).reshape(width, height)
```

La fonction d'entrée pour le prétraitement *def preprocess_input:*

C'est une procédure standard pour l'apprentissage et jugé meilleur méthode pour le modèle de réseaux de neurones dans les problèmes de vision par ordinateur

En les redimensionnant entre -1 et 1. Les images sont redimensionnées à [0,1] en le divisant par 255. De plus, la soustraction par 0,5 et la multiplication par 2 modifient le va jusqu'à [-1,1]. [-1,1]

```
def preprocess_input(x, v2=True):

    x = x.astype('float32')

    x = x / 255.0

    if v2:
```

4.4 Module de réseaux CNNs :

Ce module contient des différentes architectures testé au coure de notre études et principalement les deux architecture XCEPTION et VGG.

Il existe plusieurs techniques de programmation en python en utilisant la bibliothèque **Keras** pour construire un réseau neuronal profond et qui sont applicables à la plupart des problèmes de vision par ordinateur .Mais tous passent par une combinaison de plusieurs fonctions différentes pour construire le CNN (voire les figures 4.5 et 4.6).

File Edit Format Run Options Window Help

```
from keras.layers import Activation, Convolution2D, Dropout, Conv2D
from keras.layers import AveragePooling2D, BatchNormalization
from keras.layers import GlobalAveragePooling2D
from keras.models import Sequential
from keras.layers import Flatten
from keras.models import Model
from keras.layers import Input , Dense
from keras.layers import MaxPooling2D
from keras.layers import SeparableConv2D
from keras.layers.advanced_activations import PReLU
from keras.layers.convolutional import ZeroPadding2D
from keras.optimizers import SGD,Adadelta
from keras import layers
from keras.regularizers import l2


#***************Xception Archi*********************************
#**************************************************************

def XCNN(input_shape, num_classes):
    img_input = Input(input_shape)
    archi = Conv2D(32, (3, 3), strides=(2, 2), use_bias=False)(img_input)
    archi = BatchNormalization(name='block1_conv1_bn')(archi)
    archi = Activation('relu', name='block1_conv1_act')(archi)
    archi = Conv2D(64, (3, 3), use_bias=False)(archi)
    archi = BatchNormalization(name='block1_conv2_bn')(archi)
    archi = Activation('relu', name='block1_conv2_act')(archi)

    residual = Conv2D(128, (1, 1), strides=(2, 2),
                      padding='same', use_bias=False)(archi)
    residual = BatchNormalization()(residual)

    archi = SeparableConv2D(128, (3, 3), padding='same', use_bias=False)(archi)
    archi = BatchNormalization(name='block2_sepconv1_bn')(archi)
    archi = Activation('relu', name='block2_sepconv2_act')(archi)
    archi = SeparableConv2D(128, (3, 3), padding='same', use_bias=False)(archi)
    archi = BatchNormalization(name='block2_sepconv2_bn')(archi)

    archi = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(archi)
    archi = layers.add([archi, residual])

    residual = Conv2D(256, (1, 1), strides=(2, 2),
                      padding='same', use_bias=False)(archi)
    residual = BatchNormalization()(residual)

    archi = Activation('relu', name='block3_sepconv1_act')(archi)
    archi = SeparableConv2D(256, (3, 3), padding='same', use_bias=False)(archi)
    archi = BatchNormalization(name='block3_sepconv1_bn')(archi)
    archi = Activation('relu', name='block3_sepconv2_act')(archi)
    archi = SeparableConv2D(256, (3, 3), padding='same', use_bias=False)(archi)
    archi = BatchNormalization(name='block3_sepconv2_bn')(archi)
```

Figure 4.4 un aperçu sur le code source pour l'architecture Xception

```
File  Edit  Format  Run  Options  Window  Help
    return model

def VGGCNN2(input_shape,num_classes):

    model = Sequential()
    model.add(Convolution2D(filters=16, kernel_size=(5, 5), padding='same',
                            name='image_array', input_shape=input_shape))
    model.add(BatchNormalization())
    model.add(Convolution2D(filters=16, kernel_size=(5, 5),
                            strides=(2, 2), padding='same'))
    model.add(BatchNormalization())
    model.add(Activation('relu'))
    model.add(Dropout(.25))

    model.add(Convolution2D(filters=32, kernel_size=(5, 5), padding='same'))
    model.add(BatchNormalization())
    model.add(Convolution2D(filters=32, kernel_size=(5, 5),
                            strides=(2, 2), padding='same'))
    model.add(BatchNormalization())
    model.add(Activation('relu'))
    model.add(Dropout(.25))

    model.add(Convolution2D(filters=64, kernel_size=(3, 3), padding='same'))
    model.add(BatchNormalization())
    model.add(Convolution2D(filters=64, kernel_size=(3, 3),
                            strides=(2, 2), padding='same'))
    model.add(BatchNormalization())
    model.add(Activation('relu'))
    model.add(Dropout(.25))

    model.add(Convolution2D(filters=64, kernel_size=(1, 1), padding='same'))
    model.add(BatchNormalization())
    model.add(Convolution2D(filters=128, kernel_size=(3, 3),
                            strides=(2, 2), padding='same'))
    model.add(BatchNormalization())
    model.add(Activation('relu'))
    model.add(Dropout(.25))

    model.add(Convolution2D(filters=256, kernel_size=(1, 1), padding='same'))
    model.add(BatchNormalization())
    model.add(Convolution2D(filters=128, kernel_size=(3, 3),
                            strides=(2, 2), padding='same'))

    model.add(Convolution2D(filters=256, kernel_size=(1, 1), padding='same'))
    model.add(BatchNormalization())
    model.add(Convolution2D(filters=num_classes, kernel_size=(3, 3),
                            strides=(2, 2), padding='same'))

    model.add(Flatten())
    #model.add(GlobalAveragePooling2D())
    model.add(Activation('softmax',name='predictions'))
    return model
```

Figure 4.5 un aperçu sur le code source pour l'architecture VGG

4.5. Module d'apprentissage

Après la création de CNN, l'étape suivante consiste à faire l'apprentissage. Pour l'apprentissage du réseau, nous utilisons la fonction «fit ()» de notre modèle avec les paramètres suivants:

Données d'entraînement (Xtrain), données cibles (ytrain), données de validation et nombre d'époques.

```
model.fit_generator(data_generator.flow(xtrain, ytrain,
batch_size),
steps_per_epoch=len(xtrain) / batch_size,
epochs=num_epochs,   verbose=1, callbacks=callbacks,
validation_data=(xtest,ytest))
```

Pour nos données de validation, nous utilisons l'ensemble de test fourni dans notre ensemble de données, que nous avons divisé en Xtest et ytest.

TrainX et testX constituent les données d'image elles-mêmes, tandis que trainY et testY constituent les étiquettes.

```
- ETA: 10:04 - loss: 1.8338 - acc: 0.3016
605/897 [==================>.....] - ETA: 10:02 - loss: 1.8339 - acc: 0.3015
606/897 [==================>.....] - ETA: 10:00 - loss: 1.8341 - acc: 0.3016
607/897 [==================>.....] - ETA: 9:57 - loss: 1.8339 - acc: 0.3016
608/897 [==================>.....] - ETA: 9:55 - loss: 1.8339 - acc: 0.3018
609/897 [==================>.....] - ETA: 9:53 - loss: 1.8338 - acc: 0.3018
610/897 [==================>.....] - ETA: 9:51 - loss: 1.8330 - acc: 0.3021
611/897 [==================>.....] - ETA: 9:49 - loss: 1.8328 - acc: 0.3023
612/897 [==================>.....] - ETA: 9:47 - loss: 1.8327 - acc: 0.3023
613/897 [==================>.....] - ETA: 9:45 - loss: 1.8323 - acc: 0.3025
614/897 [==================>.....] - ETA: 9:42 - loss: 1.8321 - acc: 0.3027
615/897 [==================>.....] - ETA: 9:40 - loss: 1.8319 - acc: 0.3027
616/897 [==================>.....] - ETA: 9:38 - loss: 1.8320 - acc: 0.3026
617/897 [==================>.....] - ETA: 9:36 - loss: 1.8320 - acc: 0.3026
618/897 [==================>.....] - ETA: 9:34 - loss: 1.8317 - acc: 0.3028
619/897 [==================>.....] - ETA: 9:32 - loss: 1.8315 - acc: 0.3029
620/897 [==================>.....] - ETA: 9:30 - loss: 1.8312 - acc: 0.3030
621/897 [==================>.....] - ETA: 9:28 - loss: 1.8313 - acc: 0.3028
622/897 [==================>.....] - ETA: 9:26 - loss: 1.8309 - acc: 0.3031
623/897 [==================>.....] - ETA: 9:24 - loss: 1.8309 - acc: 0.3032
624/897 [==================>.....] - ETA: 9:22 - loss: 1.8306 - acc: 0.3036
```

Figure 4.6 : le contenu de la fenêtre de sortie pendant la phase d'apprentissage

Le nombre d'époques est le nombre de fois où le modèle parcourt les données. Plus nous avons d'époques, plus le modèle s'améliorera, jusqu'à un certain point. Ensuite, le modèle cessera de s'améliorer à chaque époque.

La figure suivante représente notre modèle qui entraîne les données, puis les valide. Une époque est le nombre de fois que le modèle s'entraîne sur l'ensemble de nos données.

Un lot peut s'expliquer comme prenant de petites quantités, s'entraînant et prenant un peu plus. Le modèle soi meilleur aussi si la perte est faible que possible.

5. L'interface de l'application FACECNN :

C'est là où nous testant nos modèles apprentis, est c'est l'interface de l'application ou on acquit les images via une webcam et le système présent les visages détectés et l'état

émotionnels de l'individu avec une fenêtre qui illuter la comparaison des résultats obtenus des deux architectures

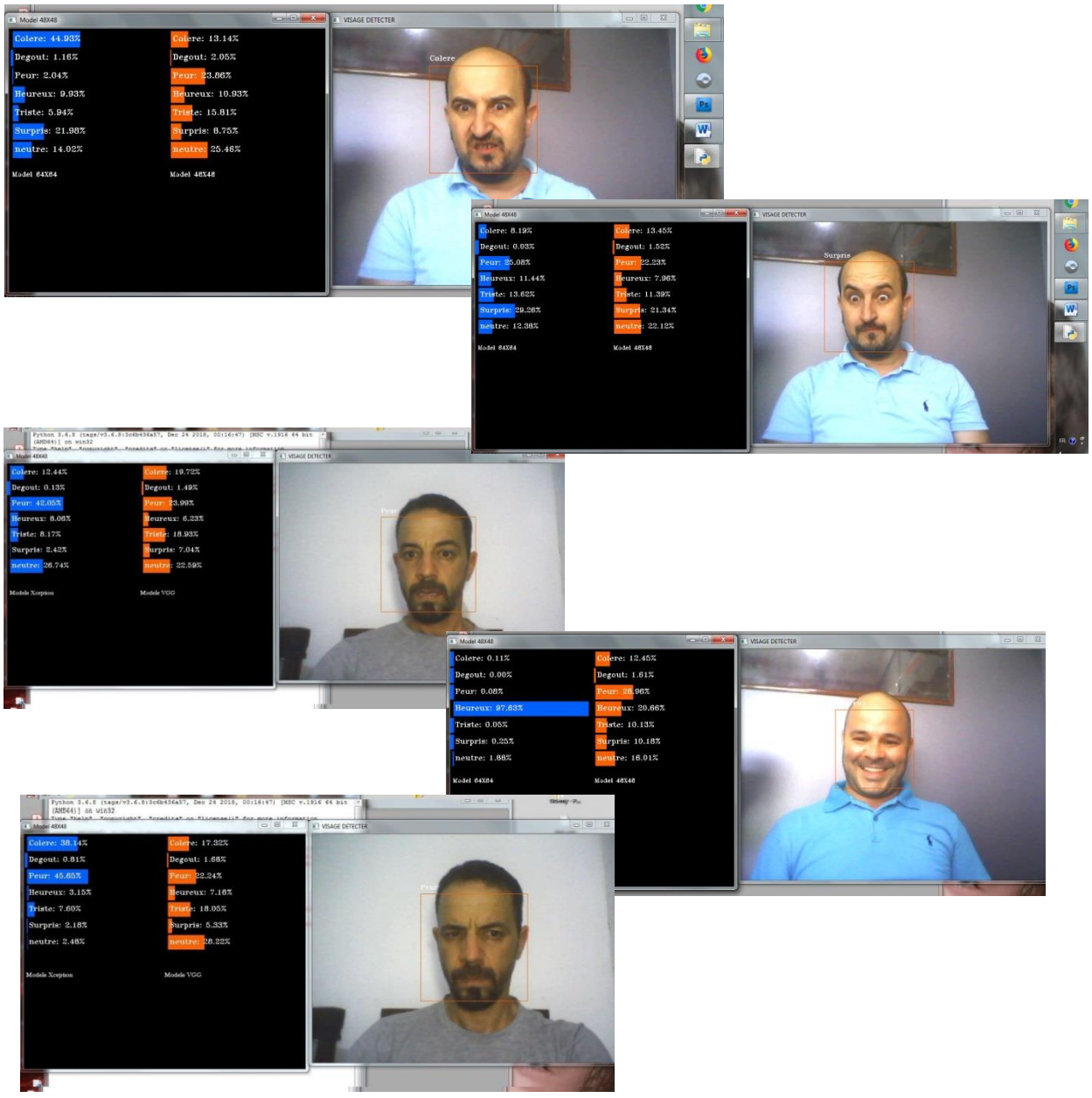

Figure 4.7 : aperçu sur FACECNN (reconnaissance de quelques expressions et affiche les résultats des deux architectures)

6.Les mesures de performance

Il y a plusieurs métriques d'évaluation qui existe pour évaluer les systèmes de reconnaissance d'expression faciale. Nous avons pris en compte trois mesures de performance **[BCK18]**:

- **Précision(Accuracy):** Elle nous indique la précision avec laquelle notre modèle permet de prédire les expressions, c'est le rapport entre le nombre des résultats correct (les résultats correct positifs et les résultats correct négatif) au nombre total de la population de la base des donnée

Accuracy (ACC) = TP + TN / Total population [BCK18]

Tel que :

TP : le nombre de résultats corrects positifs dans l'ensemble de données,

TN : le nombre de résultats corrects négatifs.

- **La matrice de confusion** Est une matrice qui représente la mesure de qualité d'un système de classification, et qui reflété la performance de l'apprentissage.

Les lignes indiquent les instances d'une classe réelle dans notre cas les six expressions universelles et les colonnes représentent les instances d'une classe prédite ou estimé.

Afin d'examiner les émotions qui sont le plus souvent confondues entre elles, cette matrice nous permet de voir facilement cette confusion entres les expressions dans notre classification ainsi le degré de confusion et donc on peut savoir si notre système de classification parvient à classifier correctement.

7. Résultats expérimentaux et discussions

Du point de vue des résultats, nous obtenons après l'apprentissage du premier modèle VGG16 une précision de test de **62**%,

Tandis que dans le cas de notre second apprentissage du modèle Xception, un taux de précision estimé à 73% est observé. Au moyen de deux graphes de figure 4.9 et figure 4.11, nous présentons le taux de précision et celui de perte obtenus :

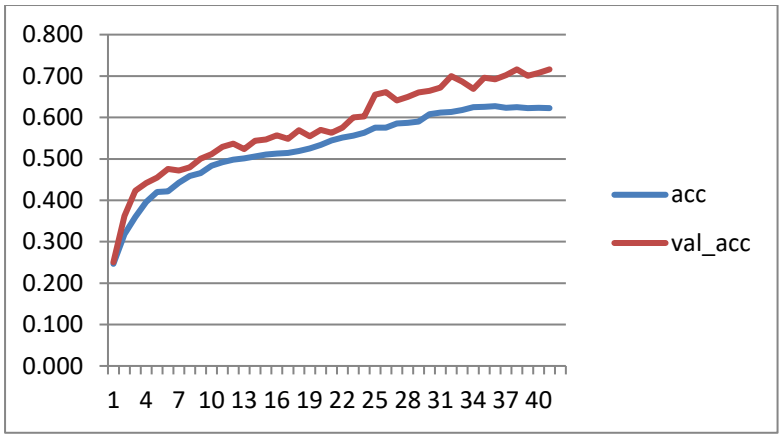

Figure 4.9 Apprentissage et validation de la précision dans VGG16 construit dans FACECNN

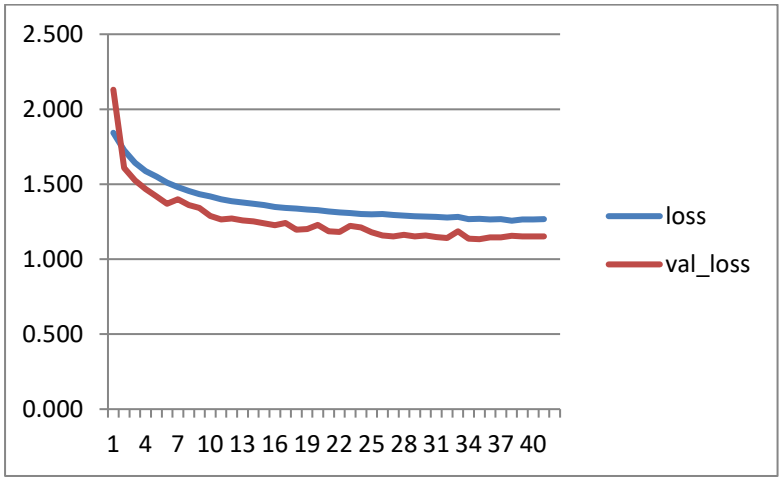

Figure 4.10 Apprentissage et validation de la perte dans VGG16 construit dans FACECNN

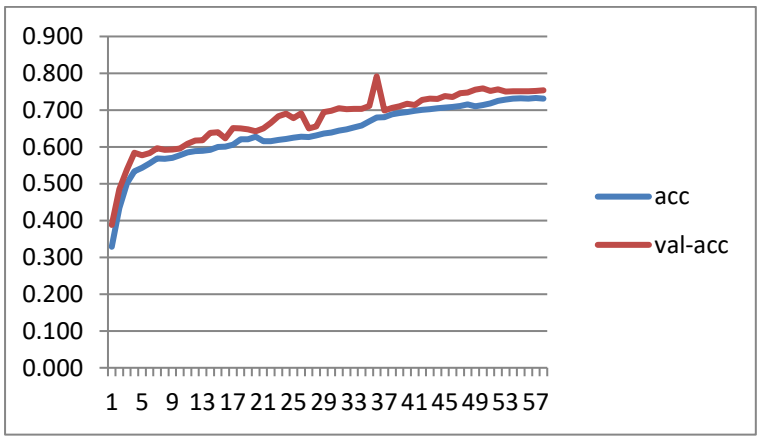

Figure 4.11 Apprentissage et validation de la précision dans Xception construit dans FACECNN

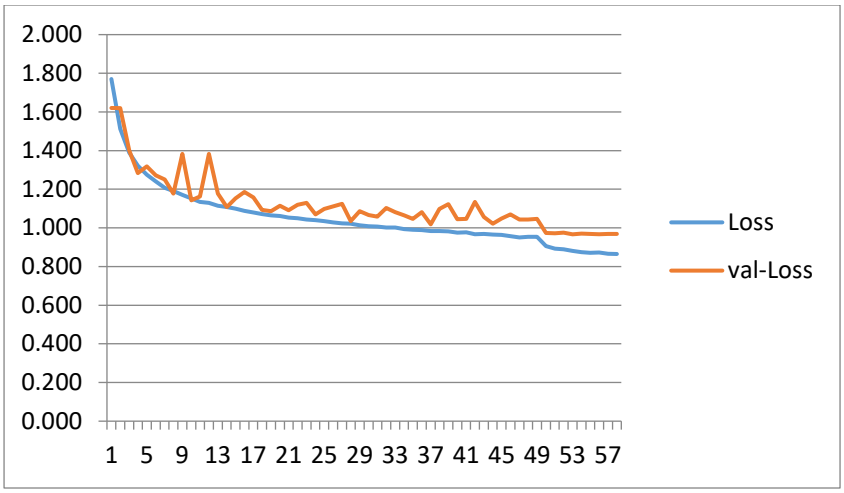

Figure 4.12 Apprentissage et validation de la perte dans Xception construit dans FACECNN

Discussion 1 :

La performance du modèle VGG16 obtenue sur la base Fer2013 est relativement faible (62%) (Compte tenu de la taille réduite de l'échantillon d'images de la base) par rapport à la performance du modèle Xception, qui est de 73% sur la même base, bien que le nombre de paramètres traité dans Xception est beaucoup moins que celui du VGG16. Cela permet de bénéficier d'un gain en temps confédérale, si on applique l'apprentissage sur une base de données avec plus d'échantillons ,

Ceci s'explique par le fait que le CNN est plus performant dans le cas d'un bon choix de technique de réduction de paramètres et non plus de les traitées au maximum.

	Nombre de paramètres	Paramètres traitable	Paramètres non traitable	précision
VGG16	559 671	557 815	1 856	0.62
Xception	58 423	56 951	1 472	0.73

Table 4.1 Comparaison entre les deux architectures

Discussion 2 : Nous constatons à travers la matrice de confusions de VGG16 que certaines classes sont plus orientées vers la classe colère.

Il y a beaucoup d'images dans la classe Triste qui sont reconnues comme des images de la classe colère. Donc, le modèle n'est pas capable de distinguer complètement les classes "colère" et "triste", et on remarque aussi que le modèle VGG16 ne reconnais pas efficacement les deux expressions dégoût et surpris.

Ces confusions sont probablement dues à la taille de la base de données.

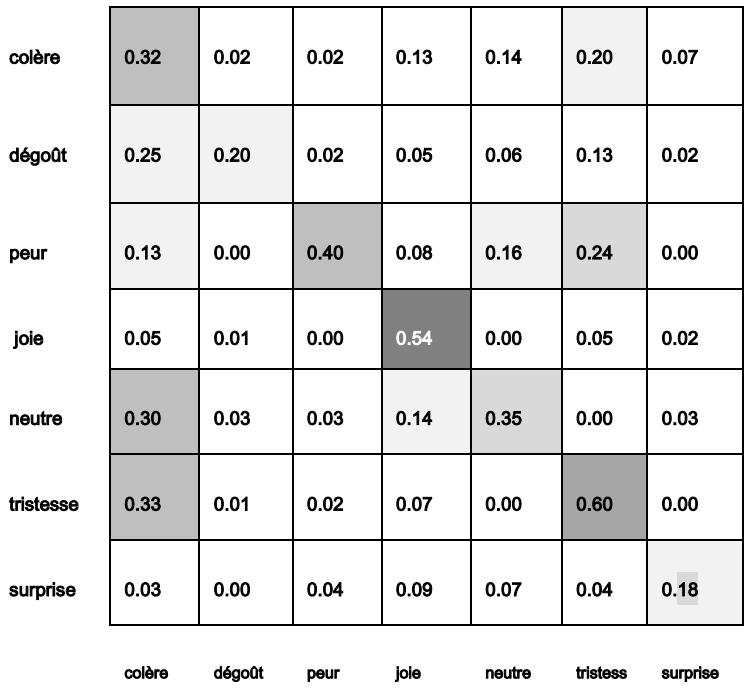

Figure 4.13 La matrice de confusion pour le model VGG16 sur la base FER2013

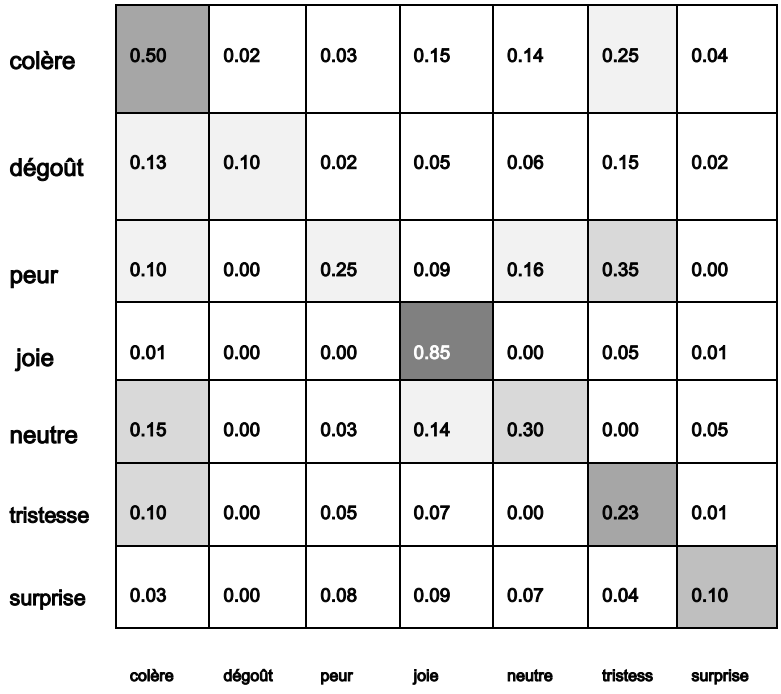

Figure 4.13 La matrice de confusion pour le model Xception sur la base FER2013

Pour la matrice de confusions du modèle Xception, il y a aussi une confusion importante entre les classes mais moins que celle du modèle VGG16. Par contre, ce modèle trouve aussi

des difficultés à reconnaitre les deux expressions dégoût et surpris. Et cela peut être dû au nombre faible d'échantillons de ces classes dans la base.

Conclusion

Dans ce chapitre, nous avons présenté notre application de reconnaissance d'expression faciale basée sur les réseaux de neurones de types CNN, nous avons utilisé deux architectures différentes la première inspirée de l'architecture VGG16 et la deuxième c'est l'architecture Xception de Google, nous avons présenté aussi les différents résultats obtenus pour chaque architecture. Notre système à été testé sur la base de données FER2013 de Kaggle. Pour conclure, les expérimentations ont montré que le modèle de Xception est le plus efficace que le VGG16 en terme gain de temps et taux de précision.

Conclusion générale

Conclusion générale et perspectives

Les réseaux de neurones convolutifs sont devenus l'algorithme principal de la vision par ordinateur ces dernières années. L'histoire de la conception des réseaux de neurones à convolution a commencé avec des modèles qui étaient de simples piles de convolutions pour les opérations d'extraction de caractéristiques et de pooling maximal pour le sous-échantillonnage spatial. Ces idées ont été affinées par la suite dans d'autres architectures, dans laquelle les opérations de convolution étaient répétées plusieurs fois entre les opérations de max-pooling, permettant ainsi au réseau d'apprendre des caractéristiques plus riches à toutes les échelles spatiales. Différentes architectures ont été proposée, parmi lesquels, nous nous sommes inspiré de l'architecture VGG proposée en 2014 et de l'architecture Xception proposée en 2018, pour proposer notre propre CNN, et le tester sur la base de données Fer2013.

L'objectif de ce projet est d'étudier l'utilisation d'un réseau de neurones convolutif et qui exploite les convolutions séparables en profondeur, tout en éliminant les couches entièrement connectées pour la reconnaissance de l'expression faciale. Nous avons montré comment les fonctions séparables ajoutées dans le CNN Xception pourraient améliorer le résultat de précision en démunies énormément de temps. La richesse des couches qui constitue le réseau VGG16 permet de traité un maximum de paramètres. Néanmoins cette technique a abouti à une précision moins que Xception, bien que ce dernier ait éliminé beaucoup de paramètres, ce qui présente un avantage par rapport à l'architecture VGG16. Le gain de performance est notable lorsque les deux modèles peuvent être combinés dans un seul, qui nous pensons surpassera les méthodes de pointe. Nos expériences démontrent un avantage évident du modèle Xception, en appliquant l'apprentissage sur l'ensemble de données FER-2013.

Nous espérons que ce travail soit une référence d'une nouvelle architecture CNN, en apliquant une hybridation des deux architectures VGG et Xception. Comme perspectives, nous souhaitons :

- Tester notre modèle sur d'autres bases des données plus volumineuse telque ck+.
- Appliquer notre modèle sur des images 3D aquises par des caméras de profondeur
- Inclure plus de classes, doc étendre le FER pour reconnaitre les micros expression, ce qui permettra plus de précision pour indiquer l'état émotionnel.

Références

“Personal beauty is a greater recommendation that any letter of reference” Fanny Brice

Références :

[AKR12 *Imagenet classification with deep convolutional neural networks* **A. Krizhevsky, I. Sutskever - G. E. Hinton** in Advances in neural information processing systems, pp. 1097–1105 - University of Toronto 2012

[BCK18] *A Brief Review of Facial Emotion Recognition Based on Visual Information* **Byoung Chul Ko** University, Daegu , Korea 2018

[BMB17] *L`apprentissage profond (Deep Learning) pour la classification et la recherche d'images par le contenu* **Boughaba Mohammed** et **Boukhris Brahim** 2017

[CMS16] **]**: *Survey on RGB, 3D, Thermal, and Multimodal Approaches for Facial Expression Recognition: History, Trends, and Affect-related Applications* Ciprian **A. Corneanu, Marc Oliu, Jeffrey F. Cohn, and Sergio Escalera** 2016.

[DKR07] *A Brief Introduction to Neural Networks* **David Kriesel** University of Bonn in Germany 2007.

[DLS16] *: Facial emotion detection usingdeep learning* **Daniel Llatas Spiers** Institution de l'information et technologie *UPPSALA UNIVERSITE 2016.*

[DPK17*]* *Adam: a method for stochastic optimization* **Diederik P. Kingma** University of Amsterdam 2017

[EKM71] *Constants across cultures in the face and emotion*. **P. Ekman and W. V. Friesen**,
Journal of personality and social psychology, vol. 17, no. 2, pp. 124–129 , 1971.

[FCH 17] *Xception: Deep Learning With Depthwise Separable Convolutions* ***Francois Chollet***; The IEEE Conference on Computer Vision and Pattern

Recognition (CVPR) pp. 1251-1258 - 2017

[FKH10 *Reconnaissance automatique des émotions par données multimodales : expressions faciales et des signaux physiologiques* **Faiza Khalfi** Université Paul Verlaine - Metz, 2010.

[GAG06] *Introduction to programming with OpenCV* **Gady Agam** Department of Computer Science University of Chicago 2006

[GBC16] Livre **:** *Deep Learning* **I. Goodfellow -Y. Bengio - A. Courville** MIT Press, 2016. http://www. deeplearningbook.org.

[IAZ13*]* *Multi-Valued and Universal Binary Neurons : Theory, Learning and Applications* **I. Aizenberg, N. N. Aizenberg, J. P. Vandewalle** Springer Science & Business Media 2013

[JAH01] *Candide-3 – an updated parameterised face* **JörgenAhlberg** Image Coding Group Dept. of Electrical Engineering, Linköping University Linköping, SWEDEN 2001

[KGH10] *Reconnaissance des Expressions Faciales àBase d'Informations Vidéo ; Estimation de l'Intensité des ExpressionsFaciales*: **KHADOUDJA GHANEMLE** Université Mentouri de Constantine 2010.

[KHZ16] *Deep residual learning for image recognition* **K. He, X. Zhang, S. Ren, J. Sun** in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778- 2016.

[KSA15] *Very deep convolutional networks for large-scale image recognition* **Karen Simonyan** , **Andrew Zisserman** et **Visual Geometry Group**, Department of Engineering Science, University of Oxford 2015

[KSI15] *Very deep convolutional networks for large-scale image recognition* **Karen Simonyan - Andrew Zisserman** + **Visual Geometry Group**,

Department of Engineering Science, University of Oxford 2015

[MBE19] *Improved Facial Expression Recognition Based on DWT Feature for Deep CNN* **Ridha Ilyas Bendjillali - Mohammed Beladgham - Khaled Merit- Abdelmalik Taleb Ahmed** University, Bechar 2019

[MDY17] *Deep Learning pour la classification des images* **Moualek Djaloul Youcef** Université Abou Bakr Belkaid Tlemcen 2017

[MDZ14] *Visualizing and understanding convolutional networks* **M. D. Zeiler - R. Fergus** in European conference on computer vision, pp. 818–833- 2014

[MMZ17] *« Classification des images avec les réseaux de neurones convolutionnels »* **Mokri Mohammed Zakaria** , Université Abou Bakr Belkaid Tlemcen 2017.

[MNT16] *Automatic Facial Emotion Recognition Method Based on Eye Region Changes* **Mina Navraan ,Tarbiat Modares , Nasrollah Moghadam Charkari**, Faculty of Electrical and Computer Engineering, University, Tehran, 2016

[MTA91] Livre *Eigedces for Recognition* **Matthew Turk- Alex Pentland** Vision and Modeling Group The Media Laboratory Massachusetts Institute of Technology
Journal of cognitive neuroscience p.71-86, 1991

[OAR17] *Real-time Convolutional Neural Networks for Emotion and Gender Classification* **Octavio Arriaga** University of Bremen Germany 2017

[PFP19] livre *Cours de Python* **Patrick Fuchs - Pierre Poulain** Université Sorbonne paris 2019

[PHM99] *The FERET Evaluation Methodology for FaceRecognition Algorithms* P **Jonathon Phillips- Hyeonjoon Moon- Syed A Rizvi - Patrick J Rauss** National Institute of Standards and Technology Gaithersburg MD jonathonnistgov University of New York 1999

[PJP96] Livre *FERET (Face Recognition Technology) Recognition Algorithm Development and Test Results* **P. Jonathon Phillips, Patrick J. Rauss, and Sandor Z. Der** / Army Research Laboratory October 1996

[SBE09] *Forêts Aléatoires: De l'Analyse des Mécanismes de Fonctionnement à la Construction Dynamique* **Simon Bernard** Université de Rouen, 2009

[SNO18] *L'observation des émotions au travers des communications non-verbales Mémoire en vue de l'obtention du diplôme de psychomotricien* **Sabine NOYER** Université Toulouse Faculté de Médecine - Institut de formation en psychomotricité 2018.

[YLC98] *GradientBased Learning Applied to Document Recognition* **Yann LeCun –Leon Bottou- Yoshua Bengio - Patrick Haner** New York University 1998

[YXC05] *AdaBoost Gabor Fisher Classifier for Face Recognition* **Shiguang Shan , Peng Yang , Xilin Chen, and Wen Gao** Institute of Computing Technology, Chinese Academy of Sciences, Beijing China 2005

Webgraphie:

[Web01] : http://visagetechnologies.com/mpeg-4-face-and-body-animation/

[Web02] : https://openclassrooms.com/en/courses/4470531-classez-et-segmentez-des-donnees-visuelles/5082166-quest-ce-quun-reseau-de-neurones-convolutif-ou-cnn

[Web03]: https://fr.wikipedia.org/wiki/M%C3%A9thode_de_Viola_et_Jones

[Web04]: https://neurohive.io/en/popular-networks/vgg16/

[Web05]: https://towardsdatascience.com/an-intuitive-guide-to-deep-network-architectures-65fdc477db41

[Web06]: https://pymotion.com/detection-objet-cascade-haar/

[Web07]:https://ch.mathworks.com/fr/solutions/deep-learning/convolutional-neural-network.html

[Web08]: http://www.linux-center.org/articles/9812/python.html

[Web09] : https://www.numpy.org/

[Web10]: https://medium.com/themlblog/how-to-do-facial-emotion-recognition-using-a-cnn-b7bbae79cd8f
[Web11]: https://zbigatron.com/has-deep-learning-superseded-traditional-computer-vision-techniques/