REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre : IA17/M2/2019

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : Intelligence artificielle

Deep Auto-Encodeur pour la Reconnaissance de Visage

Par :

BENATIA YACINE

Soutenu le 07 juillet 2019, devant le jury composé de :

BENDAHMANE Toufik	MAA	Président
AYAD Soheyb	MCB	Rapporteur
HOUADJLI Hadia	MAA	Examinateur

Remerciements

La réalisation de ce mémoire a été possible grâce au concours de plusieurs personnes à qui je voudrais témoigner toute ma gratitude.

Je voudrais tout d'abord adresser toute ma reconnaissance à mon encadreur Docteur ***AYAD*** *Soheyb, pour sa patience, sa disponibilité et ses judicieux conseils, qui ont contribué à alimenter ma réflexion.*

Je souhaiterais exprimer ma gratitude et reconnaissance au Professeur ***BELAHCENE*** *Mébarka, d'avoir suivi de près mon travail, ses précieux conseils et orientations m'ont été très bénéfiques. Je la remercie sincèrement de m'avoir permis d'évoluer dans le monde l'Intelligence Artificielle et des techniques de Deep Learning pour le traitement d'image appliquée à la biométrie. Je la remercie également pour sa patience et qualités scientifiques.*

Je tiens également à exprimer toute ma reconnaissance envers les membres du jury de ma soutenance de Master 2. Merci à Docteur ***BENDAHMANE*** *Toufik d'avoir accepté le rôle de Président de jury de ce travail. Je remercie également Docteur* ***HOUADJLI*** *Hadia d'avoir accepté d'être examinateur.*

Un grand merci au Professeur ***KAZAR*** *Okba qui m'a beaucoup aidé et pour ses conseils durant mes deux années de Master, Enfin, à tous ces intervenants, je présente mes remerciements, mon respect et ma gratitude.*

Mes plus profonds remerciements vont à mes parents pour leurs présences et leur accompagnement pendant tout mon parcours et dans les moments de doute. Tout au long de mon cursus, ils m'ont toujours soutenus et encouragés dans la poursuite de mes études. Ils ont su me donner toutes les chances pour réussir. Ils m'ont donné le goût de la connaissance. Qu'ils trouvent, dans la réalisation de ce travail, l'aboutissement de leurs efforts ainsi que l'expression de ma plus affectueuse gratitude. Je leur exprime ici toute ma gratitude de m'avoir toujours écoutée et même souvent relue avec la plus grande attention et m'apportent chaque jour tant et plus. Un grand merci à mon grand frère Mohamed Amine à qui je dois ma passion pour les sciences et l'informatique en particulier. Il reste un exemple et un modèle à suivre pour moi. Un grand aussi à merci mon frère Charaf Eddine sur lequel j'ai toujours pu compter et grâce à la solidarité j'ai pu poursuivre de longues études. Son courage et endurance pour réussir dans la vie m'ont beaucoup inspiré.

Je remercie sincèrement tous les enseignants qui ont contribué de loin ou de près à ma formation du primaire au supérieur.

Je terminerai en dédiant ce mémoire à tous mes amis aux caractères merveilleux, plein de joie et dont les sourires sont aussi agréables que communicatifs.

Dédicaces

À mes chers parents,
À mes frères,
À tous mes amis et camarades,
À tous les étudiants de la promotion 2018 / 2019,
À tous ceux qui, par un mot, m'ont donné la force de continuer….

Table des matières

Liste des tableaux

Liste des figures

Chapitre 1 Généralités sur Deep Learning et Auto-Encodeur pour le Visage

Chapitre 2 État de l'art sur le Deep Learning et les Auto-Encodeurs

Chapitre 3 Conception de la Reconnaissance de Visage basée sur les Auto-Encodeurs

Chapitre 4 Implémentation de l'Approche basée sur les Auto-Encodeurs

LISTE DES ABRÉVIATIONS ET SYMBOLES

AE	Auto-encodeur
AEC	Auto-encodeur convolutionnel
AUC	Area under the ROC Curve
ADAM	Adaptive Moments Estimation
BDD	Base de données
CNN	Convolutional Neural Network
CRC	Collaborative Representation-based Classification
DL	Deep Learning
DAE	Denoising Autoencoders - Deep Autoencoders
DBM	Deep Belief Machine
DBN	Deep Belief Network
DFD	Data Flow Diagram
EBGM	Elastic Bunch Graph Matching
FPR	False positive rate
GAN	Generative Adversarial Networks
HD-LBP	High-Dimensional Local Binary Patterns
IA	Intelligence Artificielle
LFW	Labeled Faces in the Wild
LLE	Locally Linear Embedding
LPP	La projection de préservation de la localisation
LE	Opérateur Laplaciennes
LBP	Local Binary Patterns
LGBP	Local Gabor Binary Pattern
ML	Machine learning
MNIST	Mixed National Institute of Standards and Technology
MSE	Mean Square Error
MAE	Mean Absolute Error
MDS	Méthodes de dimensionnalité
NPE	L'incorporation avec préservation de voisinage
PCA	Principal Component Analysis
PMC-MLP	Perceptron multicouches – multi layer perceptron
RV	Reconnaissance de visage
RmsProp	Root Mean Square Propogation
RBM	Restricted Boltzmann machine
ReLu	Rectified Linear Unit
SRV	Système de reconnaissance de visage
SDAE	Stacked Denoising Autoencoder
SDA	Stacked Denoising Autoencoder
SOM	Self-organizing map
SRC	Sparse representation based classification
TPR	True positive rate
VAE	Variational auto-encoder
VGG FACE	Visual Geometry Group Face
VAE	Variational AutoEncoder
VR	Verification Rate

Introduction générale

1.1 Contexte

La reconnaissance faciale est un des problèmes typiques en informatique. La vision et l'apprentissage automatique jouent un rôle important dans de nombreuses applications, telles que la vidéosurveillance, le contrôle d'accès, l'interface homme-machine et les divertissements mobiles [1]. De manière générale, un système de reconnaissance de visage conventionnel comprend quatre modules : détection de visage, alignement de visage, représentation du visage et classification d'identité. Dans ce pipeline, le composant clé pour la reconnaissance de visage est le troisième module, c'est-à-dire l'extraction de la représentation d'une face d'entrée et la réduction de données sur lequel notre travail se concentre principalement. Au cours des dernières décennies, la représentation faciale est principalement basée sur des descripteurs locaux [1,2] fabriqués à la main et sur des modèles de représentation basés sur l'apprentissage superficiel. Avec le développement de la technologie d'apprentissage en profondeur, elle devient une approche plus puissante pour l'apprentissage de la représentation faciale, en particulier dans les scénarios utilisant du monde réel. Les principaux défis de la représentation faciale résident dans la petite différence d'apparence interpersonnelle causée par des configurations faciales similaires, ainsi que dans les importantes variations d'apparence intra-personne dues à de grandes variations intrinsèques et à divers facteurs d'imagerie extrinsèques, tels que la pose de la tête, l'expression, le vieillissement et l'illumination. Par rapport à la routine artisanale précédente, la représentation des visages profonds est apprise dans un style piloté par les données, ce qui peut garantir de meilleures performances.

1.2 Problématique

Traditionnellement, la recherche sur la reconnaissance faciale s'est concentrée sur des variantes telles que la pose, l'expression, l'illumination, la qualité, le vieillissement, le déguisement et la chirurgie plastique et la plupart du temps, comme nous l'avons déjà cité ces défis sont abordés individuellement. Cependant, les applications du monde réel nécessitent que les algorithmes gèrent des scénarios sans contrainte dans lesquels les images et les vidéos comportent plusieurs variantes. Par exemple, un flou, une pose et un éclairage peuvent être présents dans une seule vidéo, ou une occlusion et une pose de profil peuvent souvent être observées sur des photographies de personnes. De nombreux efforts de recherche sont actuellement en cours pour développer des algorithmes efficaces pour des environnements non soumis à des contraintes et des projets de recherche à grande échelle

donnent une impulsion supplémentaire à ces efforts. En particulier, une représentation du visage peut être obtenue avec une grande base de données d'entraînement étiquetée (ou non) composée d'images de visage et ces représentations faciales apprises, conjointement avec un classificateur non supervisé, sont utilisées pour vérifier l'identité des sujets. En général, ces deux étapes d'apprentissage sont disjointes, c'est-à-dire que le codage des caractéristiques est indépendant de la sortie de la classification.

Les données du monde réel, telles que les images de visages et de chiffres, ont généralement une grande dimension, ce qui conduit ***à la malédiction bien connue de la dimensionnalité*** dans la reconnaissance statistique de modèles. En règle générale, il n'est pas facile de saisir de telles structures en utilisant un simple modèle paramétrique, tel que l'Analyse en Composantes Principales (ACP ou PCA). L'exploitation des relations de données s'est révélée être un moyen prometteur pour découvrir la structure sous-jacente. Par exemple, ISOMAP (méthode de réduction de dimensionnalité non linéaire) apprend une variété de faible dimension en conservant la distance géodésique entre les données paires par paires dans l'espace d'origine. Dans l'incorporation localement linéaire (LLE), chaque point de données est une combinaison linéaire de ses voisins. La relation linéaire est préservée dans l'espace de faible dimension projeté. Les cartes propres Laplaciennes (LE) permettent de créer une variété de faible dimension en minimisant la distance par paire dans l'espace projeté, pondérée par la distance correspondante dans l'espace d'origine. Ils ont une faiblesse commune de souffrir du problème hors échantillon. L'incorporation avec préservation de voisinage (NPE) et la projection de préservation de la localisation (LPP) sont des approximations linéaires de LLE et LE pour traiter le problème de manque d'échantillon, respectivement. Bien que ces méthodes exploitent la relation de données locale pour en apprendre beaucoup, la relation est fixée et définie dans l'espace de grande dimension d'origine. Une telle relation peut ne pas être valide sur la variété, par ex. le plus proche voisin géodésique sur un collecteur peut ne pas être le plus proche voisin euclidien de l'espace d'origine [3].

1.3 Motivation

Toutes ces limites des MDS (Méthodes de dimensionnalité) traditionnelles peuvent être résolues par les méthodes d'apprentissage automatique qui peuvent être la solution.

Nous affirmons ainsi que, pour être plus efficace en matière de reconnaissance / identification de visage, le cadre d'apprentissage en profondeur basé sur un auto-encodeur s'avère être un bon candidat et peut être de nature spécifique à la classification. En d'autres termes, les performances des algorithmes de reconnaissance basés sur un auto-encodeur

peuvent être une solution si les représentations apprises intègrent les informations de classe. Toutes ces raisons nous motivent à pousser notre intérêt dans cette piste qui est la reconnaissance de visage basé sur les auto-encodeurs.

En littérature [1,3,4], plusieurs modèles mathématiques ont été proposés pour traiter des vérifications récentes du visage fondées sur l'apprentissage en profondeur. Les algorithmes d'apprentissage en profondeur existants peuvent être divisés en trois groupes : réseau de neurones à convolution (CNN), auto-encodeur / machine Boltzmann restreinte (SDAE, DBM, DBN) et approches hybrides (combinant CNN et RBM). Parmi ceux-ci, les CNN sont largement utilisés et plusieurs architectures sont proposées dans la littérature, y compris celles qui sont très «profondes». D'autre part, les approches «auto-encodeur» pour la représentation d'objets se concentrent généralement sur l'apprentissage de fonctionnalités de manière non supervisée et utilisent des fonctionnalités apprises pour la reconnaissance / identification. L'algorithme auto-encodeur [4] appartient à une famille particulière de méthodes de réduction de dimensionnalité mises en œuvre à l'aide de réseaux de neurones artificiels. Il vise à apprendre une représentation comprimée pour une entrée en minimisant son erreur de reconstruction.

1.4 Structure du mémoire

Nous avons choisi d'articuler notre étude autour de **quatre chapitres** principaux.

Le **premier chapitre** est consacré à la présentation générale du Deep Learning et les auto-encodeurs.

Dans le **second chapitre** une analyse détaillée des différentes techniques développées au cours de ces dernières années dans les domaines de la reconnaissance et des auto-encodeurs en générale.

Dans le **troisième chapitre** nous étudions la conception des différentes méthodes d'optimisation et d'extraction de caractéristique ainsi que de réduction et de classification utiles pour notre auto-encodeur et système de reconnaissance de visage.

Le **quatrième chapitre** est consacré à l'implémentation de nos modèles d'application de l'auto-encodeur dans le débruitage, la reconstruction et particulièrement la réduction de données pour la reconnaissance de visage.

Enfin, la **conclusion générale** résumera nos contributions et donnera quelques perspectives sur les travaux futurs.

Introduction

La reconnaissance des formes est devenue un élément important en raison du besoin toujours plus exigeant d'apprentissage automatique et d'intelligence artificielle dans les problèmes pratiques. La reconnaissance du visage est l'un de ces problèmes confirmés par le chiffre important de travaux publiés par différents auteurs [5]. La reconnaissance dans ce domaine couvre plusieurs applications telles que la vérification de l'identité, la parenté, l'âge les jumeaux, la chirurgie esthétique du visage ainsi que la détection, le tracking, la localisation de personnes, l'analyse des émotions et la détection de maladies par l'analyse de visage qui des thématiques peu abordées. Le problème majeur dans ces investigations reste la manipulation de données gigantesques. Certaines méthodologies comme le Deep Learning (DL) et les Réseaux de Neurones Convolutionels (CNN) et ses extensions ainsi que les Auto-Encodeurs (AE) pour résoudre le problème de la dimensionnalité et la classification ; s'apprêtent bien à ce genre d'analyse de données et se rapprochent au mieux de l'intelligence humaine.

Le Deep Learning est une structure hiérarchique de réseau qui simule le cerveau humain permettant d'extraire les données d'entrée internes et externes. C'est un apprentissage approfondi basé sur des algorithmes utilisant les réseaux multicouches tels que les réseaux de neurones profonds, réseaux de neurones profonds convolutifs (CNN), réseaux profonds de croyance (DBN) réseaux de neurones récurrents (RNN) et les différents types d'auto encodeurs. Ces algorithmes d'intelligence artificielle permettent aux ordinateurs et aux machines de modéliser notre monde.

Les auto-encodeurs sont des réseaux de neurones artificiels utilisés pour apprendre un codage efficace où la couche d'entrée a le même nombre que la couche de sortie où la couche cachée a une taille plus petite [3,4,6]. Dans l'auto encodeur, la couche cachée donne une meilleure représentation de l'entrée que l'entrée brute d'origine, et la couche cachée est toujours la compression des données d'entrée qui représentent les caractéristiques importantes de l'entrée.

Ainsi, l'objectif de notre travail est l'étude et la conception d'un DeepAuto-Encodeur (DAE) dans un système de reconnaissance en apprentissage profond pour la reconnaissance de visage.

1.1 Qu'est-ce que l'apprentissage ?

L'apprentissage automatique (ou machine learning) est un processus qui crée un modèle à partir d'un jeu de données d'entraînement, dans le but, par exemple, de classifier, mesurer ou prendre des décisions sur de nouvelles données. On distingue deux cadres d'apprentissage : i) La démarche d'apprentissage supervisé, qui consiste à analyser un jeu de données empiriques, appelée base de données d'entraînement. Ces données sont étiquetées ou labellisées, c'est à dire associées à une description ou information, afin de pouvoir caractériser un nouveau jeu de données inconnu. L'algorithme apprend ainsi à partir de milliers ou de millions d'exemples étiquetés : il cherche la relation qui permet de relier les données aux labels. ii) L'apprentissage non supervisé, où l'algorithme doit opérer en l'absence d'annotations. La démarche supervisée est illustrée sur la **figure 1.1**.

Dans une grande image, nous considérons disposer d'un certain nombre de vignettes, auxquelles sont associées un label ou information. Une fois cette étape d'apprentissage terminée, dès lors que l'on présente une nouvelle vignette image à l'ordinateur, l'application de la fonction d'inférence donnera l'espèce qu'elle pense lui être associée.

Figure 1. 1 Exemple schématique de la fonction d'apprentissage [7]

Dans la littérature, il existe une grande diversité d'approches en machine learning. Le choix d'une méthode dépend du problème adressé, notamment du type de modèle d'inférence à déterminer. On distingue notamment les approches selon la complexité du modèle : spécifique ou généraliste. Un autre critère important est la taille de la base d'apprentissage. Trop petite, elle ne permet pas par exemple d'envisager les méthodes de deep learning. Par la suite, seules les méthodes les plus intéressantes pour notre problème seront exposées, en explicitant leurs avantages et inconvénients, ainsi que les difficultés rencontrées en pratique.

1.2 Apprentissage automatique

L'apprentissage automatique est un sous-domaine de l'informatique issu de l'intelligence artificielle. Cette discipline scientifique est principalement divisée en deux volets, soit l'apprentissage supervisé et non-supervisé. Tout en étant très fortement liée aux statistiques

et à l'optimisation mathématique, elle explore la construction de modèles et l'étude d'algorithmes pouvant apprendre à partir de données. Ces données sont spécifiques à une tâche précise que le modèle essaie de résoudre en apprenant à partir d'exemples. Le spectre de tâches que l'apprentissage automatique aide à résoudre est extrêmement large : la vision par ordinateur, le traitement automatique de la langue, les engins de recherche, les diagnostics médicaux, la détection de fraude, le contrôle de robots, l'analyse des marchés financiers, etc

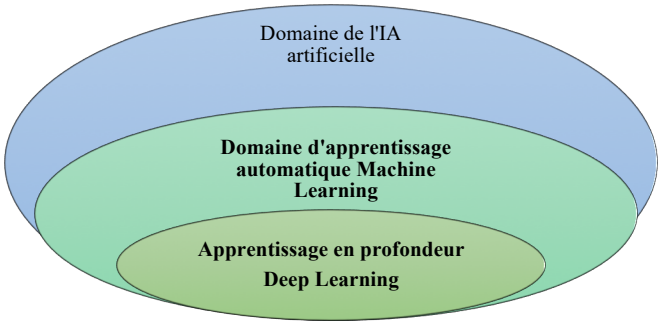


Figure 1. 2 La relation entre l'IA et l'apprentissage profond [8]

1.2.1 Apprentissage supervisé

Dans le domaine de l'apprentissage automatique, l'apprentissage supervisé est la tâche d'inférer une fonction à partir de données étiquetées. Ces données d'entraînement constituent un ensemble d'exemples composés d'un objet d'entrée, généralement un vecteur, et d'une valeur de sortie désirée, son étiquette. Un algorithme d'apprentissage supervisé analyse les données d'entraînement et produit une fonction, laquelle peut être utilisée pour générer une sortie sur de nouveaux exemples. Un scénario optimal permettra à l'algorithme de bien généraliser ce qui a été appris lors de l'entraînement, c'est-à-dire, de déterminer correctement les étiquettes pour des instances qui n'ont jamais été observées auparavant. La tâche la plus commune en apprentissage supervisé est la classification. Un exemple classique est la classification de l'ensemble de données MNIST [9] qui se compose d'images représentant des chiffres manuscrits. La tâche de classification à effectuer dans ce cas est d'assigner une classe (le chiffre que l'image représente) à chacun des éléments de l'ensemble.

1.2.2 Apprentissage non-supervisé

L'apprentissage non-supervisé, quant à lui, tente de trouver une structure cachée dans des données non étiquetées. Ce type d'apprentissage englobe de nombreuses techniques [7] visant soit à résumer, à expliquer les principales caractéristiques ou à déterminer la distribution d'un ensemble de données. Différentes méthodes incluses dans l'apprentissage non-supervisé comprennent le partitionnement de données, les modèles de Markov cachés, la réduction de dimensionnalité, l'estimation de distribution et bien d'autres.

Tableau 1. 1 Différences entre l'Apprentissage Supervisé et non Supervisé

Apprentissage Supervisé	**Apprentissage non Supervisé**
Données d'entrée sont étiquetées	Données d'entrée sont ***non étiquetées***
Utilise le jeu de données d'apprentissage	Utilise tout le jeu de données en entrée
Utilisé pour la prédiction	Utilisé pour l'analyse
Classification et régression	Regroupement, Estimation de la densité et ***Réduction de la dimensionnalité***

1.3 La reconnaissance de visage

Rien n'est plus naturel que d'utiliser le visage pour identifier une personne. Les images faciales sont probablement la caractéristique biométrique la plus communément employée par l'homme pour effectuer une identification personnelle. L'utilisation d'une caméra permet de capter la forme du visage d'un individu et d'en dégager certaines particularités. Selon le système utilisé, l'individu doit être positionné devant l'appareil ou peut être en mouvement à une certaine distance. Les données biométriques qui sont obtenues sont par la suite comparées au fichier référence. Au début des années 1970, la reconnaissance par le visage était principalement basée sur des attributs faciaux mesurables comme l'écartement des yeux, des sourcils, des lèvres, la position du menton, la forme, etc. Depuis les années 1990, les différentes technologies utilisées exploitent toutes les découvertes effectuées dans le domaine du traitement d'image, et plus récemment encore les réseaux de neurones.

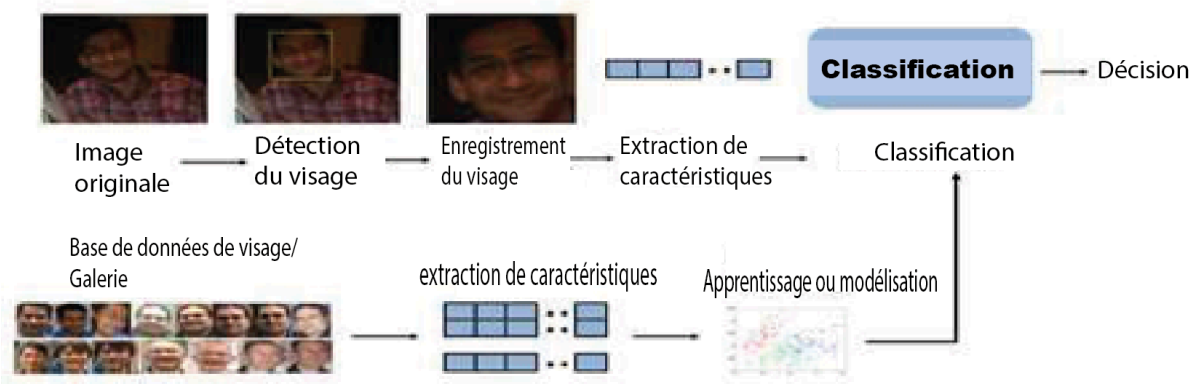


Figure 1. 3 Exemple de schéma synoptique d'un système de reconnaissance faciale [10]

Dans toutes les méthodes de reconnaissance faciale, le point le plus délicat concerne la définition et l'extraction des caractéristiques faciales les plus pertinentes [2], à savoir les caractéristiques qui représentent le mieux les informations portées par un visage.

Il existe plusieurs techniques de reconnaissance par analyse du visage (voir **figure 1.3**). Mais pour la plupart il est d'intérêt que ces techniques se basent sur des éléments du visage qui

sont le moins susceptibles aux changements : les grands traits supérieurs des orbites, les secteurs entourant les pommettes, les côtés de la bouche et d'autres caractéristiques similaires de façon à ignorer les changements comme la coupe de cheveux. Toutes ces techniques conduisent à des fonctionnements en identification sur des bases de milliers voir de centaines de milliers de personnes.

Au cours des dernières années, les techniques d'apprentissage en profondeur ont permis des progrès rapides dans la reconnaissance de visage, dépassant même les performances humaines. Dans ce travail, nous allons examiner ces progrès pour comprendre comment ils ont été réalisés grâce à un apprentissage en profondeur, ce que nous pouvons en apprendre et ce que nous pouvons en faire.

Dans ce qui suit, nous présentons brièvement quelques définitions sur le deep learning et les types d'apprentissage.

1.4 Qu'est-ce que le Deep Learning ?

Le terme Deep Learning (en français : apprentissage profond) est très en vogue ces derniers temps. C'est bien simple, lorsque l'on parle d'intelligence artificielle, on parle presque systématiquement de Deep Learning. A tel point que dans l'esprit de beaucoup, ces deux termes sont synonymes. C'est pourtant inexact.

1.4.1 L'évolution de l'intelligence artificielle

L'intelligence artificielle a connu une longue histoire puisqu'elle a été conceptualisée dès l'antiquité. Bien sûr ce n'est que suite à la création des premiers ordinateurs qu'elle a pu être mise en œuvre de façon concrète. Différents courants se sont alors développés. L'un de ces courants consistait à s'inspirer du cerveau humain afin de tenter de créer des neurones artificiels. Un neurone artificiel n'est rien d'autre qu'une opération mathématique relativement simple. La complexité repose avant tout dans l'interconnexion de plusieurs neurones. Le premier réseau de neurones artificiels a été mis au point en 1951 par Marvin Minsky et Dean Edmonds de l'Université de Harvard. Peu de temps après, en 1956, Frank Rosenblatt a mis au point le Perceptron qui a suscité un grand engouement. Les scientifiques misaient alors énormément sur les réseaux de neurones mais les résultats ont fini par décevoir.

1.4.2 L'avènement du Deep Learning

Ce n'est que récemment, grâce à l'avancée des performances de calcul des ordinateurs que s'est développé le concept de Deep Learning. Il s'agit de réseaux de neurones disposant de nombreuses couches cachées (c'est à dire de nombreuses couches de neurones situées entre les couches d'entrées, acceptant des données à traiter, et les couches de sortie, destinées à délivrer

le résultat du calcul). A la grande surprise des spécialistes, l'ajout de ces couches de neurones a eu un impact extrêmement bénéfique sur la qualité des résultats obtenus. C'est ce qui permet à l'intelligence artificielle de revenir sur le devant de la scène depuis quelques années. La plupart des acteurs du domaine **(voir figure 1.4 et 1.5)** ne jurent aujourd'hui plus que par le Deep Learning. Google, Facebook, Apple et Microsoft mettent d'ailleurs tous leur propre librairie de Deep Learning à la disposition des développeurs.

Tableau 1. 2 Historique sur le Deep Learning

`Du perceptron a l'apprentissage profond `	
1957	(Rosenblatt) Perceptron
1960	(Widrow, Hoff) ADALINE
1969	(Minsky, Papert) Probleme XOR
1986	(Rumelhart *et.* al) MLP et backpropagation
1992	(Vapnik *et.* al) SVM
1998	(LeCun *et.* al) LeNet
2010	(Hinton *et.* al) Deep Neural Networks
2012	(Krizhevsky, Hinton et al) AlexNet, ILSVRC'2012, GPU – 8 couches
2014	GoogleNet – 22 couches
2015	Inception (Google) – Deep Dream
2016	ResidualNet (Microsoft/Facebook) – 152 couches

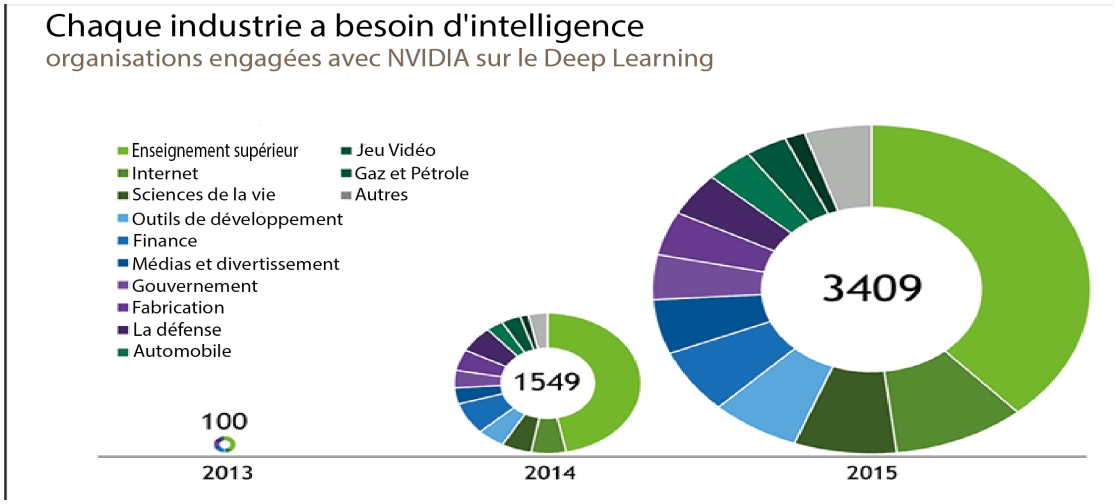


Figure 1. 4 Performances dans l'industrie et d'autres domaines [11]

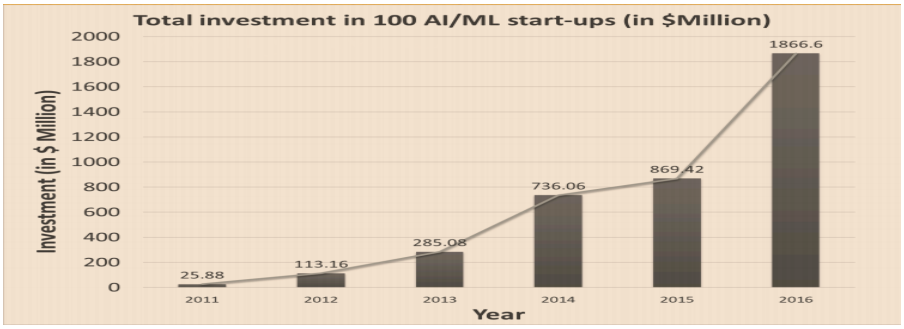


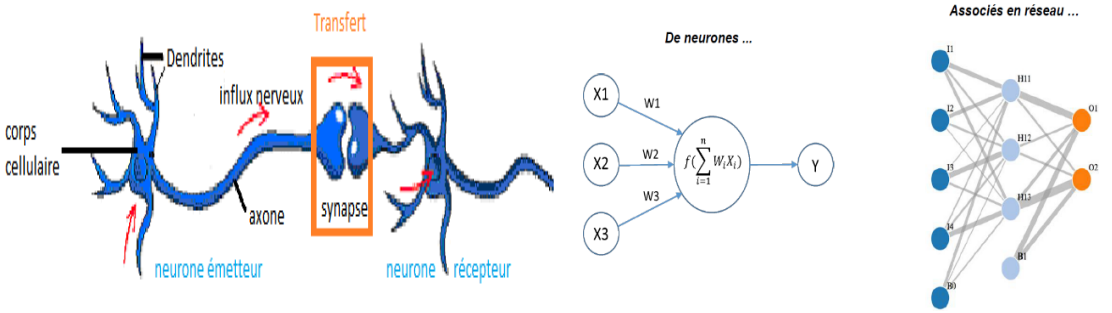
Figure 1. 5 Investissement dans le DL au cours des dernières années [8]

Le nombre de types de méthodes sur l'apprentissage profond étant très grand, notre mémoire a fait le choix d'aborder d'abord les réseaux de neurones sous leur forme traditionnelle (perceptron) avant d'envisager l'adaptation de méthodes de type Deep Learning sur nos données images et d'expliciter les difficultés rencontrées.

1.5 Rappels sur le perceptron

1.5.1 Les Réseaux de Neurones

Leur architecture s'inspire de la façon souple dont un cerveau effectue des tâches à l'aide de groupes de neurones biologiques reliés par des axones. Chaque unité neuronale est connectée à beaucoup d'autres, et les liens peuvent être contraignants ou inhibiteurs sur l'état d'activation des unités neuronales connectées (**figure 1.6 a**).



a) RN biologique **b)** RN artificiel

Figure 1. 6 Définition d'un réseau de neurones [11]

Chaque unité neuronale individuelle peut avoir une fonction de sommation qui combine les valeurs de toutes ses entrées (figure 1.6 b). Il peut y avoir une fonction de seuil ou de limitation sur chaque connexion et sur l'unité elle-même de sorte qu'elle doit la surpasser avant qu'elle puisse se propager à d'autres neurones.

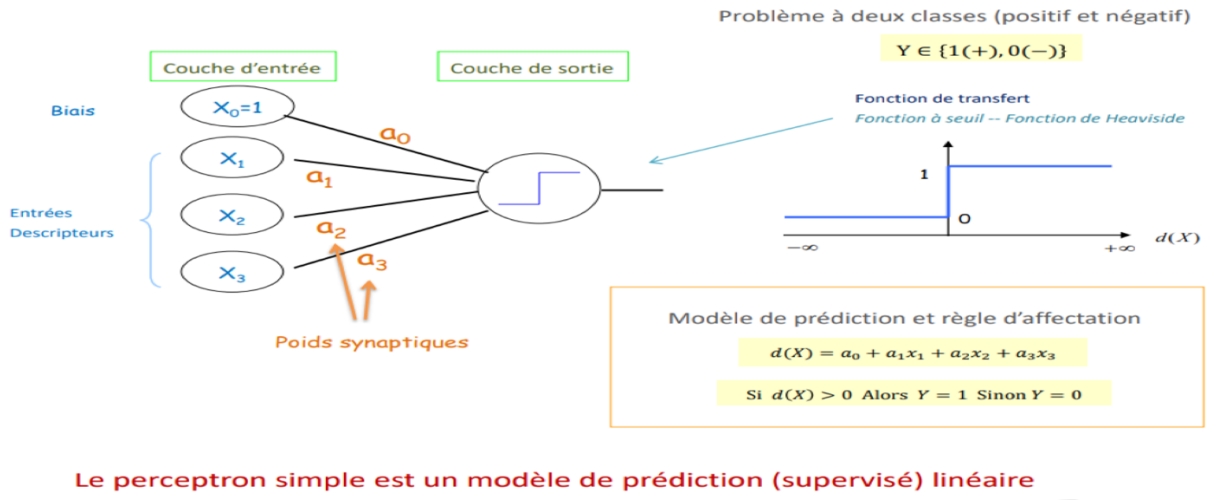


Figure 1. 7 Principe d'un réseau de neurone perceptron [12]

Ces systèmes sont entraînés sur des ensembles de données d'apprentissage, plutôt que programmés explicitement. Ils sont particulièrement intéressants lorsque la fonction recherchée entre entrées et sorties est difficile à exprimer par les méthodes informatiques traditionnelles. Les perceptrons ne sont pas capables de résoudre des taches complexes (voir figure 1.8)

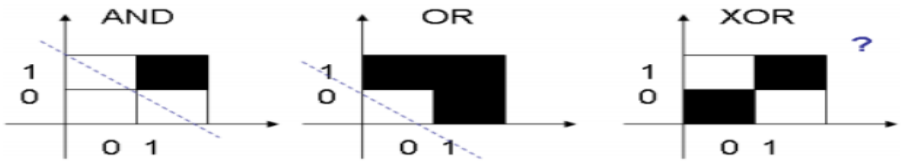


Figure 1. 8 Illustration de la limite du perceptron [12]

Pour résoudre ce problème il faut associer plusieurs perceptrons : **perceptron multi-couches** ou **réseaux de neurones**

1.5.2 Réseau de neurones artificiel ou perceptron

Les réseaux de neurones artificiels sont une famille d'algorithmes d'apprentissage statistique légèrement inspirés par les réseaux de neurones biologiques et sont utilisés pour estimer approximer des fonctions, généralement inconnues, pouvant dépendre d'un grand nombre d'entrées. Les réseaux de neurones artificiels sont généralement présentés comme des systèmes de « neurones » interconnectés pouvant calculer des valeurs à partir des différentes entrées de ceux-ci.

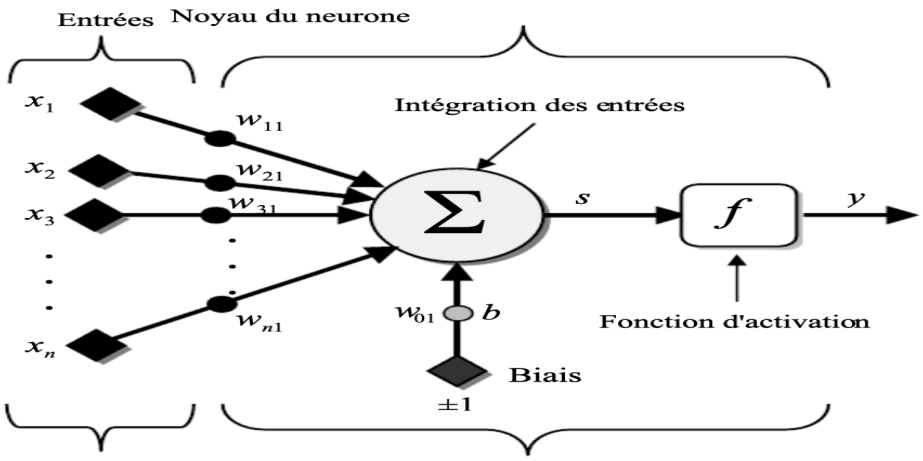


Figure 1. 9 Illustration d'un perceptron [14]

Le biais ***b*** (Figure1.9) est utilisé pour retarder le déclenchement de la fonction d'activation. D'autre part, biais est comme l'interception ajoutée dans une équation linéaire. C'est un paramètre supplémentaire du réseau de neurones utilisé pour ajuster la sortie ainsi que la somme pondérée des entrées du neurone. Par conséquent, biais est une constante qui aide le modèle à s'ajuster au mieux aux données.

Le traitement effectué par un neurone est ainsi noté :

Sortie = somme (poids * entrées) +biais

1.5.3 Types de fonction d'activation

Tous les neurones d'une couche ont la même fonction d'activation, mais cette fonction peut différer d'une couche à l'autre. Ce principe d'empilement peut être facilement généralisé à N couches. Il existe d'autres fonctions d'activation comme le montre la **figure 1.10**.

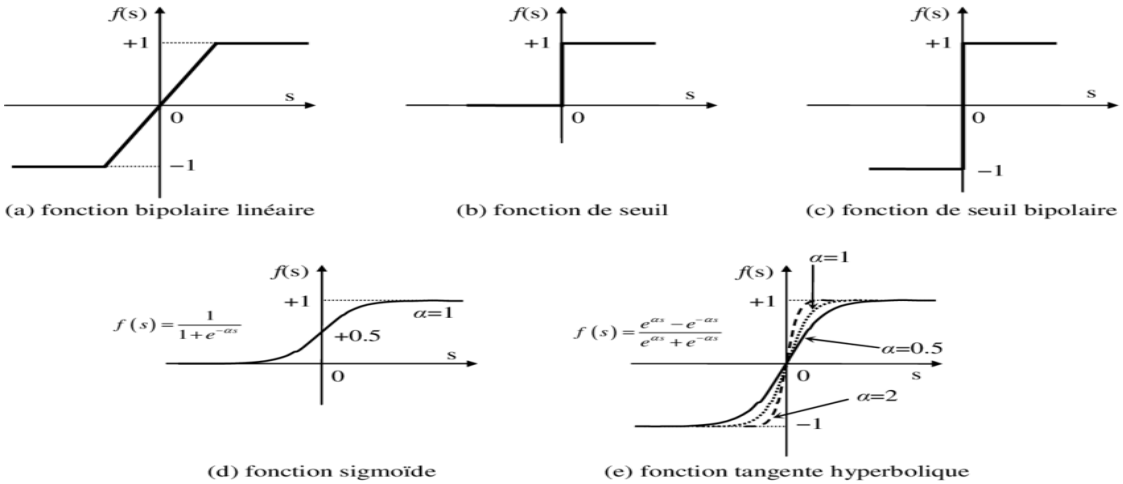


Figure 1. 10 Différents types de fonctions d'activation [13]

Le perceptron présente une inadéquation face aux tâches complexes :

- Restrictions à des calculs linéaires
- Fonction de décision discontinue

1.6 Le perceptron multi-couches (PMC ou MLP)

De quoi est composé un réseau de neurones ? La figure 1.11 suivante nous montre l'architecture d'un réseau de neurones.

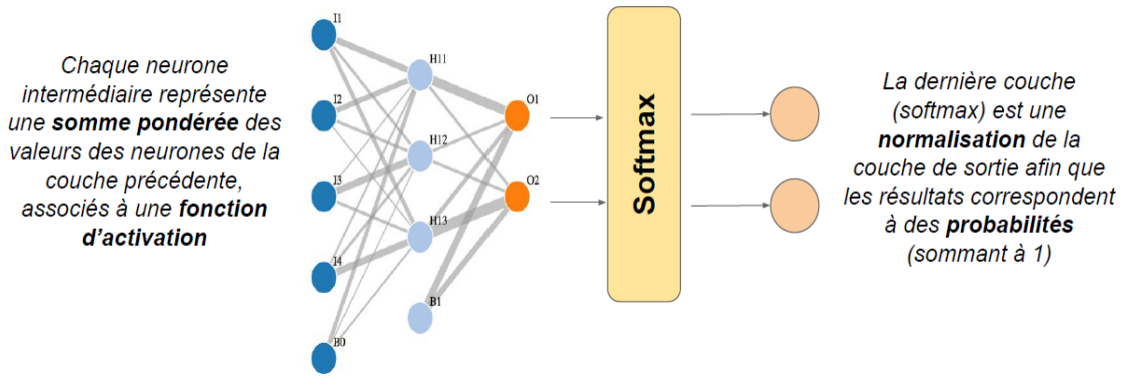


Figure 1. 11 Illustration du perceptron multi-couches (softmax) [11]

Comment fonctionne un réseau de neurone ?

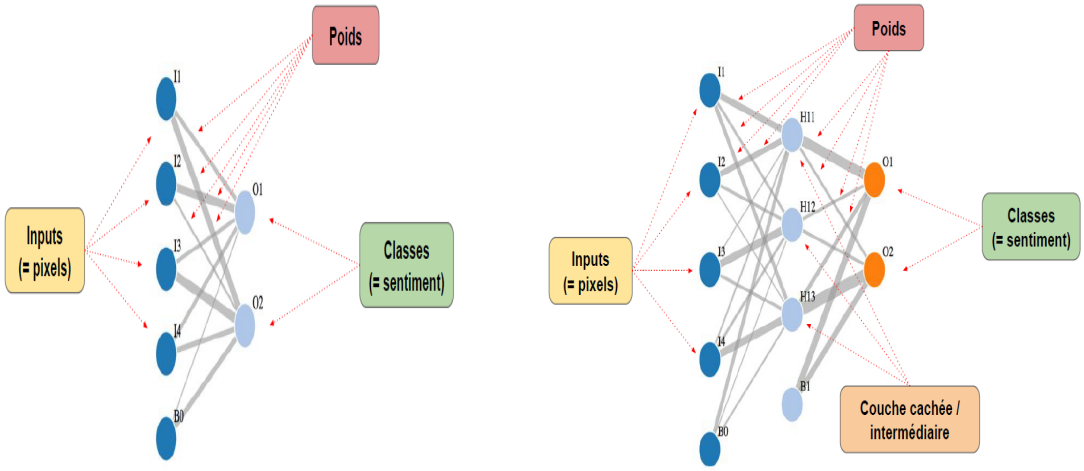


Figure 1. 12 Principe de fonctionnement d'un RN (MLP) [11]

A partir d'une architecture de perceptron choisie, il nous faut alors entraîner celui-ci à l'aide des données d'entraînement. Pour cela, on utilise la technique de rétropropagation (ou backpropagation). C'est un algorithme d'optimisation qui permet d'ajuster les paramètres de notre réseau multicouches, pour mettre en correspondance les entrées et les sorties référencées dans la base d'apprentissage. Cette optimisation se fait avec une descente de gradient, par minimisation d'une fonction d'erreur ou fonction de coût. Le choix du nombre de couches et du nombre de neurones est primordial dans un perceptron. En ajoutant des neurones ou des couches on améliore les capacités du réseau à trouver des fonctions complexes, et donc la finesse de l'approximation. Cependant l'apprentissage devient plus long (particulièrement en

augmentant le nombre de couches) et le risque de ***sur apprentissage*** augmente : le réseau devient ***surentraîné*** pour s'adapter aux données de la base d'apprentissage, mais se révèle ***mauvais*** en présence de ***nouvelles données*** et donc ***ne permet*** aucune ***généralisation*** utile.

Pour faire face à toutes ces contraintes le Deep Learning est la solution.

Le Deep Learning est aujourd'hui l'une des approches les plus prometteuses en apprentissage automatique. Cette approche constitue une forme de prolongement des réseaux de neurones traditionnels, permis par l'explosion des moyens de calculs et le développement des bases de données et d'Internet. Initialement dédiées à la reconnaissance d'images, ces méthodes sont aujourd'hui utilisées dans de nombreux domaines.

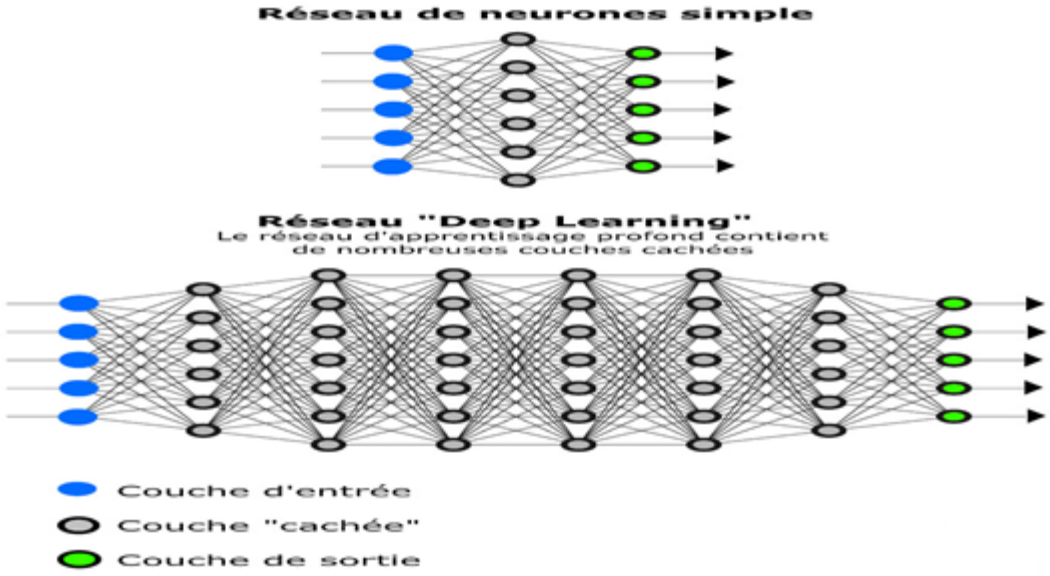


Figure 1. 13 Différence entre un réseau de neurones et le Deep Learning [15]

1.7 Deep Learning (DL)

1.7.1 Définition du Deep Learning

Le Deep Learning est un prolongement des réseaux de neurones. Les modèles de Deep Learning sont bâtis sur le même modèle que les perceptrons multicouches précédemment décrits. Les différentes couches intermédiaires sont plus nombreuses. On peut donc voir le Deep Learning comme une extension des réseaux de neurones classiques avec un très grand nombre de couches ; cependant on considère aussi souvent que ce n'est pas leur seule définition. Le réseau peut en effet avoir une architecture subdivisée en sous parties, chacune de ces sous parties traitant un sous problème plus simple, qui fournit son résultat à la couche suivante, et ainsi de suite. Cette idée n'est pas récente ; elle date des années 80, avec des travaux pionniers. Cependant, elle n'a pu être mise en place de manière très effective qu'avec l'amélioration des puissances de calcul, et également l'élaboration de bases de données d'entraînement

considérables. En 2007, le Stanford Vision Lab a ainsi créé ImageNet, une base de plusieurs millions d'images étiquetées. En 2012, le Deep Learning revient à l'honneur avec le succès de ses performances au ImageNet Large Scale Visual Recognition Challenge (ILSVRC), concours annuel de reconnaissance d'image fondé par l'université de Stanford. Depuis, les méthodes de Deep Learning se déploient pour toutes les applications. En pratique, il existe différentes architectures d'algorithmes en apprentissage profond : — Le modèle Deep Belief Networks (DBN) est une architecture d'apprentissage approfondie largement étudiée et déployée, l'une des premières développées historiquement. Il combine apprentissage non supervisé et supervisé. C'est un réseau neuronal profond, composé de multiples couches de variables latentes ("unités cachées"), avec des connexions entre les couches mais pas entre les unités au sein chaque couche. Les couches neuronales ont entraînées une par une de manière non supervisée par des machines de Boltzmann restreintes (Restricted Boltzmann Machines (RBMs)). Ces RBMs sont composées d'une couche de neurones, qui reçoit l'entrée, ainsi que d'une couche de neurones cachée, avec des neurones d'une même couche qui sont indépendants entre eux. De ce fait, les DBN enchaînent phase supervisée et phase non supervisée. — Parmi les approches supervisées, les CNN ou réseaux de convolution sont des réseaux de neurones dédiés au traitement d'images, qui ont engendré des gains de performances inégalées dans le domaine de la reconnaissance. La structure profonde des CNN permet au modèle d'apprendre des couches de descripteurs de plus en plus abstraites, ce qui permet d'avoir des représentations qui peuvent clairement stimuler les performances des classifieurs ultérieurs. Les CNN peuvent être considérés comme un empilage multicouche de perceptrons, dont le but est de pré-traiter de petites quantités d'informations. Ils incluent des couches de convolutions. — Un **auto-encodeur** ou **AE** est un réseau neuronal symétrique utilisé pour apprendre une représentation (encodage) d'un ensemble de données, généralement dans le but de réduire la dimension de cet ensemble, de manière non supervisée. Le réseau est construit en minimisant l'erreur de reconstruction entre les données d'entrée sur la couche de codage et sa reconstruction en sortie de la couche de décodage. Dans le domaine de la télédétection, l'usage du Deep Learning est encore marginal, principalement parce qu'il n'existe pas de bases de données aussi conséquentes que celles d'ImageNet. Cependant, il commence à être investigué dans ce cadre, comme dans la plupart des autres domaines.

Les algorithmes de l'Intelligence Artificielle relevant de l'apprentissage profond (Deep Learning : DL) sont réputés inexplicables. Ce « mystère » pourrait avoir pour origine l'apparition de chaos mathématiques engendrés par les systèmes itératifs de l'apprentissage

profond. Le passage continu entre le monde déterministe et le monde probabiliste est fascinant et mystérieux, mais guère rassurant !

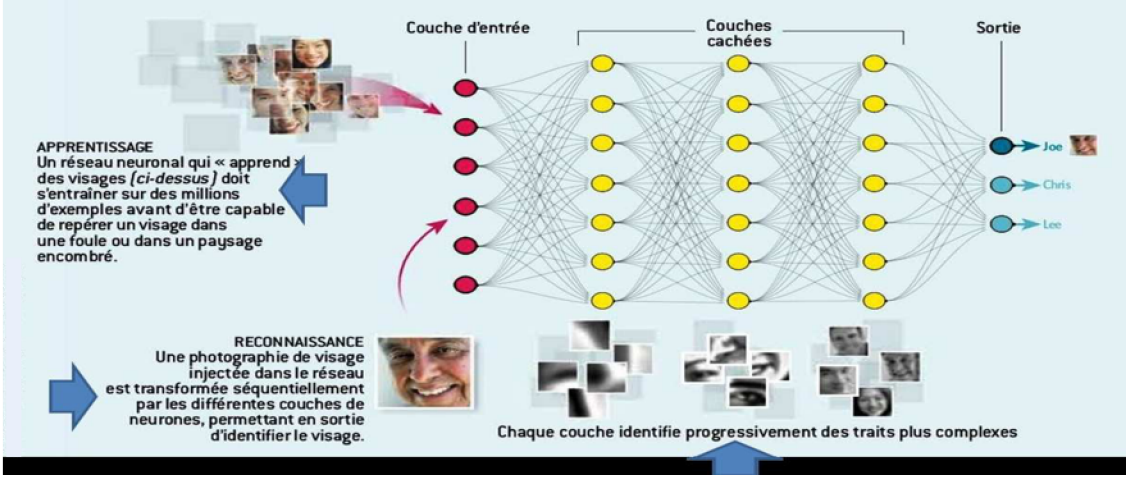


Figure 1. 14 Système de reconnaissance de visage basé sur le DL [16]

La constatation de Nozha Boujemaa [16] selon laquelle tous les algorithmes de l'intelligence Artificielle sont explicables mais, par contre ceux « relevant du Deep Learning (apprentissage profond) ne le sont pas » est troublante pour certaines applications.

Pourquoi certains, comme Paul Jorion, dans une émission de France-Culture, ont-ils l'impression que les machines prennent une certaine autonomie ? « Nous ne comprenons plus exactement ce qui se passe à l'intérieur des machines parce que nous avons créé des choses qui dépassent un peu les intuitions des programmeurs eux-mêmes, il y a des effets qu'on appelle « non linéaires » qui font qu'il y a une certaine "imprectabilité"… »

L'autonomie de la reconnaissance dans les systèmes biométriques est recherchée, ce qui justifie l'orientation des recherches actuelles vers l'apprentissage profond. ***Le terme Deep Learning, apprentissage profond en français, désigne en fait un réseau de neurones constitué de nombreuses couches. Il ne s'agit que d'une évolution des réseaux de neurones.***

1.7.2 Méthodologie de la reconnaissance de visage par le DL

La caractéristique déterminante des méthodes basées sur l'architecture profonde pour la reconnaissance faciale est en général l'utilisation d'un extracteur de caractéristiques CNN. Cette dernière est une fonction apprise obtenue en composant plusieurs opérateurs linéaires et non linéaires. Il y'a différentes méthodes pour la reconnaissance de visage comme le montre la **figure 1.15**. Nous insisterons sur les méthodes non supervisées et l'on peut citer : SOM, RBM, GAN, DBN et AE.

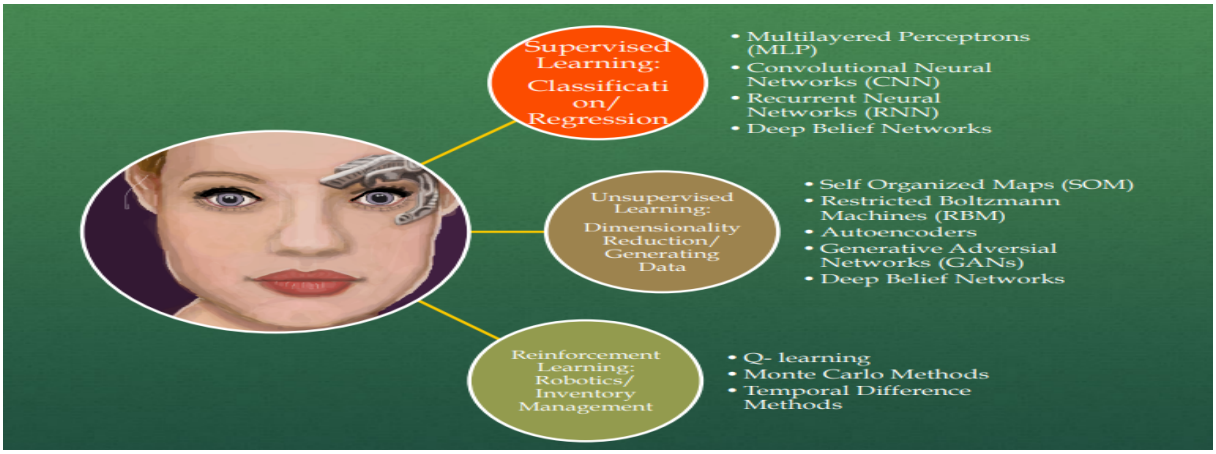


Figure 1. 15 Méthodologie de l'apprentissage automatique [8]

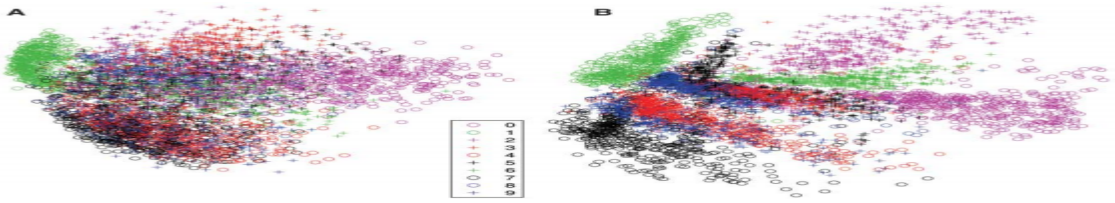
Certaines méthodes comme les auto-encodeurs peuvent jouer un rôle important dans le débruitage, la réduction de données ou la classification non supervisée. Un système représentatif de cette classe de méthodes est DeepFace, LeNet, AlexNet, VGG, GoogLeNet, ResNet…

1.8 Différence entre réseau de neurones et auto-encodeurs

Habituellement, dans un réseau de neurones, on essaie de prédire un vecteur cible y à partir de vecteurs d'entrée x. Dans un auto-encodeurs, on essaie de prédire x à partir de x.

Un réseau auto-encodeur est un type de réseau de neurones profonds dont l'objectif principal est d'extraire des caractéristiques qui aideront à reconstituer efficacement le signal d'entrée d'origine à partir de ces caractéristiques. Ainsi, un auto-encodeur possède un ensemble de couches cachées pour extraire successivement de telles caractéristiques à plusieurs niveaux et un autre ensemble de couches cachées qui suivent les couches précédentes et qui visent à reconstruire le signal d'entrée original. La formation d'un tel réseau a pour objectif que la sortie représente le plus fidèlement possible la contribution.

La figure suivante schématise un auto-encodeur simple, dont l'encodeur (encodeur) traite des images (inputs), afin de les représenter comme des points dans un espace à deux dimensions (représentation encodée), puis décode cette représentation (décodeur), afin de retrouver les données de départ (output).



a) Analyse en composantes principales **b)** Auto-encodeur

Figure 1. 16 Représentation des données latente [17]

1.8.1 L'Auto-Encodeur

L'auto-encodeur [18] peut être vu comme un cas particulier du perceptron multicouche où la valeur en entrée et la valeur désirée en sortie sont la même. Ceci implique qu'il y a autant de neurones en entrée qu'en sortie. En le configurant ainsi, le réseau de neurones doit donc apprendre à reconstruire ce qu'il reçoit en entrée. Dans le cas où les y sont une reconstruction des $\boldsymbol{x}$, ils sont dénotés $\boldsymbol{x}$ pour souligner ce fait.

Les auto-encodeurs (AE) sont des réseaux de neurones qui ont pour objectif de copier leurs entrées dans leurs sorties. Ils travaillent en comprimant l'entrée dans une représentation spatiale latente, puis en reconstruisant la sortie de cette représentation. Ce type de réseau est composé de trois parties : Encodeur, Milieu et Décodeur. Le milieu est une représentation compressée de l'entrée d'origine, créée par le codeur, qui peut être reconstruite par le décodeur.

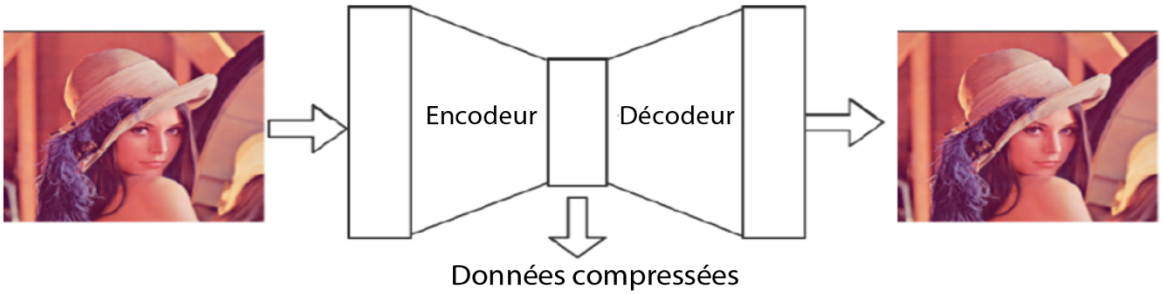


Figure 1. 17 Schéma de principe de l'auto-encodeur [19]

1. ***Encodeur :*** C'est la partie du réseau qui compresse l'entrée en une représentation en espace latent. Il peut être représenté par une fonction de codage $\boldsymbol{h = f(x)}$.
2. ***Milieu*** *:* C'est l'espace latent où l'on trouve la partie compressée des données d'entrée.
3. ***Décodeur :*** Cette partie a pour objectif de reconstruire l'entrée de la représentation de l'espace latent. Il peut être représenté par une fonction de décodage $\boldsymbol{y = g(h)}$.

1.8.2 Description d'un auto-encodeur

L'auto-encodeur dans son ensemble peut donc être décrit par la fonction $\boldsymbol{g(f(x)) = y}$ où l'on veut que la sortie $\boldsymbol{y}$ soit aussi proche que l'entrée $\boldsymbol{x}$ initiale. Un auto-encodeur est un réseau de neurones capable de réaliser l'apprentissage non supervisé.

Les réseaux de neurones sont généralement utilisés pour les problèmes d'apprentissage supervisé, en essayant de prédire un vecteur cible **y** à partir des vecteurs d'entrée **x**. Un réseau auto-encodeur tente cependant de prédire **x** à partir de **x**, sans avoir besoin d'étiquettes. Ici, le défi consiste à recréer l'information de l'entrée originale à partir de données compressées, bruyantes ou corrompues.

L'idée derrière auto-encodeur est de construire un réseau avec une couche cachée étroite entre Encodeur et Décodeur, qui sert de représentation compressée des données d'entrée.
La couche intermédiaire (« goulot d'étranglement ») est la représentation compressée des données d'entrée à partir de laquelle les données d'origine peuvent être reconstruites.
Le fait de transmettre des données d'entrée via le codeur crée une représentation compressée qui peut ensuite être transmise via un décodeur pour reconstruire l'entrée d'origine. La différence entre un auto-encodeur et un algorithme de compression technique réside dans le fait que les fonctions de compression et de décompression sont apprises à partir des données elles-mêmes, plutôt que d'être conçues et implémentées consciemment par programme (comme jpg ou zip).
L'encodeur et le décodeur sont tous deux des réseaux de neurones à feed-forward, et l'ensemble du réseau est formé en minimisant la différence entre l'entrée et la sortie, comme d'habitude. Cela signifie que les auto-encodeurs une fois formés sont assez spécifiques et auront du mal à généraliser à des ensembles de données autres que ceux sur lesquels ils ont été formés.

1.8.3 Pourquoi copier l'entrée sur la sortie ?

Si le seul but des auto-encodeurs était de copier l'entrée dans la sortie, ils seraient inutiles. En effet, nous espérons qu'en entraînant l'auto-encodeur à copier l'entrée dans la sortie, la représentation latente **h** aura des propriétés utiles.
Cela peut être réalisé en créant des contraintes sur la tâche de copie. Une façon d'obtenir des fonctionnalités utiles de l'auto-encodeur consiste à contraindre ***h*** à avoir des dimensions inférieures à ***x***, dans ce cas, l'auto-encodeur est appelé sous-complétion. En entraînant une représentation incomplète, nous forçons l'auto-encodeur à apprendre les principales caractéristiques des données d'apprentissage. Si la capacité de l'auto-encodeur est trop grande, il peut apprendre à effectuer la tâche de copie sans extraire d'informations utiles sur la distribution des données. Cela peut également se produire si la dimension de la représentation latente est identique à celle de l'entrée, et dans le cas de la sur-complétude, où la dimension de la représentation latente est supérieure à l'entrée. Dans ces cas, même un codeur linéaire et un décodeur linéaire peuvent apprendre à copier l'entrée vers la sortie sans rien apprendre d'utile sur la distribution des données. Idéalement, on pourrait former n'importe quelle architecture d'auto-encodeur avec succès, en choisissant la dimension du code et la capacité du codeur et du décodeur en fonction de la complexité de la distribution à modéliser.

1.8.4 A quoi servent les auto-encodeurs ?

De nos jours, le ***débruitage*** des données et la ***réduction de la dimensionnalité*** pour la visualisation des données sont considérés comme deux principales applications pratiques intéressantes des auto-encodeurs. Avec des contraintes de dimensionnalité et de parcimonie

appropriées, les auto-encodeurs peuvent apprendre des projections de données plus intéressantes que l'ACP (Analyse en Composantes Principales) ou d'autres techniques de base.

Les auto-encodeurs sont appris automatiquement à partir d'exemples de données. Cela signifie qu'il est facile de former des instances spécialisées de l'algorithme qui fonctionneront bien avec un type d'entrée spécifique et qu'il ne nécessite aucune nouvelle ingénierie, mais uniquement les données de formation appropriées.

Cependant, les auto-encodeurs feront un travail médiocre pour la compression d'image. Comme l'auto-encodeur est formé sur un ensemble de données donné, il obtiendra des résultats de compression raisonnables sur des données similaires à l'ensemble d'apprentissage utilisé, mais il s'agira de ***médiocres compresseurs d'images polyvalents***. Bien que les auto-encodeurs puissent être utilisés pour la compression, leurs performances sont évidemment pénalisantes, du fait de leur réduction de dimensionnalité inhérente. Cela signifie qu'ils ne sont pas aussi efficaces que les algorithmes techniques tels que JPEG, MP3, etc.

1.8.5 Domaines d'application des auto-encodeurs

Les auto-encodeurs sont utilisés pour :

- ***La réduction de la dimensionnalité non linéaire :*** l'AE encode l'entrée de la couche masquée dans une dimension inférieure à celle de l'entrée. La couche cachée est décodée ultérieurement en sortie. La couche de sortie à la même dimension que l'entrée. L'AE réduit la dimensionnalité des données linéaires et non linéaires. Il est donc plus puissant que la PCA.
- ***Les moteurs de recommandation*** *:* Ceux-ci utilisent des encodeurs profonds pour comprendre les préférences de l'utilisateur et recommander des films, des livres ou des éléments.
- ***L'extraction de caractéristiques*** : les auto-encodeurs essaient de minimiser l'erreur de reconstruction. Pour réduire l'erreur, il apprend certaines des caractéristiques importantes présentes dans l'entrée. Il reconstruit l'entrée à partir de l'état codé présent dans la couche cachée. Le codage génère un nouvel ensemble de fonctionnalités qui combine les fonctionnalités d'origine. Le codage dans les auto-codeurs permet d'identifier les caractéristiques latentes présentes dans les données d'entrée.
- ***Reconnaissance des images*** : Les auto-encodeurs empilés sont utilisés pour la reconnaissance des images. Nous pouvons utiliser plusieurs encodeurs superposés pour apprendre différentes caractéristiques d'une image.

Les auto-encodeurs sont formés pour conserver autant d'informations que possible lorsqu'une entrée est exécutée via l'encodeur, puis le décodeur, mais ils sont également formés pour que la

nouvelle représentation présente diverses propriétés intéressantes. Différents types d'auto-encodeurs visent à obtenir différents types de propriétés. Nous allons nous concentrer sur les types d'auto-encodeurs utilisés dans le domaine de l'imagerie.

1.8.6 Types d'auto-encodeurs

Il existe plusieurs types d'auto-encodeurs, quelques-uns sont décrits dans ce qui suit :

- Le *Denoising* est une utilisation plus appropriée pour un auto-encodeur. Le DAE (*Denoising Autoencoder*) est un enregistreur qui reçoit des données bruitées en entrée et qui est entraîné à prédire les données originales non bruitées en sortie.
- L'utilisation des réseaux de neurones pose un problème : ils peuvent simplement apprendre à dupliquer les données d'apprentissage au lieu d'apprendre véritablement à généraliser. Pour éviter ce problème, une technique appelée *Augmentation de données* est souvent utilisée pour compléter un ensemble d'apprentissage en créant des répliques d'observations avec un bruit aléatoire ou une "gigue" ajouté. Les auto-encodeurs peuvent être utilisés pour récupérer l'entrée originale non déformée.
- Il existe également une implémentation appelée *Sparse Autoencoder* qui peut être utilisée pour l'extraction de fonctionnalités. Celles-ci ont tendance à avoir plus d'unités cachées que d'unités d'entrée, ce qui permet une représentation fragmentée des données. Si plus d'une couche cachée est utilisée, un *auto-encodeur empilé* peut être construit, chaque couche consécutive du réseau pouvant être une projection dimensionnelle non linéaire pondérée de manière optimale des données d'apprentissage. Ceci peut être utilisé pour mettre en place un mécanisme de pré-formation pour l'apprentissage supervisé.

1.8.7 Autres types d'auto-encodeurs

Afin d'illustrer les différents types d'auto-codeur, un exemple de chacun d'entre eux est présenté. Nous pouvons également nous demander : les auto-encodeurs peuvent-ils être utilisés avec convolution au lieu de couches entièrement connectées ? La réponse est oui et le principe est le même, mais utiliser des images (vecteurs 3D).

Si nous traitons des images, nous pouvons généralement obtenir de meilleures performances en construisant un auto-encodeur convolutionnel utilisant des couches convolutives. La partie codeur du réseau sera une pyramide de convolution typique, où chaque couche de convolution est suivie par une couche de regroupement maximal afin de réduire la dimensionnalité :

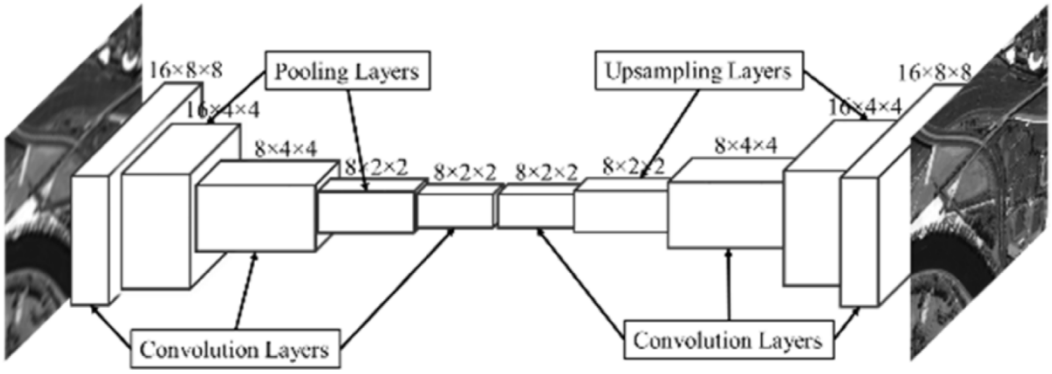


Figure 1. 18 Schéma de principe d'un auto-encodeur convolutionnel [20]

Ici, le décodeur contient un processus déconvolutionnel, qui ajoute de nouvelles données à partir de ce qu'il a appris. Ce processus peut être sujet à des artefacts visuels, de sorte qu'un redimensionnement à l'aide d'une interpolation voisine ou bilinéaire (sur-échantillonnage) peut constituer un meilleur choix que les couches déconvolutionnelles standard.

Une autre implémentation est l'Auto-encodeur variationnel (VAE), qui incorpore l'inférence Bayésienne, de sorte que la représentation comprimée produite est une distribution de probabilité. Dans sa forme la plus simple, l'auto-encodeur variationnel est un réseau à trois couches, c'est-à-dire un réseau neuronal avec une couche cachée. L'entrée et la sortie sont les mêmes et on apprend à reconstruire l'entrée, en utilisant par exemple l'optimiseur ***Adam*** et la fonction de perte d'erreur quadratique moyenne. Si une couche cachée ne suffit pas, on peut évidemment étendre l'auto-encodeur à plusieurs couches cachées (Multi layer Auto-encoder (MA)). N'importe laquelle des couches masquées peut être sélectionnée comme représentation d'entité, mais le réseau peut être symétrique et utiliser la couche la plus au centre.

1.8.8 Quelles sont les différences entre PCA et un auto-encodeur ?

Il est correct de penser aux caractéristiques extraites de la ou des couches cachées du milieu d'auto-encodeur en tant que PCA.

On peut dire que la PCA et les entités non linéaires extraites de l'auto-encodeur prennent en compte différents aspects des données d'apprentissage. Dans la pratique, l'extraction de fonctionnalités PCA et auto-encodeur sont utiles pour différents ensembles de données - l'un n'est pas meilleur que l'autre - ils représentent simplement les données différemment. Une question se pose **Peut-on remplacer une PCA par un auto-encodeur ?**

- PCA est linéaire, un auto encodeur est non linéaire.

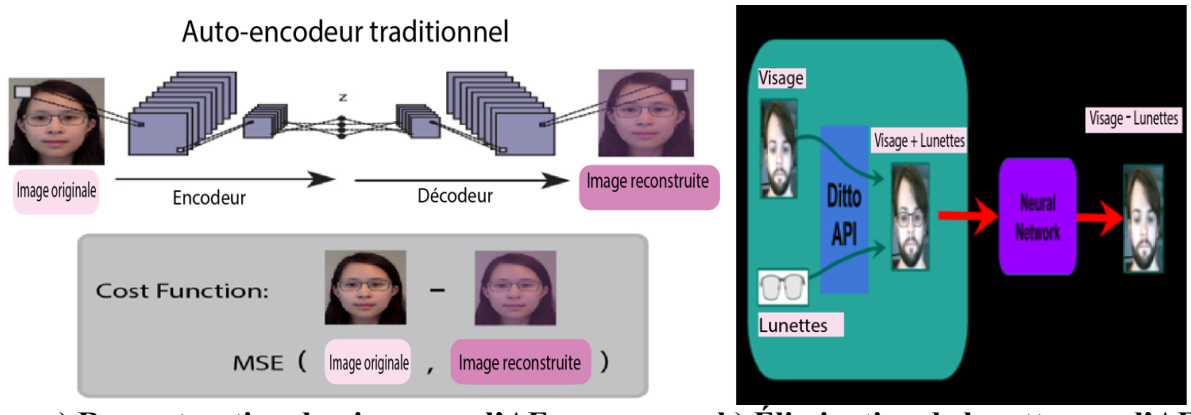
- La PCA peut fonctionner avec très peu de données, un auto-encodeur peut se sur-adapter si l'on n'a pas assez de données

Une bonne règle est d'essayer d'abord avec les fonctionnalités de la PCA et de voir ce qui se passe. Si les résultats ne sont pas assez bons, on peut essayer avec un auto-encodeur ou même combiner les fonctionnalités des deux.

Il est très important de noter qu'il n'y a pas de résultat gratuit dans l'apprentissage automatique (avec tous les algorithmes d'optimisation et tous les problèmes d'optimisation). Si les données reposent de manière inhérente sur un hyperplan approximatif à faible dimension et approximativement linéaire, les fonctionnalités PCA seront probablement meilleures que les fonctionnalités apprises par un auto-encodeur. Au contraire, si les données reposent sur une variété intrinsèque *non linéaire*, il serait probablement préférable d'utiliser un auto-encodeur. Un auto-encodeur avec des fonctions de transfert linéaires est équivalent à PCA. En regardant la fonction objective dans le problème d'optimisation décrivant les auto-encodeurs linéaires (c'est-à-dire les auto-encodeurs avec activation linéaire), on résout en gros la base qui minimise la somme des erreurs au carré ; c'est ce que fait la PCA.

1.8.9 Applications de l'Auto-Encodeur en biométrie du visage

Il existe plusieurs applications de l'auto-encodeur dans le domaine biométrique notamment la reconnaissance du visage et autres.



a) Reconstruction du visage par l'AE **b) Élimination de lunettes par l'AE**

Figure 1. 19 Exemple d'applications ISEE utilisant l'auto-encodeur [21]

1.9 Deep Auto-Encodeur pour la reconnaissance de visage

Les descripteurs de caractéristiques impliqués dans le traitement des images sont généralement choisis manuellement et de haute dimension.

La sélection des caractéristiques les plus importantes est une tâche cruciale pour des systèmes tels que la reconnaissance de l'expression faciale. Dans l'exemple de la **figure 1.20**, on étudie la performance des auto-encodeurs profonds pour la sélection des fonctionnalités et réduction des

dimensions pour la reconnaissance de l'expression faciale sur plusieurs niveaux de couches cachées. L'extraction des caractéristiques extraites de la pile de l'auto-encodeur est surperformée et sélectionnée par rapport à d'autres technologies avancées et techniques de réduction des dimensions.

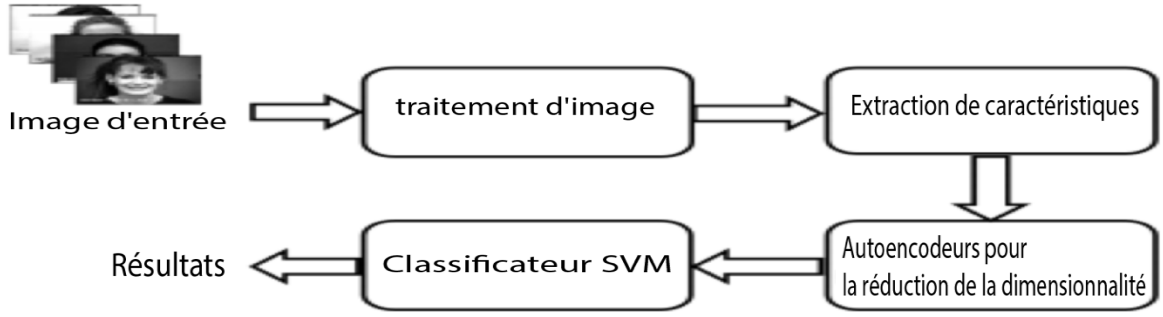


Figure 1. 20 Principe du DAE pour la reconnaissance des expressions faciales [22]

Conclusion

Dans ce premier chapitre nous avons défini l'apprentissage automatique et les types d'apprentissage. Nous avons présenté les différentes architectures de réseaux de neurones artificielles et le système de reconnaissance de visage classique ainsi que les limites de chacune d'entre elles. Nous avons aussi donné une représentation sur les auto-encodeurs et ses différents types et applications.

Le chapitre qui va suivre sera consacré à l'état de l'art des travaux récents sur les auto-encodeurs et les systèmes de reconnaissance de visage basé sur l'apprentissage profond ainsi que son utilisation dans différents domaines.

Introduction

Étant donné que les traits du visage contiennent un grand nombre d'informations d'identification et ne peuvent complètement être représentée par une seule caractéristique, la fusion de multiples caractéristiques est particulièrement importante pour obtenir une performance de reconnaissance du visage robuste, en particulier lorsqu'il existe une grande différence entre les ensembles de test et les ensembles de formation. Cela a été prouvé à la fois dans l'apprentissage traditionnel et approches en profondeur. Ces dernières années, des progrès ont été réalisés dans le traitement et reconnaissance des faces 3D-2D. L'apprentissage en profondeur a connu une croissance rapide en raison de l'émergence de jeux de données de visage à grande échelle. L'apprentissage puissant des données a entraîné une poussée de la recherche en reconnaissance faciale 2D. Il peut effectivement résoudre les problèmes des algorithmes traditionnels d'apprentissage automatique. Notre travail se concentre sur l'application de technologie d'apprentissage en profondeur à la pointe de la technologie en reconnaissance du visage.

Une revue de la littérature pertinente, du plus général au plus spécifique, sera faite et présentée dans ce chapitre.

Avant de présenter les travaux de Fukushima pour la création du modèle **Neocognitron** [23]. Nous commencerons par la présentation des architectures convolutionnelles et leurs analogies avec la vision et le traitement de l'information par l'être humain.

2.1 Les architectures convolutionnelles

Une représentation simpliste du flot d'information provenant des yeux est montrée dans la **figure 2.1**.

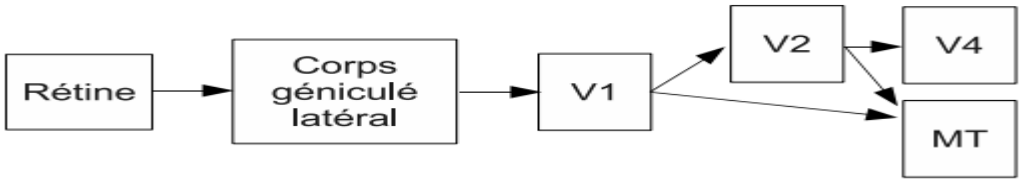


Figure 2. 1 Représentation schématique du flot de l'information provenant de la rétine [24]

Les "modules" suivant V1 exercent différentes fonctions de plus haut niveau qui seraient reliés notamment à la reconnaissance d'objet (flot dit "ventral" allant vers V4) et au positionnement et traitement du mouvement du sujet et des objets (flot dit "dorsal" allant vers MT).

Des travaux par Hubel & Wiesel en neurosciences [25] ont inspiré Fukushima pour la création du modèle **Neocognitron** [23]. Il y propose un modèle hiérarchique avec deux grands types principaux de neurones : les **cellules simples** et **cellules complexes**. Des couches alternées composées de cellules simples, puis complexes, constituent le modèle. Les cellules simples réagissent à la présence d'un motif dans leur champ réceptif. Un exemple souvent donné sera de détecter la présence de petits bouts de contours (edge) dans une orientation particulière dans leur champ réceptif. Si le motif corrèle assez bien avec l'activité dans le champ réceptif, le neurone s'activera. La réponse des cellules simples est apprise durant l'entraînement du modèle Neocognitron. On remarquera cependant que la détection de contour orientée à 45 degrés n'est pas quelque chose de spécifique à un neurone particulier, mais plutôt partagé par des neurones couvrant l'ensemble du champ visuel. C'est de cette observation que vient l'idée du partage de poids des architectures convolutionnelles : on tentera de détecter chaque motif appris à chaque position de l'image d'entrée, sans apprendre un motif séparé pour chaque position. En pratique chaque couche de cellules simples sera composée d'un certain nombre de **filtres de convolution**. En effet, pour appliquer le même motif à toute l'entrée, une opération de **convolution** du filtre fait le travail. On obtient donc, pour chaque filtre, une **"*feature map*"** de sortie donnant la réponse, à chaque position de l'entrée, de la somme des poids du filtre multipliés un à un aux pixels correspondants dans le voisinage de cette position. Le fait d'avoir plusieurs filtres nous permet d'avoir différents motifs pour une même couche, ce qui est bien sûr désirable si on veut pouvoir détecter, par exemple, plusieurs orientations de bords (dans le cas où des détecteurs de bords sont appris). On obtient donc plusieurs *feature maps*.

La réponse des cellules simples est cependant assez capricieuse sur la position du motif. S'il se déplace un peu, la réponse peut changer beaucoup. C'est là qu'interviennent les cellules complexes, en introduisant de l'invariance sur la position.

Ainsi une cellule complexe prendra en entrée l'état de quelques cellules simples à champs réceptifs proches, le ***pool*** de la cellule complexe. Cette dernière s'activera si l'une ou l'autre des cellules simples est active. On gagne donc une certaine indifférence ou invariance à la position du motif. La réponse des cellules complexes n'a pas à être apprise ; elle est prédéfinie de façon dure. On connectera donc une première couche de cellules simples à l'entrée, puis des cellules complexes à ces cellules simples. On pourra ensuite recommencer en prenant la sortie de ces premières cellules complexes comme entrées d'une prochaine couche de cellules simples. Ainsi par les couches de cellules simples on pourra détecter des motifs de plus en plus haut niveau et complexité. Par les couches de cellules complexes

on gagnera en invariance sur la position du motif et de la position relative de ses sous parties. En pratique dans ce travail on utilisera une opération de réponse maximale sur un champ réceptif ("*pool*") de taille fixe. Ainsi typiquement une cellule complexe prendra en entrée la réponse d'une zone de 2 par 2 unités d'une seule feature map et retournera la valeur maximale qui s'y trouve. Cette opération se nomme ***maxpooling***. Il s'agit d'une des façons de faire l'étape de **sous-échantillonnage** (ou **"*pooling*"**) dans un réseau convolutionnel, en pratique. Il n'y aura pas de partage d'aire couverte par deux cellules complexes adjacentes, dans l'implémentation. Le facteur de sous-échantillonnage (*pool_size*), par exemple 2, donne la dimension de la zone servant à calculer une unité d'une couche résultante du pooling. Les architectures convolutionnelles permettent donc, en résumé, d'apprendre des caractéristiques sous forme de hiérarchie, de simples à complexes, tout en ayant une certaine invariance sur leur position et taille dans l'entrée. L'idée d'entraîner de telles architectures par rétropropagation, pour des fins de classification, est apparue plus tard, avec le modèle LeNet [26]. En effet, le Neocognitron était entraîné par des critères non supervisés n'impliquant pas de gradient.

2.2 Reconnaissance en profondeur de visage

2.2.1 Deep Learning pour la reconnaissance de visage

L'apprentissage en profondeur est un domaine chaud dans la recherche sur l'apprentissage automatique en construisant des réseaux de neurones profonds pour simuler le mécanisme du cerveau humain puis interpréter et analyser des données telles que l'image, la voix et le texte. L'efficacité de l'apprentissage automatique traditionnel dépend en grande partie de la performance de représentation des caractéristiques artisanale. Au cours de ce processus, le rôle des approches d'apprentissage automatique est uniquement pour optimiser les poids d'apprentissage et finalement produire des résultats d'apprentissage optimaux. Ces méthodes traditionnelles d'apprentissage automatique ou l'apprentissage en profondeur tente de compléter le travail de représentation des données et d'extraction des caractéristiques (**figure 2.2**).

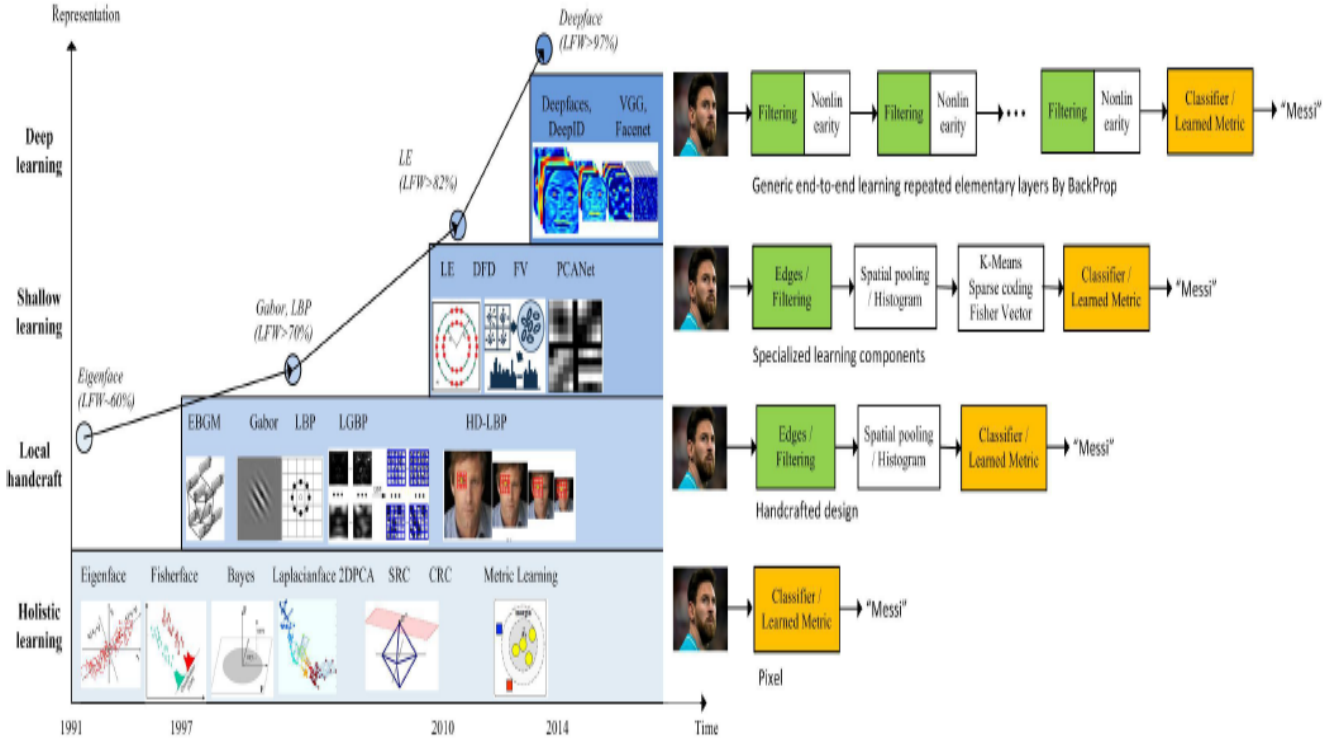


Figure 2. 2 Évolution de la représentation des fonctionnalités pour la RV [27]

Les approches holistiques ont dominé la communauté RV dans les années 1990 et 2000. Au début des années 2000 et 2010, des descripteurs locaux basés sur les caractéristiques locales et basés sur l'apprentissage ont été introduits successivement. En 2014, DeepFace [27] et DeepID **figure 2.2** ont atteint une précision irréprochable et la recherche s'est orientée vers les approches fondées sur l'apprentissage en profondeur. À mesure que le pipeline de représentation devient de plus en plus profond, les performances du LFW (Labeled Face in the Wild) s'améliorent régulièrement d'environ 60% à plus de 97%.

La puissance principale d'un réseau de neurones convolutif (CNN) se trouve dans l'architecture profonde, ce qui permet d'extraire un ensemble de caractéristiques discriminantes, représentations à plusieurs niveaux d'abstraction. Ces dernières années, CNN a été largement appliqué dans le traitement et la reconnaissance des visages en raison de ses bonnes performances. Le succès de CNN est attribué à sa capacité d'apprendre de riches fonctionnalités d'image. Cependant, la formation d'un réseau profond de CNN repose sur l'apprentissage de millions de personnes.

Les paramètres réseau nécessitent de nombreux jeux de données étiquetés pour la formation préalable. En pratique, nous ne formons généralement pas un réseau CNN complet à partir de rien, car il est impossible de recueillir un ensemble de données d'une taille suffisante

pour répondre aux besoins de CNN à chaque nouvelle tâche de reconnaissance faciale. Comme alternative, il est courant de préformer le modèle CNN sur un grand jeu de données, puis d'utiliser les poids de ce modèle en tant qu'initialisation ou en tant qu'extracteur de fonctionnalités spécifique pour la tâche qui nous intéresse, puis ajustez enfin le modèle de réseau à l'aide de l'ensemble de test. Cependant, lorsque les jeux de données de test et d'apprentissage présentent de grandes différences en termes d'éclairage, d'expression, points de vue ou d'autres facteurs, même le CNN ajusté ne peut pas obtenir de bonnes performances de reconnaissance. La cause fondamentale est que le réglage est toujours basé sur le modèle CNN préformé qui limite la performance de reconnaissance pour la tâche de reconnaissance en cours. Un réglage précis efficace nécessite que toutes les couches du modèle préformé doivent être ajustées lorsque les variations entre la source et la cible des applications sont importantes. Cependant, les données précises et limitées peuvent provoquer un sur-ajustement lorsque CNN préformé est trop optimisé. CNN assure la tâche de classification et de reconnaissance principalement par l'étude de la distribution des données. Quand il y a une grande différence entre ensemble de test et de formation, la distribution initiale des données obtenue par le réseau CNN préformé est difficile à changer, ce qui rend difficile l'amélioration de l'adaptabilité du modèle CNN, même si ce dernier modèle est affiné.

Un autre problème du réseau de formation est le grand nombre de paramètres du réseau et le fait que cela prend du temps. La raison en est que la modification des paramètres de n'importe quelle couche entraîne la modification des distributions pour les données d'entrée dans les couches suivantes, ce qui conduit le réseau de neurones en permanence au besoin de s'adapter à la nouvelle distribution des données. Ainsi, le réglage minutieux des paramètres et l'entraînement avec un taux d'apprentissage plus faible est particulièrement important. De plus, la non linéarité de l'opération d'activation rend la tâche plus difficile. Le Cun [9] a proposé que la normalisation des données peut accélérer la convergence du réseau. En 2015 dans [28], la normalisation par lots a été proposée pour résoudre la covariable interne de décalage.

2.2.2 Identification en profondeur des visages générés par CNN

Avec l'évaluation exploratoire dans [29] sur deux différents ensembles de données communes on peut soutenir les idées selon lesquelles (i) les auto-encodeurs peuvent être utilisés pour la réduction de dimension et fusion de caractéristiques et que (ii) ces auto-encodeurs peuvent être formés sur des domaines différents (mais sans doute similaires) que ceux du modèle utilisé. Les auteurs [29] affirment que les deux ensembles de données sélectionnés ne sont pas représentatifs pour une évaluation à grande échelle, mais ils fournissent une première approche pour investiguer sur une idée nouvelle dans le traitement des données d'image rares,

c'est-à-dire qu'il n'y en a pas assez données pour former un modèle, et des jeux de données d'image, où seules les fonctionnalités pour la reconstruction sont disponibles.

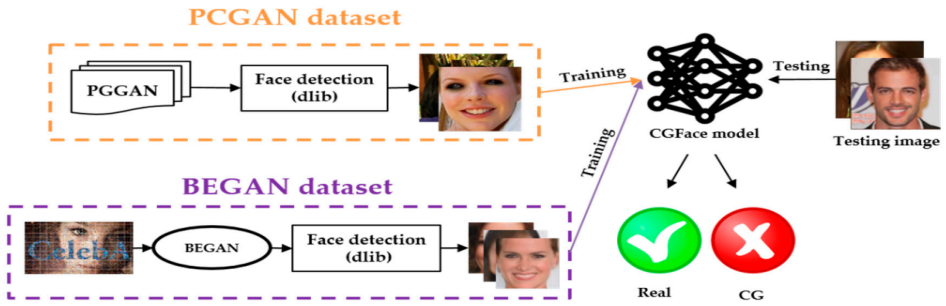


Figure 2. 3 Architecture globale de la méthode proposée [29]

La **figure 2.3** illustre un modèle de détection de visage généré par ordinateur, appelé CGFace, et présente chaque couche avec sa taille d'entrée, de noyau et de sortie correspondante.

Globalement, le modèle fonctionne bien, car on maintient un degré de diversité des visages sur toute la gamme des valeurs **γ** (le rapport entre deux pertes de reconstruction dans le temps). À faible valeur (0,3), les visages générés sont généralement uniformes. Cependant, avec une valeur plus élevée, par exemple 0,7, la variété des visages augmentent, mais les artefacts et le bruit apparaissent également plus fréquemment.

Figure 2. 4 Images générées avec différentes valeurs de l'hyperparamètre **γ** [29]
(*Certaines régions ressemblant à du bruit sont marquées en rouge*).

2.3 Deep Auto-encodeurs pour l'Image

Le domaine de la recherche d'images basée sur la similarité a connu un grand bouleversement ces derniers temps. Les fonctionnalités d'image fabriquées manuellement ont été largement surperformées par les approches basées sur l'apprentissage automatique. Les méthodes d'apprentissage en profondeur sont très bonnes pour trouver des caractéristiques

optimales de l'image, étant donné que suffisamment de données sont disponibles pour l'apprentissage.

Les auto-encodeurs ont été proposés dans [3] pour réduire la dimensionnalité de données avec des réseaux de neurones. Wang et al. [30] ont enquêté sur la différence entre les auto-encodeurs pour la réduction de la dimensionnalité et les méthodes optimales, c'est-à-dire PCA et LDA en tant que représentant de méthodes linéaires, ainsi que les méthodes non linéaires LLE et Isomap. Ils concluent que les auto-encodeurs conviennent à la détection de structures répétitives. Dans la récupération d'image basée sur le contenu, les auto-encodeurs ont été utilisés pour mapper les images en codes binaires courts.

Contrairement à ce travail, où les images complètes ont servi d'entrée, on évite d'opérer sur les pixels des images, mais sur les fonctionnalités extraites de l'image. Zhang et al. [31], proposent d'utiliser des auto-encodeurs creuses superposés (SSAE) et un cadre de fusion basé sur des graphes pour le cas d'utilisation dans l'analyse d'images histopathologiques.

Ma et al. [32] propose Réseaux profonds de fusion multi-caractéristiques (MFFDN) les travaux les plus liés. Les MFFDN sont basés sur des auto-codeurs en débruitage, utilisant les caractéristiques basées sur la couleur, la texture, la forme et le dégradé qui sont utilisées pour la classification. À notre connaissance, il y a peu de travaux dans la fusion précoce de descripteurs d'image standard avec des auto-encodeurs.

2.3.1 Travaux récents sur le Deep Auto-encodeurs pour l'Image

- **Réduction de dimension d'image hyper-spectrale par l'auto-encodeur**

La méthode non supervisée proposée par Spencer A. Thomas et al. [33], utilisant l'AE fournit à la communauté MSI (imagerie par spectrométrie de masse) une technique efficace de réduction de la dimensionnalité non linéaire qui comprend la mise en correspondance avec l'espace dimensionnel réduit. Cette méthode présente des avantages supplémentaires par rapport aux méthodes telles que la PCA en éliminant la nécessité sélectionner des caractéristiques significatives dans la liste complète des composants, en réduisant la subjectivité et les interactions humaines significatives à partir de l'analyse.

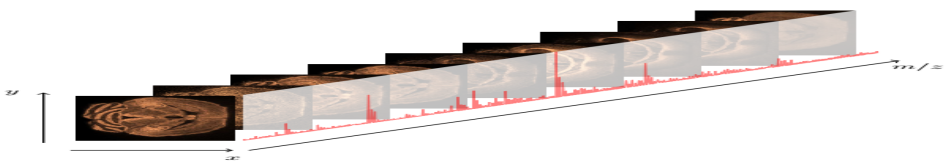


Figure 2. 5 Spectrométrie de données d'image de masse hyper-spectrale [33]

Les images dites ioniques sont créées en prenant des valeurs d'intensité de pixel à des valeurs spécifiques m / z, ce qui permet de localiser et de distribuer les espèces chimiques [33].

Dans ce travail, une image de 409x404 pixels est considérée où chaque pixel est un spectre contenant 7036 m / z. Comme les auto-encodeurs ne suppriment pas les informations pendant la compression [19] Il est préférable d'utiliser cette technique sans préavis. La réduction pour éviter la perte d'informations pouvant être présentes en composants principaux d'ordre élevé. Dans ce travail l'auto-encodeur est composé de 7036 nœuds d'entrée et 15 nœuds cachés, prenant chacun des 165 236 pixels comme exemple de formation. Les scores pour PCA sont donnés à la figure 2.6 (a) pour le premier 15 composants et la figure 2.6 (b) montre les résultats pour l'algorithme NMF avec 15 facteurs.

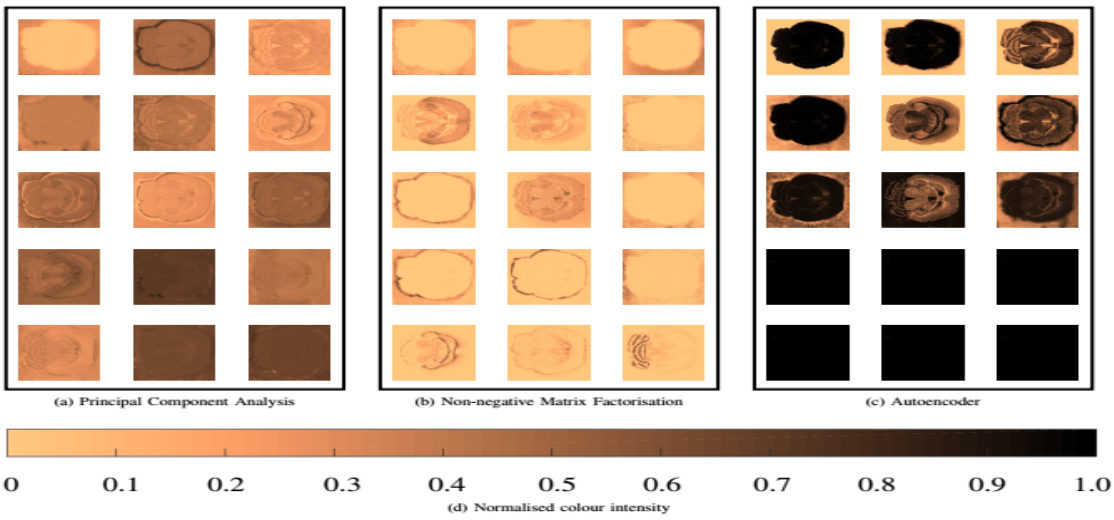


Figure 2. 6 Résultat de différentes méthodes de réduction [33]

(a) Analyse en composantes principales (PCA), (b) Factorisation matricielle non négative (NMF) et (c) Variables latentes provenant de l'auto-encodeur du jeu de données du cerveau de souris. (d) Couleur normalisée

Les intensités varient de 0 (cuivre) à 1 (noir) pour toutes les méthodes, afin de rendre la visualisation plus claire. Les caractéristiques des notes dans chaque cas ont été classées par ordre décroissant de variance.

Figure 2. 7 Comparaison du cerveau de souris et AE [33]

Comparaison de (a) régions connues d'un cerveau de souris de Allen Brain Atlas, (b) et (c) d'un seul neurone caché dans l'auto-encodeur [33]

La **figure 2.7** illustre la capacité de l'auto-encodeur pour séparer les régions anatomiques du cerveau basé sur la réduction de dimensionnalité non supervisée où la plupart des régions sont capturées par un seul nœud caché. Ceci démontre le succès de l'auto-encodeur à réduire le nombre de dimensionnalité des données MSI à quelques valeurs m / z qui fournissent informations structurelles sur l'échantillon.

Ici, l'auto-encodeur est appliqué aux données MSI pour la réduction de dimensionnalité non linéaire non supervisée. Seulement quelques-uns des travaux portent sur l'utilisation d'auto-encodeurs avec des données multidimensionnelles. Celles-ci consistent soit en des séries de données chronologiques traités comme d'avantage d'exemples de formation, plutôt que comme une dimension supplémentaire dans les données, ou un petit nombre canaux de spectres qui sont initialement réduits via PCA. Avec le succès d'appliquer un auto-encodeur au spectre de masse pour chaque pixel dans une image MS et la réduire à ses fonctionnalités principales.

Avec l'évaluation exploratoire dans [34] sur deux différents ensembles de données communes on peut soutenir les idées selon lesquelles (i) les auto-encodeurs peuvent être utilisés pour la réduction de dimension et fusion de caractéristiques et que (ii) ***ces auto-encodeurs peuvent être formés sur des domaines différents (mais sans doute similaires) que ceux du modèle utilisé.*** Les auteurs [34] affirment que les deux ensembles de données sélectionnés ne sont pas représentatifs pour une évaluation à grande échelle, mais ils fournissent une première approche pour investiguer sur une idée nouvelle dans le traitement des données d'image rares, c'est-à-dire qu'il n'y en a pas assez données pour former un modèle, et des jeux de données d'image, où seules les fonctionnalités pour la reconstruction sont disponibles. Cette méthode permet de rapidement récupérer les images dans des domaines où les données d'apprentissage sont rares.

2.3.2 Travaux récents sur le Deep Auto-Encodeurs pour la RV

2.3.2.1 Reconnaissance de Visage basé sur le Deep 2DPCA-CNN (C2D-CNN)

Dans ce travail, les auteurs [28], ont proposé une nouvelle méthode appelée C2D-CNN (analyse en composantes principales (2DPCA) - réseau de neurones de conversion (couleur à deux dimensions)). C2D-CNN combine les caractéristiques apprises à partir des pixels d'origine avec la représentation d'image apprise par CNN, puis effectue une fusion au niveau de la décision, ce qui peut améliorer considérablement les performances de la reconnaissance faciale. Ainsi, un nouveau modèle CNN est proposé :

Premièrement, on introduit une couche de normalisation dans CNN pour accélérer la convergence du réseau et raccourcir le temps de formation. Deuxièmement, la fonction d'activation en couches est introduite pour rendre la fonction d'activation adaptable aux données

normalisées. Finalement, la probabilité pool maximum est appliquée de sorte que les informations sur les caractéristiques soient préservées au maximum et étendues tout en maintenant l'invariance des caractéristiques.

La reconnaissance des faces de couleur est généralement effectuée sur chaque canal de couleur et le résultat final vient de la fusion des trois canaux de couleur. De même, les réseaux CNN utilisent pleinement la technologie trois couleurs (canaux de RVB) pour l'extraction des caractéristiques, respectivement. Tous ne prennent pas en compte les corrélations internes entre canaux de couleur. Une méthode de fusion des canaux de couleur utilisant un algorithme de réduction de la dimensionnalité conjointe a été utilisée. En revanche, l'analyse des composantes principales en deux dimensions (2DPCA) est basée sur le concept de valeur de couleur comprenant trois canaux, consacrés à la pleine utilisation des informations de couleur et de la corrélation entre les canaux de couleur. Couleur 2DPCA est conçu pour combiner des informations de couleur et spatiales. Ça encore a une bonne robustesse pour le test et la formation avec de grandes différences. Ainsi, la couleur 2DPCA fournit un moyen efficace pour améliorer la robustesse de CNN. Les contributions de [28] sont résumées comme suit :

• Proposition d'une méthode de reconnaissance de visage profond, C2D-CNN, qui combine les deux caractéristiques dans le niveau de prise de décision, avec une grande précision et un faible coût de calcul.

• Étude d'un nouveau modèle CNN. Grâce à une conception soignée, (1) la normalisation est introduite pour accélérer la convergence du réseau et raccourcir le temps de pré-formation du réseau; (2) une couche algorithme d'activation est ajouté pour améliorer la fonction non linéaire de la fonction d'activation et résoudre le problème de la saturation et de la diffusion du gradient; (3) max-pooling probabiliste est appliqué pour préserver la fonctionnalité dans la mesure maximale tout en maintenant l'invariance de la fonctionnalité.

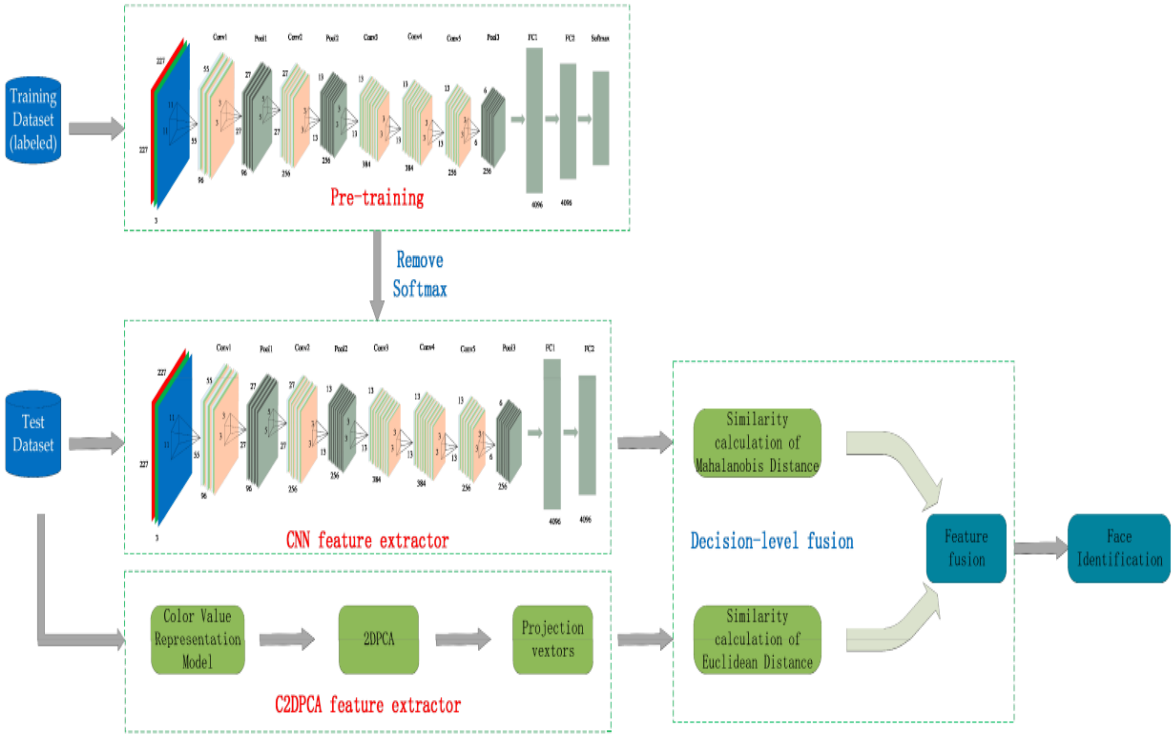


Figure 2. 8 Système de reconnaissance de visages CNN et C2DPCA (C2D-CNN) [28]

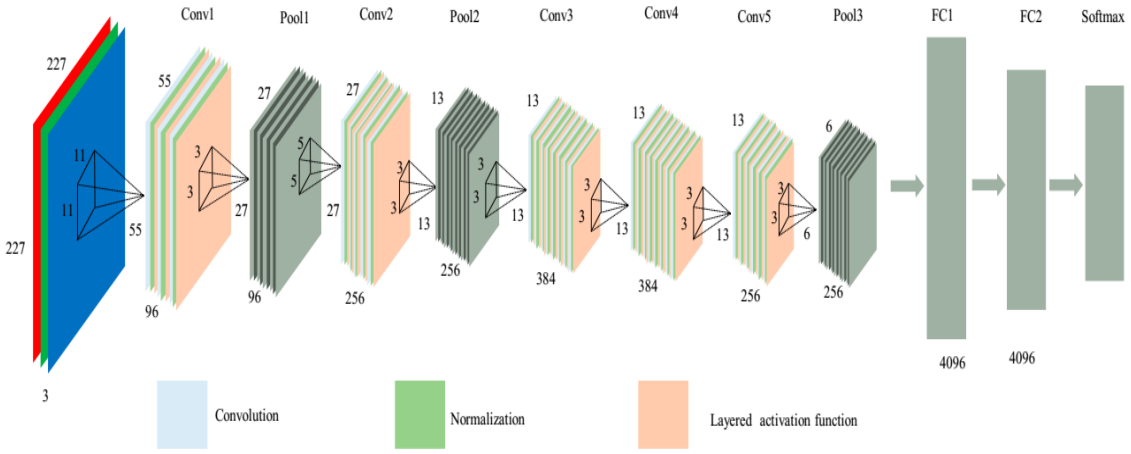


Figure 2. 9 Structure du réseau CNN basée sur la reconnaissance faciale [28]

Les résultats expérimentaux montrent que par rapport aux méthodes de pointe, cette méthode montre de meilleures performances et résout une faible précision de reconnaissance causée par la différence entre les jeux de données test et formation.

2.3.2.2 Nouvelle menace pour la reconnaissance des visages (DeepFakes)

Il devient de plus en plus facile de remplacer le visage automatiquement d'une personne dans une vidéo par le visage d'une autre personne en utilisant un réseau préformé « Generative Adversarial Network » (GAN). Scandales publics récents, par exemple les visages de célébrités

permutés sur des vidéos, des moyens automatisés de détecter ces vidéos Deepfake sont recherchés. Pour aider au développement de telles méthodes, les auteurs [35], présentent le premier ensemble de Deepfake disponible au public.

Concept de base : Le concept de Deepfakes est très simple. Disons que l'on veut transférer le visage d'une personne A vers une vidéo de la personne B. Deux cas se présentent :

i) Utilisation de l'AE pour image fixe du visage, ii) Utilisation du GAN pour les vidéos de visage

Premièrement, on recueille des centaines ou des milliers d'images pour les deux personnes. On construit un encodeur pour encoder toutes ces images en utilisant un réseau CNN à apprentissage approfondi. Ensuite, on utilise un décodeur pour reconstruire l'image. Cet **auto-encodeur** (l'encodeur et le décodeur) a plus d'un million de paramètres mais n'est même pas assez proche pour mémoriser toutes les images. Le codeur doit donc extraire les fonctionnalités les plus importantes pour recréer l'entrée d'origine.

Pour décoder les caractéristiques, on utilise des décodeurs distincts pour la personne A et la personne B. Puis le codeur et les décodeurs sont formés (à l'aide de la rétropropagation) de sorte que l'entrée corresponde étroitement à la sortie. Ce processus prend du temps. Avec une carte graphique GPU, il faut environ 3 jours pour générer des résultats décents. (après avoir répété le traitement des images environ 10 millions de fois).

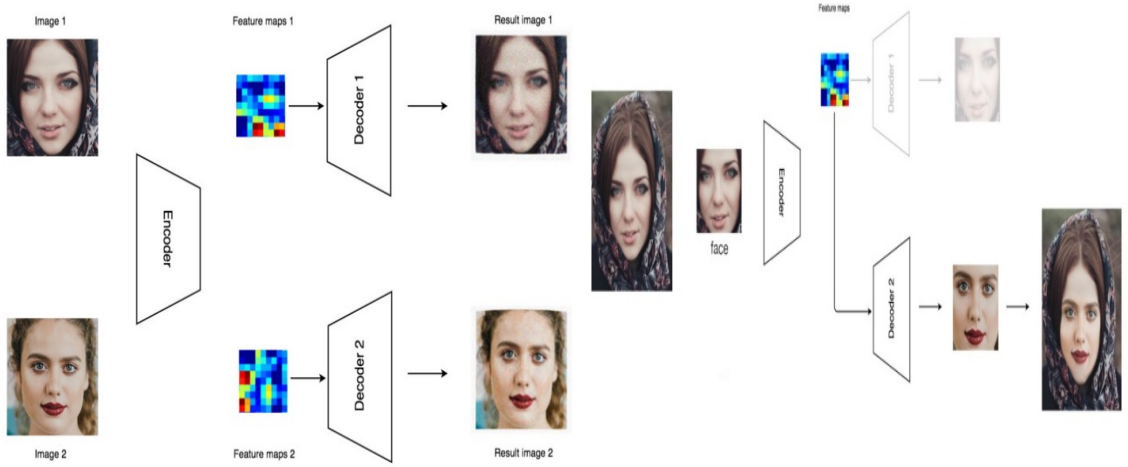


Figure 2. 10 Architecture du DeepFakes par l'AE [36]

Après la formation, la vidéo est traitée image par image pour échanger un visage avec un autre. À l'aide de la détection de visage, le visage est extrait de la personne A et introduit dans le codeur. Cependant, au lieu d'alimenter son décodeur d'origine, on utilise le décodeur de la personne B pour reconstruire l'image. C'est-à-dire que la personne B est décodée avec les

caractéristiques de A dans la vidéo d'origine. Ensuite, le nouveau visage créé est fusionné dans l'image d'origine.

2.3.2.3 Autres applications de l'AE pour la RV

- ***Reconnaissance de visage 2D en présence de poses***

Une métrique de reconstitution de caractéristiques est utilisée pour répartir un CNN en sous-réseaux d'identité et de pose (voir **figure 2.11**).

Ces réseaux multiples (réseaux à entrées multiples), correspondent à « une augmentation multiple », qui génère plusieurs images de différents patchs ou poses facilitant la reconnaissance de visage en présence de poses.

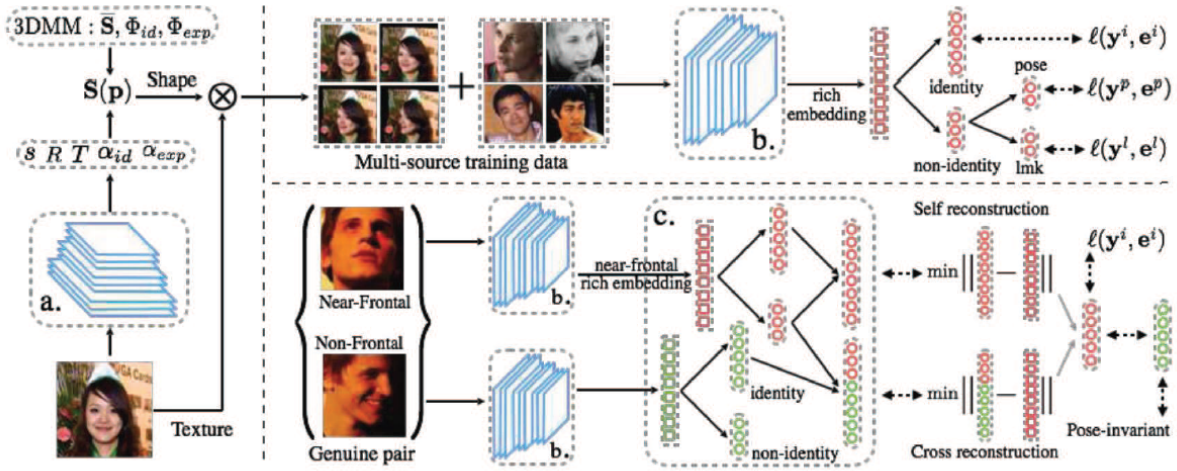


Figure 2. 11 Génération de poses par l'AE pour la reconnaissance de visage 2D [27]

- ***Reconnaissance de visage 2D avec variante âge***

Bien que des progrès considérables aient été réalisés en matière de reconnaissance des visages, la reconnaissance des visages invariante selon l'âge (AIFR) reste un défi majeur dans les applications du monde réel des systèmes de reconnaissance des visages. La principale difficulté de l'AIFR provient du fait que l'apparence du visage est sujette à d'importants changements intra-personnels causés par le processus de vieillissement au fil du temps. Afin de résoudre ce problème, dans [37] un nouveau système de reconnaissance des visages profonds est proposé pour apprendre les caractéristiques des visages profonds invariants au vieillissement par le biais d'un modèle CNN conçu avec soin (voir **figure 2.12**). Il s'agit de la première tentative visant à démontrer l'efficacité des CNN en profondeur pour faire progresser l'état de l'art des AIFR. Des expériences approfondies sont menées sur plusieurs ensembles de données de vieillissement des faces du domaine public (MORPH Album2, FGNET et CACD-VS) afin

de démontrer l'efficacité du modèle proposé par rapport à l'état de la technique. On vérifie également l'excellente généralisation de ce modèle sur le célèbre jeu de données LFW.

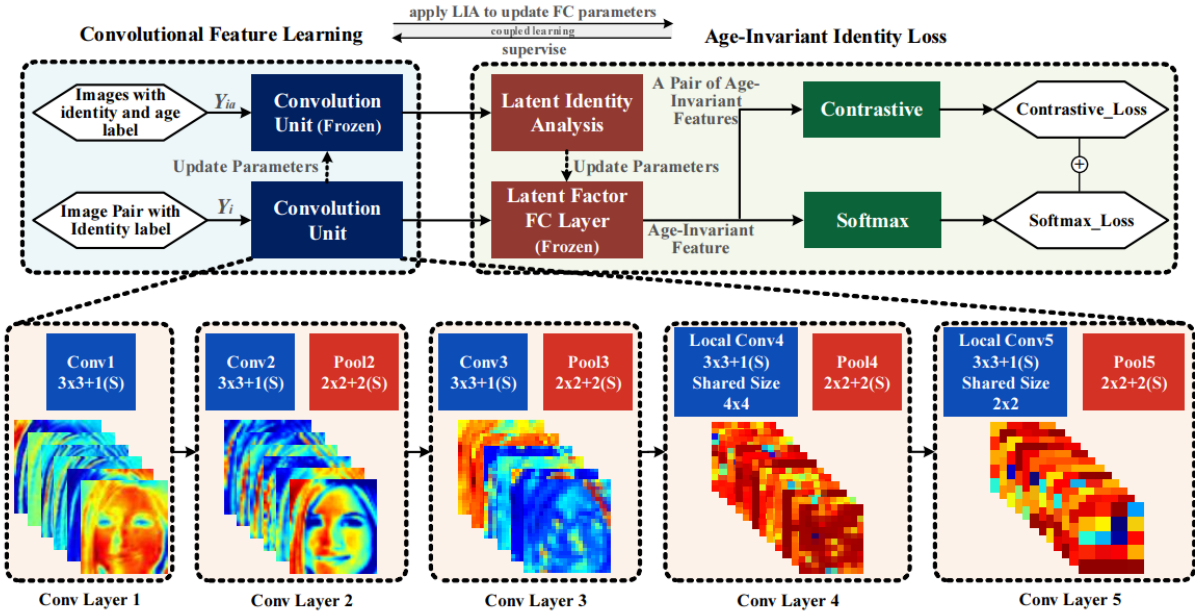


Figure 2. 12 Reconnaissance de visage 2D avec variante âge avec Latent Factor-CNNs [37]

- ***Reconnaissance de visage 2D avec variante maquillage***

Le maquillage est largement utilisé par le public aujourd'hui, mais il pose également des problèmes pour la RV en raison de changements importants de l'apparence du visage. La recherche sur les images de maquillage assorties et non maquillées attire de plus en plus l'attention. [27] ont généré des images non maquillées à partir d'images de maquillage par un réseau contradictoire à deux niveaux BLAN (Bi-Level Adversarial Network), puis ont utilisé les images non maquillées synthétisées à des fins de vérification (**figure 2.13**).

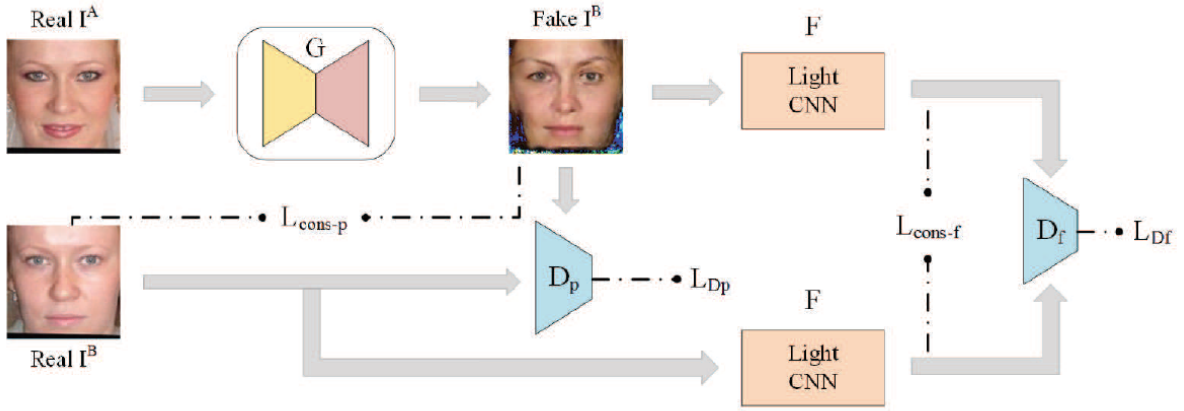


Figure 2. 13 Architecture d'un BLAN pour le visage [27]

- ***Reconnaissance de visage d'une caricature ou image robot***

L'image robot peut aider les forces de l'ordre à identifier rapidement les suspects.

Les méthodes couramment utilisées peuvent être classées en deux classes. L'une consiste à utiliser l'apprentissage par transfert pour associer directement des photos à des croquis, les réseaux profonds étant d'abord formés à l'aide d'une grande base de données de photos visage, puis affinés à l'aide d'une petite base de données de croquis. L'autre consiste à utiliser la traduction image à image, où la photo peut être transformée en croquis ou l'image robot en photo ; alors, la RV peut être effectué dans un domaine. Un réseau a été développé, celui-ci est totalement conventionnel avec perte générative et un régularisateur discriminant pour transformer les photos en croquis. Récemment, les GAN ont obtenu des résultats impressionnants en matière de génération d'images. [38] ont utilisé deux générateurs, GA et GB, pour générer des croquis à partir de photos et des photos à partir de croquis, respectivement (**figure 2.14**).

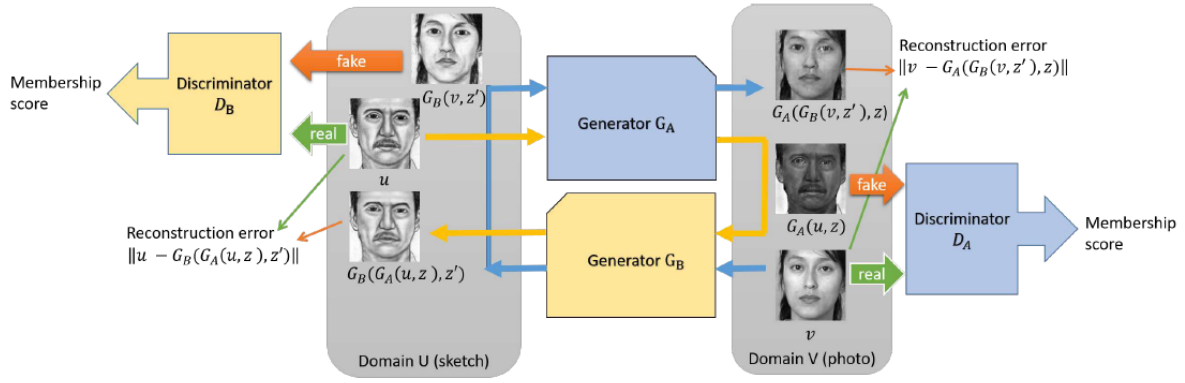


Figure 2. 14 Architecture du DualGan appliqué au visage [38]

2.3.3 Utilisation d'un AE en mode génératif

Utiliser un AE en mode génératif revient alors à échantillonner des vecteurs de variables latentes ($\lambda_1, \lambda_2, \ldots, \lambda_{50}$) puis à reconstruire les images associées au moyen du décodeur. Ce processus est créatif, si l'on veut, dans la mesure où les visages générés de cette manière seront différents de ceux du jeu d'entraînement. Il n'y a aucune raison en effet pour que les vecteurs latents choisis au hasard coïncident avec à ceux des images du jeu d'entraînement.

2.3.3.1 Les auto-encodeurs variationnels (VAE)

Les auto-encodeurs simples fonctionnent mal en pratique. Ils ne permettent pas de réaliser un encodage continu de représentations qui permettrait de se déplacer sur l'espace des visages pour faire du morphing sur des caractéristiques bien identifiées. La seule raison pour parler des AE simples était d'introduire la notion de variables latentes.

Les auto-encodeurs variationnels (VAE) rajoutent deux ingrédients supplémentaires par rapport aux AE simples qui leur permettent de réaliser la limite citée ci-dessus.

Plutôt que d'encoder chaque image de manière déterministe par un point λ dans l'espace latent, un VAE va choisir au hasard un point λ^* proche de ce point λ. Plus précisément, l'encodeur d'un VAE calcule une distribution de probabilité centrée sur λ et d'écart-type σ, les deux étant appris par l'encodeur qui comporte en l'occurrence deux modules comme l'illustre la **figure 2.15**. Le générateur d'un VAE comporte deux parties dont l'une détermine la moyenne λ et l'autre la variance σ d'une distribution normale dont on échantillonne un point λ^* au hasard dans l'espace latent.

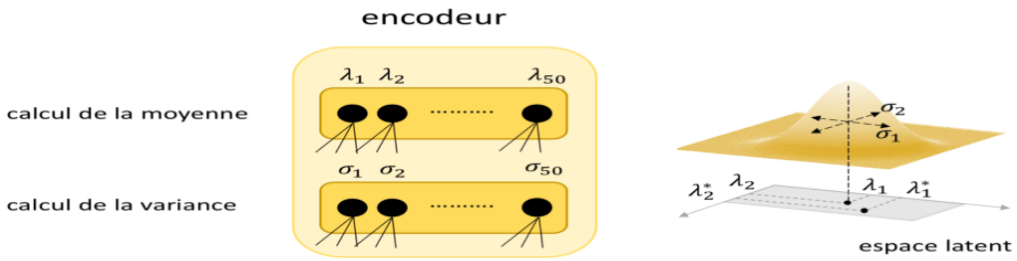


Figure 2. 15 Générateur d'un VAE [39]

En ce sens les VAE sont donc des modèles stochastiques, ils introduisent une part imprévue dans le processus de génération.

L'expérience a démontré qu'en plus propriétés de continuités déjà évoquées les VAE avait par ailleurs des propriétés arithmétiques qui les rendent très utiles pour la retouche d'image avancée. Ainsi peut-on identifier une direction « sourire » ou une direction « lunettes » dans l'espace latent qui permettent de moduler l'expression d'une photo (**figure 2.16**).

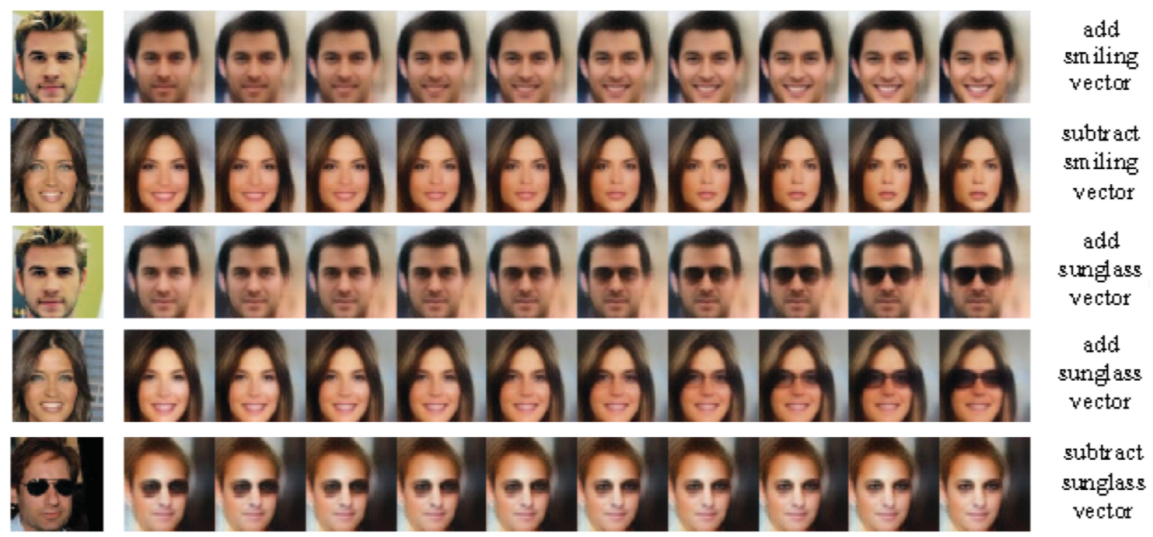


Figure 2. 16 Application du vecteur « sourire » et « lunettes » à plusieurs exemples [39]

Pour trouver ces directions dans l'espace latent, rien de plus simple : pour construire un vecteur « sourire » il suffit de soustraire le vecteur représentatif d'un visage triste à celui d'un visage souriant (ou de faire une moyenne de telles différences).

2.3.3.2 Les réseaux adverses génératifs (GAN)

L'état de l'art en matière de génération d'images réalistes ne repose plus aujourd'hui sur les VAE mais sur une nouvelle classe de modèles génératifs inventée en 2014 par Ian Goodfellow et appelés Generative Adversarial Networks (GAN). Les GAN sont intéressants pour leurs performances mais, plus encore peut-être pour l'innovation conceptuelle qu'ils représentent dans le domaine du Deep Learning. Yann LeCun, directeur de recherche du labo Facebook AI Research dit à ce propos : « *Les GAN sont l'idée la plus intéressante (en AI) que j'ai vue depuis 10 ans !* » [39].

A bien y réfléchir un faux monnayeur et un modèle génératif ont effectivement des objectifs similaires : le premier cherche à duper l'autorité en charge d'émettre la monnaie officielle en confectionnant des faux billets réalistes alors que le second doit produire des images crédibles capables de leurrer des humains. Un système GAN est constitué de deux réseaux de neurones en compétition comme l'illustre la **figure 2.17**.

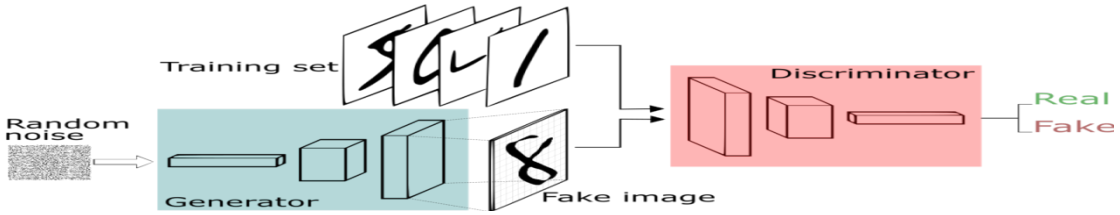


Figure 2. 17 Schéma du système GAN [39]

Le système GAN de la **figure 2.17** est composé d'un générateur chargé de créer des images réalistes de digits manuscrits par transformation d'un bruit aléatoire et d'un discriminateur chargé de distinguer les vraies images des images fictives. Le discriminateur est basé sur un réseau de convolution (CNN). Le générateur utilise une forme de CNN inversé.

Le premier, appelé le générateur, a pour objectif de créer des images fictives crédibles qui ressemblent à celles d'un ensemble d'entraînement. Ce générateur fonctionne sur le même principe que le décodeur d'un AE ou d'un VAE : il apprend à convertir du bruit (un point tiré au hasard dans l'espace latent) en une image. Ils délèguent le contrôle qualité à un second réseau de neurones, appelé le discriminateur, qui va progressivement apprendre à distinguer les vraies images des fausses sur la base d'exemples connus.

L'objectif de cette compétition, ou de ce jeu si l'on préfère, est de parvenir à un équilibre où le générateur parvient à reproduire des images indiscernables des originaux tandis que le discriminateur attribue des étiquettes correctes une fois sur deux. La **figure 2.18** montre 20 chambres à coucher synthétisées par un GAN.

Figure 2. 18 Images générées par un GAN [39]

Si les machines savent produire des images réalistes de chambres à coucher elles n'ont en revanche pas la moindre idée des usages possibles d'un lit ou d'un miroir.

2.4 Étude comparative de l'état de l'art

Dans le tableau 2.1 nous présentons l'essentiel des méthodes récentes sur l'application de l'auto-encodeur dans différentes applications.

Tableau 2. 1 Étude comparative de l'auto-encodeur et le DL dans les travaux récents

Auteurs	**Années**	**BDDs**	**Méthodes**	**Résultats**
Y.LeCun et al [9]	1998	MNIST **(Tr=60000images)**	LeNet CNN en GD	***Sans distorsions de données*** ER $_{(Test)}$ = 1 % ER $_{(Training)}$ = 0.5 % ***Avec distorsions de données*** ER $_{(Test)}$ = 0.8 %
J.WANG et al [30]	2012	MNIST	FAE avec ss CAE	Loss $_{(FAE)}$ 0.0351 Loss $_{(PCA)}$ 11.6745
W.Wang et al [3]	2014	CMU PIE	GAE	Err $_{(dGAE- MFA / dGAE-LE)}$ = 1.1% Err $_{(dGAE-PCA)}$ = 3.5%
M.Wang et al [27]	2014	LFW BDD (Guo et al)	DeepFace/DeepID BLAN	Acc = 60% à plus de 97% Acc = 94.8%

Y.Wen et al [37]	2016	MORPH **Album2 Tr : 10000 images** FGNET **82sujets (1002 images)** CACD-VS. **Tr : 400000 images** LFW **Tr : 700000 images**	LF-CNNs (**Sans réduction**)	TR = 94.4 % TR = 88.1% TR = 98.5% TR = 99.50%
X.ZHANG et al [31]	2016	120 images de tissus mammaires provenant de 40 patientes.	SSAE	Acc = 91.67%
G.Ma, et al [32]	2016	MNIST CIFAR-10 SVHN	MFFDN avec DAE	Err = 0.01 % Err = 0.22 % Err = 0.16 %
S.Thomas et al [33]	2016	MALDI	AE pour DR	-
S.Petscharnig et al [34]	2017	UCID	AE & DL pour DR	Err = 0.05%
Z.YI et al [38]	2017	LABEL-FACADES AERIAL- MAPS	DualGAN	***Segmentation accuracy pour facades→label*** Acc. $_{\text{per-pixel}}$ = 0.54 Acc. $_{\text{Per-class}}$ = 0.33 Acc. $_{\text{Class IOU}}$ = 0.19 ***Segmentation accuracy pour aerial→map*** Acc. $_{\text{per-pixel}}$ = 0.70 Acc. $_{\text{Per-class}}$ = 0.46 Acc. $_{\text{Class IOU}}$ = 0.46
P.Korshunov et al [35]	2018	VidTIMIT (Deepfake videos) Low Quality High Quality	 GAN (IQM+SVM) GAN (IQM+SVM)	 ERR= 3.33 ERR= 8.97
L.DANG et al [29]	2018	PCGAN et BEGAN dataset	CGFace	Acc = 98%
J.Li et al [28]	2018	FRGC v2.0 LFW	C2D-CNN	Acc $_{(FD=50)}$ = 91% Acc $_{(FD=50)}$ = 93% RR $_{\text{(distance euclidienne)}}$ = 95.9%

GD : gradient descent, GAN : generative adversarial network, DR : Dimension Reduction, (SSAE): Stacked Sparse Autoencoder, GAE : generalized auto-encoder, MFFDN : Multi-feature Fusion Deep Networks, DAE : denoising auto-encoder, FAE : Folded Neural Network Autoencoder, SS : symmetric structure, CAE : conventional autoencoder, UCID : Uncompressed Colour Image Database , FRGC : Face Recognition Grand Challenge, ER : Error Rate TR : taux de reconnaissance, FAR : False Acceptance Rates, Acc : Accuracy, FD : Feature Dimension, RR : Recognition Rate.

Conclusion

Plusieurs choix d'application ont été proposés concernant le Deep Learning et l'auto-encodeur notamment le problème de la reconnaissance faciale. Les méthodes superficielles ont été exclues car elles ne permettent que la transformation linéaire, tandis que les méthodes d'architecture profonde non linéaires ont été retenues car elles peuvent apprendre des représentations plus compliquées. En outre, de nombreuses conceptions d'architecture profonde ont été proposées jusqu'à présent. Malgré le grand succès des CNN, leur utilisation est déterminante. Les CNN sont principalement utilisés pour des tâches très spécifiques. En raison de sa complexité, le problème des performances de généralisation pose un défi, en particulier pour les variations d'aspect d'objet fortement non linéaires ou variant dans le temps. Il existe également de nombreux autres paramètres à gérer. Par exemple, VGG est un réseau de neurones à 16 couches, tandis que le modèle ResNet de Microsoft compte 152 couches. Cependant, le CNN est conçu pour gérer l'apprentissage supervisé, qui dépend de données étiquetées. Un autre problème est qu'elle a surtout besoin de données volumineuses pour la formation. Par exemple, DeepFace [27] a été formé avec 4 millions d'images et FaceNet a été formé avec 200 millions d'images.

D'autre part, l'auto-encodeur est un algorithme d'apprentissage qui examine les données non étiquetées. Il est simple, intuitivement compréhensible et facile à mettre en œuvre. Les auto - encodeurs sont plus généraux car ils ne spécifient rien sur la topologie du réseau. La raison supplémentaire de choisir des auto-codeurs dans ce travail est qu'ils peuvent être empilés les uns sur les autres pour former une méthode d'extraction de caractéristiques et de réduction efficace tout en conservant sa simplicité. Dans ce chapitre, nous fournissons une étude complète de l'auto-encodeur pour la reconnaissance de visage. En plus, une étude comparative de différentes méthodes de Deep Learning et de l'auto-encodeur pour différentes applications sont présentées.

Introduction

L'apprentissage automatique (ou Machine Learning) est un processus qui crée un modèle à partir d'un jeu de données d'entraînement, dans le but, par exemple, de classifier, mesurer ou prendre des décisions sur de nouvelles données. On trouve dans la littérature différents types d'apprentissage automatique : SVM, Perceptron et **Deep Learning**.

Le présent travail traite une approche spécifique d'apprentissage machine, plus spécialement une forme particulière de réseaux de neurones qui sont les Auto-Encodeurs (**AE**). Pour bien assimiler la présentation des résultats et l'originalité de la méthode, les concepts théoriques sous-jacents, assez nombreux, seront présentés ici de façon succincte.

Il existe différents types d'architectures de réseaux profonds non supervisés qui sont : les réseaux de croyances profondes (DBN), les réseaux d'interrogation générative (GAN) et les **Auto-encodeurs**. Nous nous intéressons aux auto-encodeurs et à ce niveau du travail une question se pose : **qu'est-ce qu'un auto-encodeur ?**

Le but d'un auto-encodeur est de produire une approximation de l'entrée en se concentrant uniquement sur les caractéristiques essentielles. En fait, un auto-encodeur est un ensemble de contraintes qui obligent le réseau à apprendre de nouvelles façons de représenter les données, ce qui diffère de la simple copie de la sortie.

Un auto-encodeur typique est défini avec une entrée, une représentation interne et une sortie (une approximation de l'entrée). L'apprentissage se produit dans les couches attachées à la représentation interne. En fait, il y a deux blocs principaux de couches qui ressemblent à un réseau de neurones traditionnel. La légère **différence** est que **la couche contenant la sortie doit être égale à l'entrée**. L'entrée d'origine entre dans le premier bloc appelé **encodeur**. Cette représentation interne comprime (réduit) la taille de l'entrée. Dans le deuxième bloc se produit la reconstruction de l'entrée. C'est la phase de décodage.

3.1 Conception globale de l'approche AEC pour la reconnaissance de visage

Un système de reconnaissance faciale est une technologie capable d'identifier ou de vérifier une personne à partir d'une image numérique ou d'une image vidéo à partir d'une source vidéo. Les systèmes de reconnaissance faciale fonctionnent avec plusieurs méthodes, mais en général, ils comparent des caractéristiques sélectionnées d'une image donnée à des visages dans une base de données. Il est également décrit comme une application basée sur l'intelligence artificielle biométrique qui peut identifier une personne de manière unique en analysant des motifs basés sur les textures et la forme du visage de la personne.

L'apprentissage est la phase ou le modèle apprend à partir des données d'apprentissage dont les caractéristiques du réseau sont modifiées jusqu'à ce que le comportement désiré soit obtenu. La phase test est menée grâce à des images réduites par l'auto-encodeur dans lequel ils sont récupérés de l'AE et qui ne figure pas dans la liste des échantillons utilisée pour l'apprentissage du modèle.

Chaque phase se présente sous forme de quatre modules essentiels : module de détection, module de prétraitement, module d'extraction et réduction de caractéristiques basé sur l'AE et le dernier module qui est la classification. Le modèle ainsi conçu est optimisé par des fonctions d'optimisation.

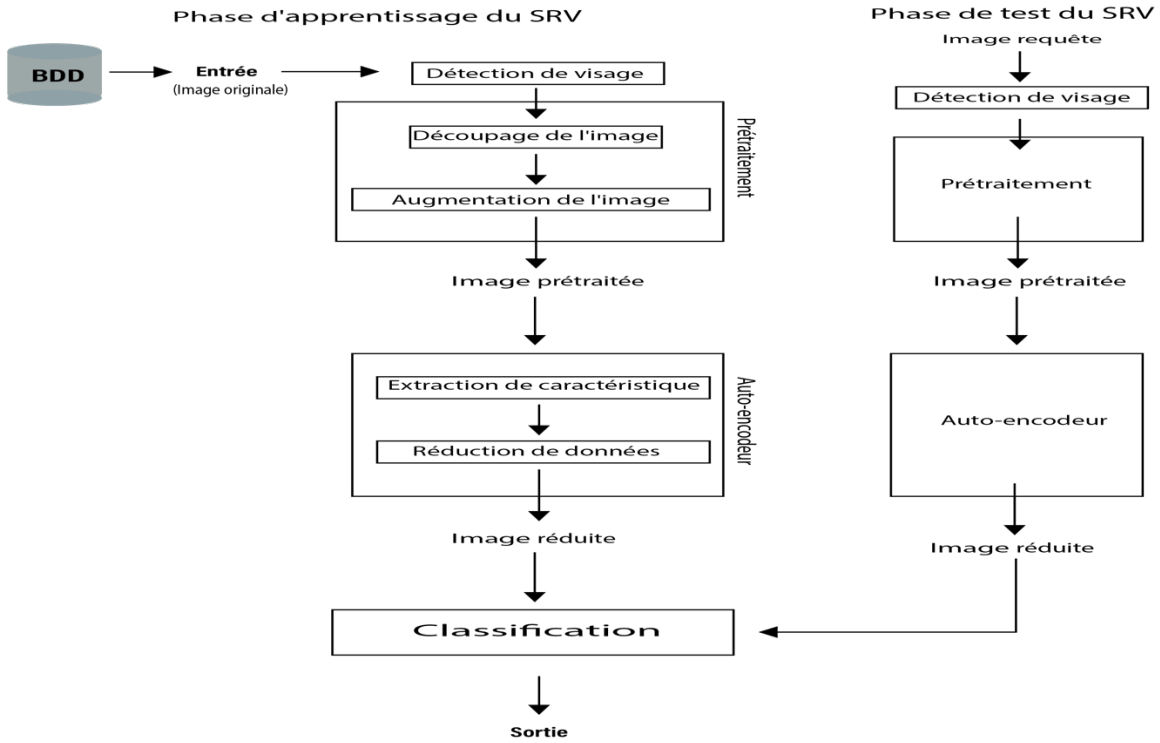


Figure 3. 1 Schéma représentatif de conception du système SRV

Module 1 détection de visage : La détection de visage est une technologie informatique utilisée dans diverses applications pour identifier les visages humains sur des images numériques. La détection des visages fait également référence au processus psychologique par lequel les humains localisent et surveillent les visages dans une scène visuelle.

Dans notre cas, la détection de visage est réalisée à l'aide de classificateurs en cascade basés sur les fonctionnalités de Haar. C'est une méthode efficace de détection proposée par Paul Viola et Michael Jones dans leur article [40] intitulé "Détection rapide d'objet à l'aide d'une cascade renforcée de fonctions simples" La fonction cascade est formée à partir de nombreuses images positives et négatives. Il est ensuite utilisé pour détecter des objets dans d'autres images.

Module 2 Phase de prétraitement : L'auto-encodeur est généralement utilisée dans la reconnaissance faciale pour extraire des caractéristiques, mais il est facilement affecté par la luminosité, l'expression du visage et d'autres raisons. Donc, avant d'extraire des caractéristiques, nous pouvons prétraiter les images de visage pour améliorer le taux de reconnaissance des visages.

Module 3 Extraction et réduction de caractéristiques basé sur l'AE : Un auto-encodeur est un excellent outil pour recréer une entrée. En un mot, la machine prend, disons une image, et peut produire une image étroitement liée. L'entrée dans ce type de réseau de neurones est non étiquetée, ce qui signifie que le réseau est capable d'apprendre sans supervision. Plus précisément, l'entrée est codée par le réseau pour se concentrer uniquement sur la fonctionnalité la plus critique. C'est l'une des raisons pour lesquelles l'auto-encodeur est populaire pour **l'extraction des caractéristiques** et la **réduction de la dimensionnalité**. De plus, les auto-encodeurs peuvent être utilisés pour produire **des modèles d'apprentissage génératifs**. Par exemple, le réseau de neurones peut être entraîné avec un ensemble de visages et peut ensuite produire de nouveaux visages. L'auto-encodeur est donc :

- Algorithme d'apprentissage machine profond et non supervisé. L'auto-encodeur n'utilise aucune donnée étiquetée.
- Réseau de neurones dirigé
- Apprentissage d'une représentation de dimension inférieure de l'entité en entrée

Module 4 Classification : La classification permet de prédire à quelle classe un élément appartient. Certains classificateurs sont binaires, ce qui entraîne une décision oui / non. D'autres sont multi-classes, capables de classer un article dans l'une de plusieurs catégories. La classification est un cas d'utilisation très courant de l'apprentissage machine : les algorithmes de classification sont utilisés pour résoudre des problèmes tels que le filtrage du courrier électronique, la catégorisation de documents, la reconnaissance vocale, la reconnaissance d'images et la reconnaissance de l'écriture manuscrite. Dans ce contexte, un réseau de neurones est l'un des algorithmes d'apprentissage automatique pouvant aider à résoudre les problèmes de classification. Sa force unique réside dans sa capacité à créer de manière dynamique des fonctions de prédiction complexes et à imiter la pensée humaine, comme aucun autre algorithme ne le peut. Il existe de nombreux problèmes de classification pour lesquels les réseaux de neurones ont donné les meilleurs résultats. Dans notre cas nous nous intéressons à la classification des images faciales en utilisant l'apprentissage en profondeur et un réseau de neurone convolutionnel (VGGFACE) qui contient des poids pré-entraînés et 22 couches et qui sert à extraire le vecteur caractéristique de l'image, pour la classification des images deux

métriques sont employées qui sont la distance euclidienne et la similarité du cosinus. Les images réduites et générées par l'AE sont utilisées pour la classification au lieu des images originales.

3.2 Détection et prétraitement du visage

La détection des visages est la première étape essentielle de la reconnaissance des visages. Elle sert à détecter les visages dans les images. Il fait partie de la détection d'objets et peut être utilisé pour détecter les visages en temps réel pour la surveillance et le suivi de personnes ou d'objets.

Le découpage, ou recadrage, est l'opération qui consiste à supprimer une partie périphérique d'une image dans le but de l'améliorer (meilleur cadrage, accentuation du sujet, suppression de défauts ou du coins sombres, etc.), de l'adapter à un usage autre que celui pour lequel elle a été réalisée, ou de modifier son format.

L'augmentation des données d'image est une technique qui peut être utilisée pour agrandir artificiellement la taille d'un jeu de données d'apprentissage en créant des versions modifiées des images du jeu de données. La formation de modèles de réseaux neuronaux d'apprentissage en profondeur sur plus de données peut donner lieu à des modèles plus performant et les techniques d'augmentation peuvent créer des variations des images susceptibles d'améliorer la capacité des modèles d'ajustement à généraliser ce qu'ils ont appris à de nouvelles images.

3.3 Extraction et réduction des données par l AE

3.3.1 Principe des Auto-Encodeurs (AE)

L'**auto-encodeur** est un outil permettant l'extraction de caractéristiques "pertinentes" à l'entrée de façon non supervise et leur réduction.

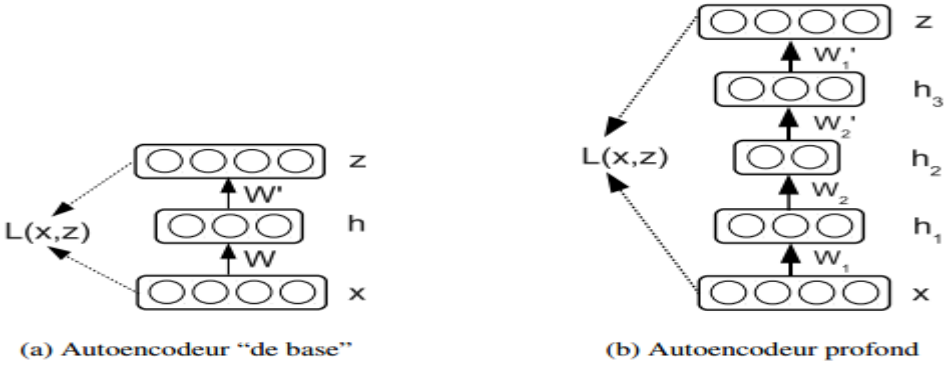


Figure 3. 2 Représentations d'un auto-encodeur ordinaire (a) et profond (b) [24]

Dans le cas de l'auto-encodeur profond, c'est ici un exemple où deux couches seraient pré-entraînées avant de former l'auto-encodeur profond, donnant des poids liés W_1 et $W_1^{'}$ ainsi que W_2 et $W_2^{'}$. L'idée est simple mais ingénieuse : on entraînera un réseau de neurones à une couche cachée h à prédire en sa sortie z son entrée x, une tâche qu'on nomme **reconstruction** voir **figure 3.2**. Le critère sera donc la minimisation d'une **erreur de reconstruction** $L(x,z)$. En principe, la couche cachée devra donc contenir de l'information pertinente à la reconstruction. Par exemple, si la couche cachée contient moins d'unités qu'il y'a d'entrées, alors logiquement pour bien reconstruire elle doit apprendre à "résumer" l'entrée, puisque l'information pour la reconstruction est entièrement contenue dans la couche cachée. Ainsi des caractéristiques "pertinentes" à la reconstruction devront être extraites.

Il faut noter que les paramètres pour passer de h à z, $W^{'}$, ont une forme matricielle qui est la transposée de la forme des paramètres W. En pratique ça signifie maintenir deux matrices distinctes en mémoire. Il est aussi possible d'effectuer la reconstruction en faisant passer l'information dans plusieurs couches cachées $\{h_1,\ldots,h_n\}$. On obtient ainsi un **auto-encodeur** profond. Comme ce type de réseau contient plusieurs couches, il sera en général préférable de pré-entraîner les couches et de les faire correspondre par paires : la première et la dernière, la seconde et l'avant dernière, etc., comme on peut voir dans la **figure 3.2** par les correspondances entre les noms des paramètres de chacune des couches.

Par contre, dans cette version "de base" de l'auto-encodeur, si le nombre d'unités dans la couche cachée est plus grand ou égal au nombre d'entrées, le modèle peut alors apprendre de façon triviale la fonction identité. C'est facile en copiant une à une les valeurs de l'entrée dans la couche cachée, par exemple. On n'obtient donc pas de caractéristiques extraites intéressantes. Pour contourner ce problème, différentes contraintes peuvent être imposées durant l'apprentissage. C'est ce qui est fait avec l'auto-encodeur débruitant. Pour des modèles effectuant de la reconstruction, on peut évaluer l'erreur de reconstruction, la différence entre l'entrée originale et l'entrée reconstruite. Cela permet de comparer les modèles entre eux. Pour le faire numériquement, dans notre travail, la MSE ou la MAE seront utilisées. Ces coûts serviront pour l'évaluation des modèles et comme fonctions objectives pour guider l'entraînement.

Les auto-encodeurs sont des modèles d'***apprentissage non supervisé*** qui vont nous permettre de comprendre dans un premier temps la notion de ***variables latentes***.

Considérons par exemple une collection d'images de visages humains d'une résolution de 1000 * 1000 pixels et posons-nous la question de savoir combien de paramètres sont nécessaires au minimum pour décrire correctement un visage en particulier. L'âge, le sexe, la coupe de

cheveux, le teint, l'aspect souriant ou non, … sont probablement indispensables mais, de tout évidence, insuffisants. Peut-être quelques dizaines de paramètres suffiront-ils si à condition d'accepter qu'ils n'aient pas tous une interprétation directe évidente. On appelle ces paramètres des variables latentes. La **figure 3.3** illustre la situation : parmi toutes les images carrées d'un million de pixels, seule une infime fraction correspondent effectivement à des visages humains.

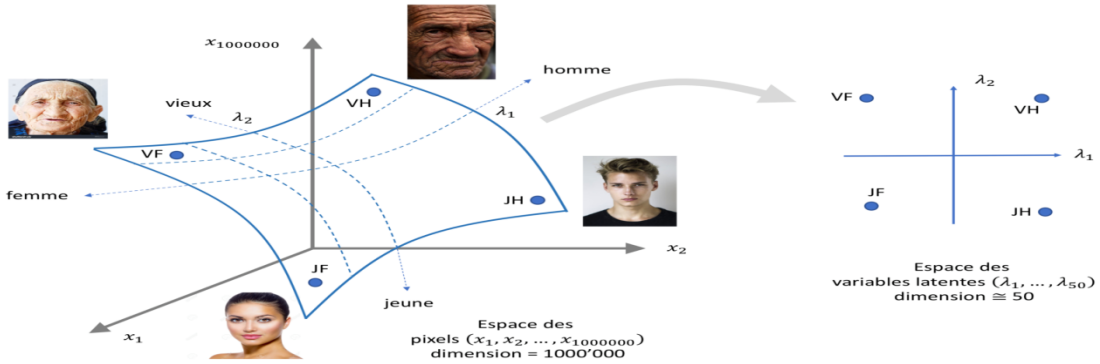


Figure 3. 3 Espace des images est de dimension 1000'000 [39]

Les images de visages constituent une infime partie de cet espace qui correspond à une surface de dimension beaucoup plus faible, peut-être 50. Les coordonnées sur cette surface de faible dimension sont les variables latentes.

Les 50 variables latentes $(\lambda_1, \lambda_2, \ldots, \lambda_{50})$ peuvent être envisagées comme une ***forme compressée*** de l'information visuelle. A, chaque point dans l'espace des variables latentes correspond un visage, si bien que parcourir cet espace latent revient à parcourir l'ensemble des visages humains. Les auto-encodeurs sont des systèmes d'apprentissage non supervisé qui permettent :

1. De **construire un espace de variables latentes**. Les différentes variables latentes que l'algorithme découvre n'ont cependant pas vocation à être directement interprétables, contrairement à celles de la **figure 3.3**.
2. De **reconstruire une image** à partir d'un point dans l'espace latent.

Leur principe de fonctionnement est illustré dans la **figure 3.4**, il utilise un réseau de neurones symétriques. Seules les connexions entre les 3 premières couches sont représentées, les autres étant symétriques.

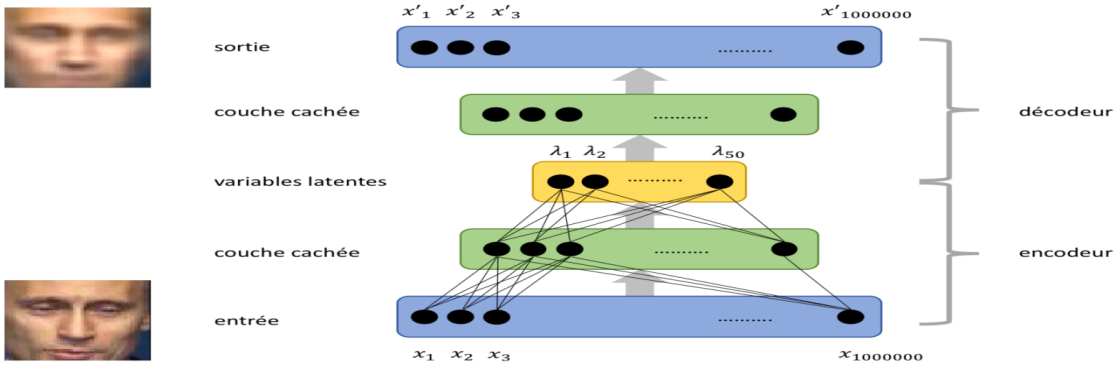


Figure 3. 4 Principe de l'auto-encodeur [39]

Un AE est un réseau de neurones constitué de plusieurs couches connectées de manière symétrique comme l'illustre la **figure 3.4**. Les couches d'entrée et de sortie sont identiques et sont celles qui possèdent le plus grand nombre de neurones, pour une image en noir et blanc chaque pixel correspond à un neurone. Le nombre de neurones de la couche centrale correspondant au nombre de variables latentes que l'on souhaite découvrir. Plus les nombres de couches et de neurones sont importants, plus l'auto-encodeur sera capable d'apprendre à encoder des images complexes. Ces nombres sont des hyper paramètres qu'il faudra ajuster par tâtonnements pour parvenir à une reconstruction optimale sans trop de ressources.

Durant l'***entraînement*** de l'AE on lui présente une liste d'images en entrée et on lui demande d'apprendre à les reconstruire aussi précisément que possible sur la couche de sortie en minimisant une ***erreur de reconstruction*** qui évalue la différence entre l'image originale et l'image reconstruite. Les couches comprises entre l'entrée et la couche de variables latentes sont alors à envisager comme un ***encodeur*** et les couches entre les variables latentes et la sortie comme un ***décodeur***. En d'autres termes, on demande à l'encodeur de compresser l'information pour la faire tenir dans un petit nombre de variables latentes et au décodeur de reconstruire cette information du mieux qu'il peut. Sur les données qu'on lui fournit l'AE ne fait donc globalement… rien ! Cette opération en apparence triviale ne l'est pas en réalité car le faible nombre de neurones de la couche centrale constitue un goulet d'étranglement qui force l'AE à trouver une représentation parcimonieuse des données qu'on lui a présenté. Il faut bien réaliser que ce mécanisme de compression ne fonctionnera correctement que pour des données de même nature que celles avec lesquelles on a entrainé l'AE. Si on présente une image de camion plutôt qu'un visage à l'AE, l'encodage n'aura aucun sens et la reconstruction sera impossible car l'AE cherchera désespérément à interpréter un camion comme un visage !

3.3.2 Architecture de l'Auto-Encodeur (AEC)

L'architecture de l'auto-encodeur convolutif est illustré par la **figure 3.4** et se présente comme suit : chaque couche est suivie d'une convolution et d'un pooling.

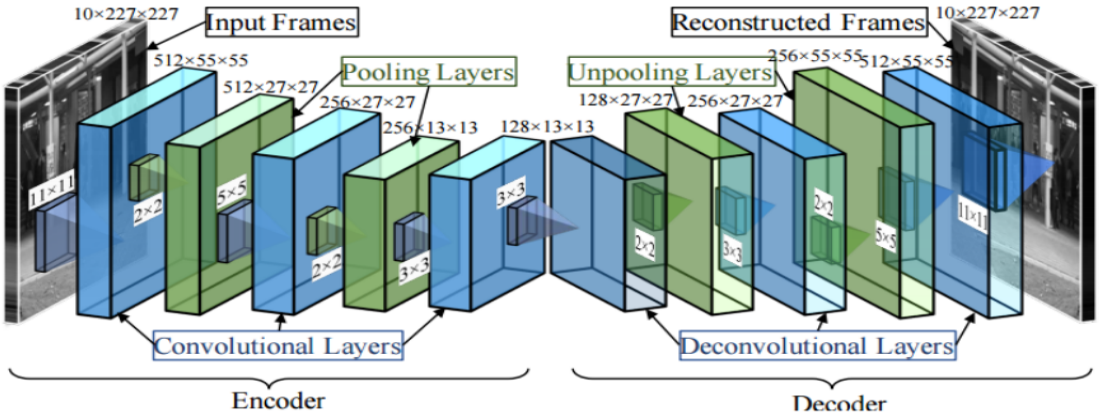
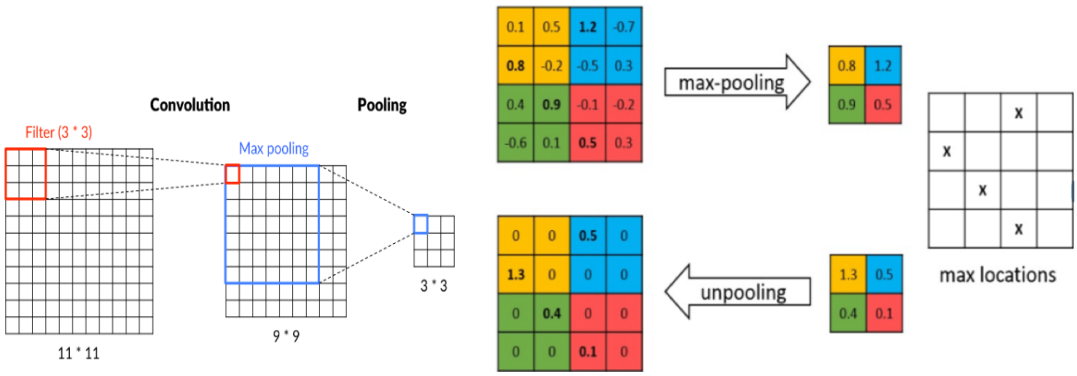


Figure 3. 5 Illustration d'un auto-encodeur convolutif [41]

Dans cet exemple de la **figure 3.5**, l'AE comprend deux couches de convolution et leurs deux couches de déconvolution correspondantes, ainsi que deux couches de mise en commun **(**pooling) "maximal" et leurs couches de "démembrement"(unpooling) correspondantes.

Examinons de plus près deux technologies fondamentales d'apprentissage en profondeur, à savoir la convolution et la mise en commun (pooling). Tout au long de cette section, des images ont été utilisées pour comprendre ces concepts.



(a) Convolution et pooling (b) pooling /unpooling

Figure 3. 6 Principe d'une couche convolution et pooling [42]

Pour chaque couche de pooling, les emplacements maximaux sont stockés comme le montre la **figure 3.6.b**. Ces emplacements sont ensuite utilisés dans la couche unpooling. Ce dernier n'est qu'une simple mise à l'échelle de l'image en la redimensionnant, donc rien d'intelligent. Afin de construire des réseaux de neurones profonds, une modification du fonctionnement convolutionnel de base s'impose, voyons comment cela fonctionne.

3.3.3 Extraction des caractéristiques par les filtres convolutifs

- **Opération de convolution :**

  La convolution de deux fonctions [43], f (t) et g (t), est donnée par :

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau \qquad (3.1)$$

Dans le cas discret, ceci est donné par :

$$(f * g)(n) = \sum_{m=-\infty}^{\infty} f(m)g(n-m) \quad (3.2)$$

Notons cependant qu'en général, les auto-encodeurs n'utilisent pas la convolution, mais utilisent plutôt une corrélation croisée. Pour les fonctions à valeurs réelles, la corrélation croisée est définie par :

$$(f * g)(n) = \sum_{m=-\infty}^{\infty} f(m)g(n+m) \quad (3.3)$$

Cette opération est appelée convolution.

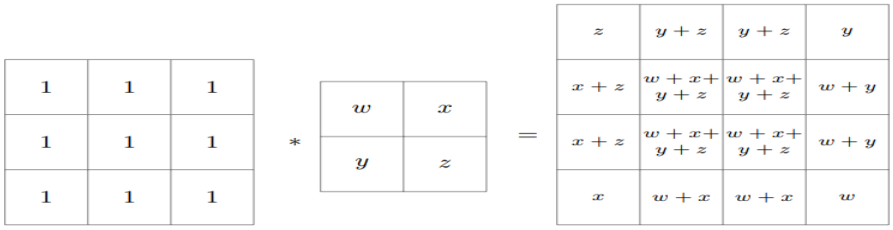
La convolution 2D est donnée par :

$$(f * g)(i,j) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(m,n)g(i+m, j+n) \quad (3.4)$$

Cela se généralise également aux dimensions supérieures. Notons aussi que ces “convolutions” ne sont pas commutatives.

Exemple : Notez que nous supposons qu’en dehors de chaque grille sont des valeurs nulles (qui ne sont pas dessinées). Maintenant, en faisant glisser la ligne du haut, nous obtenons **figure 3.7.a** :

(a) 1ère ligne (b) 2ème ligne



(c) Résultat final de la convolution

Figure 3. 7 Illustration du principe du filtre de convolution [43]

- **<u>Couche convolutive :</u>** On définit la "couche de convolution" comme une opération de convolution (généralement en 3D, car les images ont souvent une largeur, une hauteur et une profondeur pour les canaux R, V, B). La couche de convolution définit une collection de filtres (ou cartes d'activation), chacune ayant la même dimension que l'entrée. Soient (w , h, d) dimensionnalité de l'entrée, (w_f, h_f, d) dimensionnalité du filtre que le filtre

fonctionne sur une petite région de l'entrée $w_f < w$. Les profondeurs étant égales signifiant que le résultat de cette opération de convolution est 2D.

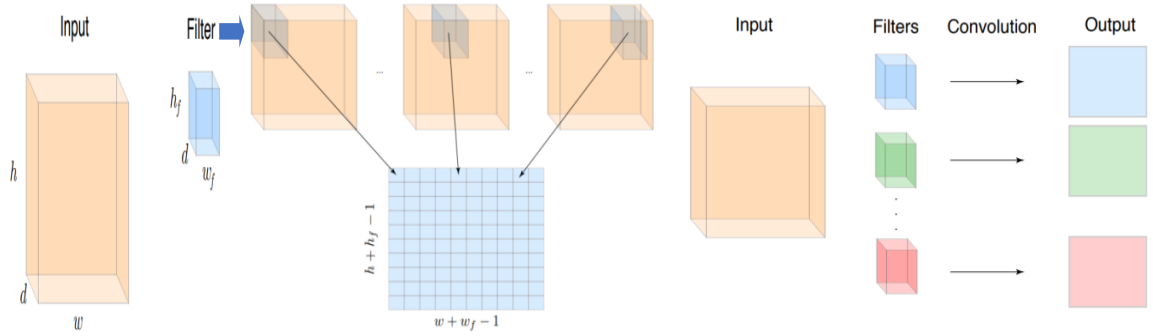
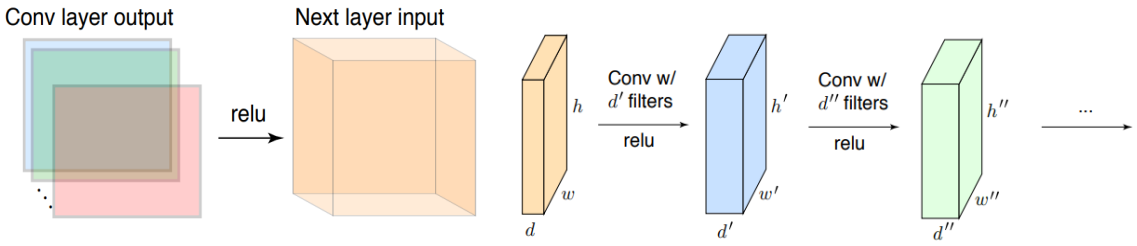


Figure 3. 8 Principe du filtre convolutif [43]

Après avoir effectué la convolution, la taille de la sortie est $(w + w_f - 1, h + h_f - 1)$. Maintenant, nous n'avons pas un seul filtre dans une couche de convolution, mais plusieurs filtres. Nous appelons chaque sortie (matrice) une tranche. Les tranches de sortie des opérations de convolution avec chaque filtre sont composées ensemble pour former un tenseur $(w + w_f - 1, h + h_f - 1, n_f)$, où n_f est le nombre de filtres. La sortie passe ensuite par une non-linéarité d'activation, telle que ReLU $(\cdot)$. Ceci agit alors comme entrée dans la couche suivante.



(a) Une couche **(b)** Plusieurs couches

Figure 3. 9 Illustration de couche(s) convolutive(s) [43]

Ces couches peuvent ensuite être composées, lesquelles comprennent une grande partie de l'AE.

- **Les couches de convolution partagent des paramètres :**

Les couches convolutives ont des paramètres partagés (ou poids liés), dans chaque neurone de sortie d'une tranche donnée, utilise le même ensemble de paramètres dans le filtre.

Exemple : considérons une entrée qui est (32 × 32 × 3). Nous avons deux architectures ;

(1) Dans l'architecture entièrement connectée, il y a 500 neurones de sortie dans la première couche.

Le réseau entièrement connecté a (32 × 32 × 3) × 500 = 1 536 000 paramètres.

(2) Dans le réseau neuronal convolutionnel, il y a 4 filtres qui sont tous 4 × 4 × 3. Chaque tranche de sortie est (32 - 4 + 1, 32 - 4 + 1), soit 841 neurones. Avec 4 tranches, il y a 3 364 neurones de sortie dans le CNN.

Le réseau neuronal convolutif a 4 × (4 × 4 × 3) = 192 paramètres.

- **Les couches convolutives peuvent gérer des entrées de taille variable :**

Les réseaux de neurones convolutifs peuvent traiter des entrées de taille ou d'étendue spatiale variable. Dans une couche entièrement connectée, chaque couche spécifie les paramètres (**W, b)** tels que **z = Wx + b**, ce qui permet de définir les échelles d'entrée et de sortie. Dans une couche convolutive, les paramètres sont des filtres convolutifs, qui sont ensuite convolués avec une entrée.

Si la taille de l'entrée change, l'opération de convolution peut toujours être effectuée.

Les dimensions de la sortie seront différentes, ce qui peut convenir en fonction de l'application. Si les dimensions de sortie ont une taille fixe, certaines modifications peuvent être apportées au réseau (par exemple, si la taille de sortie doit être réduite, les couches de mise en commun, décrites plus loin, peuvent être redimensionnées avec la taille d'entrée du réseau pour créer la taille de sortie correcte).

3.3.3.1 Convolution padding

En l'absence de toute modification, chaque couche convolutive augmente la largeur et la hauteur des données de chaque couche de la longueur du filtre moins 1. Dans la pratique, la convolution est parfois limitée aux régions où le filtre recouvre entièrement l'entrée.

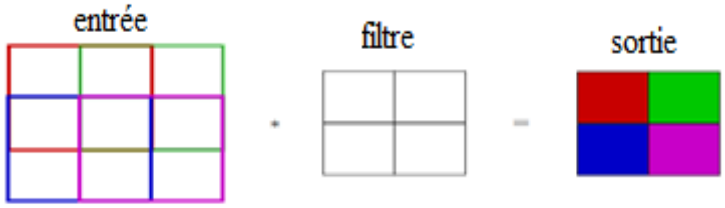


Figure 3. 10 Exemple de convolution sans padding

Cependant, nous nous heurtons maintenant à un autre problème : si nous limitons le noyau aux zones où il se chevauche complètement, le résultat obtenu sera de la dimension suivante $[w - w_f + 1, h - h_f + 1]$. Ainsi, les données d'entrée limitent le nombre de couches convolutives pouvant être utilisées. Pour le reste de la classe, nous supposerons que la convolution n'est appliquée que lorsque le filtre chevauche complètement l'entrée. Maintenant, l'entrée et la sortie resteront de la même taille si l'entrée est complétée de zéros avec un total de zéros $w_f - 1$ (c'est-à-dire, $(w_f - 1)/2$ zéros à gauche et à droite de la matrice) et la hauteur est remplie de zéros avec h_f - 1 zéros au total (c'est-à-dire $(h_f$ - 1) / 2 zéros au haut et au bas

de la matrice). Habituellement, nous spécifions un pavé variable qui correspond à la quantité de zéros à paver sur chaque bordure.

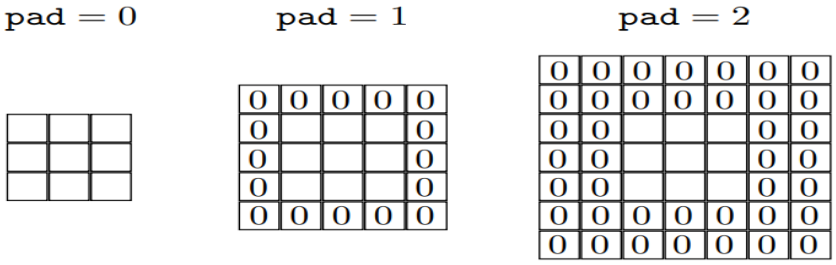


Figure 3. 11 Exemple de padding

La sortie de la convolution est maintenant $(w - w_f + 1 + 2pad, h - h_f + 1 + 2pad)$.

Il convient de noter que la quantité optimale de remplissage nul (en termes de précision du test) se situe quelque part entre pad = 0 et le pad qui fait en sorte que la sortie et l'entrée aient la même largeur et la même hauteur.

3.3.3.2 Stride (pas) convolution

Une autre variable à contrôler est la foulée (**stride**), qui définit le déplacement du filtre dans la convolution. Pour une convolution normale, stride = 1, ce qui indique que le filtre est déplacé à travers toutes les parties de l'entrée. Considérons une entrée 7 × 7avec un filtre 3 × 3. Si la convolution n'est appliquée que lorsque le noyau chevauche l'entrée, la sortie est 5 × 5. Avec stride = 2, la sortie est 3 × 3.

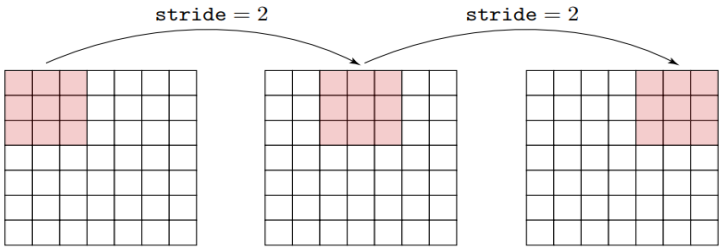


Figure 3. 12 Exemple sur le stride

Dans l'exemple de l'entrée 7 × 7 avec filtre 3 × 3, il n'est pas correct d'utiliser stride = 3 car ce n'est pas cohérent avec l'entrée. Cependant, il est possible d'appliquer des zéros sur l'entrée afin que la foulée soit appropriée. La taille de sortie après comptabilisation de la foulée et du remplissage est la suivante :

$$\left(\frac{w - w_f + 2pad}{stride} + 1, \frac{h - h_f + 2pad}{stride} + 1\right) \qquad (3.5)$$

- **Taille de sortie de la couche convolutive**

La hauteur et la largeur de sortie d'une couche convolutive sont données par :

*Size=L * H = ((Taille d'entrée - (Taille du filtre) + 2 * rembourrage) / Pas + 1.*

La formule pour la taille spatiale du volume de sortie est donc :

$$\textbf{Size} = \textbf{K} * ((\textbf{W} - \textbf{F} + 2\textbf{P}) / \textbf{S} + 1) \quad (3.6)$$

où **W** : taille du volume d'entrée, **F** : taille du filtre (champ récepteur des neurones de la couche de convolution), **S** : stride (foulée) avec laquelle ils sont appliqués, **P** : padding (la quantité de remplissage nul utilisé sur la bordure), **K** : nombre de filtres (la profondeur de la couche de convolution).

Cette valeur de la taille de la sortie doit être un entier pour que l'image soit entièrement couverte. Si la combinaison de ces options n'entraîne pas une couverture complète de l'image, le logiciel ignore par défaut la partie restante de l'image située le long des bords droit et inférieur de la convolution.

- **Nombre de neurones de la couche convolutive**

La taille de la carte qui est le produit de la hauteur et de la largeur de sortie nous donne le nombre total de neurones dans une carte de caractéristiques (map features).

Le ***nombre total de neurones*** (taille de sortie) dans une couche de convolution est égal à (***Taille de la carte * Nombre de filtres***).

Par exemple, supposons que l'image d'entrée soit une image couleur 32 sur 32 sur 3. Pour une couche convolutive avec huit filtres et une taille de filtre de 5 sur 5, le nombre de poids par filtre est 5 * 5 * 3 = 75, et le nombre total de paramètres de la couche est égal à :

(75 + 1) * 8 = 608. Si la foulée (padding) est de 2 dans chaque direction et qu'un remplissage (stride) de taille 2 est spécifié, chaque carte de caractéristiques est au format 16 sur 16. Cela est dû au fait que (32 - 5 + 2 * 2) / 2 + 1 = 16,5 et qu'une partie du remplissage de zéro le plus externe situé à droite et en bas de l'image est ignorée. Enfin, le nombre total de neurones dans la couche est de 16 * 16 * 8 = 2048. Habituellement, les résultats de ces neurones passent par une certaine forme de non-linéarité.

3.3.4 Réduction des caractéristiques par le pooling

Les AE incorporent également des couches de regroupement dans lesquelles une opération est appliquée à tous les éléments de l'étendue de filtrage. Cela correspond effectivement à un sous-échantillonnage. Le filtre de regroupement a une largeur et une hauteur (W_p, H_p) et s'applique avec une foulée donnée. Il est très courant d'utiliser l'opération max () en tant qu'opération de regroupement.

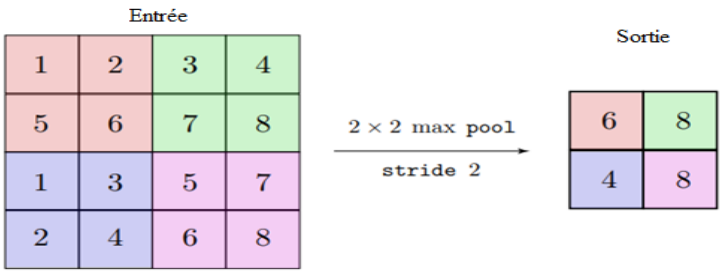


Figure 3. 13 Principe du maxpooling

3.4 Classification pour la reconnaissance de visage

Nous utilisons une classification métrique par calcul de distance pour mesurer la similarité pour la reconnaissance de visage.

- **Distance Euclidienne :** est probablement la métrique de distance la plus utilisée. C'est un cas particulier d'une classe générale de normes et se présente comme la distance entre les vecteurs x et y [44] définie comme suit :

$$d(x,y) = \|x-y\|_e = \sqrt{|x_i - y_i|^2} \qquad (3.10)$$

La distance euclidienne est à la base de nombreuses mesures de similarité et de dissimilarité.

- **Similitude de cosinus (cosine similarity) :** entre deux vecteurs est une mesure qui calcule le cosinus de l'angle entre eux. Cette métrique est une mesure de l'orientation et non de la magnitude. Elle peut être vue comme une comparaison entre des vecteurs (image) situés sur un espace normalisé, car nous ne prenons pas uniquement en considération la magnitude de chaque vecteur mais l'angle entre eux. Ce que nous devons faire pour construire l'équation de similarité cosinus est de résoudre l'équation du produit scalaire pour $\cos\boldsymbol{\theta}$. La similarité des cosinus générera une métrique indiquant la relation entre deux documents en examinant l'angle au lieu de la magnitude [44], comme dans les exemples ci-dessous :

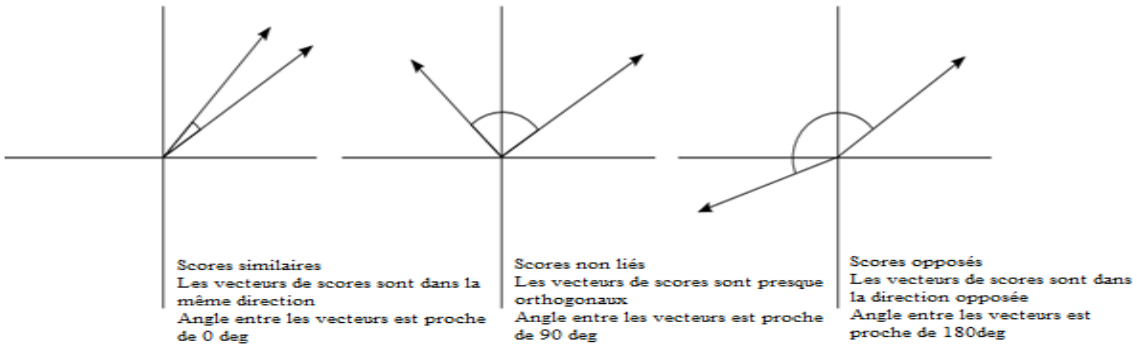


Figure 3. 14 Représentation d'exemple de similarité [44]

La similarité des cosinus [44] est calculée à l'aide de la formule suivante :

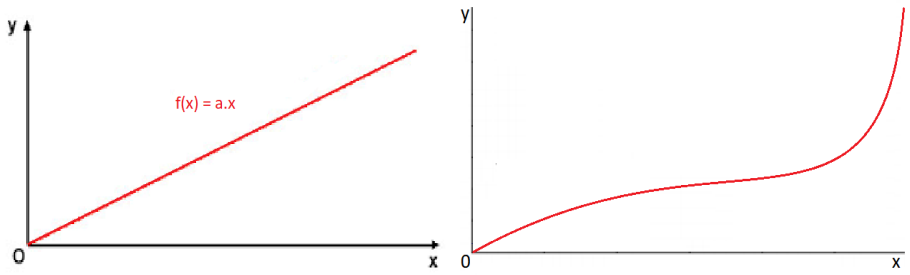
$$similarity(A,B) = \frac{A \cdot B}{||A|| \times ||B||} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}} \quad (3.11)$$

Les valeurs sont comprises entre -1 et 1, -1 étant parfaitement différent et 1 parfaitement similaire.

3.5 Optimisation du modèle

3.5.1 Optimisation par les fonctions d'activation

Les réseaux de neurones artificiels et particulièrement l'AE nécessitent une fonction 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. Un exemple de fonction d'activation est la fonction de Heaviside, qui renvoie tout le temps 1 si le signal en entrée est positif, ou 0 s'il est négatif dans notre travail nous avons utilisé la fonction ReLu, Sigmoide ainsi que Softmax. L'activation a pour but de transformer le signal de manière à obtenir une valeur de sortie à partir de transformations complexes entre les entrées. Pour ce faire la fonction d'activation doit être non linéaire. C'est cette non-linéarité qui permet de créer de telles transformations. Citons brièvement la différence entre une fonction d'activation linéaire et non linéaire.



(a) **(b)**

Figure 3. 15 Exemple de fonction (a) linéaire et (b) non linéaire

- **Fonction d'activation linéaire :** C'est une fonction simple de la forme f(x) = a*x où f(x) = x (**figure 3.15 (a)**). En gros, l'entrée passe à la sortie sans une très grande modification ou alors sans aucune modification. On reste ici dans une situation de proportionnalité.
- **Fonction d'activation non linéaire :** Les situations de proportionnalité sont généralement des cas particuliers. Les fonctions non linéaires (**figure 3.15 (b)**)

permettent de séparer les données non linéairement séparables. Elles constituent les fonctions d'activation les plus utilisées. L'utilisation de fonctions d'activation non linéaires est tout simplement indispensable pour la simple et bonne raison que les *fonctions linéaires* ne fonctionnent qu'avec une *seule couche de neurone*. Car au-delà d'une couche de neurones, l'application récurrente d'une même fonction d'activation linéaire n'aura plus aucun impact sur le résultat. Autrement dit, afin de résoudre des problèmes complexes, l'utilisation de fonctions non linéaires est obligatoire. Autre point très important, les données traitées par les neurones peuvent atteindre des valeurs très grandes. L'utilisation d'une fonction linéaire, ne modifiant pas la sortie, les valeurs des données transmises de neurones en neurones peuvent devenir de plus en plus grandes et rendant les calculs beaucoup plus complexes. Afin d'y remédier, les fonctions d'activation non linéaires réduisent la valeur de sortie d'un neurone le plus souvent sous forme d'une simple probabilité. Étudions maintenant les fonctions d'activation les plus utilisées.

3.5.1.1 Fonction Sigmoïde

La fonction Sigmoïde prend une valeur réelle en entrée et génère une autre valeur comprise entre 0 et 1. Elle est facile à utiliser et présente toutes les propriétés intéressantes des fonctions d'activation : non linéaire, continuellement différentiable, monotone et plage de sortie fixe.

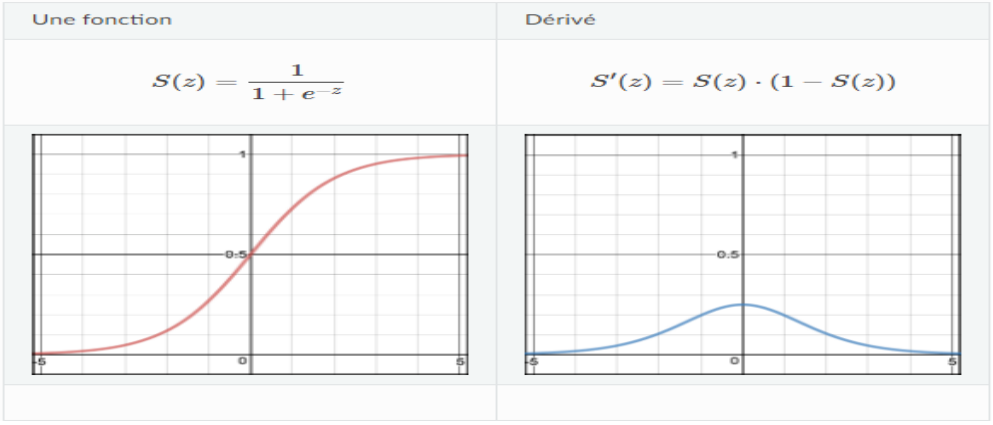


Figure 3. 16 Principe de la fonction Sigmoïde [45]

Avantages

- Non linéarité ;
- Activation analogique ;
- Bon pour un classificateur ;
- La sortie de la fonction d'activation sera toujours dans la plage (0 , 1) par rapport à (-inf , +inf) de la fonction linéaire.

Inconvénients

- Sa sortie n'est pas centrée à zéro (0 <sortie <1), ce qui rend l'optimisation plus difficile.
- Le réseau refuse d'approfondir ses connaissances ou est extrêmement lent (le calcul n'est pas affecté par les limites des valeurs en virgule flottante).

3.5.1.2 Fonction ReLU

Une invention récente qui représente les unités linéaires rectifiées. La formule est simple : max (0,z). Malgré son nom et son apparence, elle n'est pas linéaire et offre les mêmes avantages que Sigmoïde, mais avec de meilleures performances.

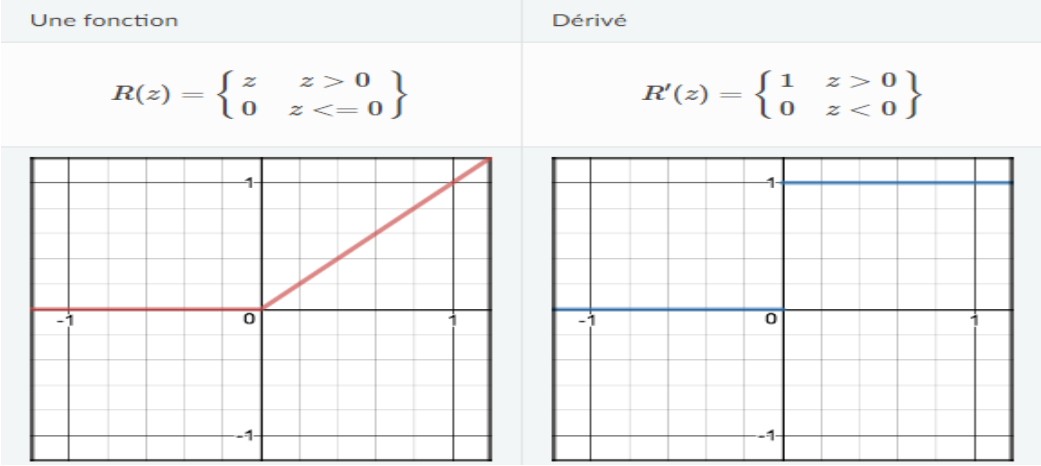


Figure 3. 17 Principe de la fonction ReLu [44]

Avantages

- **ReLu** est moins coûteux en calcul que **tanh** et **sigmoid**, car il implique des opérations mathématiques plus simples.

Inconvénients

- L'une de ses limites est qu'il ne doit être utilisé que dans les couches cachées d'un modèle de réseau neuronal.
- ReLu pourrait entraîner la mort de neurones.
- La plage de ReLu est [0, inf[. Cela signifie problème d'activation.

3.5.1.3 Fonction Softmax

La fonction Softmax (voir **figure 3.18**) calcule la distribution des probabilités de l'événement sur '**n**'différents événements. En règle générale, cette fonction calcule les probabilités de chaque classe cible sur toutes les classes cibles possibles. Plus tard, les probabilités calculées seront utiles pour déterminer la classe cible pour les entrées données.

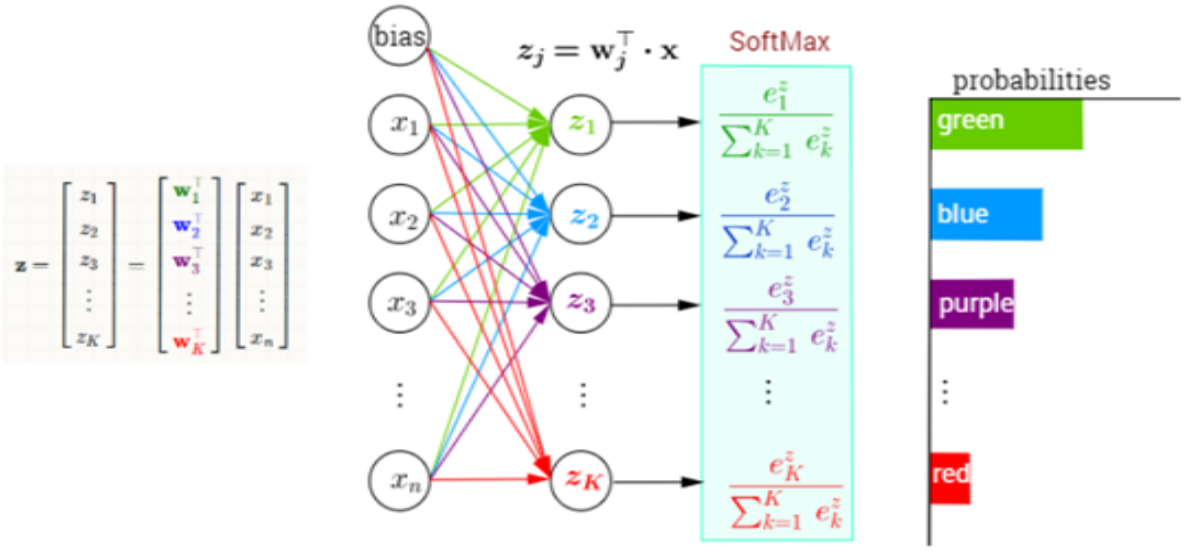


Figure 3. 18 Exemple de classification avec Softmax [46]

3.5.2 Optimisation de fonctions de perte et de précision

Les fonctions de perte fournissent plus qu'une représentation statique de la performance du modèle, elles permettent également aux algorithmes de mieux adapter les données. La plupart des algorithmes d'apprentissage automatique utilisent une sorte de fonction de perte dans le processus d'optimisation ou de recherche des meilleurs paramètres (poids) de données. Il existe différentes variétés de fonctions de perte pour différents problèmes, tout comme différents optimiseurs. La fonction de perte et l'optimiseur fonctionnent en parallèle pour adapter l'algorithme aux données de la meilleure façon possible.

3.5.2.1 Algorithmes d'optimisation de la précision du modèle

- **Descente de Gradient :** est un algorithme d'optimisation utilisé pour minimiser une fonction en se déplaçant de manière itérative dans la direction de la descente la plus raide, définie par le négatif du gradient.

En apprentissage machine, on utilise la descente de gradient pour mettre à jour les paramètres de notre modèle. Les paramètres font référence aux coefficients de la régression linéaire et aux poids dans les réseaux de neurones. Le processus se poursuit de manière itérative jusqu'au bas du graphique ou à un point où l'on ne peut plus descendre, un minimum global.

La descente de gradient utilise la dérivée partielle de la fonction de perte ou d'erreur afin de propager les mises à jour aux poids de neurones. Afin de trouver le gradient pour le neurone de sortie, nous devons faire la dérivée partielle de la fonction d'activation utilisée. La **figure 3.19** montre comment la méthode de descente de gradient recule dans la dérivée afin de trouver le minimum.

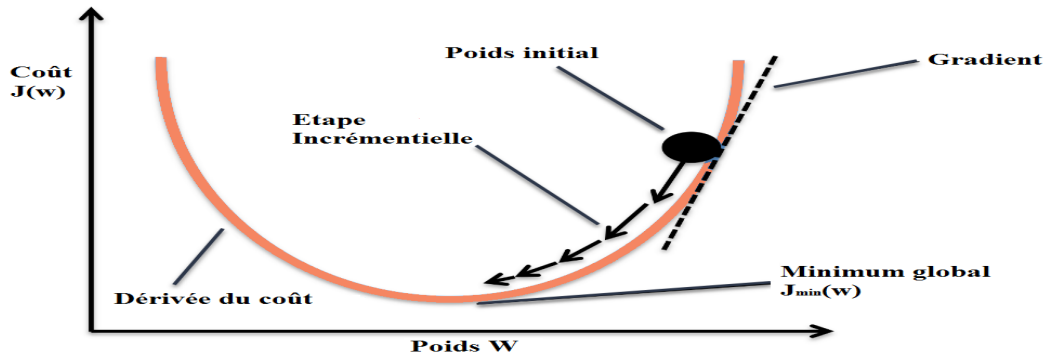
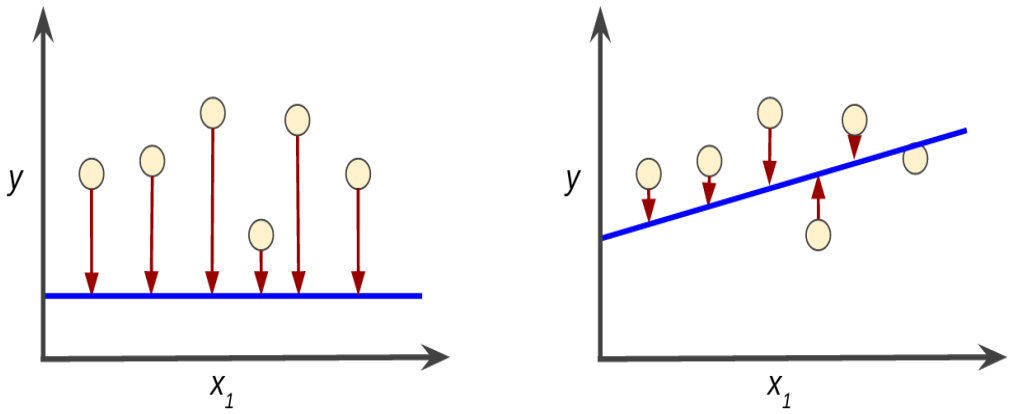


Figure 3. 19 Recherche du coût minimum par la descente du gradient [47]

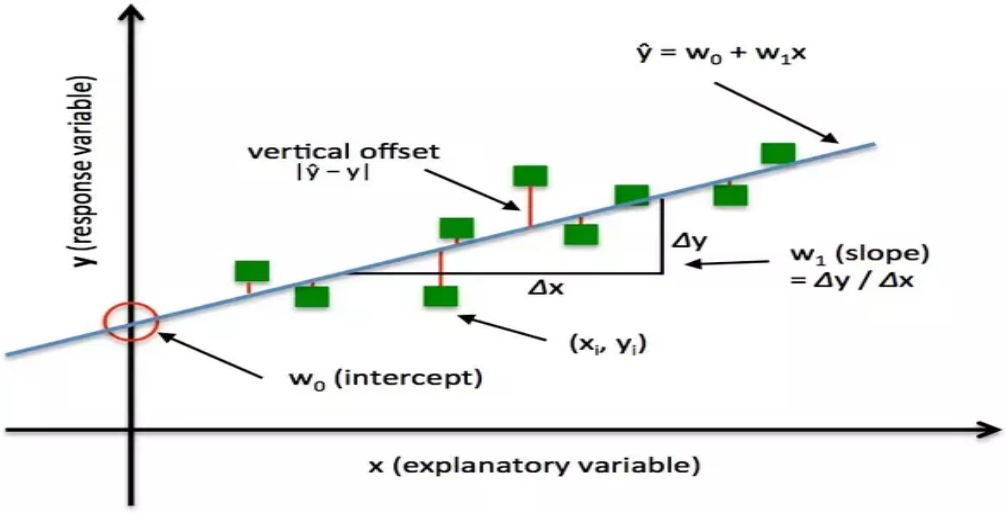
- **RMSProp :** Une autre façon d'améliorer les performances de la descente de gradient consiste à appliquer la stratégie RMSProp - Root Mean Squared Propagation - qui est l'un des optimiseurs les plus fréquemment utilisés. Ceci est un autre algorithme qui utilise des moyennes pondérées de manière exponentielle. De plus, il est adaptatif - il permet un ajustement individuel du taux d'apprentissage pour chaque paramètre du modèle. Les valeurs de paramètre suivantes sont basées sur les valeurs de gradient précédentes calculées pour un paramètre particulier.
- **Adam :** C'est un algorithme qui, comme RMSProp, fonctionne bien dans un large éventail d'applications. Il tire parti des plus grands avantages de RMSProp et les combine avec des idées issues de l'optimisation de momentum. Le résultat est une stratégie qui permet une optimisation rapide et efficace.

3.5.2.2 Algorithmes d'optimisation de perte du modèle

Pour un modèle, l'**apprentissage** signifie déterminer les bonnes valeurs pour toutes les pondérations et le biais à partir d'exemples étiquetés. Dans l'apprentissage supervisé ou non supervisée, un algorithme de Machine Learning crée un modèle en examinant de nombreux exemples, puis en tentant de trouver un modèle qui minimise la perte. Ce processus est appelé **minimisation du risque empirique**. La perte [48] correspond à la pénalité pour une mauvaise prédiction. Autrement dit, la **perte** est un nombre qui indique la médiocrité de la prévision du modèle pour un exemple donné. Si la prédiction du modèle est parfaite, la perte est nulle. Sinon, la perte est supérieure à zéro. Le but de l'entraînement d'un modèle est de trouver un ensemble de pondérations et de biais pour lesquels la perte, en moyenne sur tous les exemples, est faible. Par exemple, la **figure 3.20** présente à gauche (a) un modèle dont la perte est élevée, et à droite (b) un modèle dont la perte est faible. À noter concernant cette **figure 3.20** que les flèches rouges représentent les pertes et la ligne bleue représente les prédictions.



(a) perte élevée (b) perte faible



(c) Principe de perte (loss)

Figure 3. 20 Définition de perte (loss) [48]

Notez que les flèches rouges dans le graphique de la **figure 3.20 (a)** sont plus longues que celles de l'autre graphique **figure 3.20 (b)**. Il est clair que la ligne bleue dans le modèle de la **figure 3.20 (b)** correspond à un modèle prédictif plus performant que celui représenté dans le graphique de la **figure 3.20 (a)**. Il existe différents calculs de l'erreur :

- L'**erreur quadratique moyenne** (**MSE**) correspond à la perte quadratique moyenne pour chaque exemple. Pour calculer l'erreur MSE, il faut additionner toutes les pertes quadratiques de chaque exemple, puis diviser cette somme par le nombre d'exemples :

$$MSE = \frac{1}{N}\sum_{(x,y)\in D}(y - prediction(x))^2 \quad (3.7)$$

Où (x,y) est un exemple dans lequel x est l'ensemble des caractéristiques de l'image que le modèle utilise pour réaliser des prédictions , y est l'étiquette de l'image, prediction (x) est une fonction des pondérations et biais en combinaison avec l'ensemble des caractéristiques x, D est un ensemble de données contenant de nombreux exemples étiquetés, qui sont des paires (x,y)

et N est le nombre d'exemples dans D. Bien que l'erreur MSE soit couramment utilisée dans le Machine Learning, ce n'est ni la seule fonction de perte pratique, ni la meilleure fonction de perte pour toutes les circonstances.

- L'**erreur quadratique moyenne (RMSE : Root Mean Square Error)** est une règle de notation quadratique qui mesure également la magnitude moyenne de l'erreur. C'est la racine carrée de la moyenne des différences au carré entre la prévision et l'observation réelle.

$$RMSE = \sqrt{\frac{1}{N}\sum_{(x,y)\in D}(y - prediction(x))^2} \quad ou \sqrt{MSE} \tag{3.8}$$

Le problème avec les MSE est qu'elles conduisent à des résultats peu corrélés avec la perception que nous avons de la qualité d'image. De plus, cela peut conduire à des artefacts dans les zones lisses (la peau, ...) et reconstruit assez mal les détails des parties des images (les yeux, la bouche,...).

- L'**erreur absolue moyenne (MAE** : Mean Absolute Error**)** fera en sorte que le modèle reconstruise des images plus nettes et moins floues. MAE mesure la magnitude moyenne des erreurs dans un ensemble de prévisions, sans tenir compte de leur direction. Il s'agit de la moyenne sur l'échantillon test des différences absolues entre la prévision (image prédite) et l'observation réelle (image originale).

$$\frac{1}{N}\sum_{(x,y)\in D}|y - prediction(x)| \tag{3.9}$$

Si la valeur absolue n'est pas prise (les signes d'erreur ne sont pas supprimés), l'erreur moyenne devient l'erreur de biais moyenne (MBE) et est généralement destinée à mesurer le biais moyen du modèle. MBE peut transmettre des informations utiles, mais doit être interprété avec prudence, car les erreurs positives et négatives s'annuleront.

Similarités : MAE et RMSE expriment toutes les deux l'erreur de prédiction moyenne du modèle en unités de la variable d'intérêt. Les deux mesures peuvent aller de 0 à 1 et sont indifférentes au sens des erreurs. Ce sont des scores orientés négativement, ce qui signifie que les valeurs basses sont meilleures. On peut citer d'autres calculs d'erreur :

- **Log Loss (Cross Entropy Loss)** *:* Est une fonction de perte également utilisée fréquemment dans les problèmes de classification.

- **Adagrad :** Algorithme complexe de descente de gradient qui redimensionne les gradients de chaque paramètre en attribuant à chacun des paramètres un taux d'apprentissage indépendant.

Conclusion

Dans ce chapitre nous avons présenté l'architecture de l'auto-encodeur ainsi que la présentation du principe des composants de ses couches. Nous avons défini les filtres de convolution, la mise en commun (pooling), le rembourrage (padding) et le pas de convolution (stride). Comme pour les réseaux de neurones les fonctions d'activation sont fondamentales, nous avons été amené à donner un aperçu sur les fonctions d'activation utilisées dans notre travail et nous pouvons dire à ce stade qu'il sera le plus souvent préférable d'utiliser la fonction ReLU.

Nous utilisons les auto-encodeurs dans trois applications : le débruitage, la reconstruction d'image de visage et principalement la reconnaissance de visage. Nous avons besoin dans les deux premiers cas de calculer la perte (loss) et dans le cas de la reconnaissance de visage, la classification est essentielle. Pour toutes ces raisons, nous avons consacré la dernière partie du chapitre pour définir toutes ces notions qui nous serons très utile pour le chapitre 4 dans lequel nous présentons l'implémentation de notre approche.

Introduction

Traiter une quantité énorme de données de haute dimension, telles que le schéma climatique mondial, l'expression des gènes humains. Grâce aux avis des clients sur les grands sites Web, les scientifiques se confrontent quotidiennement au problème de la réduction de la dimensionnalité et espèrent ainsi découvrir des méthodes efficaces pour transformer les données de grande dimension à une expression plus compacte et significative dans un espace de faible dimension. Les techniques classiques de réduction de la dimensionnalité, notamment l'analyse en composantes principales (PCA) et l'échelle multidimensionnelle (MDS) sont largement utilisées dans ces domaines pour leur facilité de mise en œuvre et leur efficacité informatique. Une fois les vrais modèles des données incorporées sont trouvés, les données peuvent être compressées sans trop de perte. Cependant, dans de nombreux cas, la propriété non linéaire des données rendra la PCA et la MDS inefficaces. Les deux techniques ne permettent pas de détecter le véritable degré de liberté d'observations naturelles complexes, telles que le visage humain sous différentes conditions d'observation. Par conséquent, un cadre global inspiré de l'apprentissage profond pour la réduction de la dimensionnalité : l'auto-encodeur AE est proposé pour surmonter cette limitation en combinant les filtres convolutionnels et le pooling profonds. Cette technique est largement utilisée pour la compression des données, ce qui permet de réduire l'utilisation de la mémoire, le temps nécessaire de calcul, l'amélioration des performances en supprimant les variables redondantes, en utilisant des données de grande dimension et en supprimant le bruit des données d'origine. Nous nous demandons si nous pouvons exploiter la puissance du réseau de neurones à convolution pour améliorer les performances à l'aide de simple auto-encodeur. Dans ce chapitre, nous présentons un auto-encodeur utilisant des couches de convolution. Nous utilisons l'auto-encodeur dans différentes tâches : compression d'image, suppression du bruit et dans la reconnaissance de visage. Nous utilisons les bases de données universelles CASIA 2D et Faces 95 pour valider nos travaux utilisant l'auto-encodeur pour trois différentes applications.

4.1 Description des bases de données utilisées

4.1.1 Faces 95

À l'aide d'une caméra fixe, une séquence de 20 images par individu a été prise. Au cours de la séquence, le sujet fait un pas en avant vers l'appareil photo. Ce mouvement est utilisé pour introduire des variations importantes de la tête (échelle) entre les images du même individu. Il y a environ 0,5 seconde entre les images successives de la séquence. La base de données Faces 95 contient : un nombre de 72 individus, une résolution de l'image de 180 x 200 pixels (format

portrait). Elle contient des images d'hommes et de femmes. Les images ont été prises en une seule session.

Plusieurs Variation des images individuelles sont prises en compte comme le montre la **figure 4.1** : variation de l'arrière-plan est provoquée par les ombres lorsque le sujet avance, grande variation d'échelle de la tête, inclinaison de la tête, position du visage dans l'image, variation de l'éclairage de l'image en raison de la disposition de l'éclairage artificiel, variation d'expression.

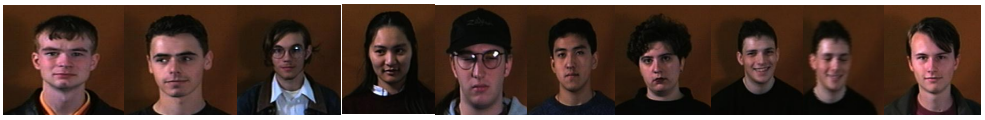


Figure 4. 1 Échantillons de la BDD Faces 95

4.1.2 VGGFACE

Le jeu de données comprend 2 622 identités, un échantillon est présenté par la **figure 4.2**. Chaque identité est associée à un fichier texte contenant des URL pour les images et les détections de visage correspondantes. Des modèles pré-entraînés à l'aide de ces données sont disponibles. La BDD VGGFace a été créée par Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman.

Figure 4. 2 Échantillons de la BDD VGG-Face [49]

4.1.3 CASIA 2D V4

Parmi toutes les fonctions biométriques, la reconnaissance faciale reste l'un des sujets de recherche les plus actifs en reconnaissance de formes. Au cours des dernières décennies, la plupart des travaux se sont concentrés sur la source d'images 2D d'intensité ou en couleur. La précision de la reconnaissance de visage 2D étant influencée par les variations de poses, d'expressions, d'illuminations et rotations de la tête haute vers le bas et le contraire (voir **figure 4.3)**.

Figure 4. 3 Échantillons de la BDD CASIA2DV4

4.2 Approche proposée

L'approche que nous proposons pour aborder la biométrie du visage utilise essentiellement l'auto-encodeur, tout d'abord pour débruiter l'image, puis la reconstruction du visage et finalement la réduction des caractéristiques du visage en vue de reconnaissance qui est notre approche proposée nommée RVAEC (voir la **figure 4.4**). Cette dernière application est notre objectif principal.

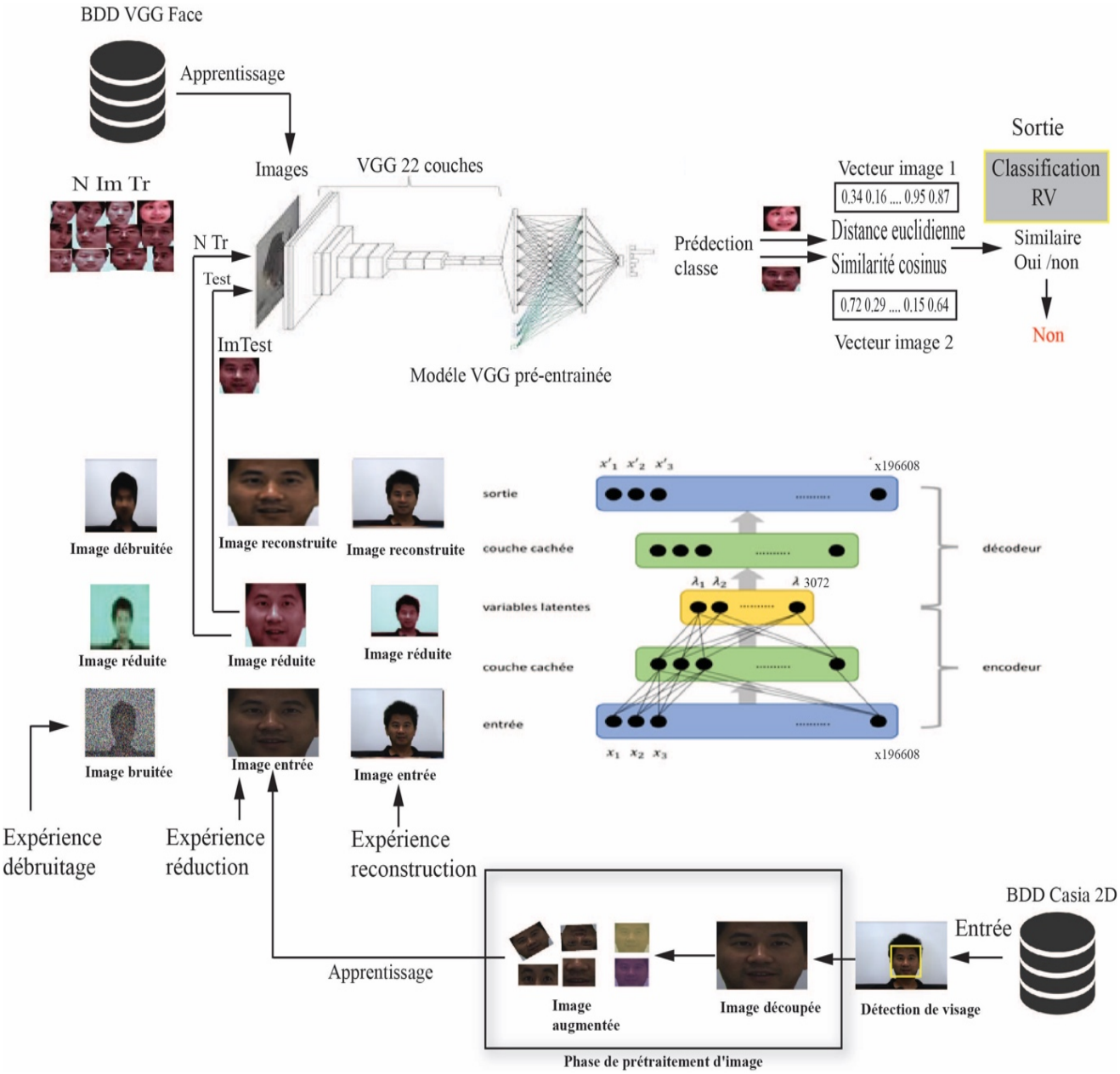


Figure 4. 4 Schéma de principe de notre approche RVAEC

Dans ce qui suit nous jugeons bon à ce stade du travail de définir : 1) la répartition du jeu de données, 2) les différents paramètres du modèle AE et 3) les outils utiles pour notre conception et expériences.

Par la suite, l'étude expérimentale est structurée en trois parties :

1) Le débruitage sur la BDD CASIA2DV4 sur un échantillon de 10 personnes différentes avec un nombre de 359 images avec plusieurs variantes (poses, expressions, illuminations…) ;

2) La reconstruction ou compression sur les BDDs Faces 95, CASIA2DV4 et différentes images sur le net dans les milieux incontrôlés. La taille des échantillons d'images de visage est variable d'une expérience à une autre ;

3) La reconnaissance de visage sur la totalité de la BDDs CASIA2DV4.

4.3 Répartition du jeu de données

4.3.1 Généralisation et séparation du jeu de données

Une question fondamentale est donc de savoir si les prédictions du modèle sont bonnes pour des données qu'il n'aura jamais vues ? C'est le problème de la généralisation.

Le problème survient lorsque la fonction à modéliser est "complexe" et le nombre de dimensions de l'entrée est grand. Quand c'est le cas, choisir simplement le voisin qui minimise une métrique simple peut devenir une très difficile, et cette approche généralise mal à de nouveaux exemples. On voit donc que le fait de trop se fier aux exemples de l'ensemble d'**apprentissage** peut être trompeur. On nomme sur-apprentissage (over-fitting) le fait d'apprendre une réponse généralisant mal, mais performant bien sur les exemples d'apprentissage. C'est pourquoi on réservera une partie du jeu de données, ne servant pas durant l'apprentissage, pour estimer l'erreur de généralisation. C'est le jeu de données de **test**, ou ensemble de test.

4.3.2 Hyper paramètres et ensemble de validation

Le sur-apprentissage est un phénomène qui peut se produire si on donne trop de flexibilité au modèle, lui permettant d'ajuster ses paramètres de façon trop précise aux exemples d'entraînement. On dira que le modèle a une trop grande capacité. Pour éviter ce problème, une approche simple consiste à réduire le nombre de paramètres du modèle (pour les modèles le permettant). Ce nombre est un hyper paramètre, c'est-à-dire qu'il s'agit d'une valeur qui n'est pas apprise au même titre que d'autres paramètres, durant l'apprentissage, mais contrôle de façon plus globale la forme du modèle et le déroulement de l'apprentissage. D'autres exemples d'hyper paramètres sont la durée de l'apprentissage, les coefficients (poids) de certains termes dans la fonction objective, etc. C'est l'existence d'hyper paramètres qui justifie une autre séparation du jeu de données, le jeu de données de **validation**, ou ensemble de validation. Typiquement, on ne connaîtra pas la valeur optimale des hyper paramètres menant à l'apprentissage du

"meilleur" modèle. On devra donc essayer plusieurs combinaisons possibles d'hyper paramètres. Chacun des modèles sera entraîné sur l'ensemble d'entraînement en optimisant ses paramètres (et non pas hyper paramètres) selon cet ensemble. On réservera donc une partie du jeu de données, l'ensemble de validation, pour évaluer la performance de chaque modèle et choisir le meilleur, selon cette mesure. Cette opération se nomme sélection de modèle. On utilisera l'ensemble de test, une autre sous-partie réservée mais distincte, introduite plus haut, comme mesure de la performance finale de l'apprentissage et de la sélection de modèle, pour pouvoir comparer une approche (un type de modèle) à d'autres publiées dans la littérature pour le même ensemble de données.

4.4 Paramètres prise en compte par l'AE

4.4.1 Définition du lot de données (batch)

Pour ne pas transférer l'ensemble du jeu de données dans le réseau neuronal en une fois. **Le jeu de données** peut être **divisé en un nombre de lots.** La taille du lot est un hyper paramètre qui définit le nombre d'échantillons à traiter avant la mise à jour des paramètres de modèle internes.

Les prédictions sont calculées sur un ou plusieurs échantillons du lot. À la fin de celui-ci, les prévisions sont comparées aux variables de sortie attendues et une erreur est calculée. À partir de cette erreur, l'algorithme de mise à jour est utilisé pour améliorer le modèle, par exemple pour descendre le long du gradient d'erreur. Un ensemble de données d'apprentissage peut être divisé en un ou plusieurs lots. Le **batch ou lot** est donc un terme utilisé dans l'apprentissage automatique et désigne le nombre d'exemples de formation utilisés dans une itération. La taille du lot peut être l'une des trois options suivantes :

1. ***Descente de gradient par lots (batch mode)*** : où la taille du lot est égale à l'ensemble de données total, rendant ainsi les valeurs d'itération et d'époque équivalentes.

   Taille du lot = Taille de l'ensemble d'apprentissage.

2. ***Descente de gradient par mini-lots (mini-batch mode*)** : où la taille du lot est supérieure à un mais inférieure à la taille totale du jeu de données. Généralement, un nombre pouvant être divisé en la taille totale du jeu de données.

   1 <Taille du lot <Taille de l'ensemble d'apprentissage.

3. ***Descente de gradient stochastique (stochastic mode)*** : où la taille du lot est égale à un. Par conséquent, les paramètres de gradient et de réseau neuronal sont mis à jour après chaque échantillon.

   Taille du lot = 1.

Que se passe-t-il si le jeu de données ne se divise pas de manière égale par la taille du lot?
Cela peut arriver et arrive souvent lors de la phase d'apprentissage d'un modèle. Cela signifie simplement que le dernier lot contient moins d'échantillons que les autres.

Solution

- Supprimer certains échantillons du jeu de données
- Modifier la taille du lot de sorte que le nombre d'échantillons dans le jeu de données se divise de manière égale par la taille du lot.

Quelques algorithmes d'optimisation du modèle [50] sont largement utilisés par la communauté du Deep Learning pour résoudre les challenges cités précédemment : i) RMSProp (Root Mean Square Prop) est un algorithme d'optimisation du taux d'apprentissage adaptatif, ii) Adam (Adaptive Moment Estimation) combine les idées de RMSProp et de Momentum, iii) Adadelta est une extension d'Adagrad qui cherche à : réduire son taux d'apprentissage.

4.4.2 Définition d'une époque

Une **époque** est un terme utilisé dans l'apprentissage automatique et indique le **nombre de passages** dans l'ensemble du jeu de données d'apprentissage.

Autrement dit, une époque est lorsqu'un ensemble de données entier est passé en avant et en arrière à travers le réseau de neurones qu'une seule fois.

Étant donné qu'une époque est trop importante pour être transmise à l'ordinateur en une fois, nous la divisons en plusieurs lots plus petits. Le passage de l'ensemble du jeu de données via un réseau de neurones ne suffit pas. Et nous devons transmettre l'ensemble de données complet plusieurs fois au même réseau de neurones. Mais gardons à l'esprit que nous utilisons un jeu de données limité et que nous optimisons l'apprentissage et le graphique par la descente du gradient, qui est un processus ***itératif***. Donc, *mettre à jour les poids avec un seul passage ou une époque ne suffit pas. Une époque conduit à un sous-ajustement de la courbe dans le graphique de la* ***figure 4.5***.

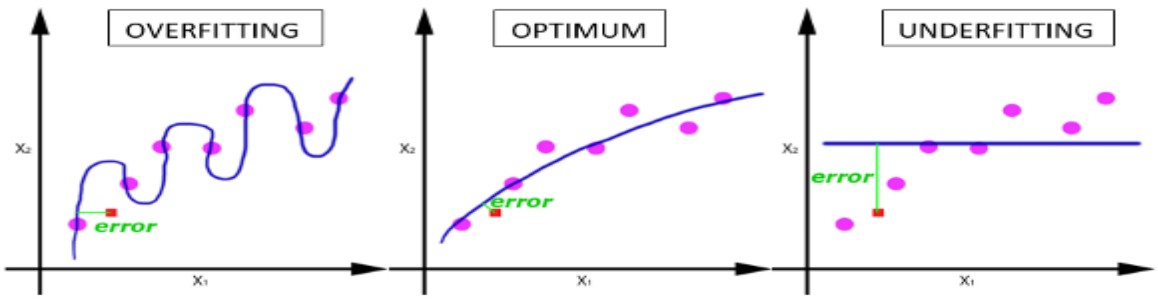


Figure 4. 5 Différents cas d'optimisation d'apprentissage (GD) [51]

Comme le nombre d'époques augmente (X_1), plus le nombre de fois le poids (X_2) sont modifiés dans le réseau neuronal et la courbe va de **sous-apprentissage** à **optimal** à **sur- apprentissage**.

Pour des résultats optimaux, on doit choisir le nombre d'époques convenable. Malheureusement, il n'y'a pas un nombre fixe. La réponse est différente pour différents ensembles de données, mais on peut dire que le nombre d'époques est lié à la diversité des données.

4.4.3 Définition d'une itération

Les itérations sont le nombre de lots nécessaires pour terminer une époque.

Exemple : la division de l'ensemble de données de 2000 échantillons en lots de 500, il lui faudra 4 itérations pour terminer une époque.

4.5 Paramètres de performance de la biométrie du visage utilisant l'AEC

4.5.1 Définition de la précision (Accuracy)

Une fois que le modèle est réalisé, il est important de vérifier si le modèle fonctionne bien avec des exemples de test non utilisés pour l'apprentissage du modèle. Le système de reconnaissance biométrique utilise un certain nombre de métriques pour mesurer la précision prédictive d'un modèle. Le choix de la métrique de précision dépend de l'application. Il est important d'examiner ces mesures pour déterminer si le modèle fonctionne bien.

4.5.2 Définition du paramètre d'ajustement du modèle

Le paramètre d'ajustement validation_split du modèle AEC divise les données en deux parties pour chaque époque, à savoir les données d'apprentissage et les données de validation. Il forme le modèle sur les données d'apprentissage et valide le modèle sur les données de validation en vérifiant sa perte et sa précision. Généralement, à chaque époque, la perte diminue et la précision augmente. Mais avec val_loss (validation de perte) et val_acc (validation de précision), de nombreux cas sont possibles :

1. Augmentation de **val_loss** avec diminution de **val_acc** (signifie que le modèle écrase les valeurs sans apprendre)
2. Augmentation de **val_loss** avec augmentation de **val_acc** (il peut s'agir d'un surajustement ou de diverses valeurs de probabilité dans les cas où softmax est utilisé dans la couche de sortie)
3. Diminution **val_loss** avec augmentation de **val_acc** (Correct, signifie que la construction du modèle apprend et fonctionne correctement)

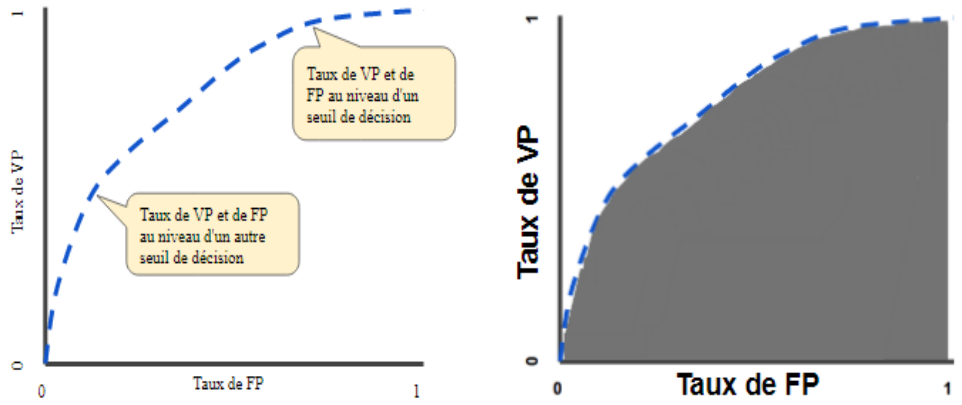
4.5.3 Définition des courbes ROC et AUC

Une **courbe ROC** (**Receiver Operating Characteristic**) est un graphique représentant les performances d'un modèle de classification pour tous les seuils de classification. Cette courbe trace le taux de vrais positifs en fonction du taux de faux positifs :

Le **taux de vrais positifs** (**TVP**) et le **taux de faux positifs** (**TFP**) sont définis comme suit :

$$TVP = \frac{VP}{VP + FN} \qquad TFP = \frac{FP}{FP + VN}$$

Une courbe ROC trace les valeurs TVP et TFP pour différents seuils de classification. Diminuer la valeur du seuil de classification permet de classer plus d'éléments comme positifs, ce qui augmente le nombre de faux positifs et de vrais positifs. La **figure 4.6.a** ci-dessous représente une courbe ROC classique.



a) Courbe ROC b) AUC (aire sous la courbe ROC)

Figure 4. 6 Taux de VP et de FP pour différents seuils de classification

AUC signifie "aire sous la courbe ROC". Cette valeur mesure l'intégralité de l'aire à deux dimensions situées sous l'ensemble de la courbe ROC (par calculs d'intégrales) de (0,0) à (1,1). L'AUC présente les avantages suivants :

- L'AUC est invariante d'échelle. Elle mesure la qualité du classement des prédictions.
- L'AUC est indépendante des seuils de classification. Elle mesure la qualité des précisions du modèle quel que soit le seuil de classification sélectionné.

4.6 Présentation des outils

4.6.1 Software

Plusieurs framework (infrastructure logicielle) open source sont disponibles dans la littérature, la grande majorité supporte le langage Python. Voici une liste non exhaustive de quelques-uns :

- **Theano :** Crée par Frédéric Bastien et l'équipe de recherche derrière le laboratoire de l'Université de Montréal, MILA. Il a l'avantage d'être codé en Python et performant s'il est utilisé correctement. Mais il a les inconvénients : i) support du Multi GPU nécessite une solution de contournement, ii) temps de compilation très lent pour les grands modèles.

- **TensorFlow :** crée par l'équipe Google Brain pour mener des recherches sur le ML et le Deep Learning. Il est considéré comme une version moderne de Theano. Son avantage est le fait d'être codé en Python, il est supporté par Google en ayant une grande communauté et supporte un multi-GPU. Les inconvénients sont la lenteur et les couches de réseaux de neurones.
- **Keras :** c'est le framework de plus haut niveau, le plus convivial de la liste. Il permet aux utilisateurs de choisir si les modèles qu'ils construisent sont exécutés sur Theano ou TensorFlow. Il est écrit et entretenu par Francis Chollet, un autre membre de l'équipe Google Brain. Ses avantages en plus du fait d'être codé en Python, Il a un arrière-plan par excellence pour Theano ou TensorFlow et c'est une interface haut niveau. Par contre, il est moins flexible que les autres interfaces de programmation applicative (API)
- **PyTorch :** c'est le dernier né de la liste PyTorch est soutenu par l'équipe de recherche en intelligence artificielle de Facebook. Il permet le traitement des entrées et des sorties de longueur variable (dynamic computation graphs), ce qui est utile lorsque l'on travaille avec des RNN. Une caractéristique absente dans TensorFlow et Theano. Il a les avantages d'être codé en Python, soutenu par Facebook, mélange d'API de haut niveau et de bas niveau, support des graphes dynamiques et c'est le meilleur outil pour les chercheurs. Malheureusement, il est beaucoup moins mûr que ses concurrents et limité en références / ressources limitées en dehors de la documentation officielle

Autres frameworks : CNTK : par Microsoft, **Neon** : par Nervana Systems. Il a récemment été classé comme le framework le plus rapide dans plusieurs catégories, **Deeplearning4j** : Il supporte le langage java, **Caffe** : par Berkeley Vision and Learning Center50

Après ce petit tour d'horizon des différents frameworks disponibles, notre choix s'est porté sur le framework **Keras** et TensorFlow (**figure 4.6**) de Google. La raison principale de ce choix c'est la très grande et aussi très active communauté qui est derrière cette librairie.

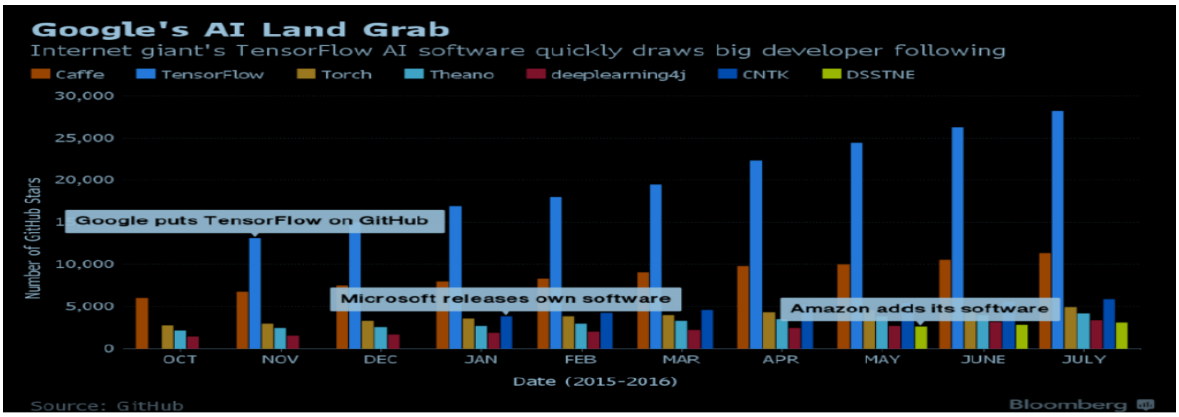


Figure 4. 7 Croissance de la popularité de TensorFlow [52]

4.6.2 Hardware

Le Deep Learning est un domaine avec des exigences en calculs intenses et la disponibilité des ressources (surtout en GPU) dédiés à cette tâche vont fondamentalement influencer sur l'expérience de l'utilisateur car sans ces ressources, il faudra trop de temps pour apprendre de ses erreurs ce qui peut être décourageant. Les expérimentations ont toutes été effectuées sur une machine qui offre des performances acceptables Dont voici les caractéristiques :

Tableau 4. 1 Caractéristiques de la machine utilisée

Processeur : Intel® Core™ i7-3632QMCPU @ 2.20 GHz
GPU : AMD Radeon HD 7650M
RAM : 6GB
Système d'exploitation : 64 bits, processeur x64

4.7 Étude expérimentale

Dans nos expériences nous commençons par valider l'AEC pour le débruitage des images, ensuite la reconstruction pour tester l'encodage et le décodage du modèle afin de pouvoir l'appliquer principalement à la réduction pour une bonne reconnaissance dans le domaine de la biométrie du visage.

4.7.1 Application débruitage

Expérience 1

La première expérience est consacrée au débruitage des images de visage pour les préparer à être analysées en vue d'une reconnaissance. L'auto-encodeur pour le débruitage (DAE) ainsi utilisé, nous permet de se familiariser avec les auto-encodeurs d'une part et de fixer convenablement les hyper paramètres afin de bien déterminer les fonctions de pertes et la précision pour une meilleure optimisation en choisissant les lots de données (batch size) et le nombre d'époques (epoch). Dans cette première expérience nous utilisons l'auto-encodeur pour le débruitage des images de la BDD CASIA2DV4. L'expérience est menée sur un échantillon de 10 personnes au total un échantillon de 359 images. La BDD de test représente 0.2% de la totalité des images choisi d'une façon aléatoire, le reste des images est consacré à la BDD apprentissage. Dans ce cas nous procédons comme suit :

1. Nous appliquons une matrice de bruit gaussien d'un facteur de bruit égale à 0.5. Le résultat est présenté par la **figure 4.8**.

(a) image originale (b) image bruitée

Figure 4. 8 Utilisation du bruit gaussien sur une image

2. Nous soumettons l'image bruitée au système DAEC. L'architecture de ce modèle est composée :
   i. D'une image d'entrée redimensionner 256x256x3 qui définit la largeur, la hauteur de l'image ainsi que les trois canaux de couleur R, G, B ;
   ii. La partie encodeur contient trois couches composées de convolution et de mise en commun (Pooling) chacune.

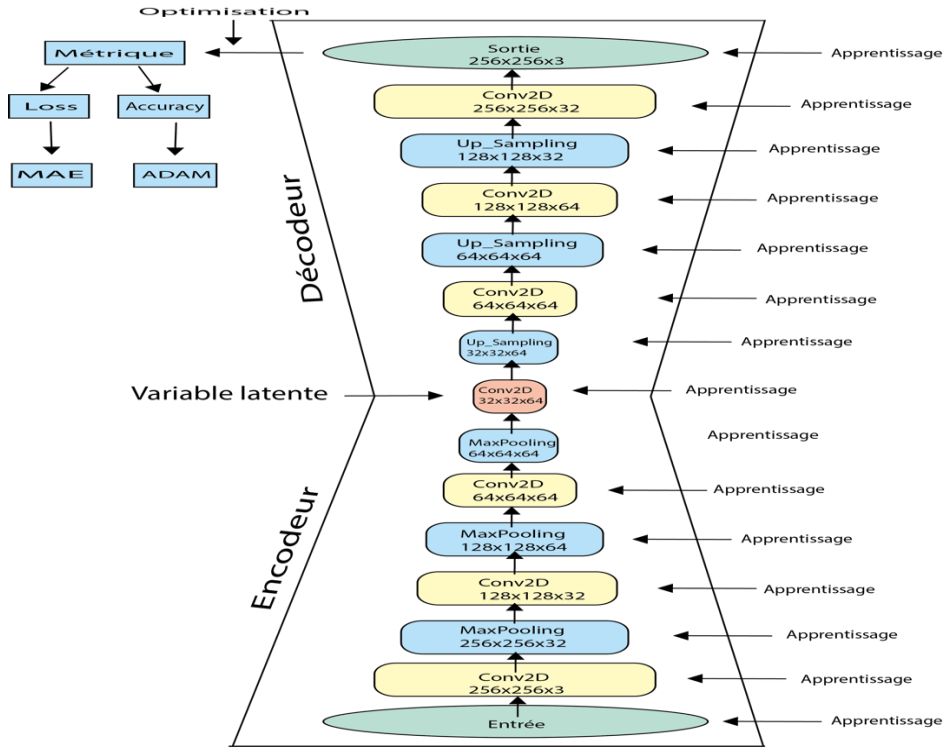


Figure 4. 9 Architecture du DAE utilisé

Les couches convolutives utilisent un filtre de dimension (3,3) et une fonction d'activation ReLu et un padding (same), c'est-à-dire la taille de sortie est la même que la taille d'entrée

Pour les couches de pooling nous avons utilisé un filtre de (2,2) dimension et un padding (same).

3. La partie décodeur est composée de 3 couches : 3 couches de déconvoultion et 3 couches de démembrement (unpooling).

Elle prend comme entrée les données encodées de la partie encodeur pour reconstruire l'image sortie débruitée. Les résultats de nos expériences pour différentes fonctions d'optimisation sont donnés par le **tableau 4.2**

Tableau 4. 2 Résultats du DAE versus fonctions d'optimisation BDD CASIA2DV4

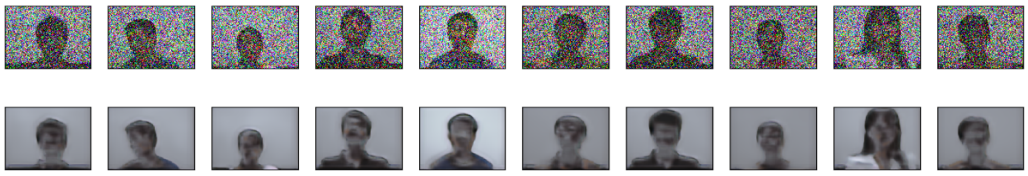
Opt.Precision/Perte	Accuracy	Loss	Epoch	Batch Size	Val/Acc	Val/loss
RmsProp/MSE	0.9018	0.0200	100	32	0.904	0.0260
RmsProp/MAE	0.901	0.0210	100	32	0.905	0.0260
Adam/MSE	0.902	6.000e-4	100	32	0.904	7.000e-4
Adam/MAE	0.9074	0.0155	100	32	0.9018	0.0158
Adam/MAE	0.905	0.0160	50	15	0.895	0.0170
Adam/MAE	**0.903**	**0.0140**	**50**	**1**	**0.886**	**0.0165**

D'après le **tableau 4.2** nous pouvons dire que les meilleurs résultats sont obtenus pour les méthodes d'optimisation ***Adam*** et ***MAE*** notamment dans le cas de l'expérience avec 50 époques et 1 batch.

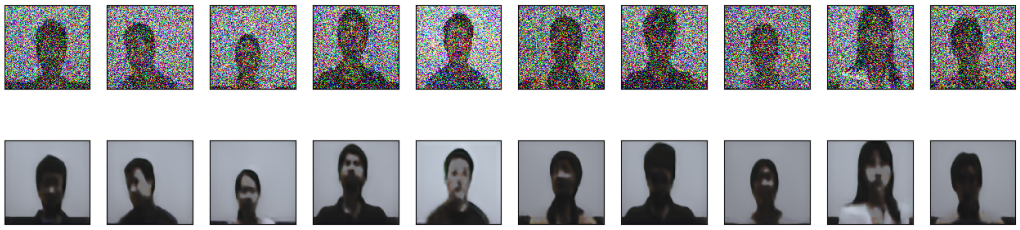
Résultats

- **Pour 100 epochs et 32 batchs:**

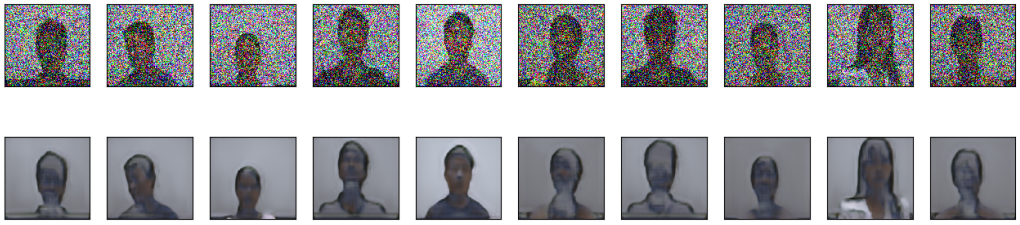
a) Adam/MAE



b) RmsProp/MSE



c) RmsProp/MAE



d) Adam/MSE

Figure 4. 10 Résultats débruitage DAE pour différents optimiseurs (100 epoch, 32 batchs)

Notons que pour les meilleurs résultats **figure 4.10.a)** sont obtenus dans le cas de la combinaison Adam/MAE. La qualité de l'image débruitée reste moyenne et ceci pour un nombre d'époques et de lots importants. Pour cette raison nous réduisons la taille du lot pour que le modèle puisse apprendre sur la totalité de la base de données dans chaque époque, ce qui permet un meilleur apprentissage et pourrait améliorer les performances.

- **Pour 50 epochs et 15 batch :**

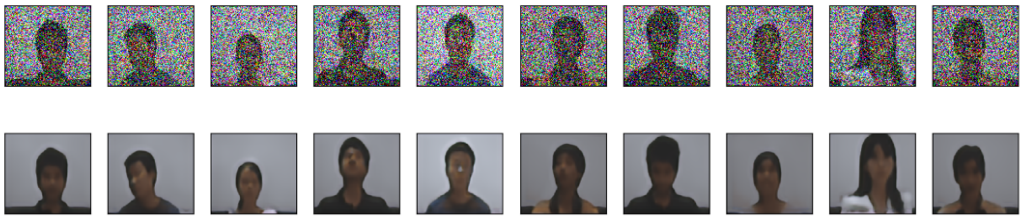


Figure 4. 11 Résultats débruitage DAE utilisant **Adam/MAE** (50 epoch, 15 batch)

- **Pour 50 epochs et 1 batch :**

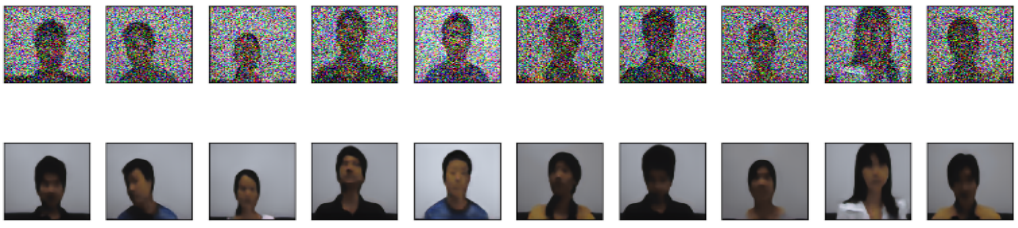


Figure 4. 12 Résultats débruitage DAE utilisant **Adam/MAE** (50 epoch, 1 batch)

D'après les résultats de la **figure 4.12**, nous constatons que la qualité de l'image débruitée est la meilleure de tous les cas. Suite aux résultats obtenus l'algorithme d'optimisation ***Adam*** est retenu ainsi que la fonction d'estimation de perte ***MAE.*** Après **50 époques** et un **batch = 1**, nous arrivons à avoir des images reconstruites non bruitées (voir **figure 4.11**) avec des paramètres appréciables : un **loss = 0.0140** et un **accuracy = 0.903**.

L'un des meilleurs moyens d'améliorer les performances d'un modèle d'apprentissage en profondeur consiste à ajouter davantage de données à l'ensemble de formation. Il existe de nombreuses façons d'augmenter les jeux de données existants et de produire des modèles plus robustes. Dans le domaine des images, elles sont utilisées pour exploiter toute la puissance de l'AEC.

Dans ce qui suit nous présentons les courbes de chacun des paramètres de performance calculés voir la **figure 4.13** et **4.14** pour un nombre d'époques égale à 50 et la taille du lot de données égale à 1.

Figure 4. 13 Courbes de performance du modèle avant l'augmentation des données

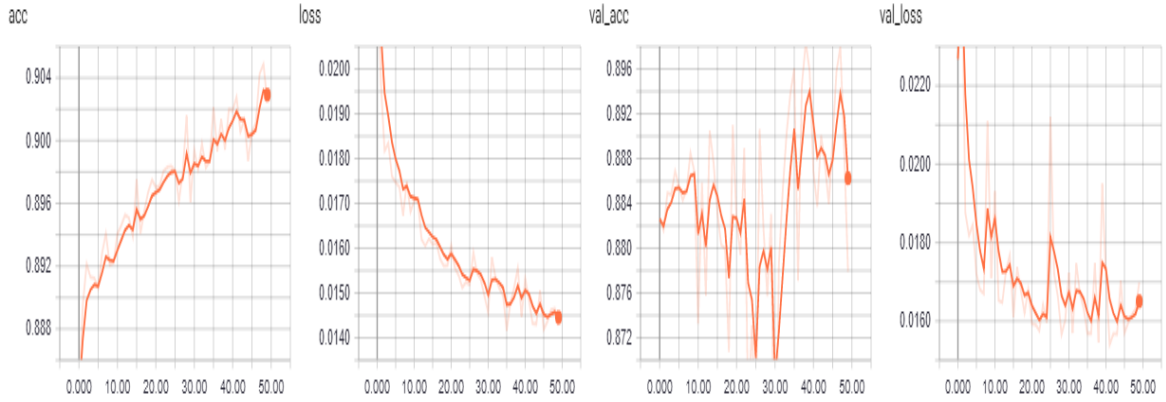


Figure 4. 14 Courbes de performance du modèle après l'augmentation des données

La **figure 4.14** représente le résultat d'apprentissage final après l'augmentation des données montre bien la qualité de résultats puisque les courbes sont plus stable moins de fluctuations et convergente. Le taux de précision et de perte sont meilleurs avec des valeurs **Acc=90%** et **loss=0.01%**. Les courbes de validation du modèle sont acceptables (val_acc et val_loss). La **figure 4.14** montre bien que les meilleurs résultats sont obtenus pour 50 époques.

D'après les résultats obtenus comme la montre la **figure 4.12** sur la BDD CASIA2D on voit bien qu'il n'est pas trop nécessaire de faire le débruitage sur ce type d'images acquises dans des circonstances contrôlées, car ce type d'images ne contient que peu de bruit.

Mais l'utilisation de DAE reste recommandée dans le cas des images bruitées telles que les images médicales ou acquises dans l'espace…

4.7.2 Application reconstruction

Nous avons utilisé dans l'application reconstruction d'image la BDD Faces95 pour essayer de valider le modèle AEC. Les images de la BDD Faces95 sont toutes frontales, de mauvaise qualité et de dimension 256x256. Pour ces raisons nous l'avons utilisé pour valider l'architecture de l'AEC. Comme, cette dernière contient moins de couches que celle du débruitage, ce qui nous permet d'utiliser plusieurs époques.

Nous menons nos expériences sur 3 personnes avec un échantillon de 60 images.

Les différentes expériences menées pour la reconstruction de l'image de visage par AEC avec différents optimiseurs sont présentées par le **tableau 4.3**.

Tableau 4. 3 Résultats de reconstruction de l'image par AEC BDD Faces95

Expériences	Optimisation	Nombre d'époques
Image 1	Mse/Adadelta	500
Image 2	Mse/Adadelta	500
Image 3	Mse/RmsProp	500
Image 4	Mse/RmsProp	500
Image 5	Mse/RmsProp	1000
Image 6	Mse/RmsProp	1000
Image 7	Crossbinary/Adadelta	500

Résultats

- ***Expérience sur Faces95***
  - ***Cas de différents époques et optimiseurs***

Voici quelques résultats de reconstruction obtenus par l'AEC illustré par la **figure 4.15** :

Image 1 Image 2 Image 3 Image 4 Image 5 Image 6 Image 7

Figure 4. 15 Reconstruction de visage par AEC pour BDD Faces95

(1ère ligne : Images originales, 2ème ligne : Images reconstruites)

D'après les expériences menées, nous remarquons que plus le nombre d'époques augmente et plus la reconstruction est mauvaise pour la raison de sur-apprentissage. Le choix du nombre

d'époques reste empirique et dépend de la perception visuelle. ***Les meilleurs résultats sont obtenus par les fonctions d'optimisation MSE et RmsProp.***

Dans ce qui suit, nous choisissons le nombre d'époques égale à 50 et les optimiseurs MAE et Adam voir **figure 4.12**, car d'après les expériences précédentes de débruitage les meilleurs résultats sont obtenus pour ce cas.

- ***Cas de 50 époques avec l'optimiseur MAE et Adam***

Les résultats de reconstruction obtenus par l'AEC pour le cas de 50 époques et un lot de données égales à 1 illustré par la **figure 4.16** ci-dessous :

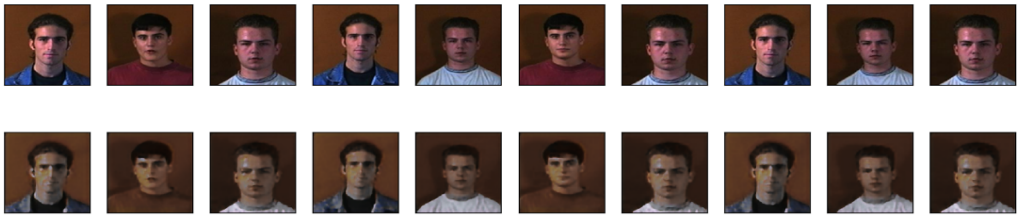


Figure 4. 16 Reconstruction de visage utilisant **Adam/MAE** (50 epoch, 1 batch) Faces95 (1ère ligne : Images originales, 2ème ligne : Images reconstruites) 50epoques 1 batch

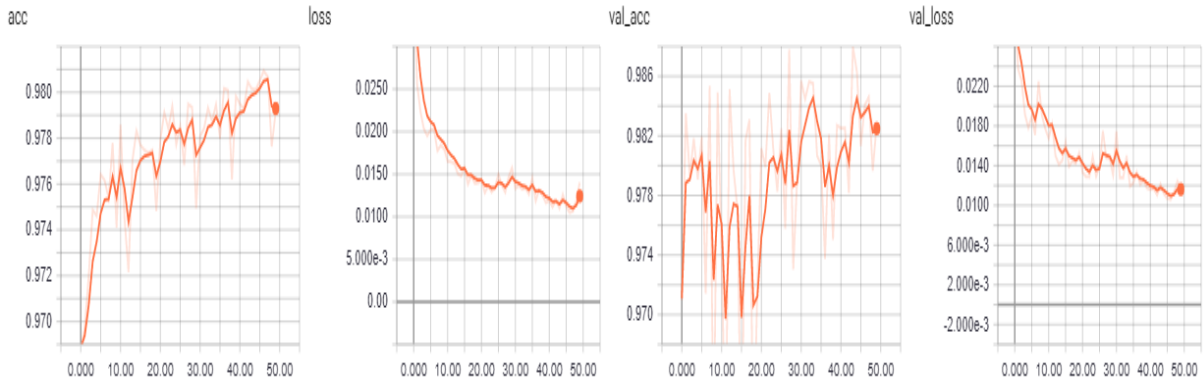


Figure 4. 17 Courbes de performance du modèle avec augmentation des données **Adam/MAE** (50 epoch, 1 batch) Faces95

Résultats

- ***Expérience sur Faces95***
  - ***Cas de 50 époques et 1 batch***

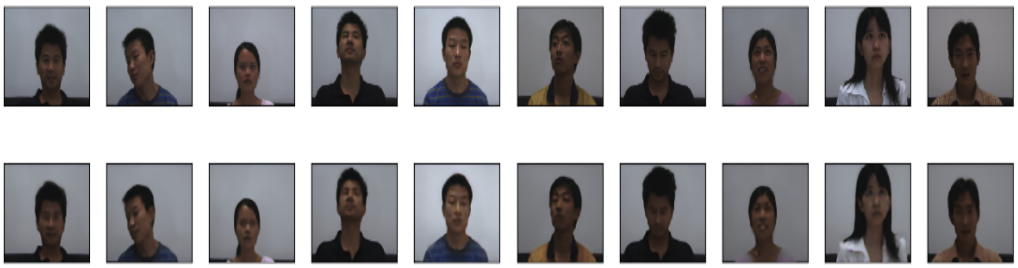
(Expérience utilisant 3 échantillons de la BDD)

Pour 50 époques et 1 batch (voir **figure 4. 17**) nous obtenons un **loss = 0.0140**, **acc = 0.9792**, **val_loss: 0.0116** et **val_acc: 0.9828**. D'aprés les résultats obtenus dans cette expérience nous pouvons dire il y'a une nette amélioration relativement aux expériences précédentes menées au début et dont les résultats sont présentés par la **figure 4.15**. Donc, nous retenons ce résultat à ce stade du travail.

- ***Expérience sur CASIA2DV4***

Dans cette expérience nous utilisons l'auto-encodeur pour la reconstruction des images de la BDD CASIA2DV4. L'expérience est menée sur un échantillon de 10 personnes au total un échantillon de 359 images. Un nombre d'expériences est réalisé et le meilleur résultat est présenté par la **figure 4.16**.

- **10 epoch 1 batch**



- **50 epoch 1 batch**

Figure 4. 18 Reconstruction de visage par AEC pour BDD CASIA2DV4

Pour la même architecture précédente utilisée dans le débruitage et les mêmes fonctions d'activation ainsi que les fonctions d'optimisations, nous obtenons de très bons résultats de reconstruction d'images de visages.

Les meilleurs résultats sont obtenus par les fonctions d'optimisation MAE et Adam.

Dans ce qui suit nous présentons les courbes de chacun des paramètres de performance du modèle de reconstruction par l'AEC calculés voir la **figure 4.19** et **4.20** pour un nombre d'époques égale à 50 et la taille du lot égale à 1.

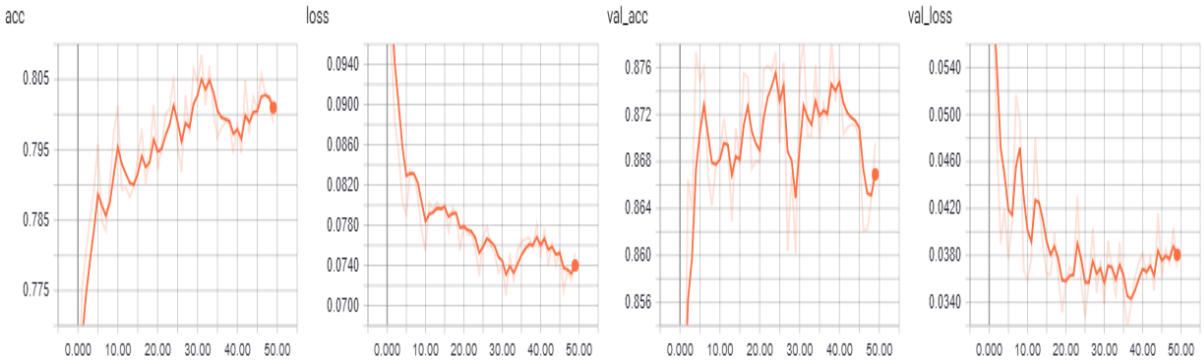


Figure 4. 19 Courbes de performance du modèle de reconstruction (données non augmentées)

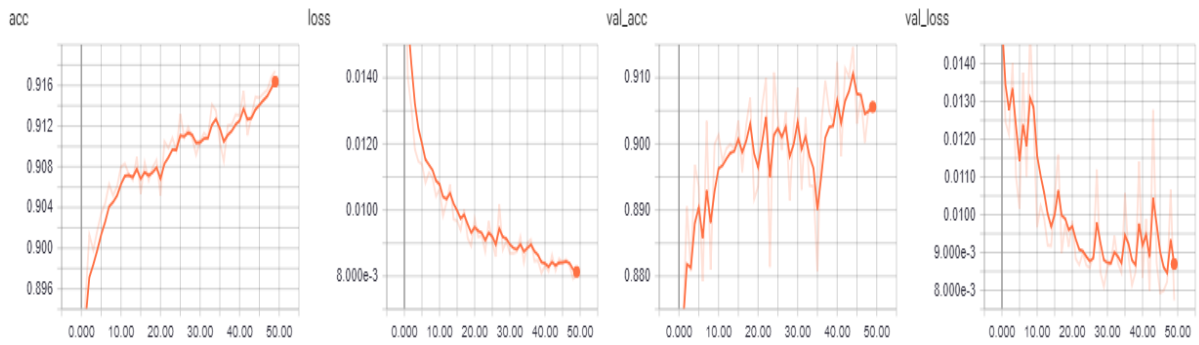


Figure 4. 20 Courbes de performance du modèle de reconstruction (données augmentées)

La **figure 4.20** qui est le résultat de l'apprentissage final après l'augmentation des données est de qualité puisque les courbes sont convergentes et plus stables présentant moins de fluctuations. Le taux de précision et de perte sont meilleurs avec des valeurs **Acc=91%** et un **loss** inférieur à **0.01%**. Les courbes de validation du modèle (val_acc et val_loss) sont acceptables. La **figure 4.20** montre bien que de bons résultats sont obtenus à partir de pour les 10 époques et le meilleur résultat est obtenu pour 50 époques.

Cette expérience est utilisée pour valider le modèle et ensuite récupérer les données de l'espace latent pour la prochaine étape.

4.7.3 Application Réduction pour la Reconnaissance de Visage

Notre système de reconnaissance de visage proposé comprend 3 étapes (comme illustré a la **figure 4.21**) : 1) étape de détection et prétraitement, 2) étape d'extraction et de réduction de données réaliser par l'auto-encodeur, 3) étape de classification.

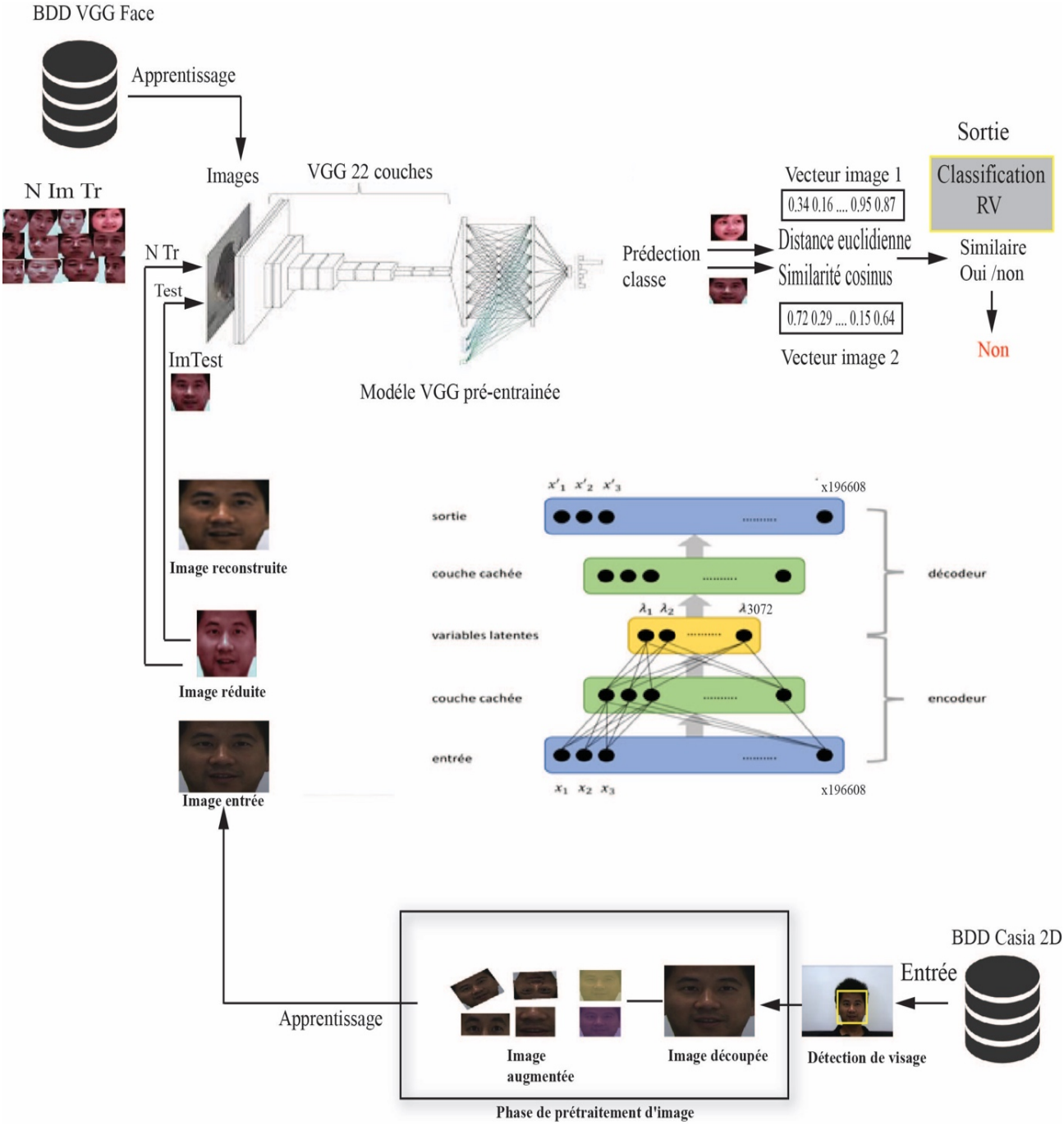


Figure 4. 21 Schéma représentatif de la réduction et l'extraction de caractéristiques basé sur l'AEC

4.7.3.1 Détection et prétraitement du visage

La première étape de la **figure 4.22** concerne la détection des visages par la méthode Viola Jones [40] pour la détection et l'extraction des visages, ce travail est mené grâce aux classificateur Haar qui est utilisé pour détecter des objets particuliers à partir de la source. Le haarcascade_frontalface_default.xml est une cascade de Haar conçue par OpenCV pour détecter la face frontale. Cette cascade de Haar est disponible sur GitHub. Une cascade de Haar fonctionne en formant la cascade sur des milliers d'images négatives avec l'image positive

superposée. Il est capable de détecter des caractéristiques de la source. La bibliothèque Dlib disponible sur python nous permet de faire la détection de visage. Dlib est un détecteur facial avec des modèles pré-formés. Le Dlib est utilisé pour estimer l'emplacement de 68 coordonnées (x, y) qui cartographient des points sur le visage d'une personne.

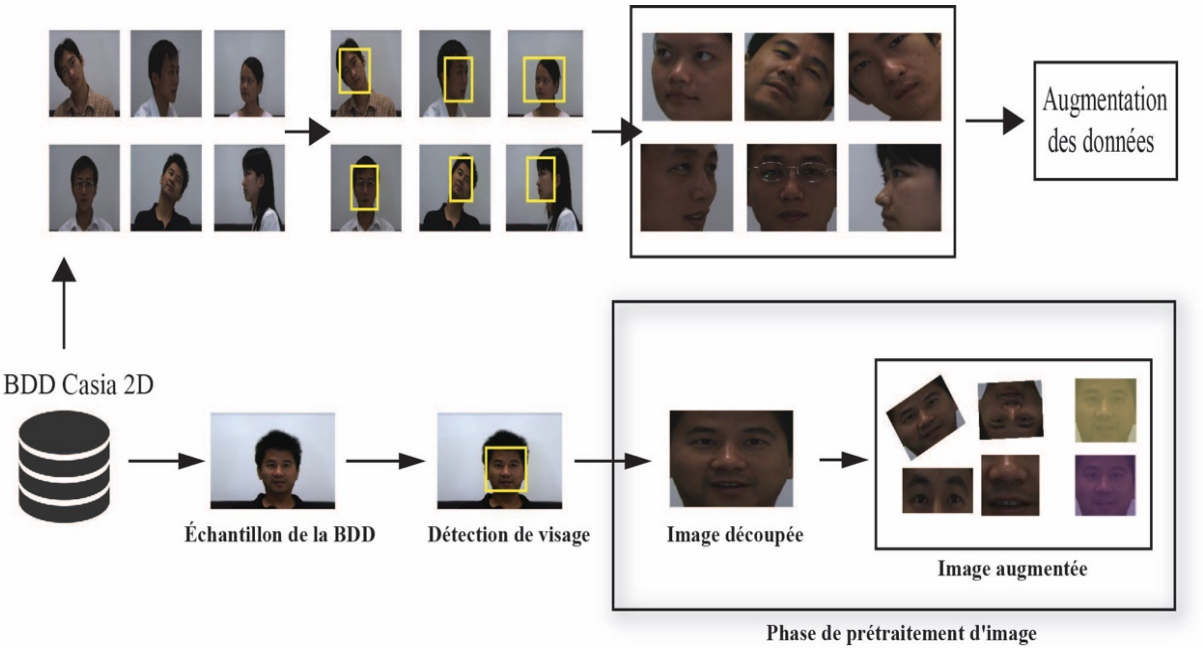


Figure 4. 22 Illustration de la phase de détection et prétraitement sur la BDD CASIA2DV4

Comme le montre la **figure 4.22** la deuxième étape concerne le découpage et l'augmentation de l'image pour une meilleure optimisation de la BDD, le découpage est utilisé pour faire face aux informations qui ne sont pas utiles tels que l'arrière-plan et la lumière générer par l'arrière-plan et qui permet une réduction tout en se basant seulement sur le visage de l'échantillon et non l'image complète, la bibliothèque PIL disponible sur python est employée pour faire cette tâche. L'augmentation des données est une technique permettant de créer artificiellement de nouvelles données d'entraînement à partir de données d'apprentissage existantes. Pour ce faire, On applique des techniques spécifiques à un domaine aux exemples tirés des données de formation qui créent des exemples de formation nouveaux et différents. L'augmentation des données d'image est peut-être le type d'augmentation de données le plus connu. Elle consiste à créer des versions transformées d'images dans le jeu de données d'apprentissage appartenant à la même classe que l'image d'origine. Les transformations comprennent une gamme d'opérations dans le domaine de la manipulation d'images, telles que les décalages, les retournements, les zooms, etc.

La bibliothèque de réseaux neuronaux d'apprentissage approfondi de Keras permet d'adapter des modèles utilisant l'augmentation de données d'image via la classe ImageDataGenerator.

4.7.3.2 Implémentation de l'AEC pour l'extraction et la réduction de caractéristiques

La troisième étape c'est l'utilisation de l'auto-encodeur pour l'extraction des caractéristiques et la réduction des données.

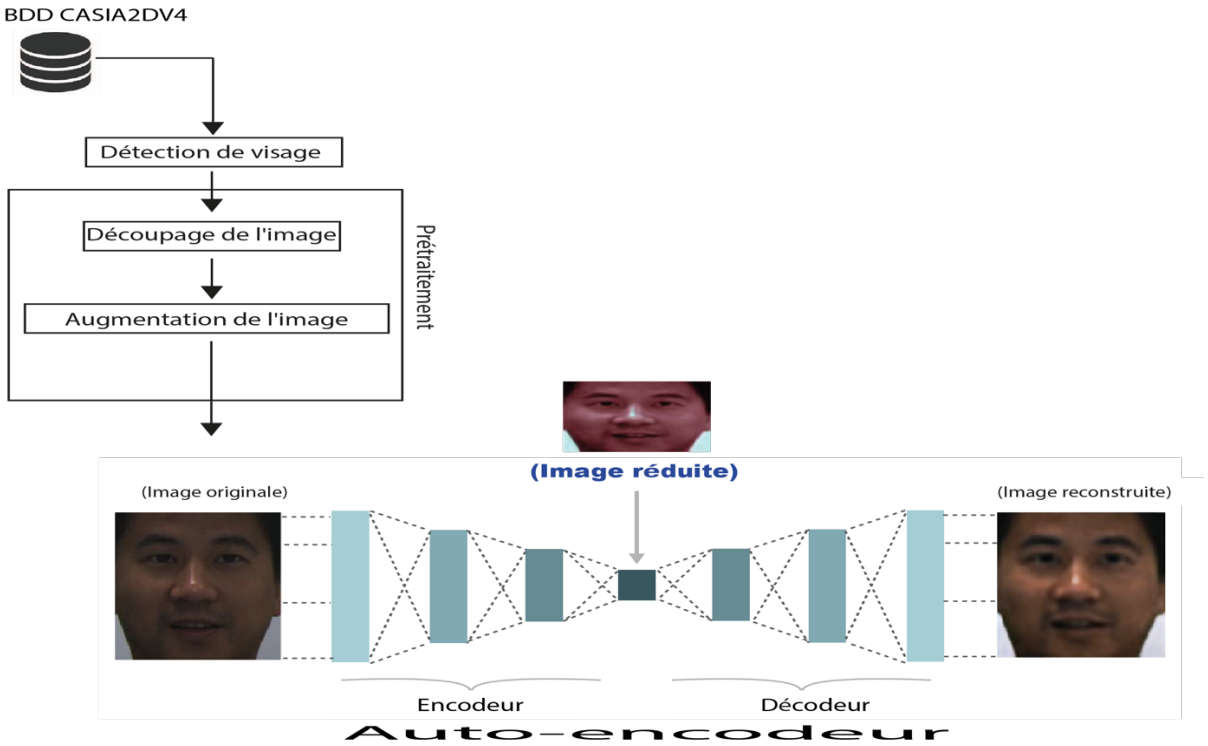


Figure 4. 23 Présentation de l'approche AEC pour la réduction

Les expériences sont menées sur un échantillon de 10 personnes au total un échantillon de 314 images. La BDD de test représente 0.2% de la totalité des images choisi d'une façon aléatoire, le reste des images est consacré à la BDD apprentissage et une expérience finale sur la totalité de la base.

4.7.3.3 Application de l'AEC pour l'extraction et la réduction sur CASIA2DV4

4.7.3.3.1 Résultats de l'AEC sans détection et découpage

- ***Cas de l'architecture trois couches de l'encodeur***

L'architecture de ce modèle est composée :

i. D'une image d'entrée redimensionner 256x256x3 qui définit la largeur, la hauteur de l'image ainsi que les trois canaux de couleur R, G, B ;

ii. La partie encodeur contient trois couches composées de convolution et de mise en commun (Pooling) chacune.

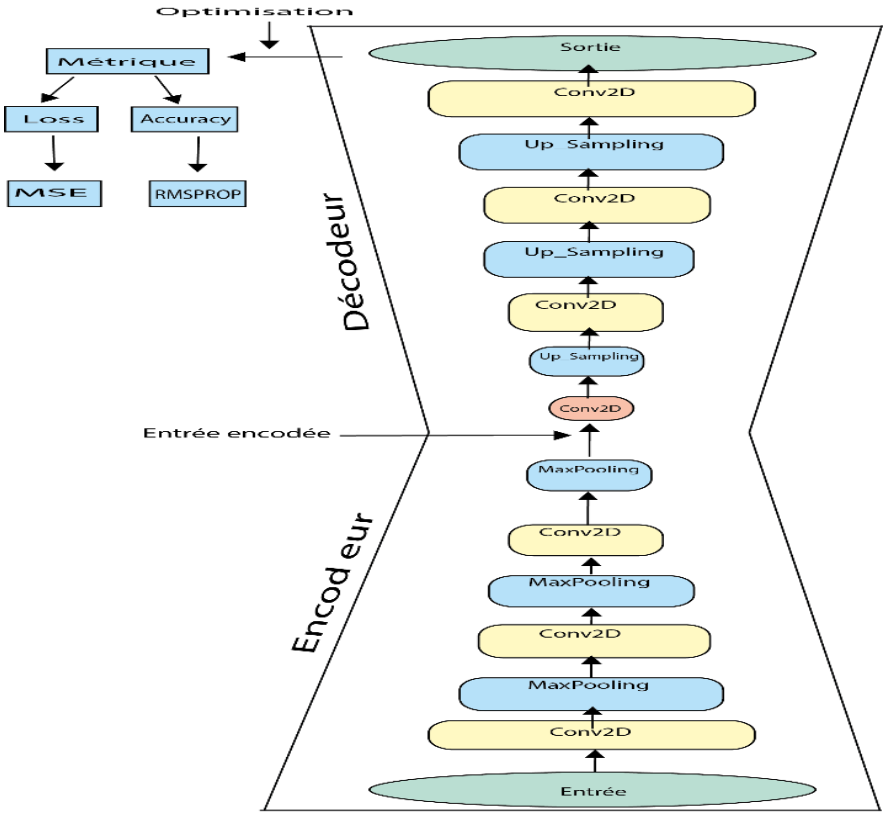


Figure 4. 24 Architecture de L'AEC (image codée)

Les couches convolutives utilisent un filtre de dimension (3,3) et une fonction d'activation ReLu et un padding (same), c'est-à-dire la taille de sortie est la même que la taille d'entrée. Pour les couches de pooling nous avons utilisé un filtre de (2,2) dimension et un padding (same).

La partie décodeur est composée de 3 couches : 3 couches de déconvoultion et 3 couches de démembrement (unpooling).

Elle prend comme entrée les données encodées de la partie encodeur pour reconstruire l'image sortie. Les optimiseurs MSE et RMSPROP sont appliquée dans cette expérience.

Résultats : Les résultats de reconstruction obtenus par l'AEC pour le cas de 50 époques et un lot de données égales à 1 illustré par la **figure 4.25** ci-dessous :

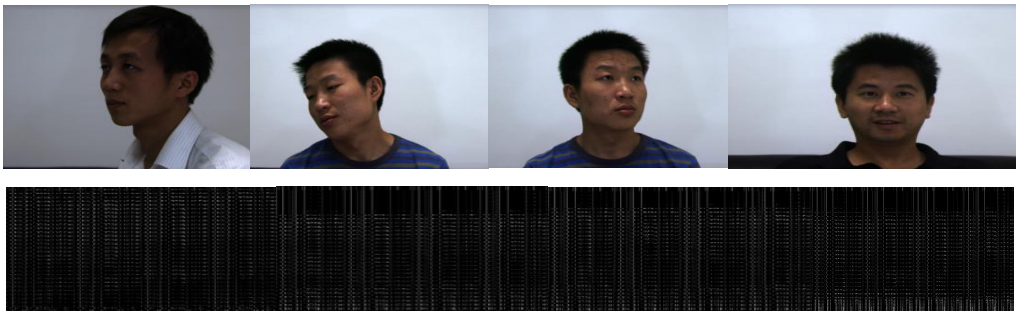


Figure 4. 25 Résultat pour la sortie codée (50epoch,1batch) sur BDD CASIA2DV4

D'après l'expérience menée, nous remarquons que le résultat est une image codée en niveaux de gris car la sortie de l'encodeur est générée à partir de la couche pooling qui permis à avoir

une image codée et non une carte de caractéristique (image) avec les canaux RGB. Cela nous à données un résultat moyen du taux de reconnaissance.

- ***Cas de l'architecture quatre couches de l'encodeur***

L'architecture de ce modèle est la même que l'architecture du modèle précèdent sauf qu'une couche de convolution est ajoutée avec 3 filtres de dimension (3,3) pour obtenir une image couleur. Les expériences dans ce cas sont validées sur la BDD CASIA2DV4.

Résultats : Les résultats de cette expérience sont illustrés par la **figure 4.26** ci-dessous:

Figure 4. 26 Images originales et images réduites par L'AEC

D'après les résultats de la **figure 4.26**, nous constatons que les images réduites apparaissent contrairement à la figure 4.25 mais nécessite encore des améliorations sur le modèle pour avoir de meilleures performances. Les images d'entrée sont d'une taille de 256x256x3 et les images sorties sont de 32x32x3. Suite aux résultats obtenus l'algorithme d'optimisation ***Adam*** est appliqué ainsi que la fonction d'estimation de perte ***MAE.*** Après **100 époques** et un **batch = 1**. ***Nous avons ainsi réalisé plusieurs expériences dans le souci d'amélioration des performances du système de reconnaissance de visage toujours sur la même BDD CASIA2DV4, en jouant sur plusieurs paramètres fondamentaux suivants :***

- ***Cas de l'expérience 50 epoch et 4 batch*** en utilisant les optimiseurs ***Adam*** et ***MAE*** avec ajout du paramètre « ***Dropout*** » après chaque couche cachée de la partie encodeur. Les résultats sont présentés par la **figure 4.27**

Nous rappelons que le paramètre Dropout sert à éviter les problèmes dus au sur-apprentissage

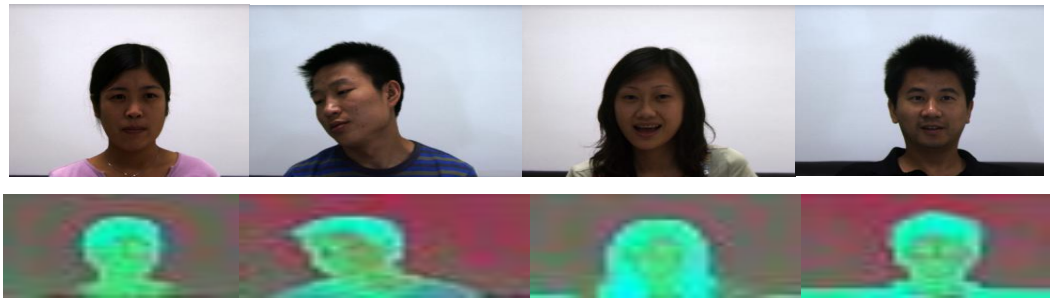


Figure 4. 27 Résultat pour 50 epoch ,4 batch avec dropout et (MAE, Adam)

- ***Cas de l'expérience 10 epoch et 1 batch*** en utilisant les optimiseurs ***Adam*** et ***MAE*** avec ajout du paramètre « ***Dropout*** » après une seule couche cachée de la partie encodeur. Les résultats sont présentés par la **figure 4.28**

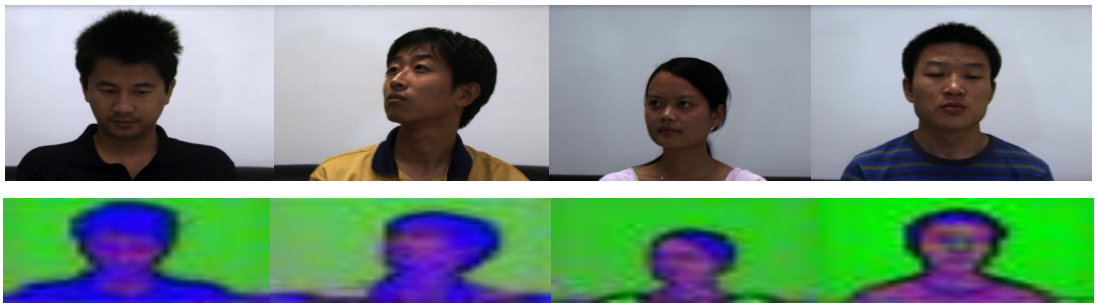


Figure 4. 28 Résultat pour 10 epoch ,1 batch avec dropout et (MAE, Adam)

- ***Cas de l'expérience avec une sortie*** utilisant la fonction ***softmax*** et un filtre de ***convolution*** de dimension ***(3x3)*** ainsi que ***5 epoch et 1 batch*** en appliquant les optimiseurs ***Adam*** et ***MAE*** avec ajout du paramètre « ***Dropout*** ». Les résultats sont présentés par la **figure 4.29**

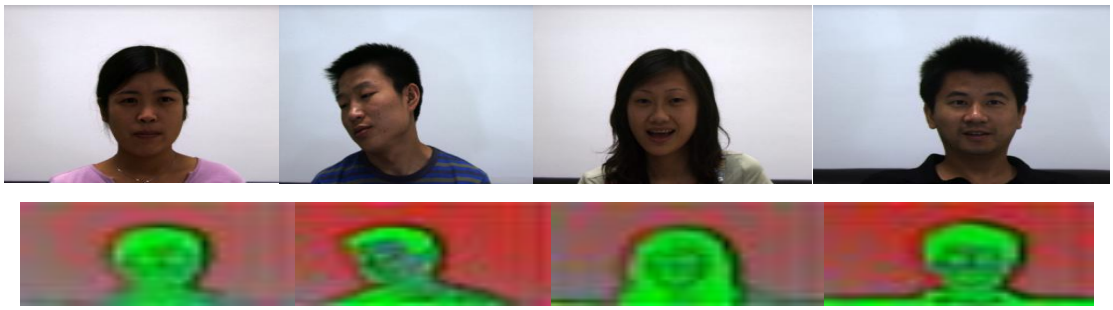


Figure 4. 29 Résultat pour la sortie softmax et 5 epoch ,1 batch avec dropout et (MAE, Adam)

- ***Cas de l'expérience avec une sortie*** utilisant la fonction ***ReLu*** et ***5 epoch et 1 batch*** en appliquant les optimiseurs ***Adam*** et ***MAE*** avec ajout du paramètre « ***Dropout*** ». Les résultats sont présentés par la **figure 4.30**

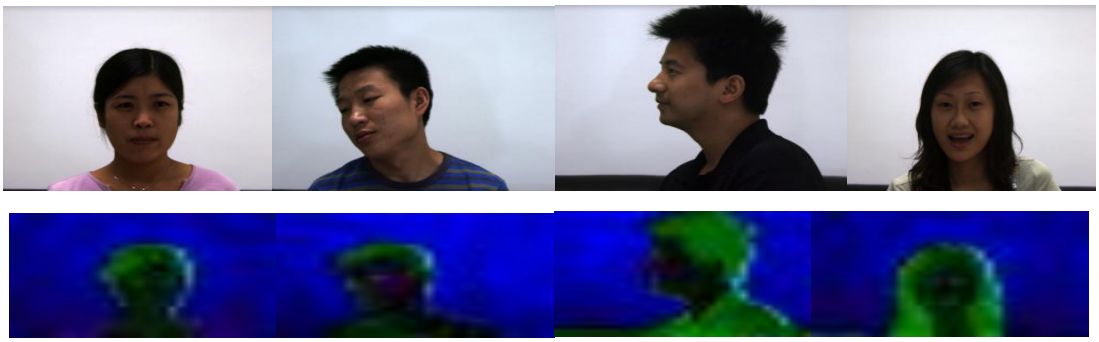


Figure 4. 30 Résultat pour la sortie ReLu et 5 epoch ,1 batch avec dropout et (MAE, Adam)

- ***Cas de l'expérience avec une sortie*** utilisant la fonction ***sigmoid*** et un filtre de ***convolution*** de dimension ***(3x3)***, ***50 epoch et 1*** batch en utilisant les optimiseurs ***Adam*** et ***MAE*** avec ajout du paramètre « ***Dropout*** » après chaque couche cachée de la partie encodeur. Les résultats sont présentés par la **figure 4.31**

Figure 4. 31 Résultat pour la sortie sigmoid et 50 epoch ,1 batch avec dropout et (MAE, Adam)

- ***Cas de l'expérience avec une sortie*** utilisant la fonction ***sigmoid*** et un filtre de ***convolution*** de dimension ***(3x3), 1 epoch et 1 batch*** en utilisant les optimiseurs ***Adam*** et ***MAE*** avec ajout du paramètre « ***Dropout*** » après une seul couche cachée de la partie encodeur. Les couches cachées utilisent des filtres de convolution comme suit (***128=>64=>32***) Les résultats sont présentés par la **figure 4.32**

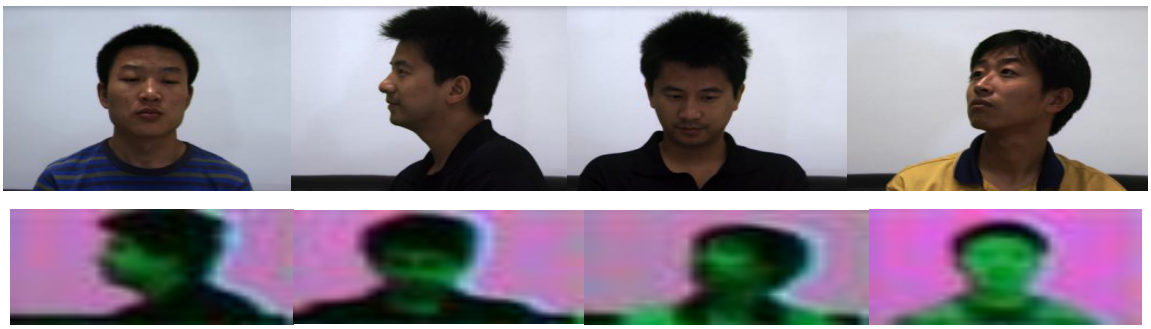


Figure 4. 32 Résultat pour la sortie sigmoid et 1 epoch ,1 batch avec dropout et les optimiseurs MAE et Adam (nombre de filtres = 128, 64 et 32)

- ***Cas de l'expérience avec une sortie*** utilisant la fonction ***sigmoid*** et un filtre de ***convolution*** de dimension ***(3x3)***,***1 epoch et 1 batch*** en utilisant les optimiseurs ***Adam*** et ***MAE*** avec ajout du paramètre « ***Dropout*** » après une seule couche cachée de la partie encodeur. ***Nous faisons varier le nombre des filtres de convolution*** dans les couches cachées comme suit : De ***(128=>64=>32)*** à ***(128=>256=>256)***. Les résultats sont présentés par la **figure 4.33**

Figure 4. 33 Résultat pour la sortie sigmoid et 1 epoch ,1 batch avec (dropout, MAE, Adam) et (nombre de filtres = 128, 256 et 256)

Discussion

Les meilleurs résultats dans toutes ces expériences est le cas illustré par la **figure 4.34**. Ces résultats restent à améliorer puisque les images réduites obtenues par l'AEC sont floues et leur qualité visuelle reste médiocre. Ceci est dus surement à l'arrière-plan de l'image qui a un impact sur la qualité de l'image et sur la vérification. L'arrière-plan contient beaucoup d'information redondante qui est inutile et induit des résultats erronés pour la classification et la reconnaissance. Ce qui nous a poussé à éliminer l'arrière-plan par la détection et le découpage des images de visage.

4.7.3.3.2 Résultats de l'AEC sur les images détectées et découpées

A ce stade du travail trois expériences sont menées sur un échantillon de 10 personnes au total un échantillon de 314 images. La BDD de test représente 0.2% de la totalité des images choisi d'une façon aléatoire, le reste des images est consacré à la BDD apprentissage et une expérience finale sur la totalité de la base (comme dans le cas des expériences précédentes). La BDD est soumise à un prétraitement dont le but de garder seulement les image de visage contenant les informations essentielles. Dont on utilise en entrée des images de taille 256x256x3 qui définit la hauteur, la largeur et les trois canaux de couleur R, G, B de l'image. Les étapes de cette phase sont comme suit :

1. Une phase de détection et de prétraitement
2. Nous soumettons l'image au système AEC. L'architecture de ce modèle est composée :
    i. D'une image d'entrée redimensionnée 256x256x3
    ii. La partie encodeur contient quatre couches composées de convolution et de mise en commun (Pooling) chacune.

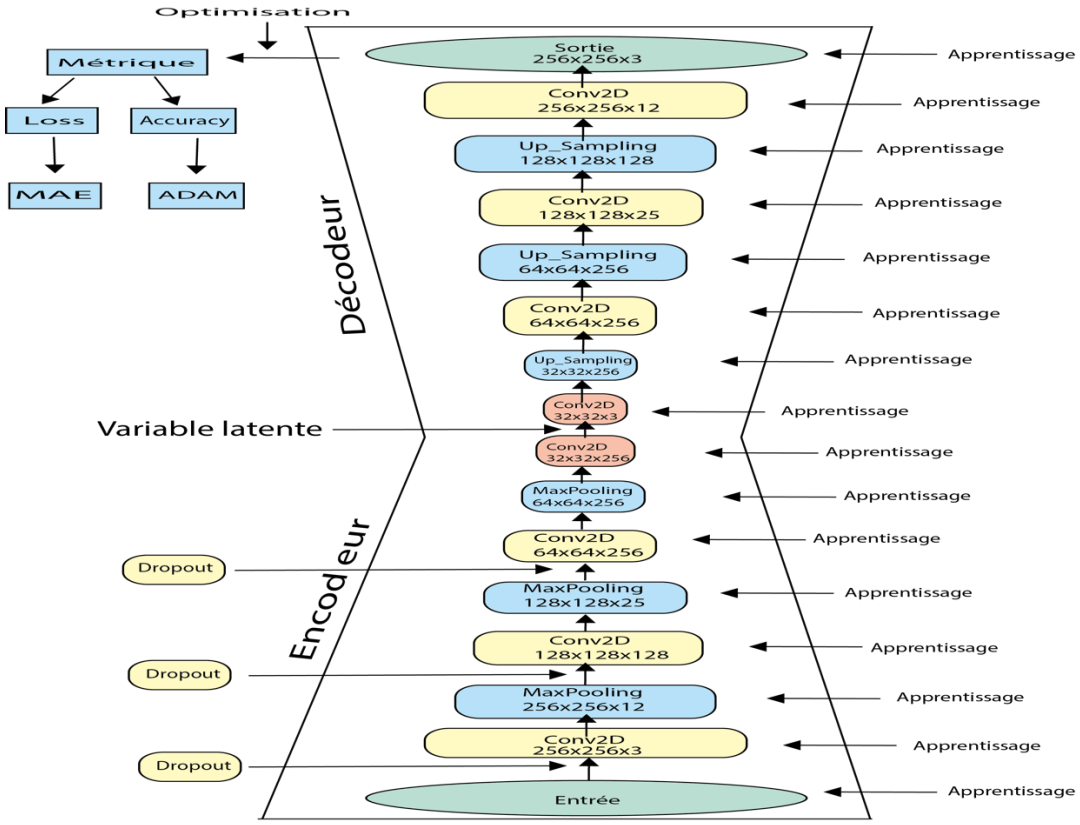


Figure 4. 34 Architecture de L'AEC avec ajout d'une couche et dropout

Les couches convolutives utilisent un filtre de dimension (3x3) et une fonction d'activation ReLu et un padding (same), c'est-à-dire la taille de sortie est la même que la taille d'entrée

Pour les couches de pooling nous avons utilisé un filtre de dimension (2x2) et un padding.

Pour éviter le sur-apprentissage la méthode de régularisation (dropout) est utilisée.

Une couche de convolution est ajoutée pour avoir comme sortie une image avec les trois canaux de couleur.

3. La partie décodeur est composée de 3 couches : 4 couches de déconvoultion et 3 couches de démembrement (unpooling). Elle prend comme entrée les données encodées (image réduite) de la partie encodeur pour reconstruire l'image sortie tout en utilisant des fonctions d'optimisation tels que MAE et ADAM en comparant l'image prédite avec l'image d'entrée. Ce qui nous intéresse c'est la sortie de l'encodeur qui contient les images réduites dont la partie du décodeur est utilisée pour forcer l'encodeur à bien converger c'est à dire donner de bons résultats de réduction et d'extraction de caractéristiques ce qui diffère d'un simple réseau de neurones convolutionnel qui utilise une architecture ressemblant à la partie encodeur.

Les résultats de l'expérience sont donnés par la **figure 4.35**

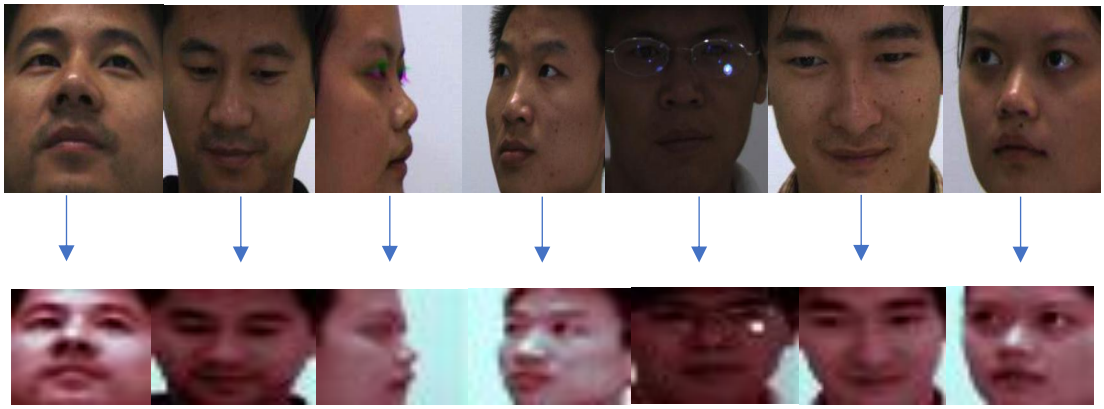


Figure 4. 35 Résultats de réduction (256x256x3 vers 32x32x3) BDD CASIA2DV4

Figure 4. 36 Résultats de réduction (256x256x3 vers 64x64x3) BDD CASIA2DV4

Figure 4. 37 Résultats de réduction (256x256x3 vers 128x128x3) BDD CASIA2DV4

Maintenant que nous avons réussi à améliorer la qualité de l'image par la détection et le découpage les **figures 4.35, 4.36 et 4.37** le confirment bien. Nous pouvons dire que l'AEC joue bien son rôle dans la réduction des données en conservant une qualité exceptionnelle de l'image et avec les paramètres de performances suivants : accuracy = 0.9262, val_acc=0.9711, loss = 0.0230, val_loss : 0.0221 avec un taux de réduction = 1koctets / 922koctets = 0.00108459869 ce qui est très satisfaisant dans un temps de calcul = 75.95132803916931 seconds.

4.7.3.4 Extraction vectorielle par VGG et classification pour la reconnaissance

Fondamentalement, nous appliquons l'apprentissage par transfert et nous utilisons les poids préformés du modèle VGG Face. De plus, nous utilisons le modèle comme encodeur

automatique pour représenter les images sous forme de vecteurs. VGG-Face est plus profond que le DeepFace de Facebook, il comporte 22 couches et 37 unités profondes. La structure du modèle VGG-Face est illustrée par la **figure 4.38**.

Figure 4. 38 Structure du modèle VGG Face [53]

Notez que les images d'entrée pour le modèle VGG sont de taille 224x224x3. Ici, les 3D dimension font référence aux nombres de canaux ou de couleurs RGB. De plus, nous utilisons la fonction preprocess_input qui normalise les entrées afin que les images d'entrées soient compatibles avec le modèle VGG sans affecter la taille et la qualité de l'image. L'architecture VGG-Face est visualisée par la **figure 4.39** pour plus de clarté.

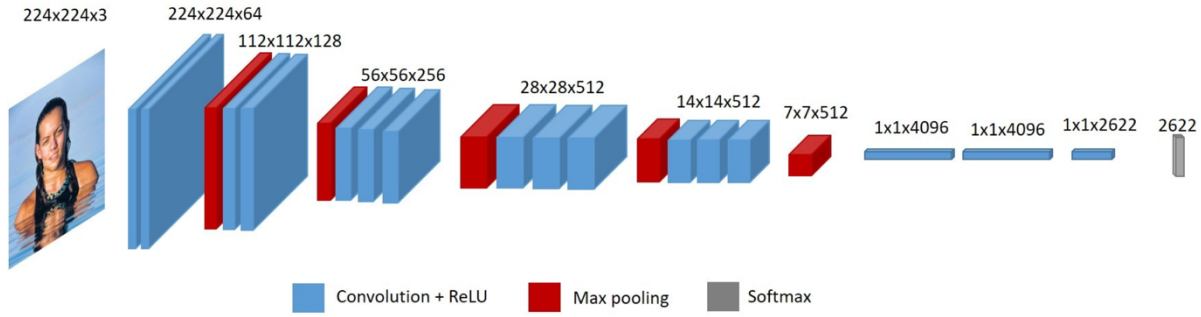


Figure 4. 39 Architecture de VGG-Face [53]

Les données VGGFace sont disponibles sous le chemin *vgg_face_matconvnet / data / vgg_face.mat* et sont compatibles matlab. Les poids sont pré-entraînés pour Keras et on peut les trouver dans le lien *model.load_weights* ('vgg_face_weights.h5'). De cette manière, nous pouvons représenter les images sous forme de vecteur à 2622 dimensions, comme illustré ci-dessus. Enfin, nous utilisons la couche précédente de la couche de sortie pour la représentation. De cette manière, nous pouvons représenter les images sous forme de vecteurs à 2622 dimensions.

Les images sont donc représentées sous forme de vecteurs. Nous déciderons que les deux images sont identiques ou non, en comparant ces représentations vectorielles par la mesure de similarité. Si les deux images représentent la même personne, la mesure doit être petite. Sinon, la mesure devrait être grande si deux images sont de différentes personnes. Pour cela, nous devons trouver la distance entre ces vecteurs. Il existe deux manières courantes de trouver la

distance de deux vecteurs : la distance cosinus et la distance Euclidienne. La distance cosinus est égale à 1 moins la similarité. Quelle que soit la mesure que nous adaptons, elles servent toutes à trouver des similitudes entre les vecteurs. La similarité du cosinus devrait être inférieure à 0,16 ou bien la distance euclidienne devrait être inférieure à 100 sur la base de nos observations. Les seuils peuvent être ajustés en fonction du problème traité. C'est à nous de choisir la bonne mesure de similarité.

Pour la reconnaissance de visage nous menons plusieurs expériences et celles qui présentent les meilleurs résultats sont retenues. Dans ce qui suit, nous présentons l'essentiel de ces expériences.

4.7.3.5 Résultats de la reconnaissance de visage par VGG sans détection et découpage

Dans toute la série d'expériences suivantes nous travaillons sur les images originales brutes sans détection et découpage avec une réduction de taille 32x32x3.

- ***Cas de l'architecture trois couches de l'encodeur***

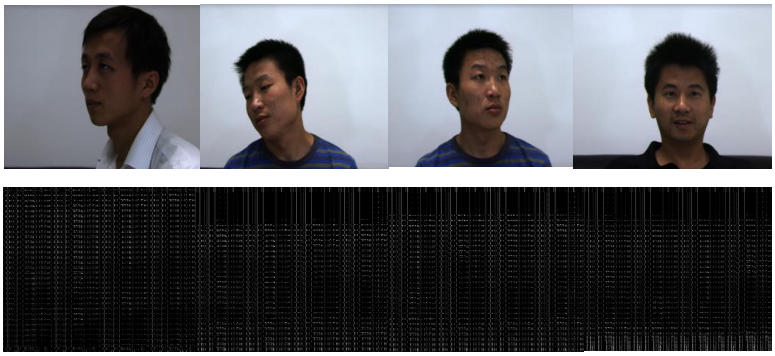


Figure 4. 40 Résultat pour la sortie codée (50epoch,1batch) par l'AEC

Nous remarquons que l'image réduite encodée est en niveaux de gris et non descriptive sur le plan qualité visuelle.

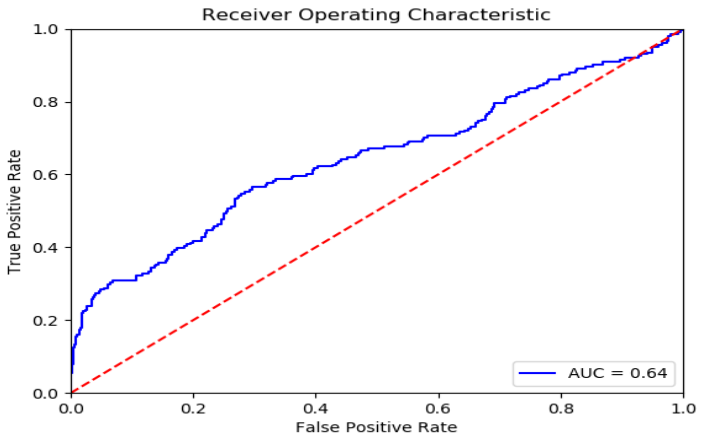


Figure 4. 41 Courbe ROC du SRV avec AEC trois couches par VGG FACE

Tableau 4. 4 Temps d'exécution et AUC par l'AEC trois couches BDD CASIA2DV4

Temps d'exécution	AUC
t = 43,88 seconds	0.64

Discussion

Dans cette expérience, la qualité visuelle de l'image réduite et codée est mauvaise, donc la majorité de l'information n'est pas conservée ce qui explique la courbe ***ROC*** obtenue ***(AUC=64%)***. Cette expérience est couteuse en temps d'exécution égale à ***43,88 secondes***. Pour toutes ces raisons, nous avons pensé à approfondir l'apprentissage de notre modèle en rajoutant une couche a l'architecture de l'AEC.

- ***Cas de l'architecture quatre couches de l'encodeur***

Dans ce cas, nous utilisons un lot de données égale à 2 et un nombre d'époque égale à 10.

Figure 4. 42 Résultats de la réduction avec 2 batch et 10 epoch par l'AEC

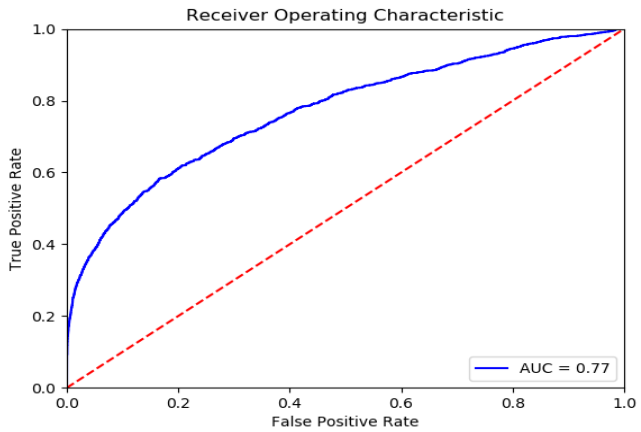


Figure 4. 43 Courbe ROC du SRV basée sur l'AEC avec 2 batch et 10 epoch par VGG FACE

Tableau 4. 5 Résultats du temps d'exécution Et l'AUC par l'AEC (sans détection et découpage taille 32x32x3) BDD CASIA2DV4

Temps d'exécution	AUC
t = 26,004 seconds	0.77

Discussion

Cette expérience montre une amélioration de la qualité de l'image réduite et codée ainsi qu'une meilleure courbe ***ROC*** avec ***AUC=77%***. Le temps d'exécution dans ce cas est à *26,004* secondes.

- ***Cas de réduction par l'AEC avec (1batch,1epoch, 1dropout)***

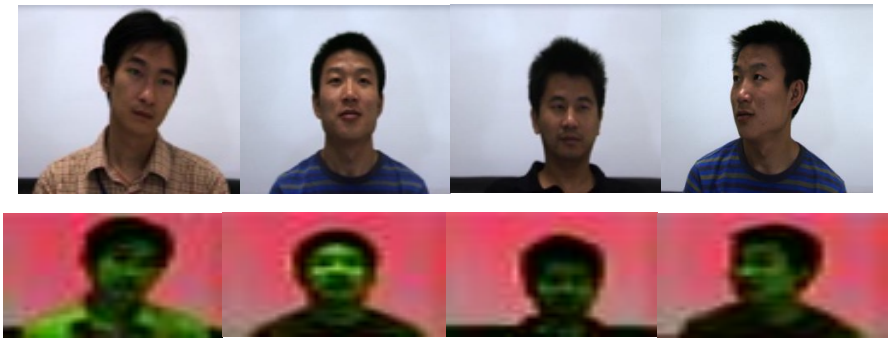


Figure 4. 44 Résultats de la réduction avec (1batch,1epoch,1dropout)

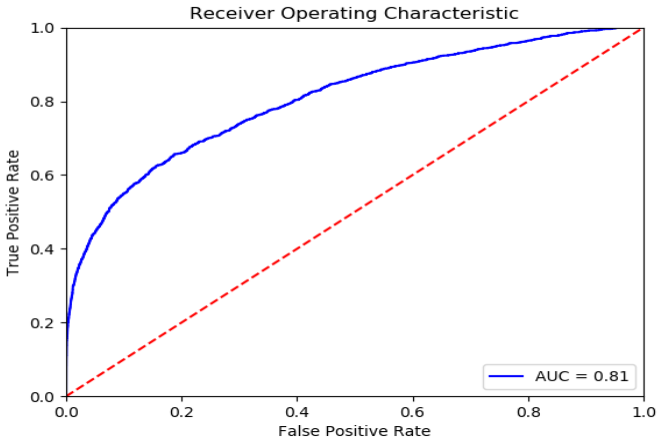


Figure 4. 45 Courbe ROC du SRV basée sur l'AEC avec (1batch,1epoch,1dropout) par VGG FACE

Nous remarquons qu'en réduisant le nombre d'époques et de batchs à 1 en rajoutant un dropout le paramètre de performance **AUC = 81%** pour la reconnaissance de visage augmente mais reste insuffisant. Pour cela, dans les expériences suivantes, nous procédons à la détection et au découpage des images pour éliminer l'arrière-plan des images de visage qui présente une information inutile affectant le taux de reconnaissance.

4.7.3.6 Résultats de la reconnaissance de visage par VGG avec détection et découpage

Dans toutes les expériences suivantes nous travaillons sur les images détectées et découpées ayant des tailles varient entre 32x32x3 et 128x128x3. Nous tenons à préciser que les paramètres (1batch,1epoch et le dropout) de l'AEC demeurent la même dans tous ce qui suit.

- ***Cas d'une image de taille** 32x32x3 **réduite par l'AEC***

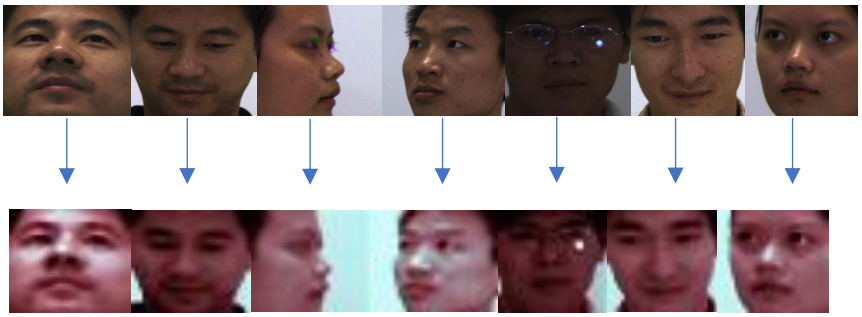


Figure 4. 46 Résultats de la réduction AEC taille (32x32x3)

Nous obtenons dans ce cas une courbe ***ROC acceptable** et **AUC = 0.85***. D'après les résultats présentés par la figure 4.46; nous pouvons dire que les ***données*** sont effectivement ***bien réduites***, mais le système de reconnaissance de visage reste ***moins performant*** *voir tableau 4.6*.

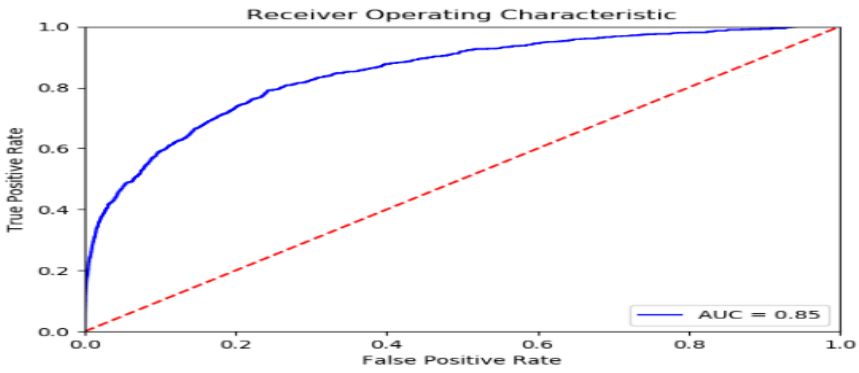


Figure 4. 47 Courbe ROC du SRV basée sur l'AEC (32x32x3) par VGG FACE

Tableau 4. 6 Résultats du temps d'exécution Et l'AUC par l'AEC (32x32x3) BDD CASIA2DV4

Temps d'exécution	AUC
t =11,54 seconds	**0.85**

- ***Cas d'une image de taille** 64x64x3 **réduite par l'AEC***

Figure 4. 48 Résultats de la réduction AEC taille (64x64x3)

Les résultats montrent une augmentation de taux de reconnaissance une courbe ROC plus performant et **AUC = 96%** ainsi que ***la qualité visuelle de l'image est réduite et plus nette*** D'après les résultats présentés par la **figure 4.48**; nous pouvons affirmer que les ***données*** sont effectivement **très *bien réduites et l'extraction des caractéristiques et très bien faite mais le temps d'exécution reste plus élevé que l'expérience précédente*** *voir tableau 4.7.*

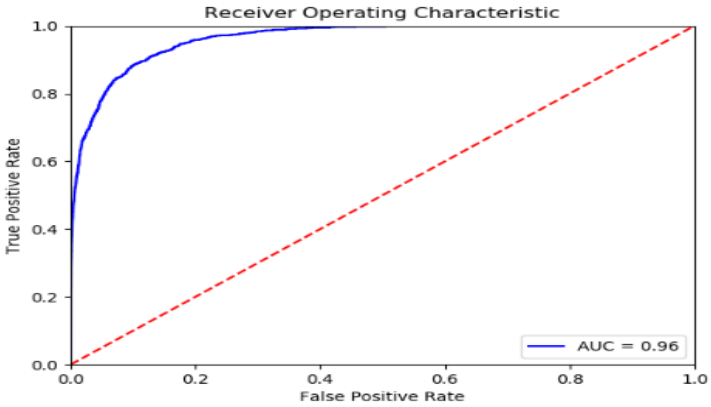


Figure 4. 49 Courbe ROC du SRV basée sur l'AEC (64x64x3) par VGG FACE

Tableau 4. 7 Résultats du temps d'exécution Et l'AUC par l'AEC (64x64x3) BDD CASIA2DV4

Temps d'exécution	AUC
t =19,42 seconds	0.96

- ***Cas d'une image de taille** 128x128x3 **réduite par l'AEC***

Figure 4. 50 Résultats de la réduction AEC taille (128x128x3)

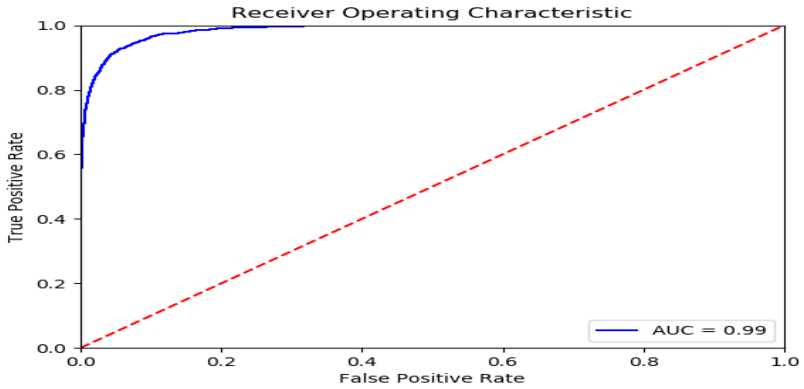


Figure 4. 51 Courbe ROC du SRV basée sur l'AEC (128x128x3) par VGG FACE

Dans cette expérience l'image est prétraitée de tels sorte quel soit compatible avec l'entrée du modèle VGG ainsi la valeur du seuil est modifiée de 0.16 à 0.18 ce qui permet d'avoir un temps inferieur par rapport aux expérience précédente et une courbe ROC excellente et ***AUC=99%*** en gardant les images bien réduites et une qualité visuelle de l'image plus nette avec un taux de reconnaissance plus performant voir tableau 4.8. Le temps de vérification du visage est nettement diminué avec t = 9.14 secondes.

Tableau 4. 8 Résultats du temps d'exécution et l'AUC par l'AEC (64x64x3) BDD CASIA2DV4

Temps d'exécution	AUC
t = 9.14 secondes	***0.99***

4.8 Étude comparative avec des travaux récents

4.8.1 Application du DAEC pour le débruitage des images

Dans le tableau 4.9 nous comparons les résultats obtenus par notre approche basée sur le DAE avec certains travaux récents sur le débruitage d'image.

Tableau 4. 9 Étude comparative de l'expérience débruitage avec l'état de l'art

Auteurs	Années	Méthodes	Base de données	Résultats
W. Wang et al [3]	2014	CMU PIE	GAE	Err $_{(dGAE- MFA / dGAE-LE)}$ = 1.1% Err $_{(dGAE-PCA)}$ = 3.5%
L.Gondara [54]	2016	CDAE	(MMM) et (DX)	Loss$_{(training/100epoch-10batch)}$ = 2.7% Val_loss$_{(test/100epoch-10batch)}$ = 2.75%
Notre approche	***2019***	***CASIA2DV4*** ***(tr=80% test=20%)***	***DAEC***	***Loss$_{(training/100epoch-10batch)}$ = 1.40%*** ***Val_loss$_{(test/100epoch-10batch)}$ = 1.65%*** ***Accuracy = 90.3%***

AECRV : Auto-encodeur convolutionnel pour la reconnaissance de visage; MMM : Mini-MIAS database of mammo- grams; DX : Dental radiography database; CDAE : (convolutional denoising autoencoders); Val_loss : Validation loss ; tr : training set.

4.8.2 Application du AEC pour la reconstruction des images

Dans le tableau 4.10 ci-dessous nous présentons une étude comparative des résultats de notre travail avec ceux de l'état de l'art.

Tableau 4. 10 Étude comparative de l'expérience reconstruction avec l'état de l'art

Auteurs	Années	Base de données	Méthodes	Résultats
C. TAN et al [55]	2010	ORL (tr=50 test=50)	SAE-MBP PCA	$Loss_{(training/230epoch)}$ = 10 % $Val\_loss_{(test/230epoch)}$ = 9.1092 % $Acc_{(SAE-MBP)}$ = 80.5 % Loss = 4.921 % $Acc_{(PCA)}$ = 88 %
J. WANG et al [30]	2012	MNIST (tr=70% test = 30%)	FAE UFAE PCA	$Loss_{(training/1000epoch)}$ = 3.51% $Val\_loss_{(test/1000epoch)}$ = 35.1535% $Loss_{(training/1000epoch)}$ = 3.23% $Val\_loss_{(test/1000epoch)}$ = 32.0443% $Loss_{(training/1000epoch)}$ = 11.0443% $Val\_loss_{(test/1000epoch)}$ = 19.2997%
Notre approche	***2019***	**Faces95** ***CASIA2DV4*** ***(tr=80% ; test=20%)***	***RAEC***	**$Loss_{(training/50epoch-1batch)}$ = 1.40%** **$Acc_{(training/50epoch-1batch)}$ = 97.92%** **$Val\_loss_{(test/50epoch-1batch)}$ = 1.16%** **$Val\_Acc_{(test/50epoch-1batch)}$ =98.28%** **$Acc_{(training/50epoch-1batch)}$ = 91%** **$Loss_{(training/50epoch-1batch)}$ = 1%** **$Val\_loss_{(test/50epoch-1batch)}$ = 0.09%** **$Val\_Acc_{(test/50epoch-1batch)}$ =90%**

SA-MBP : Stacked autoencoder multiple backpropagation ; FAE : Folded autoencoder ; UFAE : Unfolded autoencoder

Nous avons montré que les auto-encodeurs peuvent être utilisés efficacement pour des applications de reconstruction et de reconnaissance d'images. Les résultats expérimentaux montrent que les auto-encodeurs sont plus performants que la PCA surtout dans le cas ou les échantillons d'entraînement fournis sont volumineux. Ils peuvent même s'adapter à tous les types de base de données : petite taille, grand taille, environement réel (in the wild)…etc

4.8.3 Application du AEC pour la Reconnaissance des images

4.8.3.1 Étude comparative de l AEC avec la PCA

Nous rappelons que dans la première étape du travail, nous procédons à la réduction des images par l'AEC et dans la figure 4.52 montre une comparaison entre notre méthode proposée et la PCA de l'état de l'art .

(a) Image originale; (taille = 175KB) (b) Image réduite; (taille = 107KB) (c) Image réduite (taille = 101KB) (a) Image originale; (taille = 156KB) (b) Image réduite ; (taille = 101KB)

Figure 4. 52 Réduction par **PCA** [56]

(a)Image originale (***taille = 922koctets***) (b) Image détectée et découpée (c) Image réduite (***taille = 1koctets***)

Figure 4. 53 Notre approche de réduction par **AEC**

D'aprés les figures nous constatons que l'AEC est trés efficace pour la réduction. La figure 4.53 montre bien la qualité de l'image réduite avec des taux extraordinaire.

4.8.3.2 Étude comparative de l'AEC pour la reconnaissance avec l'état de l'art

De nombreux travaux récents sur la reconnaissance des visages ont proposé de nombreuses variantes d'architecture basée sur l'apprentissage profond pour les visages, et nous évaluons certains de ces choix de modélisation.

Tableau 4. 11 Étude comparative de l'expérience RV basée sur l'AEC avec l'état de l'art

Auteurs	Année	Base de données	Méthode	Résultats
P.Hiremath et al [57]	2013	CASIA3D	Depth+PCA+ AdaBoost	VR= 99.50 %
K.Simonyan et al[58]	2013	LFW	Fisher Vector Faces	Acc = 93.10%
W. Wang et al [3]	2014	CMU PIE	GAE	Err $_{(dGAE- MFA / dGAE-LE)}$ = 1.1% Err $_{(dGAE-PCA)}$ = 3.5%
O.Parkhi et al [59]	2015	LFW (4Mimages)	DeepFace	Acc = 97.35%
F.Schroff [60]	2015	LFW (200Mimages)	FaceNet	Acc = 98.87%
F.Schroff [60]	2015	LFW (200Mimages)	FaceNet+Alignement	Acc = 99.63%
S.Hafez et al [61]	2015	CASIA 2D Face tr=4x123 test=3x123 ORL Cropped YaleB	Selected Gabor+LDA	RR= 99.17% RR= 98.33% RR=99.33%
F.Li et al [62]	2017	ORL	DAE	Loss = 5% RR = 97.5%
J.Li et al [28]	2018	FRGC v2.0 LFW	C2D-CNN	Acc $_{(FD=50)}$ = 91% Acc $_{(FD=50)}$ = 93% RR $_{(distance\ euclidienne)}$ = 95.9%
Notre approche	***2019***	***CASIA2DV4 (tr=80% test =20%)*** ***CASIA2DV4*** **Tr= 4 Test=9**	***AECRV***	***Loss$_{(training/1epoch-1batch)}$ = 2%*** ***Acc$_{(training/1epoch-1batch)}$ = 92%*** ***Image réduite : 32x32x3 AUC = 91%*** ***Image réduite : 64x64x3 AUC = 96%*** ***Image réduite : 128x128x3 AUC = 99%*** ***Image réduite : 64x64x3 AUC = 100%***

Premièrement, le tableau 4.11 montre que les résultats de l'auto-encodeur diffèrent en effet des autres méthodes de réduction de dimensionnalité. Dans certains cas, l'auto-encodeur réduit non seulement la dimension, mais peut également détecter des structures répétitives. Nous pensons qu'il s'agit d'une bonne propriété pour de nombreuses applications. Probablement les situations

avec des structures répétitives, le codage automatique est plus approprié. Nous obtenons des résultats comparables à l'état de la technique tout en nécessitant moins de données (que DeepFace et FaceNet) et en utilisant une architecture de réseau plus simple (que DeepID-2,3). Ainsi les résultats DeepID3 s'appliquent au jeu de tests et corrigent les erreurs d'étiquette qui n'a pas été fait par une autre méthode. Certes, le temps de calcul rapide est également une préoccupation.

Conclusion

Dans ce chapitre, nous commençons par l'implémentation d'un auto-encodeur convolutif destiné au débruitage, à la reconstruction et à la reconnaissance de visage. Tout d'abord, nous menons une analyse des défauts d'architecture du modèle AEC est menée en observant son comportement grâce l'évolution des graphes caractérisant son bon fonctionnement. Les paramètres sont ajustés un à un afin de comprendre l'impact de chacun d'eux sur la performance finale du système. Après avoir trouvé l'architecture convenable nous optimisons au maximum le réseau en augmentant le nombre d'époques pour un meilleur apprentissage afin d'obtenir la performance maximale que peut atteindre notre modèle.

Ensuite, un système de reconnaissance des visages basé sur l'apprentissage en profondeur a été mis en place. Dans le système proposé, AECRV (Auto encodeur Convolutionnel pour la Reconnaissance de Visage), un réseau en profondeur a été créé pour apprendre la représentation des caractéristiques des images de visage par extraction automatique des caractéristiques du visage et la réduction de dimensionnalité pour avoir un temps d'exécution minimum.

Pour atteindre un système robuste, deux classificateurs ont été utilisés, calcul de métrique (distance Euclidienne) et mesure de similarité. Les expériences de la méthode proposée fournissent des résultats de précision comparables en reconnaissance faciale et peuvent être généralisés à d'autres tâches de reconnaissance grâce à sa capacité à extraire des caractéristiques robustes.

Conclusion générale

Grâce aux Deep Learning et les auto-encodeurs, l'avenir de l'intelligence artificielle et le domaine de la reconnaissance de visage est prometteur.

Dans ce travail, nous nous interessons a l auto-encodeur qui est un puissant outil de réduction de la dimensionnalité, et proposons un système de reconnaissance de visage basée sur un auto-encodeur convolutionnel. La structure de niveau d'implémentation ainsi que les algorithmes sont présentés et analysés de manière détaillée.

Afin d'aboutir aux résultats attendus, nous avons passé beaucoup de temps dans la recherche et l'étude des mémoires de Master 2, des publications et articles pour voir ce qui se fait de mieux en matière dans la reconnaissance de visage et la réduction de dimensionnalité pour pouvoir choisir et concevoir notre propre modèle.

Les travaux réalisés dans ce projet de fin d'étude Master 2 sont décrits selon trois implémentations comme suit :

1. ***Débruitage***

   Le débruitage des images est une étape de prétraitement importante dans l'analyse d'images. Différents algorithmes ont été proposés au cours des trois dernières décennies avec des performances de débruitage variables. Plus récemment, après avoir surpassé toutes les méthodes conventionnelles, les modèles fondés sur l'apprentissage en profondeur ont été très prometteurs. Ces méthodes sont toutefois limitées pour les besoins en grands échantillons d'entraînement et pour des coûts de calcul élevé. Dans ce travail, nous montrons qu'en utilisant une petite taille d'échantillon, des auto-codeurs de débruitage construits à l'aide de couches convolutives peuvent être utilisés pour un débruitage efficace des images. Des images hétérogènes peuvent être combinées pour augmenter la taille de l'échantillon et améliorer les performances de réduction de bruit. Le plus simple des réseaux peut reconstruire des images avec des niveaux de bruitée si élevés que le bruit et le signal ne sont pas différenciables à l'œil humain. Contrairement à ce que nous pensons, nous avons montré qu'une bonne performance de débruitage pouvait être obtenue à l'aide de petits jeux de données d'entraînement. Des échantillons d'entraînement d'environ 300 suffisent à une bonne performance.

2. ***Reconstruction***

   Un auto-encodeur consiste à copier son entrée dans sa sortie mais la chose la plus importante dans un auto-encodeur c'est de réduire la dimensionnalité des données en entrée avant d'être recréer. La reconstruction est utile dans certains travaux tels que les Deep Fake

et la génération des nouvelles images à partir d'autres en appliquant certains bruits dans l'espace latent de l'auto-encodeur ou bien de créer des nouvelles bases à partir d'autres bases. Notre approche dans la reconstruction consiste simplement à recréer l'image originale afin de se familiariser avec l'auto-encodeur et bien déterminer les paramètres essentiels et l'architecture du modèle, certains travaux sont étudiés et comparés avec notre approche.

3. Reconnaissance de Visage

Dans ce travail, nous avons exploré le domaine de la reconnaissance de visage qui comme tous les autres domaines de l'intelligence artificielle ont connu une évolution majeure depuis l'apparition du Deep Learning.

Dans cette partie, nous réalisons les tâches suivantes :

- La proposition d'un système complet et robuste de reconnaissance des visages, qui inclut également la détection des visages et peut traiter divers jeux de données de visages contenant des visages sans contraintes avec des variations de pose, d'occlusions, d'expression et d'éclairage ;
- La prise en compte de la propriété Dropout qui réduit efficacement le nombre de neurones dans le processus de formation, de sorte que la possibilité de sur-apprentissage est considérablement réduite dans un modèle entraîné. Dans cette étude, cette contrainte est spécifiée pour que le modèle entraîné repose sur le point d'équilibre qui n'est ni insuffisant ni excessif. Ceci réduit également considérablement les coûts de calcul.
- L'extraction des caractéristiques plus robustes des variances des images de visage, un auto-encodeur optimisé est construit en plus du processus de débruitage.

On peut constater que nous obtenons des résultats comparables à ceux d'autres réseaux CNN profonds et aux méthodes de pointe (FaceNet et DeepFace [27]) en utilisant beaucoup moins de données, même lorsque les données d'apprentissage proviennent d'une autre base de données. Cette conclusion peut s'appliquer à de nombreuses autres tâches.

Notre méthode proposée est évaluée sur l'ensemble de données universelles et publics CASIA2DV4 en utilisant deux classificateurs robustes ; distance Euclidienne et la similarité pour obtenir des résultats plus précis.

Pour finir, avant de passer aux perspectives, ce travail nous a permis d'enrichir nos connaissances sur les réseaux de neurones artificiels, le Deep Learning particulierement l'AEC et ses applications notamment dans le domaine de la reconnaissance de visage ainsi que les

différentes méthodes de réduction de dimensionnalité. Le temps passé à lire des articles nous a servi d'une bonne initiation à la recherche.

Nous comparons les performances de notre approche avec celle de l'état de l'art. Nous validons notre travail sur le jeu de données universelle CASIA2DV4. Les résultats de l'implémentation confirment que la structure de L'AEC pourrait non seulement réduire le temps de calcul, mais également améliorer les performances de généralisation. Il existe plusieurs axes de recherche futurs intéressants sur ce sujet. Premièrement, notre travail est principalement axé sur le développement d'un nouveau cadre et de nouveaux algorithmes pour la réduction de la dimensionnalité, ainsi que l'amélioration d'un système de reconnaissance de visage. Le cadre proposé devrait être approfondi sous l'angle de l'analyse théorique, comme la capacité d'apprentissage, la complexité et la robustesse de l'apprentissage. Deuxièmement, bien que l'implementation présentée dans ce travail montre des résultats prometteurs, des expériences à grande échelle, des ensembles de données plus complexes ainsi que des tests statistiques sont nécessaires pour évaluer pleinement la capacité du nouveau cadre. Troisièmement, notre discussion ne se limite qu'à la technologie auto-encodeur dans ce travail. De nombreux nouveaux cadres et techniques ont été développés et appliqués avec succès dans la réduction de la dimensionnalité, tels que le réseau de convolution, l'analyse multidimensionnelle MPCA…. Par conséquent, il serait très intéressant de voir comment le cadre proposé fonctionne lorsqu'il est intégré à ces techniques.

Comme perspectives nous pouvons citer :

- Implémentation de l'auto-encodeurs dans le domaine du Big data ainsi que l'imagerie médicale et l'imagerie faciale 3D.
- Tester sur de nouvelles bases en environnement réel.
- Segmentation et génération des images.
- Association de l'AEC à la multidimensionnalité.

Bibliographie

[1] X. Liu et al. VIPLFaceNet: an open source deep face recognition SD,Front, Comput. Sci, 2017, 11: 208.

[2] MA. Yahiaoui. Le Tenseur Hybride Local + Gabor pour la Reconnaissance de Visage, mémoire de master, Université Mohamed Khider Biskra,2017

[3] W. Wang et al. "Generalized Autoencoder: A Neural Network Framework for Dimensionality Reduction," *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, Columbus, OH, 2014, pp. 496-503. doi: 10.1109/CVPRW.2014.79

[4] A. Majumdar et al. "Face Verification via Class Sparsity Based Supervised Encoding," in *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 1 June 2017, vol. 39, no. 6, pp. 1273-1280.

[5] M. Belahcene. *Authentification et identification en biométrie*. Thèse de doctorat, Université Mohamed Khider Biskra, 2013.

[6] P. Baldi. "Autoencoders, unsupervised learning, and deep architectures." Proceedings of ICML workshop on unsupervised and transfer learning. 2012.

[7] G. Brigot. Prédire la structure des forêts à partir d'images PolInSAR par apprentissage de descripteurs LIDAR. Traitement du signal et de l'image. Université Paris-Saclay Français. <NNT : 2017SACLS584>, 2017.

[8] A. Kapoor et al. Convolutional Neural Networks, Nurture AI, 2018, 10.13140/RG.2.2.33935.28328.

[9] Y. Lecun et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998, vol. 86, no 11, p. 2278-2324.

[10] A. Nehemiah. Face Recognition, MathWorks, extrait de https://www.mathworks.com/discovery/reconnaissance-faciale.html

[11] Y. Benoit. Deeplearning, Investor Relations, Apr 7, 2018 exrait de https://www.slideshare.net/featured/category/investor-relations

[12] R. Rakotomalala. Deeplearning Perceptrons simples et multicouches, Cours en graduation université lumière lyon 2, extrait de https://eric.univ-lyon2.fr/~ricco/cours/slides/reseaux_neurones_perceptron.pdf

[13] A. Bechouche. *Utilisation des techniques avancées pour l'observation et la commande d'une machine asynchrone : application à une éolienne*, PhD Thesis. Université Mouloud Mammeri, 2013.

[14] Y. Djeriri. Les Réseaux de Neurones Artificiels, Université Djilali Liabès Sidi- Bel-Abbès. extrait de https://www.researchgate.net/publication/319939107_Les_Reseaux_de_Neurones_Artificiels, Septembre 2017

[15] S. Nachez. Directeur de la publication Actu IA ,Qu'est-ce que le Deep Learning. Extrait de https://www.actuia.com/actualite/quest-deep-learning/, 5 février 2018

[16] JP. Forestier. Algorithmes DL inexplicables, Extrait de http://www.beaubiophilo.com/2018/08/ia.07-le-chaos.html, 25 Aout 2018.

[17] GE. Hinton et al. Reducing the dimensionality of data with neural networks. *science*, 2006, 313.5786: 504-507

[18] M. Germain et al. Made: Masked autoencoder for distribution estimation. In: *International Conference on Machine Learning*. 2015. p. 881-889.
[19] Y. Zhang. A Better Autoencoder for Image: Convolutional Autoencoder.
[20] A. Azarang et al. Convolutional Autoencoder-Based Multispectral Image Fusion. *IEEE Access*, 2019, 7: 35673-35683.
[21] M.Runfeldt, iSee: Using deep learning to remove eyeglasses from faces , extrait de https://blog.insightdatascience.com/isee-removing-eyeglasses-from-faces-using-deep-learning-d4e7d935376f Melissa Runfeldt, 2016.
[22] M. Usman et al. Using deep autoencoders for facial expression recognition. In: *2017 13th International Conference on Emerging Technologies (ICET)*. IEEE, 2017. p. 1-6.
[23] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 1980, 36.4: 193-202.
[24] F. Savard. Réseaux de neurones à relaxation entraînés par critère d'autoencodeur débruitant, en vue de l'obtention du grade de Maître ès sciences (M.Sc.) en informatique, université de Montréal, Aout 2011.
[25] D. Hubel et al. Uniformity of monkey striate cortex: a parallel relationship between field size, scatter, and magnification factor. *Journal of Comparative Neurology*, 1974, 158.3: 295-305.
[26] Y. Lecun et al. LeNet-5, convolutional neural networks. *URL: http://yann. lecun. com/exdb/lenet*, 2015, 20.
[27] M. Wang, Mei et al. Deep face recognition: a survey. *arXiv preprint arXiv:1804.06655*, 2018.
[28] J. LI et al. Robust face recognition using the deep C2D-CNN model based on decision-level fusion. *Sensors*, 2018, 18.7: 2080.
[29] L. Dang et al. Deep Learning Based Computer Generated Face Identification Using Convolutional Neural Network. *Applied Sciences*, 2018, 8.12: 2610.
[30] J. Wang, et al. A folded neural network autoencoder for dimensionality reduction. *Procedia Computer Science*, 2012, 13: 120-127.
[31] X. Zhang et al. Fusing heterogeneous features from stacked sparse autoencoder for histopathological image analysis. IEEE journal of biomedical and health informatics, 2016, 20.5: 1377-1383.
[32] G. Ma et al. Multi-feature fusion deep networks. *Neurocomputing*, 2016, 218: 164-171.
[33] T. Spencer et al. Dimensionality reduction of mass spectrometry imaging data using autoencoders. In: *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2016. p. 1-7.
[34] S. Petscharing et al; CHATZICHRISTOFIS, Savvas. Dimensionality reduction for image features using deep learning and autoencoders. In: *Proceedings of the 15th International Workshop on Content-Based Multimedia Indexing*. ACM, 2017. p. 23.
[35] P. Korshunov et al. DeepFakes: a New Threat to Face Recognition? Assessment and Detection. *arXiv preprint arXiv:1812.08685*, 2018.

[36] J. Hui. How deep learning fakes videos (Deepfake) and how to detect it, 2018, extrait de https://medium.com/@jonathan_hui/how-deep-learning-fakes-videos-deepfakes-and-how-to-detect-it-c0b50fbf7cb9
[37] Y. Wen et al. Latent factor guided convolutional neural networks for age-invariant face recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016. p. 4893-4901.
[38] Z. Yi et al. Dualgan: Unsupervised dual learning for image-to-image translation. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017. p. 2849-2857.
[39] P. Lemberger et al. LIVRE AU-DELÀ DES CLICHÉS, L'INTELLIGENCE ARTIFICIELLE, EDITEUR WEAVE, Paris, 2017, extrait de https://weave.eu/app/uploads/2017/11/livre-blanc-2018-1.pdf
[40] P. Viola et al. Rapid object detection using a boosted cascade of simple features. *CVPR (1)*, 2001, 1: 511-518.
[41] KM. Manash. Mettre en œuvre des autoencodeurs PCA, Feedforward et Convolutional et les utiliser pour la reconstruction, la récupération et la compression d'images. (9 janvier 2018). extrait de https://blog.manash.me/implementing-pca-feedforward-and-convolutional-autoencoders-and-using-it-for-image-reconstruction-8ee44198ea55
[42] D. Omid et al. Deeppainter: Painter classification using deep convolutional autoencoders. In: *International conference on artificial neural networks*. Springer, Cham, 2016. p. 20-28.
[43] J. Kao, UCLA University of California, Los Angeles , CNN article extrait de https://seas.ucla.edu/~kao/nndl/lectures/cnn.pdf
[44] Perone CS. Machine learning: Cosine similarity for vector space models (part iii). Pyevolve. sourceforge. net/wordpress. 2013.
[45] https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html
[46] http://rinterested.github.io/statistics/softmax.html
[47] M. Lanham. *Learn ARCore-Fundamentals of Google ARCore: Learn to build augmented reality apps for Android, Unity, and the web with Google ARCore 1.0*. Packt Publishing Ltd, 2018.
[48] https://blog.algorithmia.com/introduction-to-loss-functions/ , APRIL 30, 2018
[49] X. Liu et al. Hard negative generation for identity-disentangled facial expression recognition. *Pattern Recognition*, 2019, 88: 1-12.
[50] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.
[51] S. Shams . article Overfitting and Regularization, En provenance du cours de Stanford University Machine Learning Extrait de https://machinelearningmedium.com/2017/09/08/overfitting-and-regularization/
[52] DY. Moualek. Deep Learning pour la classification des images. 2017. PhD Thesis. 07-03-2017
[53] Deep Face Recognition with Keras ,2018, Extrait du Papier originale PARKHI, Omkar M, et al. Deep face recognition. In: *bmvc*. 2015. p. 6., Extrait du site https://sefiks.com/2018/08/06/deep-face-recognition-with-keras/

[54] L. Gondara. Medical image denoising using convolutional denoising autoencoders. In: *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2016. p. 241-246.
[55] CC. Tan et al. Reconstruction and recognition of face and digit images using autoencoders. *Neural Computing and Applications*, 2010, 19.7: 1069-1079.
[56] S. Ng. Principal component analysis to reduce dimension on digital image. *Procedia computer science*, 2017, 111: 113-119.
[57] P. Hiremath et al. 3D face recognition based on depth and intensity gabor features using symbolic PCA and AdaBoost. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 2013, 6.5: 1-12.
[58] K. Simonyan et al. Fisher vector faces in the wild. In: *BMVC*. 2013. p. 4.
[59] M.Parkhi et al. Deep face recognition. In: *bmvc*. 2015. p. 6.
[60] F. Schiroff et al. Facenet: A unified embedding for face recognition and clustering. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015. p. 815-823.
[61] S. Hafez et al. 2d face recognition system based on selected gabor filters and linear discriminant analysis lda. *arXiv preprint arXiv:1503.03741*, 2015.
[62] F. Li et al. Face Recognition Based on Deep Autoencoder Networks with Dropout. In: *2017 2nd International Conference on Modelling, Simulation and Applied Mathematics (MSAM2017)*. Atlantis Press, 2017.

Résumé

La reconnaissance de visage est un sujet brûlant toujours dans la recherche de nouvelles technologies, en raison de nombreux défis de variation, notamment la différence de poses, d'illumination, d'expression, d'occlusion et de scènes. Récemment, les méthodes d'apprentissage en profondeur ont donné des résultats remarquables dans les domaines de la représentation et de la reconnaissance d'images. Ces méthodes extraient automatiquement les caractéristiques pertinentes des images pour réduire la dimension et obtenir une représentation plus utile des données brutes. Dans ce travail, le système proposé est basé sur les auto-encodeurs utilisant la technologie de réseau de neurones profonds pour les tâches de débruitage, reconstruction et reconnaissance. L'auto-encodeur nous permet l'extraction des caractéristiques et leur réduction d'une façon automatique. Dans le système de reconnaissance de visage proposé, il est utilisé à la base pour construire un réseau de neurones qui apprend une approximation d'une fonction d'identité en plaçant les individus sous différentes variantes pour apprendre de fines représentations des entrées. De cette façon, nous produisons des résultats plus significatifs. Pour la tâche de classification, le modèle VGGFACE est utilisé et la classification se base sur deux classifieurs, à savoir la distance Euclidienne et la similarité du cosinus. Les résultats expérimentaux sur des bases de données universelles CASIA2DV4, Faces95, presentant differentes variantes de visage, montrent que le système proposé offre des performances en taux de reconnaissance et en temps de calcul prometteuses. Les résultats sont encourageants et présentent une précision comparable à celle de l'état de l'art récent dans le domaine.

Mots clés : Biométrie ; Deep Learning ; Auto-encodeur ; Classification ; Réseaux de Neurones

ملخص

يعد التعرف على الوجوه دائمًا موضوعًا ساخنًا في البحث عن تقنيات جديدة ، نظرًا للتحديات الكثيرة المتعلقة بالاختلاف ، بما في ذلك الفرق في الأشكال والإضاءة والتعبير والانسداد والمشاهد. في الآونة الأخيرة ، أسفرت أساليب التعلم العميق عن نتائج ملحوظة في مجالات تمثيل الصور والتعرف عليها. تستخرج هذه الطرق تلقائيًا الخصائص ذات الصلة بالصور لتقليل الحجم والحصول على تمثيل أكثر فائدة للبيانات الخام. في هذا العمل ، يعتمد النظام المقترح على أجهزة التشفير التلقائي التي تستخدم تقنية الشبكة العصبية العميقة في مهام تقليل الضوضاء و إعادة التشكيل و التعرف. يسمح لنا التشفير التلقائي باستخراج الميزات وتقليلها تلقائيًا. في نظام التعرف على الوجوه المقترح ، يتم استخدامه في الأساس لبناء شبكة عصبية تتعلم تقريبًا لوظيفة الهوية عن طريق وضع الأفراد تحت أشكال مختلفة لتعلم تمثيلات دقيقة للمدخلات. بهذه الطريقة ، ننتج نتائج أكثر وضوحا. بالنسبة لمهمة التصنيف ، يتم استخدام نموذج فيجيجي فايس ويستند التصنيف إلى مصنفين ، هما المسافة الإقليدية وتشابه جيب التمام تُظهر النتائج التجريبية على قواعد البيانات العالمية ذات المتغيرات المختلفة للوجه كاسيا2دف4 و فايسس95 أن النظام المقترح يوفر الأداء من حيث معدل التعرف ووقت الحساب. النتائج مشجعة ولها دقة مشابهة لتلك الموجودة في المجال العلمي الحديث

الكلمات الرئيسية :القياسات الحيوية ; التعلم العميق ; التشفير التلقائي ; تصنيف الشبكات العصبية

Summary

Face recognition is always a hot topic in the search for new technologies, because of the many challenges of variation, including the difference in poses, illumination, expression, occlusion and scenes. Recently, deep learning methods have yielded remarkable results in the areas of image representation and recognition. These methods automatically extract the relevant characteristics of the images to reduce the size and obtain a more useful representation of the raw data. In this work, the proposed system is based on auto-encoders using deep neural network technology for denoising, reconstruction and recognition tasks. The auto-encoder allows us to extract features and reduce them automatically. In the proposed face recognition system, it is used at the base to build a neural network that learns an approximation of an identity function by placing individuals under different variants to learn fine representations of inputs. In this way, we produce more meaningful results. For the classification task, the VGGFACE model is used and classification is based on two classifiers, namely the Euclidean distance and the cosine similarity. Experimental results on CASIA2DV4, Faces95, universal databases with different face variants, show that the proposed system offers performance in terms of recognition rate and computation time. The results are encouraging and have a precision comparable to that of the state of the art in the field

Key words : Biometrics; Deep Learning; Auto-encoder ; Classification; Neural Networks