République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieure et de la Recherche Scientifique

Université Abderrahmane Mira

Faculté de la Technologie

Département d'Automatique, Télécommunication et d'Electronique

Projet de Fin d'Etudes

Pour l'obtention du diplôme de Master

Filière : Automatique

Spécialité : Automatique et informatique industriel

Thème

Reconnaissance automatique de formes

Préparé par :

SID Cylia

Bouhzila Madjid

Dirigé par :

Mme GAGAOUA

Examiné par :

Mme. Mezzah

Mr. Nait Mohand

Année universitaire : **2021/2022**

Remerciement

Tout d'abord nous remercions le bon dieu qui a éclairé mon chemin et qui m'a donné la foi, la santé et le courage pour réaliser ce travail.

Nous remercions, pour la qualité de son encadrement, sa patience et sa disponibilité durant ma préparation de ce mémoire **Madame GAGAOUA.**

Nos vifs remerciements vont à mes parents qui m'ont toujours encouragé dans la poursuite de mes études, ainsi que pour leur aide, leur compréhension et leur soutien.

Nous tiendrons à exprimer nos gratitudes et nos remerciements au membre de jury **Madame Mezzah** et **Mr. Nait mohand** pour avoir accepté de participer et d'examiner notre mémoire. Veuillez accepter dans ce travail nos sincères respects et notre profonde reconnaissance.

Nous souhaitons adresser nos remerciements les plus sincères aux personnes qui nous ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire ainsi qu'à la réussite de cette formidable année universitaire.

Finalement, nous remercions l'Université de Bejaia et tous les enseignants du département ATE.

Dédicace

À ma source de motivation et d'inspiration, mes très chers parents. Nulle dédicace ne saurait être assez diserte pour vous formuler toute mon affection et mon immense gratitude. Vous m'avez élevé et protégé contre vents et marées et m'avez soutenu de ma naissance jusqu'à l'âge mûr, en peignant de mille joies, horizon de ma vie. Rien au monde ne vaut vos sacrifices et votre dévouement, recevez ici, mes plus sincères reconnaissances.

À mes très chers frères **Soufiane** et **Baby**. Je vous dédie ce travail en témoignage de mon amour et de mon grée à vous qui vous êtes toujours présent, que dieu vous garde et vous protège.

À mes sœurs de cœur **Dyhia**, **Sabine**, **Chanez**, **Nessrine** et **Katia** Pour votre appui, votre assistance sans faille et pour vos encouragements sincères et sans oublié mon meilleure ami **Amine** et en particulier à mon cher fiancé **Yanis** qui m'a soutenus pendant tous ce parcours avec ses encouragements, sa patience, et son sentiments d'amour aux moments les plus difficiles.

À tous ceux qui m'aiment et que j'aime.

CYLIA

Dédicace

C'est avec profonde gratitude et sincères mots, que je Dédie ce modeste travail de fin d'étude à :

Mes chers parents :

Les mots ne sauraient exprimer mon respect, source de vie, d'amour et d'affection, l'estime et ma considération pour tous les sacrifices que vous avez consenti pour mon instruction et mon bien être. Je vous dédie ce travail, car c'est grâce à votre soutien durant tout mon parcours que cela a été réalisable. Je ne vous remercierai jamais assez pour vos efforts de jour comme de nuit pour mon éducation et ma formation. J'espère qu'un jour, je pourrais vous rendre un peu de ce que vous avez fait pour moi, que dieu vous prête bonheur et longue vie

A mes chères frère : issame ,lounis,adam et mon petit frère mahd

Vous êtes formidable

. A mes amis :

Khaled, daho ,rafik, benyamine ,celine , soufainevous étés dans mon cœur

. Pour ma binôme Cylia : Une partenaire assidue, belle et timide, mais spéciale

À tous ceux qui m'aiment et que j'aime.

Madjid

Table des matières

Remerciement.....
Dédicace.....
Acronyme.....
Liste des figures.....
Liste des tableaux.....
Introduction Générale..... 1

Chapitre I : La reconnaissance de formes

I.1 Introduction..... 3
I.2 La reconnaissance de forme..... 3
I.2.1 Définition..... 3
I.2.2 Historique..... 3
I.3 Les méthodes de reconnaissance de forme..... 4
I.4 Applications typiques de la reconnaissance de forme..... 5
I.5 Schéma générale d'un système de reconnaissance de forme..... 6
I.5.1 Préparation des données..... 6
I.5.2 Apprentissage..... 7
I.5.3 Classification..... 7
I.6 Conclusion..... 8

Chapitre II: L'apprentissage profond

II.1 Introduction..... 9
II.2 L'apprentissage profond..... 9
II.2.1 Définition..... 9
II.2.2 Historique..... 10
II.2.3 Le mode de fonctionnement..... 10
II.2.4 Le champ d'application..... 11
II.2.5 Méthodes de Deep Learning..... 11
II.3 Les fonctions d'activations..... 12
II.3.1 Définition..... 12
II.3.2 les différents types..... 14
II.4 Les réseaux de neurones..... 15
II.4.1 Définition..... 15
II.4.2 Les types des réseaux de neurones..... 15
II.5 Les réseaux de neurones convolutionnels..... 16

II.5.1 Définition 16
II.5.2 Historique 17
II.5.3 Architecture de réseau CNN 20
II.5.4 Les couches de réseau CNN 20
II.5.4.1 La couche de convolution 20
II.5.4.2 La couche de Pooling 21
II.5.4.3 La couche de correction ReLU 23
II.5.4.4 La couche Fully-Connected 23
II.5.5 Les avantages de réseau CNN 24
II.6 Conclusion 24

Chapitre III : Reconnaissance de formes géométriques avec DL et l'attribut HMB

III.1 Introduction 26
III.2 HMB (Histogram of marqued background) 26
III.2.1 Marquage du fond 27
III.2.2 Exemples de marquage 28
III.3 Conception 30
III.3.1 Training 30
III.3.2 Test 31
III.4 Logiciel et librairies utilisé dans l'implémentation 31
III.4.1 Python 31
III.4.1.1 Historique 31
III.4.1.2 Définition 31
III.4.1.3 Ses bibliothèques 31
III.4.2 Keras 32
III.4.2.1 Définition 32
III.4.2.2 Le mode de fonctionnement 32
III.4.2.3 Les avantages et les inconvénients de Keras 32
III.4.2.4 Les structures de données de base de keras 33
III.4.2.5 Les modules de keras utilisés 33
III.4.3 TensorFlow 33
III.4.3.1 Définition 33
III.4.3.2 Le mode fonctionnement 33
III.4.3.3 Les avantages et les inconvénients de tensorflow 34
III.5 configuration utilisée dans l'implémentation 35

III.6 La base de données 36
III.6.1 Définition 36
III.7 Architecture du réseau CNN 37
III.7.1 Réseau neuronal convolutif à 2 couches 37
III.7.2 Réseau neuronal convolutif à 4 couches 39
III.7.3 Réseau neuronal convolutif à 6 couches 40
III.8 Code source de notre architecture 42
III.9 Résultats et discussion 42
III.9.1 Sans binarisation et marquage 42
III.9.1.1 Résultats obtenue pour le modèle à 2 couches 44
III.9.1.2 Résultats obtenue pour le modèle à 4 couches 46
III.9.1.3 Résultats obtenue pour le modèle à 6 couches 48
III.9.2 Avec binarisation et marquage 50
III.9.2.1 Résultats obtenue pour le modèle à 2 couches 51
III.9.2.2 Résultats obtenue pour le modèle à 4 couches 53
III.9.2.3 Résultats obtenue pour le modèle à 6 couches 56
III.10 Conclusion 58
Conclusion générale 59
Liste bibliographique .....
Résumé .....

Acronymes

RdF :Reconnaissance**d**e**F**orme

DL :Deep**L**earning

ML :Machine**L**earning

IA :Intelligence**A**rtificiel

CNN : Convolutional**N**eural **N**etwork

MLP : Multi **L**ayer**P**erceptron

FC : Fully **C**onnected

ReLU : Rectified**L**inear**U**nit

CONV : Couche de **C**onvolution

POOL : Pooling **L**ayer

HMB : Histogram of **M**arqued**B**ackground

AVG : Average

CPU : Central **P**rocessing **U**nit

GPU : Graphics **P**rocessing**U**nit

Liste des figures

FigureI. 1 : Schéma général d'un système de reconnaissance des formes 6

FigureII. 1 : L'intelligence artificielle et ses sous-domaines 9
FigureII. 2 : Structure simplifié d'un neurone artificiel 13
FigureII. 3 : l'architecture d'un neurone biologique et artificiel 16
FigureII. 4: l'architecture du réseau convolutif LeNet 17
FigureII. 5 : l'architecture du réseau convolutif AlexNet 18
FigureII. 6 : l'architecture du réseau convolutif ZFNet 18
FigureII. 7 : l'architecture du réseau convolutif GoogleNet 19
FigureII. 8 : l'architecture du réseau convolutif ResNet 19
FigureII. 9 : Architecture d'un réseau CNN 20
FigureII. 10 : Exemple simpliste des valeurs des pixels d'une image 5x5 et de valeurs d'une matrice utilisée comme filtre 21
FigureII. 11 :Parcours de la fenêtre de filtre sur l'image 21
FigureII. 12 : Action de la couche de pooling 22
FigureII. 13 : illustration de l'opération de pooling 22
FigureII. 14 : la représentation de la couche de correction ReLu 23
FigureII. 15 : Représentation de la couche fully-connected 24

FigureIII. 1 : Configuration avec 4 directions 27
FigureIII. 2 : Toutes les possibilités de marques avec quatre directions et leur code binaire ainsi que leurs signes décimaux correspondants. 28
FigureIII. 3 : Exemple de marquage d'un pixel de fond ayant un seul voisin d'encre 28
FigureIII. 4 : exemples d'images de caractères après le processus de binarisation et marquage de la base de caractère forme 29
FigureIII. 5 : Schéma générale de notre système 30
FigureIII. 6 : La structure de données de la bibliothèque Keras 33
FigureIII. 7 : la structure générale de notre base de données des formes géométriques 36
FigureIII. 8 : Architecture de modèle à 2 couches 37
FigureIII. 9 : configuration du modèle à 2 couches 38
FigureIII. 10 : Architecture de modèle à 4 couches 39
FigureIII. 11 : configuration du modèle à 4 couches 40
FigureIII. 12 : Architecture de modèle à 6 couches 40
FigureIII. 13 : configuration du modèle à 6 couches 41
FigureIII. 14 : Précision et erreur du modèle à 2 couches (4 époques) 44
FigureIII. 15 : Précision et erreur du modèle à 2 couches (8 époques) 44
FigureIII. 16 : Précision et erreur du modèle à 2 couches (12 époques) 45
FigureIII. 17 :Précision et erreur du modèle à 4 couches (4 époques) 46
FigureIII. 18 : Précision et erreur du modèle à 4 couches (8 époques) 47
FigureIII. 19 : Précision et erreur du modèle à 4 couches (12 époques) 47
FigureIII. 20 : Précision et erreur du modèle à 6 couches (4 époques) 48

FigureIII. 21 : Précision et erreur du modèle à 6 couches (8 époques) 49
FigureIII. 22 : Précision et erreur du modèle à 6 couches (12 époques) 49
FigureIII. 23 : Précision et erreur du modèle à 2 couches (4 époques) 51
FigureIII. 24 : Précision et erreur du modèle à 2 couches (8 époques) 52
FigureIII. 25 : Précision et erreur du modèle à 2 couches (12 époques) 53
FigureIII. 26 : Précision et erreur du modèle à 4 couches (4 époques) 53
FigureIII. 27: Précision et erreur du modèle à 4 couches (8 époques) 54
FigureIII. 28 : Précision et erreur du modèle à 4 couches (12 époques) 55
Figure III. 29 : Précision et erreur du modèle à 6 couches (4 époques) 56
FigureIII. 30 : Précision et erreur du modèle à 6 couches (8 époques) 56
FigureIII. 31: Précision et erreur du modèle à 6 couches (12 époques) 57

Liste des tableaux

TableauII. 1 : Les différentes fonctions d'activation 14

TableauIII. 1 : la différence entre TensorFlow et Keras..... 35
TableauIII. 2 : configuration utilisée dans l'implémentation 35
TableauIII. 3 : comparaison des 3 modèles sans binarisation et marquage..... 43
TableauIII. 4 : comparaison des 3 modèles avec binarisation et marquage 51

Introduction Générale

La reconnaissance de formes est un domaine qui a beaucoup évolué ces dernières décennies et si on regarde l'histoire moderne de la reconnaissance de forme (RDF), disons depuis les années 1980, son développement récent est bien sûr lié au développement des ordinateurs, on voit qu'il est lié avant tout à l'automatisation des tâches de perception. L'une des applications phares dans ce domaine est la reconnaissance de l'écriture manuscrite, qui reste aujourd'hui un domaine d'application et de recherche actif. Au-delà, la vision, en reconnaissant les objets dans les images, ainsi que la reconnaissance auditive et automatique de la parole, sont des applications emblématiques et historiques de la reconnaissance des formes. C'est pourquoi la reconnaissance des formes a longtemps été considérée comme faisant partie intégrante du domaine de l'intelligence artificielle (IA). Le rôle initial de RDF est de générer tous les algorithmes nécessaires pour percevoir abstraitement l'environnement (obstacles, personnes, etc.) à partir de capteurs du monde extérieur.

Peu à peu, RDF s'est débarrassé des conseils d'IA. En se concentrant sur n'importe quelle forme ou modèle, elle peut résoudre d'autres tâches de classification et a été étendu à d'autres cadres d'apprentissage supervisé tels que la régression. Aujourd'hui, RDF est une discipline fondatrice dans le domaine d'apprentissage automatique, aux côtés des algorithmes, de la complexité, de la cryptographie, de la logique, de l'optimisation, de la physique statistique, des probabilités, des sciences cognitives, des statistiques, de l'évolution et d'autres disciplines.

Dans le cadre de la réalisation de ce travail, nous avons choisi de nous intéresser à la **reconnaissance des formes** par **Deep Learning en utilisant un sous-type de réseau de neurone appelé réseau de neurone convolutif (CNN)**, convient bien aux taches permettant d'extraire et de classifier des caractéristiques des caractères, permettant ainsi la reconnaissance automatique des formes.

Pour aboutir aux objectifs fixés, ce présent mémoire sera organisé en trois chapitres précédé de cette introduction et suivie d'une conclusion générale :

Pour commencer, le chapitre 1 avec l'intitulé de la reconnaissance de forme est consacré à la représentation des concepts généraux liés à la RDF, citant les principales étapes sur les quelles cette reconnaissance est basée ainsi que ses divers domaines d'applications.

Passons au 2ème chapitre qui est dédié à la présentation des réseaux de neurones profonds, en particulier le CNN ainsi que ces fonctions d'activation et aussi les différentes couches.

Enfin, le 3ème chapitre se portera sur la simulation et l'interprétation des différents résultats obtenue suite à l'implémentation des différents algorithmes de Deeplearning en combinaison avec l'attribut HMB pour une base de données des formes géométrique.

Chapitre I

La reconnaissance de formes

I.1 Introduction

La reconnaissance de formes est étroitement liée à l'intelligence artificielle et l'apprentissage de la machine, ainsi que des applications telles que l'exploration de données et la découverte de connaissances dans les bases de données.

Dans ce chapitre, nous définissons le domaine de reconnaissance de formes et ses caractéristiques principales ainsi que son domaine d'application et nous introduisons les méthodes traitées par cette branche.

I.2 La reconnaissance de forme

I.2.1 Définition

Le domaine de la reconnaissance de formes concerne la découverte automatique des régularités dans les données grâce à l'utilisation d'algorithmes informatiques et l'utilisation de ces régularités pour mener des actions telles que la classification des données en différentes catégories.

On désigne par reconnaissance de formes (ou parfois reconnaissance de motifs) un ensemble de techniques et méthodes visant à identifier des motifs à partir de données brutes afin de prendre une décision dépendant de la catégorie attribuée à ce motif. On considère que c'est une branche de l'intelligence artificielle qui fait largement appel aux techniques d'apprentissage automatique et aux statistiques [1].

Le problème à résoudre par la reconnaissance des formes est d'associer une classe à un modèle ou une forme inconnue (qui n'a pas encore de classe associée). La reconnaissance des formes est souvent considérée comme un problème de classification : il s'agit de trouver la fonction qui attribue à toute forme inconnue la classe la plus pertinente. Elle fait partie intégrante de la prise de décision de tout système intelligent [1] [2].

I.2.2 Historique

Pour déchiffrer un texte dactylographié ou manuscrit, et pour compter des chromosomes, reconnaître une tumeur, un char ou un avion de guerre, la compréhension de l'image, sa classification passe toujours par la reconnaissance d'une forme. « Plusieurs approches théoriques ont été développées », explique Olivier Faugeras [1].

« Les premières consistaient à faire des calculs à partir de l'image et construire des représentations symboliques de plus en plus complexes, d'abord en deux dimensions tel que

sur l'image, puis tridimensionnelles, pour tenter de restituer une description proche de notre propre vision. » Un peu partout dans le monde, les chercheurs ont mis au point des méthodes mathématiques permettant de détecter les contours des objets à partir des changements rapides de contraste dans l'image, des ombres et des lumières, des régions homogènes en couleur, en intensité, en texture [1].

« Dès 1964, des chercheurs français, Georges Matheron (1930-2000) et Jean Serra, ont développé une autre approche théorique (baptisée morphologie mathématique) et un outil spécifique (l'analyseur de texture breveté en 1965, ndlr) d'abord pour analyser des microphotographies de terrain et évaluer des teneurs en minerai, puis pour d'autres applications comme la cytologie (caractérisation et comptage de cellules) » rappelle Olivier Faugeras. En 1968, ils créent le Centre de morphologie mathématique de l'Ecole des Mines de Fontainebleau. Leurs outils d'analyse et d'interprétation d'images sont longtemps restés franco-français, jusqu'à ce qu'un américain, Robert Haralick (Université du Kansas à cette époque, de Seattle actuellement), en fasse une large publicité dans les années 1980, en les adaptant à de nombreuses applications : industrielles comme l'inspection radiographique des ailes d'avions de Boeing, aériennes ou médicales [1] [3].

D'autres chercheurs, comme les américains Marvin Minsky et Seymour Papert du MIT (Massachussets Institute of Technology) ont considéré le problème dans l'autre sens, en cherchant à formaliser et à faire reproduire par l'ordinateur notre propre processus de reconnaissance d'images, donc notre propre vision. Cette démarche était dans l'air du temps, au cœur des promesses de « l'intelligence artificielle » qui devait permettre de mettre l'intelligence en équations et doter les ordinateurs de toutes les capacités humaines de raisonnement, mémoire, perception. Or la vision s'est révélée être un domaine particulièrement complexe à modéliser tant elle est basée sur une quantité phénoménale de connaissances à priori fondée sur notre intelligence et notre expérience [1] [4].

I.3 Les méthodes de reconnaissance de forme

La reconnaissance des motifs peut être effectuée à l'aide de divers algorithmes d'apprentissage automatique, tels que les algorithmes suivants :

- ✓ Les réseaux de neurones
- ✓ L'analyse statistique
- ✓ L'utilisation des modèles de Markov cachés
- ✓ La recherche d'isomorphisme de graphes ou de sous-graphes

La forme recherchée peut être géométrique et peut être décrite par une formule mathématique, comme :

- Cercles ou ellipses
- Courbes de Bézier, splines
- Lignes droites

Elles peuvent également être de nature plus complexe :

- Lettres
- Chiffres
- Empreinte digitale

L'algorithme de reconnaissance fonctionne sur des images en noir et blanc, où le contour de l'objet dans l'image est blanc, Ces images sont le résultat d'algorithmes de détection des bords. Ils peuvent également travailler sur des régions prédéfinies de l'image, qui sont générées par la segmentation de l'image [1].

I.4 Applications typiques de la reconnaissance de forme

- **Marketing :** la reconnaissance des formes est souvent utilisée pour classer les consommateurs en fonction des produits qu'ils sont susceptibles d'acheter. Elle est également utilisée par les sociétés de vente pour classer les clients selon qu'ils sont de bons ou de mauvais payeurs, ou qu'ils vont passer à un concurrent [1] [5].
- **Finance :** les systèmes de reconnaissance des formes sont utilisés pour détecter les transactions bancaires frauduleuses et les prédire ainsi que les prévisions d'insolvabilité [1] [6].
- **Usinage :**la qualité d'un produit dépend souvent d'une mesure correcte, et la relation exacte entre la qualité et les valeurs des paramètres n'est pas claire. Les systèmes de reconnaissance des formes sont utilisés pour classer les paramètres en fonction de la qualité du produit qu'ils peuvent produire. Cela permet de réduire le nombre de tests et d'économiser du temps et de l'argent [1] [7].
- **Energie :** des systèmes de reconnaissance de formes sont utilisés pour prédire la consommation d'énergie (faible, normale, élevée), permettant aux clients de réduire leur consommation si nécessaire, et aussi aux producteurs de mieux gérer leurs unités de production [1] [8].

- **Lecture automatisée :** les systèmes de reconnaissance de formes permettent de numériser et d'archiver des documents et des archives anciens, non pas sous forme d'images, mais sous forme de texte [1] [9].
- **Sécurité :**la reconnaissance de la voix et de la rétine est un exemple d'application typique de la reconnaissance des formes pour l'authentification. La vérification des signatures est également très populaire [1] [10].

I.5 Schéma générale d'un système de reconnaissance de forme

La plupart des systèmes reconnaissance de forme disposent des options de fonctionnement montrées dans la figure I.1 ci-dessous :

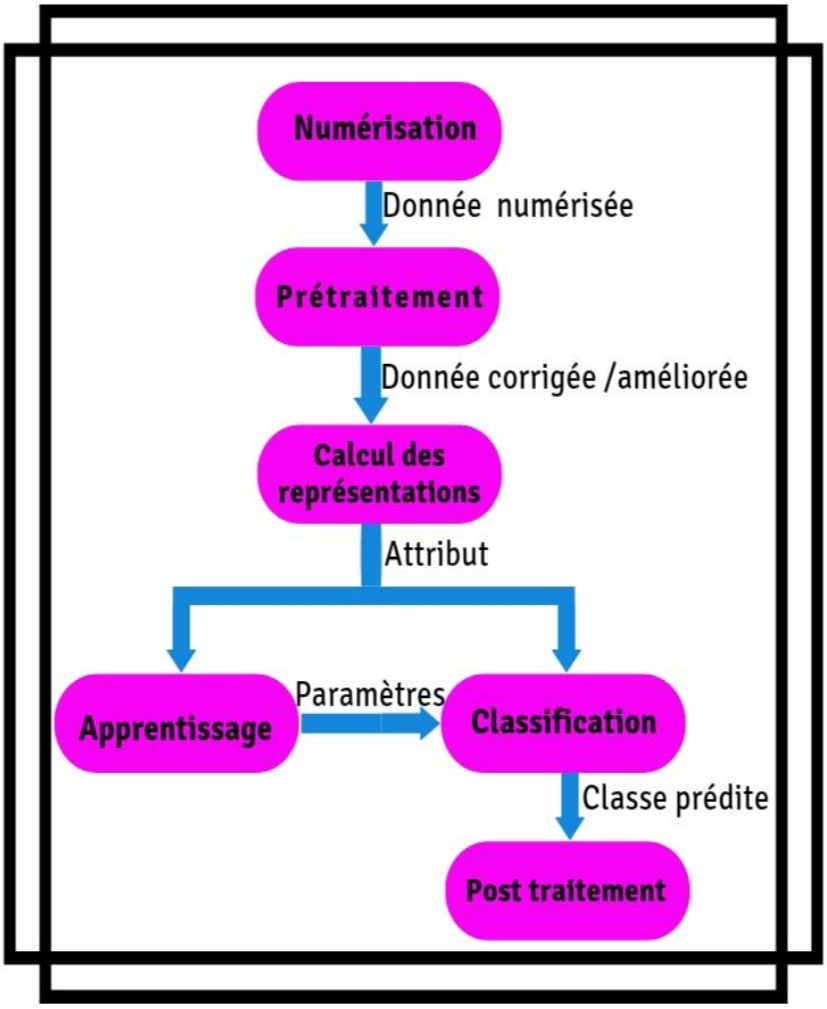


Figure I. 1 : Schéma général d'un système de reconnaissance des formes

I.5.1 Préparation des données

a- Numérisation : à partir des informations sur le monde physique, on construit une représentation des données qui peuvent être manipulées directement par des machines. Les

capteurs (microphones, caméras, instruments de mesure) convertissent les signaux reçus du monde réel en représentations numériques discrètes. L'espace résultant, appelé espace de représentation, a une très grande dimension **r**, ce qui lui permet d'avoir le maximum d'informations sur la forme numérisée [1].

b- Prétraitement : il s'agit de sélectionner les informations requises par le domaine d'application dans l'espace de représentation. Cette sélection implique souvent l'élimination du bruit, la normalisation des données et suppression des contenus redondants. Le nouvel espace de représentation a une dimension **r'** beaucoup plus petite que **r**, mais reste un espace à haute dimension et contient des informations qui sont encore très primitives [1].

c- Calcul de représentation : il s'agit de la dernière étape de la préparation des données. Elle fournit un certain nombre de caractéristiques ou de paramètres (appelés attributs), en utilisant un algorithme de sélection et/ou d'extraction d'attributs. Comme le nombre d'attributs est fini, l'espace des paramètres résultant a une très petite dimension **p** par rapport à **r'**[1].

I.5.2 Apprentissage

L'apprentissage ou l'entrainement est une partie importante d'un système de reconnaissance.Un classificateur étant généralement une fonction paramétrée, l'entraînement permettra d'optimiser les paramètres du classificateur pour le problème à résoudre à l'aide de données d'entraînement. Lorsque cesdernièressont classées au préalable, l'apprentissage est appelé apprentissage supervisé, sinon il s'agit d'apprentissage non supervisé [1] [11].

I.5.3 Classification

Cette étape est le cœur de la reconnaissance des formes. À l'aide de modèles (paramètres) obtenus lors de l'apprentissage, le classificateur attribue à chaque forme inconnue sa ou ses formes les plus probables [1].

I.5.4 Post traitement

L'objectif de cette étape est de corriger les résultats de la classification à l'aide d'outils spécifiques à l'application. Par exemple, pour un système de reconnaissance de texte manuscrit, le classificateur classe chaque caractère individuellement et le post-processeur applique un correcteur orthographique à l'ensemble du texte pour vérifier et éventuellement corriger les résultats de la classification. Bien que facultative, la qualité de la reconnaissance est grandement améliorée à ce stade [1].

I.6 Conclusion

Les systèmes de reconnaissance des formes peuvent être considérés comme des systèmes qui attribuent des étiquettes à des valeurs d'entrée données. Les méthodes de reconnaissance des formes visent généralement à fournir une réponse raisonnable à toutes les entrées possibles et à faire correspondre les entrées "les plus probables", en tenant compte de leur variation statistique.

La reconnaissance des formes comprend plusieurs étapes : la segmentation des objets (analyse de l'image), l'extraction de caractéristiques (géométriques, invariantes, etc…) et la classification (supervisée ou non supervisée, méthodes probabilistes, statistiques, etc…). Les applications sont très diverses et nécessitent une connaissance approfondie du domaine d'application. Il existe de nombreuses méthodes, mais les principes de base sont similaires.

Dans le prochain chapitre, nous allons décrire l'apprentissage profond avec ses différents domaines d'applications ainsi que son principe de fonctionnement, tous en se basons sur les réseaux de neurones plus spécialement le réseau de neurone convolutif CNN avec ses différentes couches.

Chapitre II

L’apprentissage profond

II.1 Introduction

Un jour, un esprit brillant en a eu assez de devoir expliquer sans cesse aux machines comment elles devaient procéder pour accomplir une tâche et s'est dit : "Eh bien, voyons comment fonctionne le cerveau humain et adaptons ce mécanisme aux ordinateurs du mieux que nous pouvons." Bien que nous ne comprenions toujours pas une grande partie des mystères de notre cerveau et que les adaptations informatiques qui en résultent soient en fait très éloignées du fonctionnement biologique du cerveau, cette approche a porté ses fruits, donnant naissance au Deep Learning des décennies plus tard.

Ce chapitre va nous permettre de situer le contexte de notre travail qui est l’apprentissage profond avec son mode de fonctionnement et ses différents champs d’applications tous ont citons les réseaux de neurone convolutionnel et les couches de convolution utilisées.

II.2 L’apprentissage profond

II.2.1 Définition

Le Deep Learning ou l'apprentissage profond est une forme d'intelligence artificielle dérivée de l'apprentissage automatique (machine learning), où les machines ordinateurs sont capables d'apprendre par elles-mêmes, plutôt que de simplement suivre des règles prédéterminées comme elles sont programmées pour le faire, pour conclure on résume cette description par la figure II.1 représenter ci-dessous qui montreRelation entre intelligence artificielle, machine Learning et l’apprentissage profond [12].

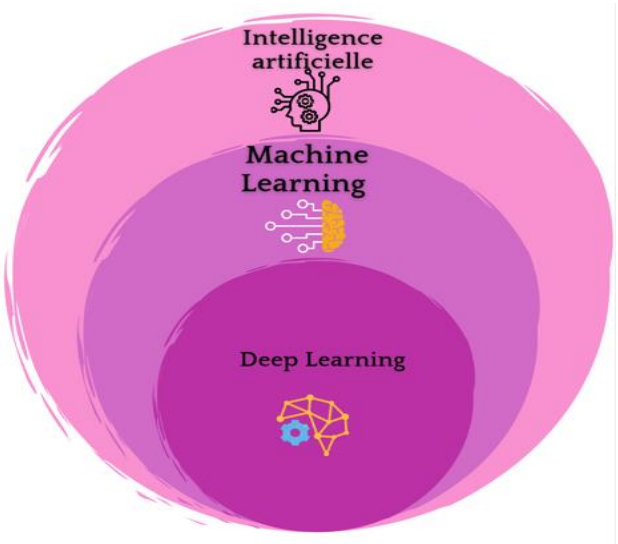


Figure II. 1 : L'intelligence artificielle et ses sous-domaines

II.2.2 Historique

L'histoire de l'apprentissage profond remonte à un domaine qui a changé de nom pour devenir aujourd'hui la cybernétique. Elle a commencé dans les années 1940 avec McCulloch et Pitts avec l'idée des neurones qui sont des unités de seuil avec des états de marche et d'arrêt.

En 1947, Donald Hebb a eu l'idée que les neurones biologiques apprennent en modifiant la force des connexions entre eux. C'est ce qu'on appelle l'hyper apprentissage, où si deux neurones sont activés ensemble alors la connexion entre eux augmente et s'ils ne le sont pas alors la connexion diminue. Puis il a été développé en 1948 par Norbert Wiener avec l'idée d'avoir des systèmes avec des capteurs et des actionneurs ; par contre en 1957, Frank Rosenblatt a proposé le Perceptron qui est un algorithme d'apprentissage qui modifie les poids de réseaux neuronaux très simples.

Durant les années 1960 le DL s'est éteint pour quelques raisons mais vite il a pris un nouvel essor en 1985 avec l'apparition de la rétro propagation. En 1995, le domaine a périclité à nouveau et la communauté de l'apprentissage machine a abandonné l'idée des réseaux neuronaux. Début 2010, les gens ont commencé à utiliser les réseaux de neurones dans la reconnaissance vocale avec une amélioration considérable des performances, et plus tard, ils ont été largement déployés dans le domaine commercial. En 2013, la vision par ordinateur a commencé à passer aux réseaux de neurones. En 2016, la même transition a eu lieu dans le traitement du langage naturel. Bientôt, des révolutions similaires se produiront dans la robotique, le contrôle et bien d'autres domaines [13].

II.2.3 Le mode de fonctionnement

L'apprentissage profond repose sur un réseau de neurones artificiels inspiré du cerveau humain. Ce réseau est constitué de dizaines, voire de centaines de "couches" de neurones, chaque couche reçoit et interprète les informations de la couche précédente.

Le système par exemple, apprend à reconnaître les lettres avant de traiter les mots dans un texte, ou encore il détermine s'il y a un visage sur une photo avant de découvrir de quelle personne s'agit-il.

Grâce à un processus d'auto-apprentissage, l'apprentissage profond est capable d'identifier un chat dans une photographie. Chaque couche du réseau de neurones correspond à un aspect spécifique de l'opération.

À chaque étape, les réponses "fausses" sont éliminées et renvoyées à chaque couche, pour ajuster le modèle mathématique. Au fil du temps, le programme réorganise l'information

en blocs plus complexes. Lorsque ce modèle est ensuite appliqué à d'autres cas, il est généralement capable de reconnaître un chat sans qu'on lui dise qu'il n'avait jamais encore appris le concept de chat. Les données de départ sont cruciales (plus le système a accumulé différentes expériences, plus il sera performant) [14].

II.2.4 Le champ d'application

Parmi les domaines d'application d'apprentissage profonds on cite les suivants :

- Robotique
- Dans le domaine de l'agriculture
- Imagerie médicale et domaine de la santé
- Le Deep Learning dans le secteur du marketing et de la vente
- Pour la conduite autonome
- (Cyber) sécurité
- Le Deep Learning au service de la rédaction de textes

Comme les modèles d'apprentissage profondtraitent les informations d'une manière similaire à celle du cerveau humain, ils peuvent être appliqués à un grand nombre de tâches que les gens effectuent. L'apprentissage profond est actuellement utilisé dans la plupart des outils de reconnaissance d'images, des logiciels de traitement du langage naturel et de reconnaissance vocale. Ces outils commencent à apparaître dans des applications aussi diverses que les voitures et les services de traduction linguistique.

II.2.5 Méthodes de Deep Learning

Différentes méthodes peuvent être utilisées pour créer de puissants modèles d'apprentissage profond. Ces techniques comprennent la réduction des taux d'apprentissage, l'apprentissage par transfert, la formation à partir de zéro et les abandons [12].

a.Le taux d'apprentissage diminue : Le taux d'apprentissage est un hyper-paramètre, un facteur qui définit le système ou fixe ses conditions de fonctionnement avant le processus d'apprentissage - qui contrôle la quantité de variation que le modèle subit en fonction de l'erreur d'estimation chaque fois que les poids du modèle sont modifiés. Un taux d'apprentissage trop élevé peut conduire à un processus d'apprentissage instable ou à l'apprentissage d'un ensemble de poids sous-optimal. Un taux d'apprentissage trop faible peut entraîner un long processus de formation qui stagne.

b. La méthode de réduction du taux d'apprentissage : également appelée recuit du taux d'apprentissage ou taux d'apprentissage adaptatif - consiste à adapter le taux

d'apprentissage pour améliorer les performances et réduire le temps d'apprentissage. L'adaptation la plus simple et la plus courante des taux d'apprentissage pendant la formation consiste en des techniques qui réduisent le taux d'apprentissage au fil du temps.

c. Transfert de l'apprentissage : Ce processus consiste à affiner un modèle déjà formé ; il nécessite une interface avec un réseau préexistant. Tout d'abord, l'utilisateur fournit au réseau existant de nouvelles données contenant des classifications précédemment inconnues. Une fois que le réseau a été réglé, la nouvelle tâche peut être exécutée avec des capacités de classification plus spécifiques. L'avantage de cette approche est qu'elle nécessite beaucoup moins de données que les autres méthodes, ce qui réduit le temps de calcul à quelques minutes ou quelques heures.

d. Formation à partir de zéro : Cette approche exige du développeur qu'il collecte une grande quantité de données étiquetées et qu'il configure une architecture de réseau capable d'apprendre des caractéristiques et des modèles. Cette technique est particulièrement utile pour les nouvelles applications et pour les applications comportant un grand nombre de catégories de sortie. Cependant, en général, il s'agit d'une approche moins courante car elle nécessite trop de données, ce qui fait que la formation prend des jours ou des semaines.

e. Abandon : Cette approche tente de résoudre le problème de l'overfitting dans les réseaux comportant un grand nombre de paramètres en supprimant de manière aléatoire des unités et leurs connexions du réseau neuronal pendant la formation. Il a été démontré que la méthode d'abandon améliore les performances des réseaux neuronaux dans les tâches d'apprentissage supervisé dans des domaines tels que la reconnaissance vocale, la classification des documents et la biologie informatique.

II.3 Les fonctions d'activations

II.3.1 Définition

Une fonction d'activation est une équation mathématique qui détermine la sortie dechaque élément (perceptron ou neurone) dans le réseau neuronal. Comme illustré en figure II.2, il prend l'entrée de chaque neurone et la transforme en sortie.

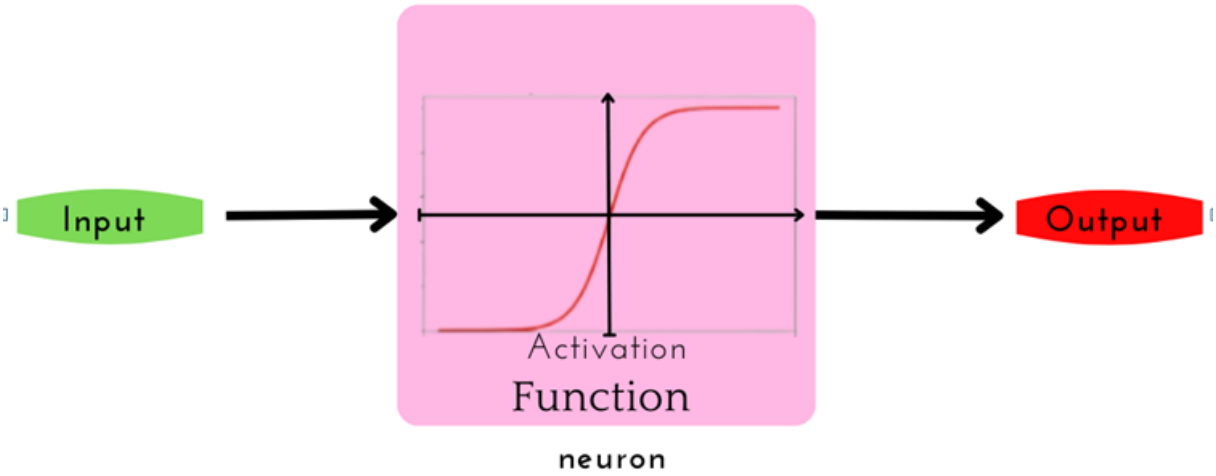


Figure II. 2: **Structure simplifié d’un neurone artificiel**

Dans un réseau neuronal, les entrées, qui sont généralement des valeurs réelles, sontintroduites dans les neurones du réseau. Chaque neurone a un poids, et les entrées sont multipliées par le poids et introduites dans la fonction d'activation, tel que cette dernièreest une fonction qui permet de traiter l'information qui arriveà un neurone artificiel en machine Learning, comme le fait ceux du cerveau avec les signauxélectriques qu'ils reçoivent [15].

La sortie de chaque neurone est l'entrée des neurones dans la couche suivante du réseau, et donc les entrées se succèdent à travers de multiples fonctions d'activation jusqu'àce que finalement, la couche de sortie génère une prédiction. Les réseaux de neuronesreposent sur des fonctions d'activation non linéaires, la dérivée de cette fonction aide le réseau à apprendre à travers le processus de rétro propagation [15].

II.3.2 les différents types

Le Tableau II.1 ci-dessous décrit les principales fonctions d'activation existantes :

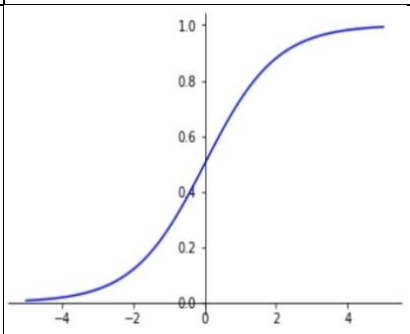
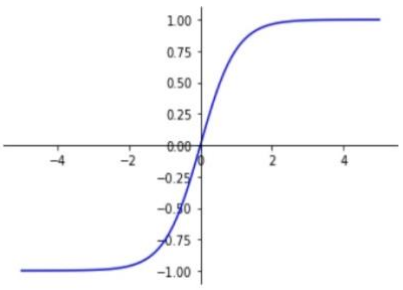
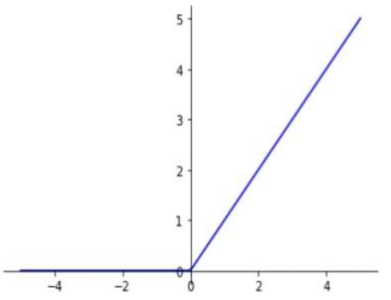
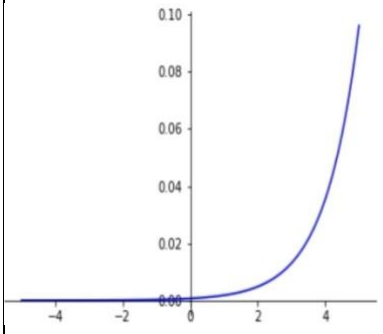
Noms	Description	Expression	Graphe
Sigmoïde	Elle génère des valeurs comprises entre zéro et un. Pour des valeurs très élevées ou basses des paramètres d'entrée, le réseau peut être très lent à atteindre une prédiction.	$f(x) = \frac{1}{1+e^{-x}}$	
TanH	Centrée sur zéro, ce qui facilite lamodélisation d'entrées fortementnégatives, fortement positives ou neutres.	$Tanh(x) = \frac{\sinh(x)}{\cosh(x)}$ $= \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{e^x - 1}{e^x + 1}$	
ReLu	Très efficace en termes de calcul mais n'est pas en mesure de traiter des entrées qui approchent de zéro ou négatives.	$f(x) = \begin{cases} 0 \; si \; x < 0 \\ x \; si \; x > 0 \end{cases}$	
Softmax	Utilisée pour les neurones des sorties. Elle normalise les sorties pour chaque classe entre 0 et 1 et renvoie la probabilité que l'entrée appartient à une classe spécifique.	$\sigma(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}}$	

Tableau II. 1 : Les différentes fonctions d'activation

II.4 Les réseaux de neurones

II.4.1 Définition

Les réseaux de neurones sont largement considérés comme une avancée majeure dans le domaine au moment de leur invention. Inspirée de la fonction des neurones de notre cerveau, l'architecture des réseaux de neurones introduit une méthode, un algorithme, qui permet à un ordinateur d'améliorer sa prise de décision, c'est-à-dire son apprentissage.

Un algorithme d'apprentissage automatique avancé, connu sous le nom de réseau neuronal artificiel, constitue la base de la plupart des modèles d'apprentissage profond. L'apprentissage profond peut parfois être appelé apprentissage neuronal profond ou réseaux neuronaux profonds.

Les réseaux neuronaux impliquent un processus d'essais et d'erreurs et à ce titre, ils nécessitent de grandes quantités de données pour la formation. Ce n'est pas une coïncidence si les réseaux neuronaux ne sont devenus populaires qu'après que la plupart des entreprises adopté l'analyse de données à grande échelle et accumulé une grande quantité de données. Comme la première itération d'un modèle implique des suppositions éclairées sur le contenu d'une image ou d'une partie du discours, les données utilisées dans la phase de formation doivent être étiquetées pour que le modèle puisse voir si ses suppositions sont correctes. Cela signifie que les données non structurées sont moins utiles, même si de nombreuses entreprises qui utilisent beaucoup de données en ont beaucoup. Les données non structurées ne peuvent être analysées par des modèles d'apprentissage profond qu'après avoir été entraînés à un niveau de précision acceptable, mais les modèles d'apprentissage profond ne peuvent pas être entraînés sur des données non structurées [16].

II.4.2 Les types des réseaux de neurones

- **Les neurones biologiques**

Un neurone formel est une représentation artificielle et schématique d’un neurone biologique : les synapses sont modélisées par des poids, le soma ou corps cellulaire est modélisé par la fonction de transfert, appelé aussi fonction d’activation. Ons'inspire du fonctionnement du cerveau humain on a pu créer un neurone artificiel [17].

- **Les neurones artificiels**

Les réseaux d'apprentissage profond sont formés sur les structures de données complexes qu'ils rencontrent. Ils construisent des modèles informatiques composés de plusieurs couches de traitement afin de créer plusieurs niveaux d'abstraction pour représenter les données [18].

Par exemple, les modèles d'apprentissage profond connus sous le nom de réseaux neuronaux convolutifs peuvent être entraînés à l'aide d'un grand nombre (des millions) d'images, telles que des images de chats. Ce type de réseau neuronal apprend à partir des pixels contenus dans l'image reçue. Il peut classer des groupes de pixels sur la base des caractéristiques des chats (par exemple : pattes, oreilles, yeux…) qui indiquent la présence d'un animal dans une image [18].

La figure II.3 ci-dessous nous montre les différentes architectures d'un neurone biologique et artificiel.

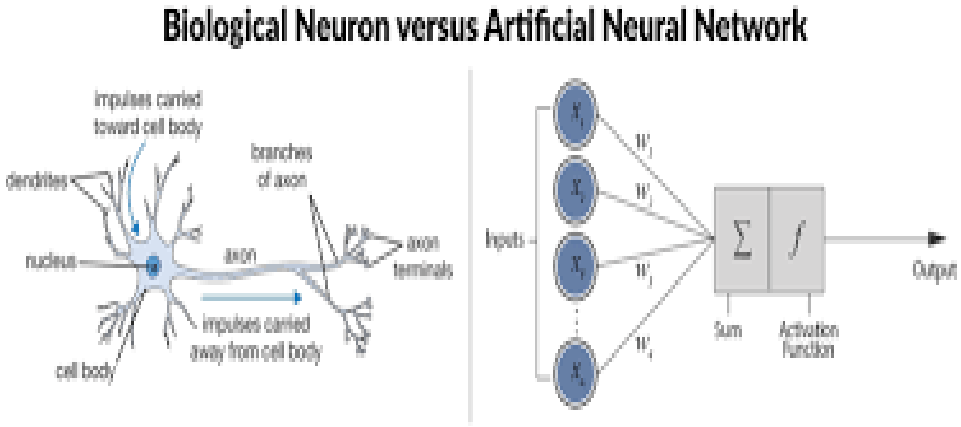


Figure II. 3: **l'architecture d'un neurone biologique et artificiel**

II.5 Les réseaux de neurones convolutionnels

II.5.1 Définition

Les réseaux neuronaux convolutifs ou CNN pour « Convolutional Neural Network »sont des extensions des perceptrons multicouches ou MLP pour « Multi Layer Perceptron »qui remédie efficacement aux principaux inconvénients de ces derniers. Ils sont conçus pour extraire automatiquement les caractéristiques de l'image d'entrée, sont insensibles aux images et mettent en œuvre le concept de partage des poids, ce qui permet de réduire considérablement le nombre de paramètres du réseau. Ce partage des poids permet également de prendre en compte les associations locales contenues dans une image. Les réseaux

neuronaux convolutifs ont été inspirés à l'origine par les neurones localement sensibles et sélectifs de la direction trouvés dans le système visuel des chats par Hubel et Wiesel [19] [20].

II.5.2 Historique

- **LeNet**

La première application réussie des réseaux convolutifs a été développée par Yann LeCun dans les années 1990. La plus célèbre d'entre elles est l'architecture LeNet. Lire les codes postaux, les chiffres, et ainsi de suite, comme représenter dans la figure II.4 qui décrit l'architecture du ce réseau convolutifLeNet.

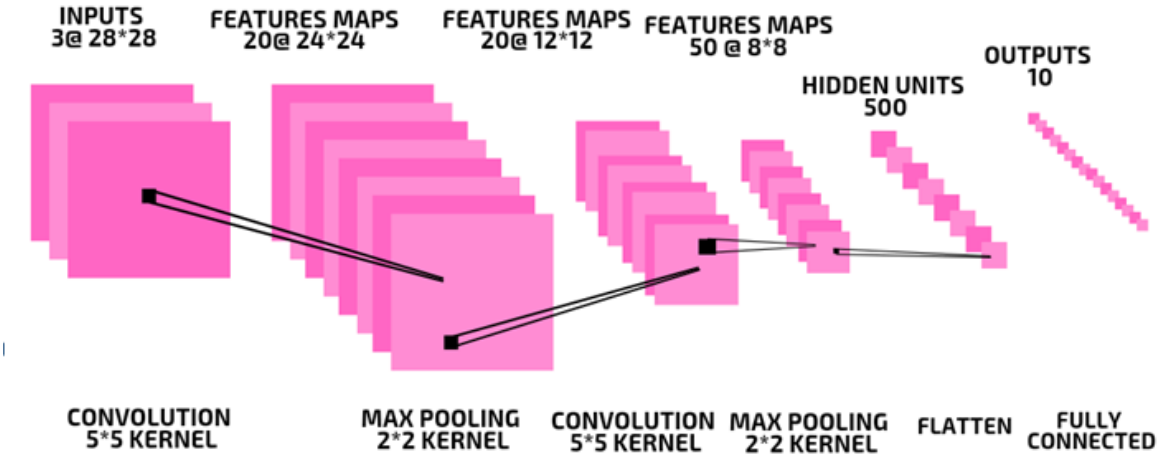


Figure II. 4: l'architecture du réseau convolutifLeNet

- **AlexNet**

Le premier ouvrage à populariser les réseaux convolutifs dans le domaine de la vision par ordinateur est AlexNet, développé par Alex Krizhevsky, Ilya Sutskever et Geoff Hinton.

AlexNet a été soumis à la defineImageNet ILSVRC en 2012 et a obtenu des résultats significativementmeilleurs que ses concurrents. L'architecture du réseau est très similaire à celle de LeNet, mais elle est plus profonde et plus grande, et présente les caractéristiques suivantes35 couches convolutionnelles empilées les unes sur les autres (auparavant, il était fréquent d'avoir couches convolutionnelles suivant toujours immédiatement une couche d'ensemble) ; qui est bien représenter dans la figure II.5 suivante avec l'intitulé d'architecture du réseau convolutifAlexNet.

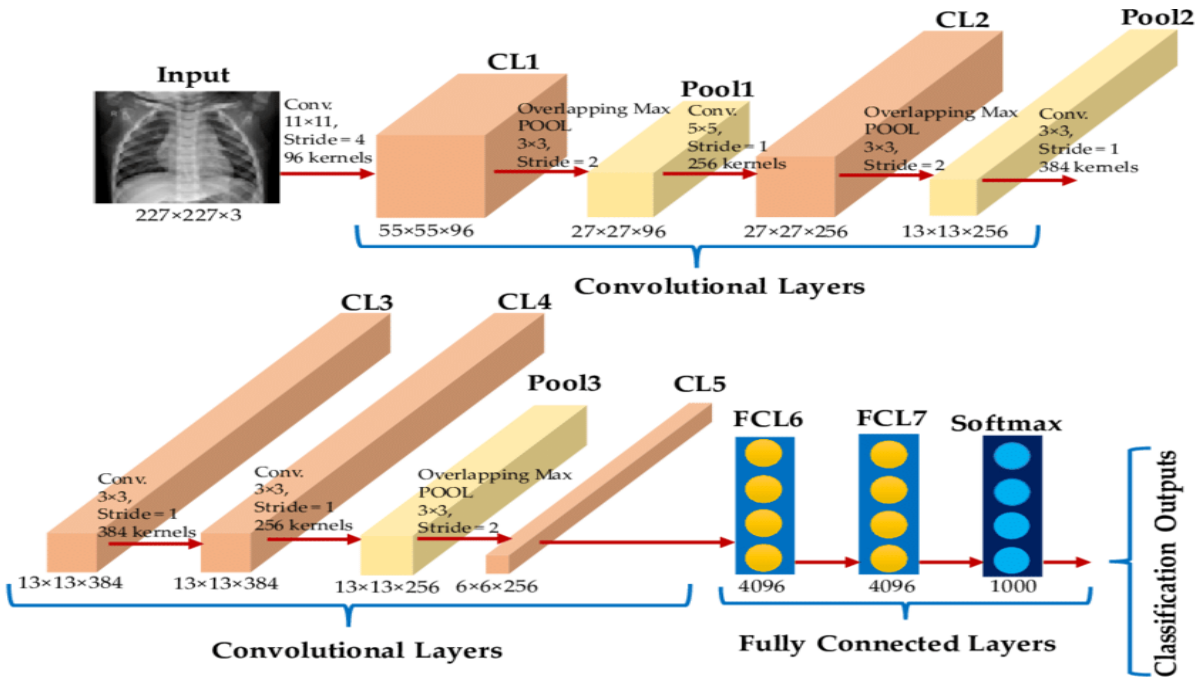


Figure II. 5 : l'architecture du réseau convolutifAlexNet

- **ZFnet**

Le gagnant du défi ILSVRC 2013 était le réseau convolutif de Matthew Zeiler et Rob Fergus. Il est devenu ZFNet (abréviation de Zeiler and Fergus Net). Il s'agit d'une amélioration obtenue en modifiant les hyper paramètres de l'architecture, notamment en augmentant la taille de l'AlexNet. Couches convolutionnelles et en réduisant la taille des noyaux sur la première couche, tous en représentent son architecture par la figure II.6 suivante :

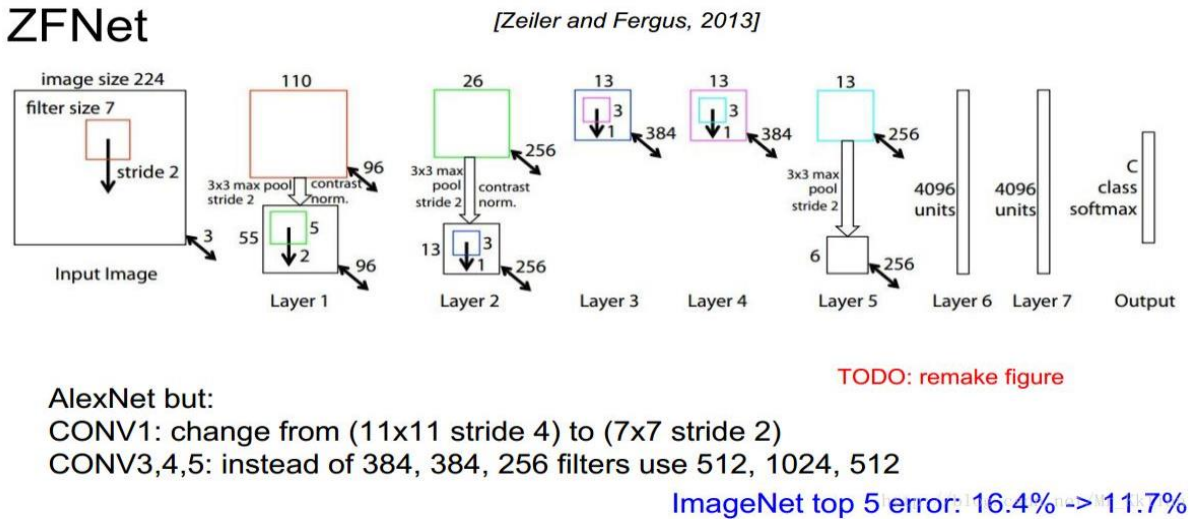


Figure II. 6 : l'architecture du réseau convolutifZFNet

- **GoogLeNet**

Le gagnant du défi ILSVRC 2014 était un réseau convolutif de Szegedy et al. De Google. Sa principale contribution a été le développement d'un module initial qui a permis de réduire considérablement le nombre de paramètres du réseau (4M, contre 60M pour AlexNet). En outre, le module utilise un pool AVG global, plutôt que le PMC dans la version finale. Ce qui permet d'éliminer un grand nombre de paramètres. Il existe également plusieurs versions de GoogLeNet, y compris Inception-v4. En décrit son architecture par la figure II.7 qui est la suivante :

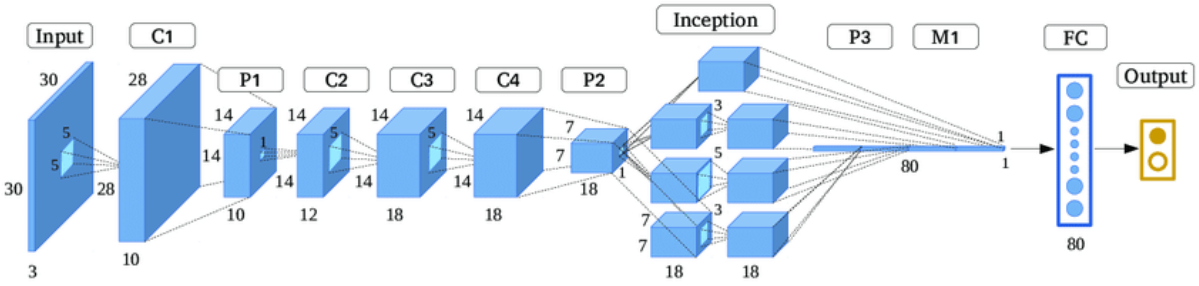


Figure II. 7 : l'architecture du réseau convolutifGoogleNet

- **ResNet**

Le réseau résiduel développé par Kai-Ming He et al. Est le gagnant du concours ILSVRC 2015. Il se caractérise par des sauts de connexion et une utilisation intensive de la normalisation des lots. Il utilise également des jeux d'AVG globaux plutôt que des PMC finaux. Comme montré dans la figure II.8 qui suit :

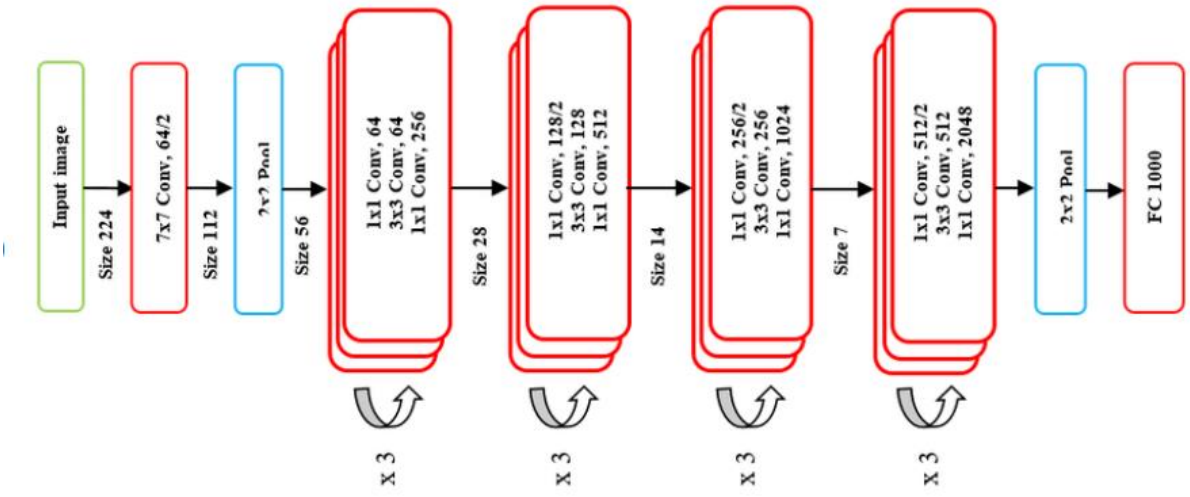


Figure II. 8 : l'architecture du réseau convolutifResNet

II.5.3 Architecture de réseau CNN

Le réseau neuronal convolutif est basé sur le perceptron multicouche (MLP), qui s'inspire du comportement. Il est très bien adapté au traitement des images.

Les MLP ont de grandes difficultés à traiter les grandes images en raison de la croissance exponentielle du nombre de connexions avec la taille de l'image[21].

Les réseaux neuronaux convolutifssont constitués de plusieurs couchescomme le montre la Figure II.9 :

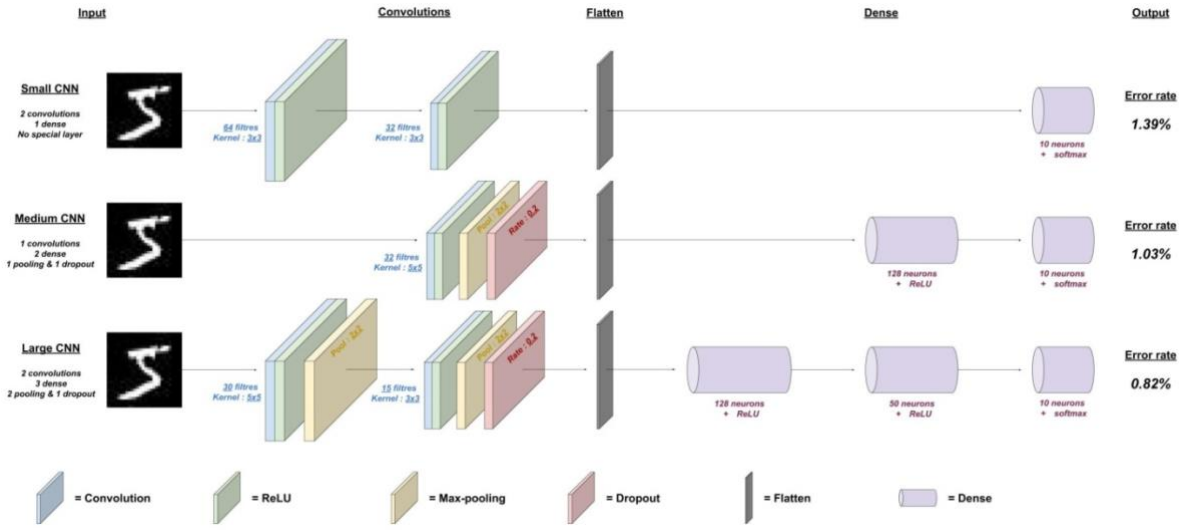


Figure II. 9 : Architecture d'un réseau CNN

II.5.4 Les couches de réseau CNN

II.5.4.1 La couche de convolution

La convolution, d'un point de vue simple, est l'application d'un filtre mathématique à une image. D'un point de vue plus technique, il s'agit de faire glisser une matrice sur l'image, et pour chaque pixel, d'utiliser la somme de ce pixel multipliée par la valeur de la matrice.

Cette technique nous permet de trouver les parties de l'image qui peuvent présenter un intérêt Intéressant [22].Prenons la Figure II.10 ci-dessous à gauche comme exemple d'image et la Figure à droite comme exemple de filtre :

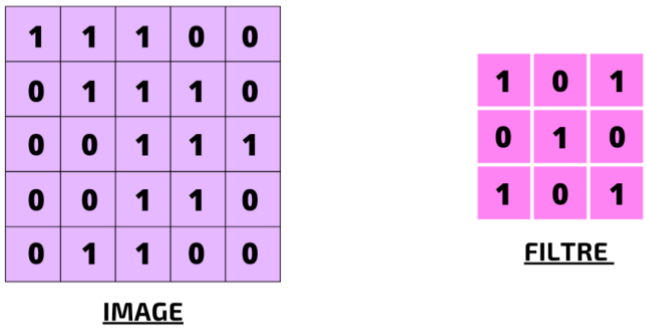


Figure II. 10 : Exemple simpliste des valeurs des pixels d'une image 5x5 et de valeurs d'une matrice utilisée comme filtre

La figure II.11 nous montre le parcours suivi des couches de la fenêtre de filtre sur une image quelconque avec un certain nombre de pas.

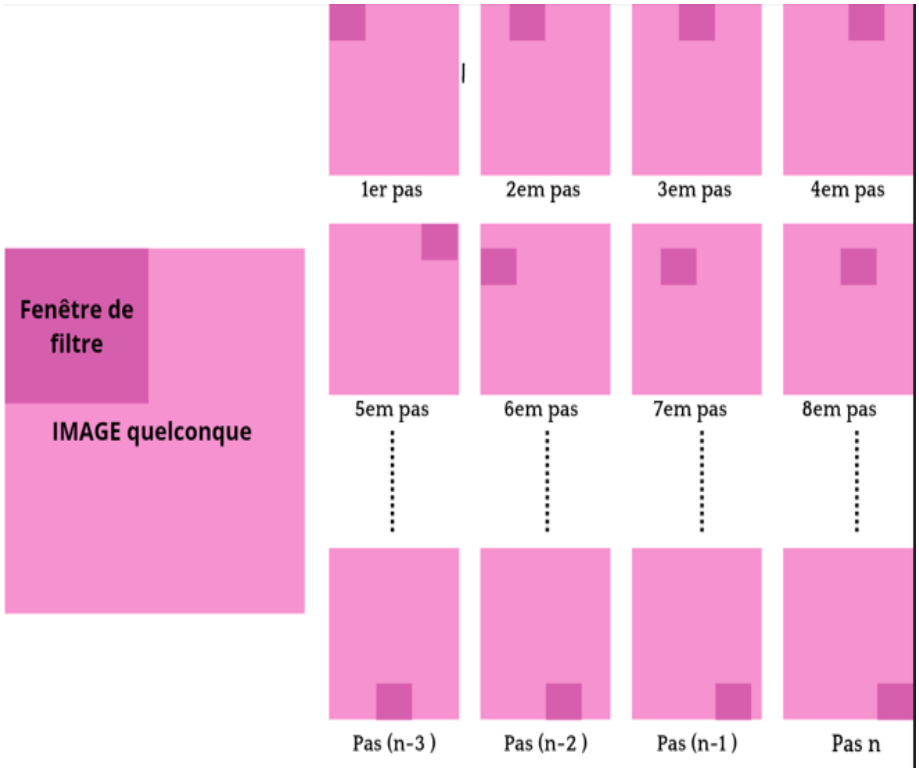


Figure II. 11:Parcours de la fenêtre de filtre sur l'image

II.5.4.2 La couche de Pooling

Appelée aussi la couche de mise en commun qui est une couche placée généralement entre deux couches convolutionnelles: il reçoit plusieurs cartes de caractéristiques en entrée et effectue une opération de mise en commun sur chacune d'entre elles [22] [23].

L'opération de pooling (ou sous-échantillonnage) consiste à réduire la taille de l'image tout en conservant ses caractéristiques importantes, comme elle est bien démontrée dans la figure II.12 suivante :

La figure II.13 nous montre l'illustration de la couche de pooling avec ses différents types (le Mean et le Max pooling) :

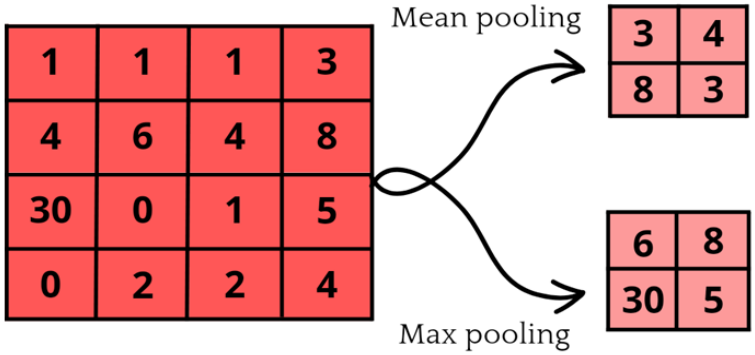


Figure II. 12: illustration de l'opération de pooling

- ✓ **Max Pooling**

Est un processus de convolution où le Kernel (noyau) extrait la valeur maximale de la zone du featuremap. Il est de loin le plus utilisé car son calcul est immédiat et les images sont efficacement simplifiées.

- ✓ **MeanPooling**

Appelé aussi averagepooling, calcule la moyenne des éléments présent dans le featuremap couverte par le filtre. Alors que le max pooling donne la feature la plus importante dans un patch particulier de la featuremap, le meanpooling donne la moyenne des fonctionnalités présentes dans un patch.

II.5.4.3 La couche de correction ReLU

L'efficacité du traitement est améliorée en intercalant entre les couches de traitement une couche qui va opérer une fonction mathématique (fonction d'activation) sur le signal de sortie [23].

ReLU (rectifiedlinear unit) se réfère à la fonction réelle non linéairedéfini par l'équation (II.1) suivante :

$$\mathbf{ReLU(x) = max\ (0,x)}(\text{II.1})$$

La couche de correction ReLU remplace donc par zéro toutes les valeurs négatives reçues en entrée,Elle agit comme une fonction d'activation; comme illustré et présenter dans la figure II.14 qui suit :

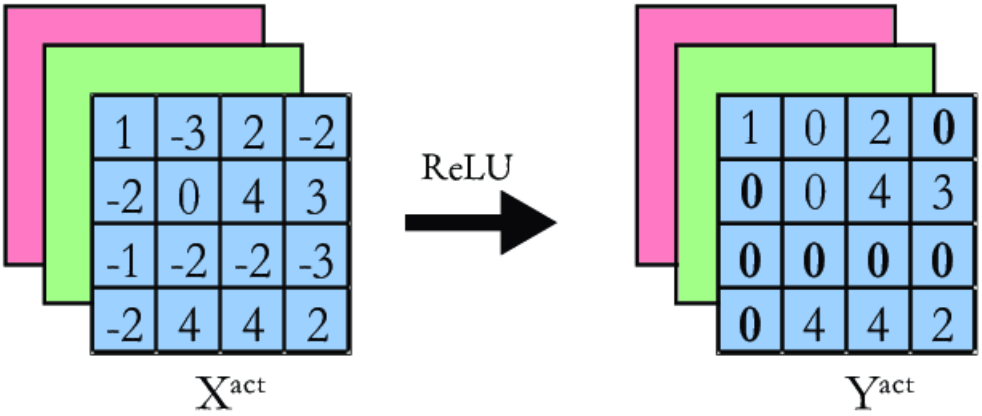


Figure II. 13 : la représentation de la couche de correction ReLu

- **But :**

Son but est d'améliorer l'efficacité du traitement en intercalant entre les couches de traitement une couche qui va opérer une fonction mathématique.

II.5.4.4 La couche Fully-Connected

Est la dernière couche d'un réseau de neurones, convolutif ou non, ce type de couche reçoit un vecteur en entrée et produit un nouveau vecteur en sortie.

Elle applique une combinaison linéaire puis éventuellement une fonction d'activation aux valeurs reçues en entrée.

La figure II.15 représente l'architecture de manière générale de la couche complètement connectée (Fully-connected) :

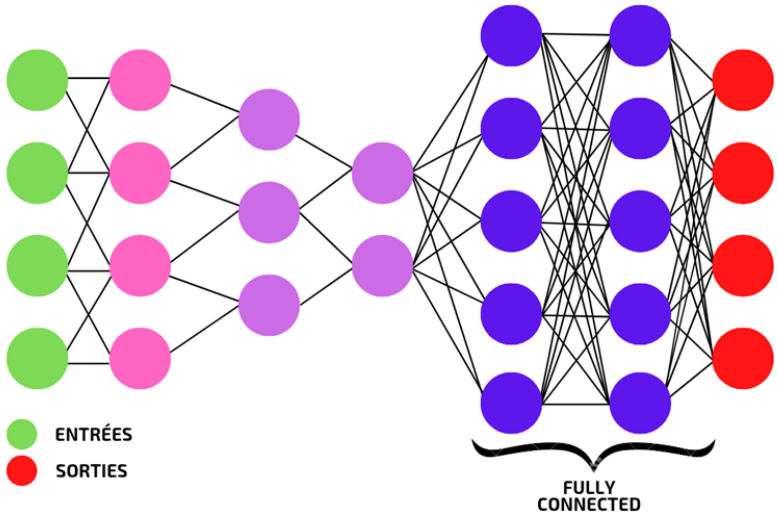


Figure II. 14 : Représentation de la couche fully-connected

- **But :**

Son but est de classifier l'image en entrée du réseau : elle renvoie un vecteur de taille N, ou N est le nombre de classes dans notre problème de classification d'image. Chaque élément du vecteur indique la probabilité pour l'image en entrée d'appartenir à une classe. Elle détermine le lien entre la position des features dans l'image et une classe.

II.5.5 Les avantages de réseau CNN

Un avantage majeur des réseaux convolutifs est l'utilisation d'un seul poids associé au signal entrant dans tous les neurones d'un seul noyau convolutif.

Cette approche réduit l'empreinte mémoire, améliore les performances et permet l'invariance de la traduction. C'est le principal avantage des réseaux neuronaux convolutifs par rapport aux perceptrons multicouches, qui considèrent que chaque neurone est indépendant et attribuent donc des poids différents à chaque signal entrant[24].

II.6 Conclusion

Après avoir approfondis les connaissances en Deep Learning on remarque qu'il développe l'intelligence artificielle et il est parmi les domaines évolués et nécessaire pour persévérée dans le domaine du machine Learning.

On à pu voir et traité les réseaux de neurones artificiels utilisées en DL et particulièrement les réseaux de neurones convolutionnels, en détaillant leur conception et leur utilisation dans le traitement de la reconnaissance, ainsi que leurs différentes couches.

Chapitre II L’apprentissage profond

Dans le chapitre qui suit nous allons mettre en œuvre nos connaissances en apprentissage profond et aussiprocéder à l’application de ces réseaux de neurone convolusionnels dans la reconnaissance de formes géométriques. Ces réseaux sont capables d’extraire des caractéristiques d’images présentées en entrée et de les classifier. Ils implémentent l’idée de partage des poids permettant ainsi de réduire d’une part le nombre de paramètres libres de l’architecture, d’autre part de réduire les temps de calcul, l’espace mémoire nécessaire, et améliorer ainsi les capacités de généralisation du réseau.

Chapitre III

Reconnaissance de formes géométriques avec DL et l'attribut HMB

III.1 Introduction

Le Deep Learning a montré de bonnes performances dans de nombreuses taches d'intelligence artificielle et d'apprentissage automatique ; Le système de reconnaissance de forme fait face à plusieurs défis, y compris la variation illimitée des formes existantes et des grandes bases de données publique.

L'extraction d'attributs à partir d'images peut être effectuée de deux manières. Premièrement, la manière la plus classique consiste à concevoir des attributs spécialement adaptés à la tâche et parfois même adaptées à un ensemble de données spécifique (écriture, formes, etc.). Ces attributs sont appelés manuels (handcrafted) puisqu'un algorithme a été conçue manuellement pour les extraire, incorporant des informations à priori sur les spécificités des données parmi lesquels on trouve l'attribut HMB (Histogram of marqued background) [25]. La deuxième catégorie d'attributs consiste à apprendre automatiquement les attributs à partir des images en utilisant l'apprentissage automatique. C'est une solution qui est de plus en plus utilisée depuis l'avènement du Deep Learning avec les Réseau neuronal convolutif (CNN) [26] [27].

Dans le but de palier à l'inconvénient principal de l'utilisation du DeepLearning qui réside dans la nécessité de disposer de beaucoup de données d'apprentissage, nous proposons de combiner entre l'attribut HMB et le CNN pour la reconnaissance des formes géométrique.

Ce chapitre est destiné à présenterla conception de nos différentes architectures de CNN pour la base de données des formes géométriques de 12000 images pour le fichier train et 4000 pour le test, tous en combinent entre l'attribut HMB et le CNN.

III.2 HMB (Histogram of marqued background)

Le principe général de l'attribut HMB est d'injecter des informations discriminantes dans l'image du texte et cela en attribuant des marques aux pixels du fond de l'image du texte. Les pixels de fond sont marqués selon la répartition spatiale des pixels d'encre dans leur voisinage. L'attribut HMB est extrait en deux étapes : marquage du fond de l'image et calcul des histogrammes [25]. Dans notre mémoire on utilise uniquement la phase de marquage d'image.

III.2.1 Marquage du fond

L'idée principale de HMB est d'enrichir l'image du texte pour qu'elle soit plus représentative en intégrant des informations contextuelles dans le fond de l'image du texte. Pour atteindre cet objectif, des marques sont attribuées pour les pixels de fond ayant des pixels d'encre dans leur voisinage. Pour Ceci, les auteurs ont considéré un rayon « r » pour délimiter la région dans laquelle les voisins d'encre sont identifiés et des directions « d » dans lesquelles ils cherchent la présence de pixels d'encre voisins [25].

Dans notre cas nous avons utilisé un rayon de 4 pixels et les directions montrées dans la figure III.1 :

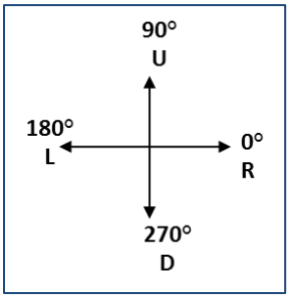


Figure III. 1 : Configuration avec 4 directions

Si on considère une image d'un mot « I » après nettoyage et binarisation définie par l'équation (1) qui suit :

$$I(i,j) = \begin{cases} 1 & if I(i,j)\ encre \\ 0 & otherwise \end{cases} \qquad (1)$$

Pour chaque pixel de fond I (i,j)=0, les auteurs [25] déterminent une marque M (i,j). Pour ce faire, ils considèrent un cercle « Cr » avec un rayon de « r » pixels centré au pixel de fond I (i,j) à marquer. Ensuite, ils cherchent la présence de pixels d'encre dans chaque direction montrée dans la figure III.1. Enfin, la marque M (i, j) est définie par les directions dans lesquelles ils ont trouvé des pixels d'encre dans le cercle "Cr". L'image marquée est définie par l'équation (4).

$$I_{marked}(i,j) = \begin{cases} 1 & if I(i,j)\ encre \\ \mathrm{M(i,j)} & otherwise \end{cases} \qquad (4)$$

La figure III.2 montre toutes les possibilités de marques en binaire et en décimal en utilisant 4 directions.

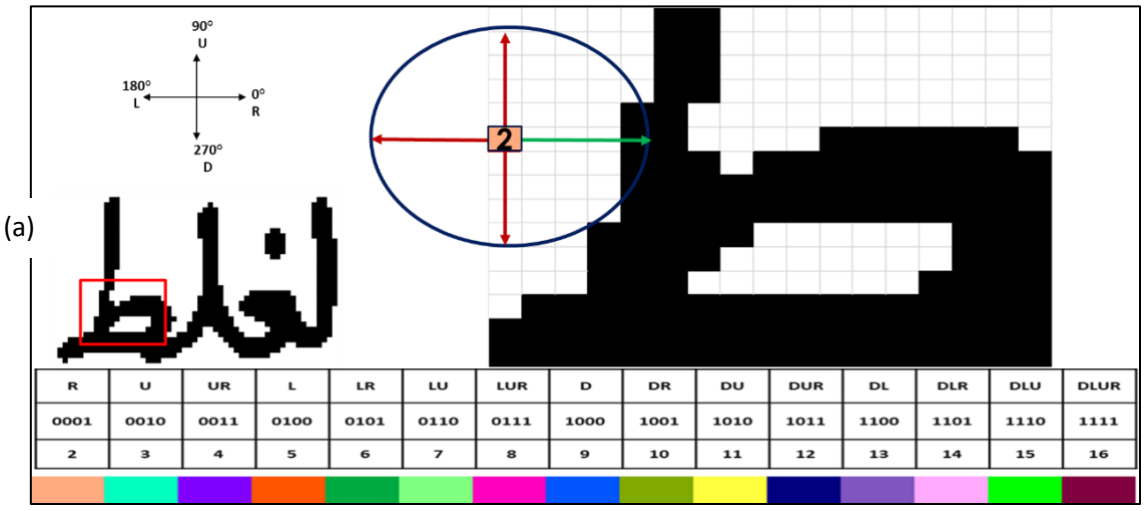
R	U	UR	L	LR	LU	LUR	D	DR	DU	DUR	DL	DLR	DLU	DLUR
0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Figure III. 2 : Toutes les possibilités de marques avec quatre directions et leur code binaire ainsi que leurs signes décimaux correspondants.

III.2.2 Exemples de marquage

Dans cette section nous présentons un exemple illustrant le marquage des pixels de fond d'une image d'un mot suivant les directions (0°,90°,180°,270°) et délimité par un rayon de 4 pixels. Une couleur différente est utilisée pour chaque pour chaque marque.

Dans la Figure 3 le pixel de fond mis en évidence est marqué avec la marque **R** qui correspond au code binaire (0001) et la valeur décimale **2** car il a un seul voisin d'encre dans la région délimitée par le cercle de rayon de 4 pixels et dans la direction droite (R) illustré avec la flèche verte [25].



R	U	UR	L	LR	LU	LUR	D	DR	DU	DUR	DL	DLR	DLU	DLUR
0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Figure III. 3 : Exemple de marquage d'un pixel de fond ayant un seul voisin d'encre

La figure III.4 montre un exemple d'images après le processus de binarisation et marquage de quelques caractères la base des formes :

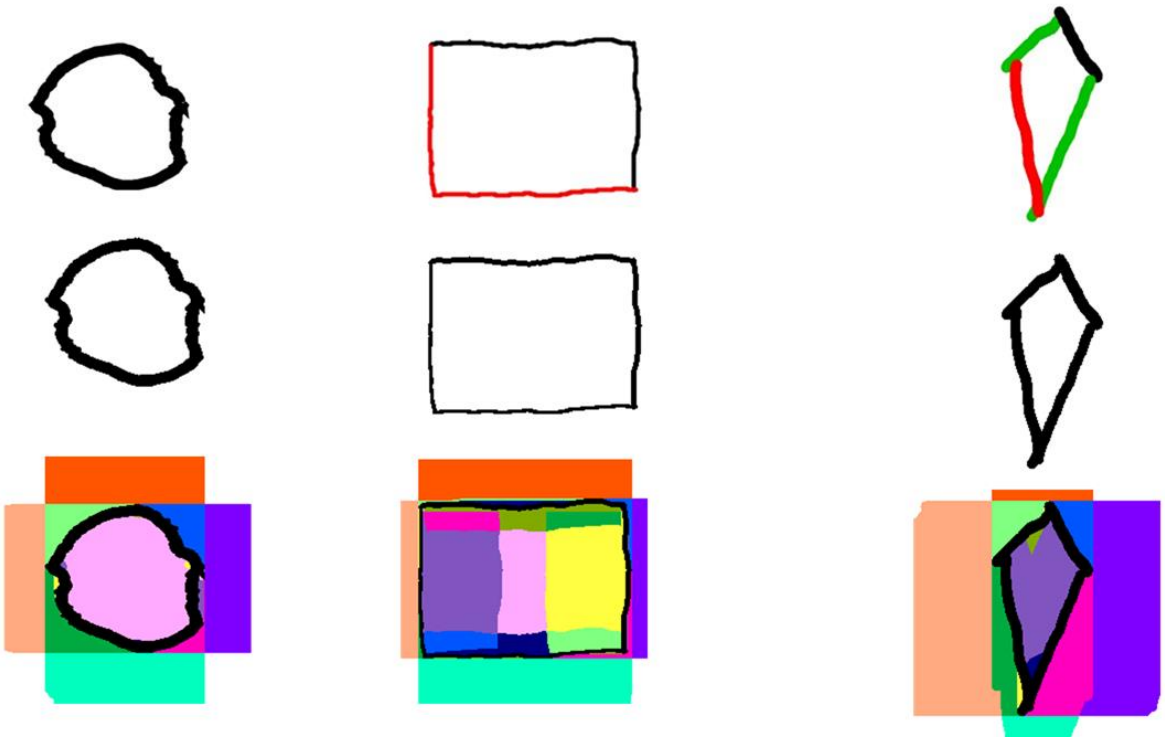


Figure III. 4 : exemples d'images de caractères après le processus de binarisation et marquage de la base de caractère forme

L'image des formes passe par l'étape de binarisation puis par le marquage du fond en utilisant l'attribut HMB. Puis l'image marquée serra passé au processus d'apprentissage automatique par le Deep Learning en utilisant les réseaux CNN.

La figure III.5 présente le schéma général de notre système pour la reconnaissance des formes géométrique.

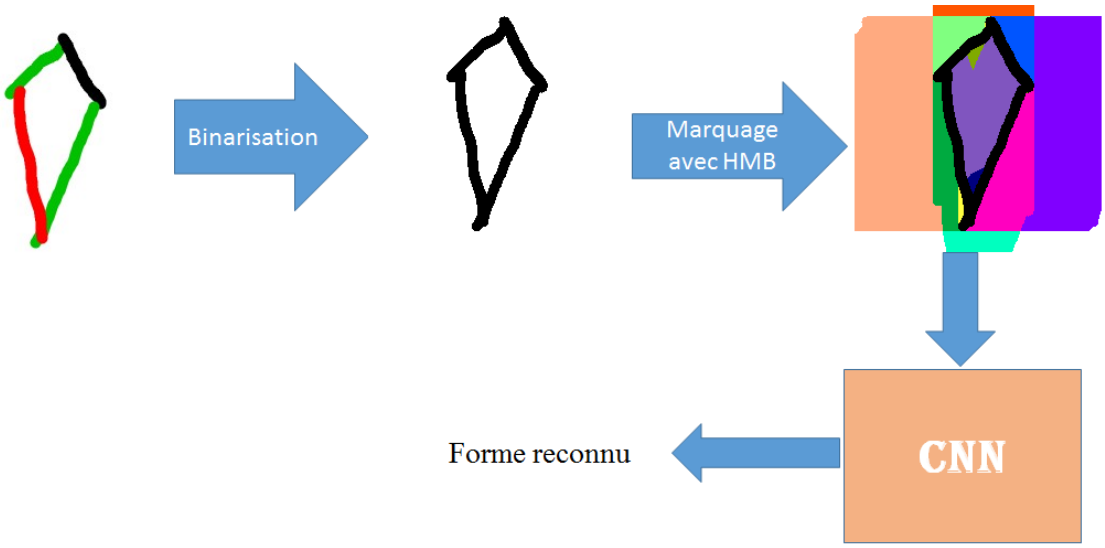


Figure III. 5 : Schéma générale de notre système

III.3 Conception

Avant d'entrer dans l'implémentation, on va expliquer comment concevoir notre programme.

Il y a deux grands processus qui englobent notre conception

III.3.1 Training

C'est le processus le plus important parce qu'on va créer notre modèle grâce à des configurations précises.

- **La base de données :** c'est une base de données d'images répertoriées en classes. Par exemple, Si on prend la base de données de formes géométriques utilisées durant notre simulation on peut trouver des classes comme 'circle' 'triangle' 'kite' ... etc.
  C'est une base de données de 8 classes, pour 12000 images pour le fichier du traitement (tain) avec 1500 images par classe.
- **labels_classes :** c'est un fichier texte qui portera les noms des classes de notre base de données (dataset).
- **CNN et paramètres :** c'est notre algorithme de création d'un réseau neurones convolutionels qui sera configuré avec des paramètres, par exemple : nombre d'époques, nombre de filtre, nombre de couche ...etc. On va exécuter la dataset (la base de données) sur notre algorithme CNN qui sera paramétrée pour générer un modèle puis on va utiliser ce modèle pour le Test.

III.3.2 Test

Dans le processus Test on retrouve :

- **La base de données :** c'est une base de données d'images répertoriées en classes contenant un nombre inférieur d'images par rapport au training.
  C'est une base de données de 8 classes, pour 4000 images pour le fichier du test avec 500 images par classe.
- **Le modèle :** c'est un fichier généré dans notre training.
- **L'affichage de la classification :** son nom résume son travail, on va afficher le résultat sorti du modèle qui est le nom d'une classe.

III.4 Logiciel et librairies utilisé dans l'implémentation

III.4.1 Python

III.4.1.1 Historique

Le langage de programmation Python a été créé en 1989 aux Pays-Bas par Guido van Rossum. Le nom Python vient d'un mot de la langue anglaise, Hommage à la série télévisée Monty Python's Flying Circus, dont G.van Rossum était un fan. La première version publique du document La langue a été publiée en 1991[28].

III.4.1.2 Définition

Python est le langage de programmation open source le plus utilisé par les informaticiens. Il est devenu un langage de premier plan pour la gestion des infrastructures, l'analyse des données et le développement de logiciels [29].
Est un langage interprété qui n'a pas besoin d'être compilé pour être utilisé.

III.4.1.3 Ses bibliothèques

- Numpy et Scipy **:** pour le calcul scientifique
- BeautifulSoup et Scrapy **:** pour l'exploitation de données.
- Pandas, Matplotlip, Seaborn et Plotly **:** pour l'exploitation et la visualisation de données (dataviz).
- Scikitlearn, Pycat, Keras, PyTorch et Tensorflow **:** pour la machine Learning.

III.4.2 Keras

III.4.2.1 Définition

Il s'agit d'une bibliothèque open source écrite en Python (sous la licence MIT). L'objectif de cette bibliothèque est de permettre la composition rapide de réseaux de neurones. Dans ce cas, Keras n'est pas utilisé comme son propre cadre, mais comme une interface de programmation d'applications (API) pour accéder à différents cadres d'apprentissage automatique et les programmer [30].

Theano, Microsoft Cognitive Toolkit (anciennement CNTK) et TensorFlow font partie des frameworks pris en charge par Keras.

III.4.2.2 Le mode de fonctionnement

Keras est une bibliothèque de niveau modèle : elle fournit des modules pour développer des modèles complexes d'apprentissage profond. Contrairement aux cadres autonomes, cette plateforme ne s'occupe pas des opérations de bas niveau, mais utilise les bibliothèques des cadres d'apprentissage automatique connexes comme moteur de base pour les réseaux neuronaux à mettre en œuvre. Selon le principe de la modularité, les couches nécessaires sont connectées. Cependant, il n'est pas nécessaire de connaître l'infrastructure réelle du framework choisi et les utilisateurs de Keras ne doivent pas le lancer directement.

III.4.2.3 Les avantages et les inconvénients de Keras

- **Les avantages :**

Le cadre le plus élevé et le plus convivial de la liste. Il permet aux utilisateurs de choisir si les modèles qu'ils construisent fonctionnent sur Theano ou TensorFlow. Il est écrit et maintenu par Francis Jollet, un autre membre de l'équipe Google Brain.

- ✓ Écrit avec python.
- ✓ Excellent backend pour Theano ou TensorFlow.
- ✓ Interface intuitive de haut niveau

- **Les inconvénients :**

- ✓ Moins flexible que les autres API.

III.4.2.4 Les structures de données de base de keras

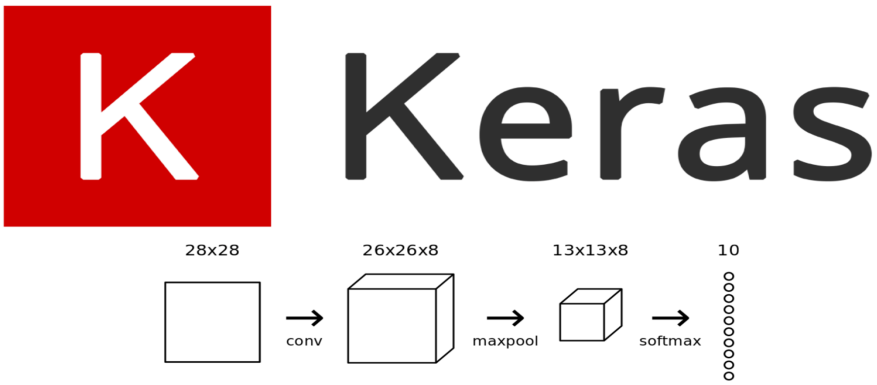


Figure III. 6 : La structure de données de la bibliothèque Keras

III.4.2.5 Les modules de keras utilisés

Les modules de Keras qu'on a utilisés sont :

- **Keras.models :**Sequential
- **Keras.layers :** Conv2D, MaxPooling2D, Flatten, Dense et Dropout
- **Keras.preprocessing.image :**ImageDataGenerator.

III.4.3 TensorFlow

III.4.3.1 Définition

TensorFlow est une plateforme open source de bout en bout dédiée à l'apprentissage automatique. Elle fournit un écosystème complet et flexible d'outils, de bibliothèques et de ressources communautaires qui permettent aux chercheurs en apprentissage automatique de faire progresser le domaine de Deep Learning et aux développeurs de créer et de déployer facilement des applications qui exploitent cette technologie [31].

Plus qu'un simple cadre d'apprentissage profond. Il s'agit d'un cadre informatique polyvalent permettant d'effectuer des opérations mathématiques générales de manière parallèle et distribuée [31].

III.4.3.2 Le mode fonctionnement

Tensorflow fonctionne sur le principe des graphiques de flux de données. Pour effectuer un calcul, il y a deux étapes [32] :

- Représentez le résultat du calcul sous forme graphique.
- Exécutez le graphe.

Comme tout graphe dirigé, un graphe Tensorflow est constitué de nœuds et d'arêtes directionnelles.

Node: les nœuds sont également appelés Op (pour opération). Un nœud peut avoir plusieurs arêtes entrantes, mais un seul front sortant.

Bord: indique les données entrantes ou sortantes d'un nœud. Dans ce cas, les entrées et sorties de certains nœuds (Op).

Chaque fois que nous parlons de données, nous faisons référence à un vecteur à n dimensions appelé tensor. Un tenseur a trois propriétés : rang, forme et type.

- **Le rang :** fait référence au nombre de dimensions du tenseur (un cube ou un carré a un rang de 3).
- **La forme :** fait référence à la valeur de ces dimensions (une boîte peut être 1x1x1 ou 2x5x7).
- **Le type :** fait référence au type de données dans chaque coordonnée tensorielle.

III.4.3.3 Les avantages et les inconvénients de tensorflow

❖ **Les avantages:**

- ✓ Est écrit en Python
- ✓ Propulsé par Google
- ✓ Une très grande communauté
- ✓ Support multi-GPU [32]

❖ **Les Inconvénients :**

- ✓ Plus lent que les autres frameworks**(cadre)** dans de nombreux benchmarks, bien que Tensorflow le compense, il rattrape le temps perdu.
- ✓ La prise en charge du RNN est toujours dépassée par Theano [32].

- **La différence entre TensorFlow et keras**

Tensorflow et Keras sont deux modules d'apprentissage automatique bien connus, utilisés dans le domaine de la science des données, dans le tableau III.1 qui suit nous allons examiner les différences entre TensorFlow et Keras [33] :

	TensorFlow	Keras
1	TensorFlow est écrit en C++, CUDA et python.	Keras est écrit en python.
2	TensorFlow est utilisé pour les grands ensembles de données et les modèles hautes performances.	Keras est généralement utilisées pour les petits ensembles de données.
3	TensorFlow est un framwork qui propose des API de haut niveau.	Keras est une API de haut niveau.
4	TensorFlow est utilisé pour les modèles hautes performances.	Keras est utilisé pour les modèles peu performants.
5	Dans TensorFlow, le débogage entraîne des complexités.	Dans le framwork Keras, il n'y a qu'une exigence minimale pour le débogage des réseaux simples
6	TensorFlow a une architecture complexe et pas facile à utiliser.	Keras a une architecture simple et facile à utiliser.
7	TensorFlow à été développé par l'équipe Google Brain.	Keras a été développé par François Chollet alors qu'il travaillait dans le cadre de l'effort de recherche du projet ONEIROS.

Tableau III. 1 : la différence entre TensorFlow et Keras

III.5 configuration utilisée dans l'implémentation

La configuration utilisée du matériel dans notre configuration est :

Appareil utilisé	DELL-Pc
Processus	Wireless 1705 802.11 b/g/n (2,4 GHz)
RAM	4 GO
Système d'exploitation	Windows 10 professionnel N
Le navigateur utilisé	Anaconda avec la plateforme Spyder
Le type du système	Système d'exploitation 64 bits, processur*64

Tableau III. 2 : configuration utilisée dans l'implémentation

III.6 La base de données

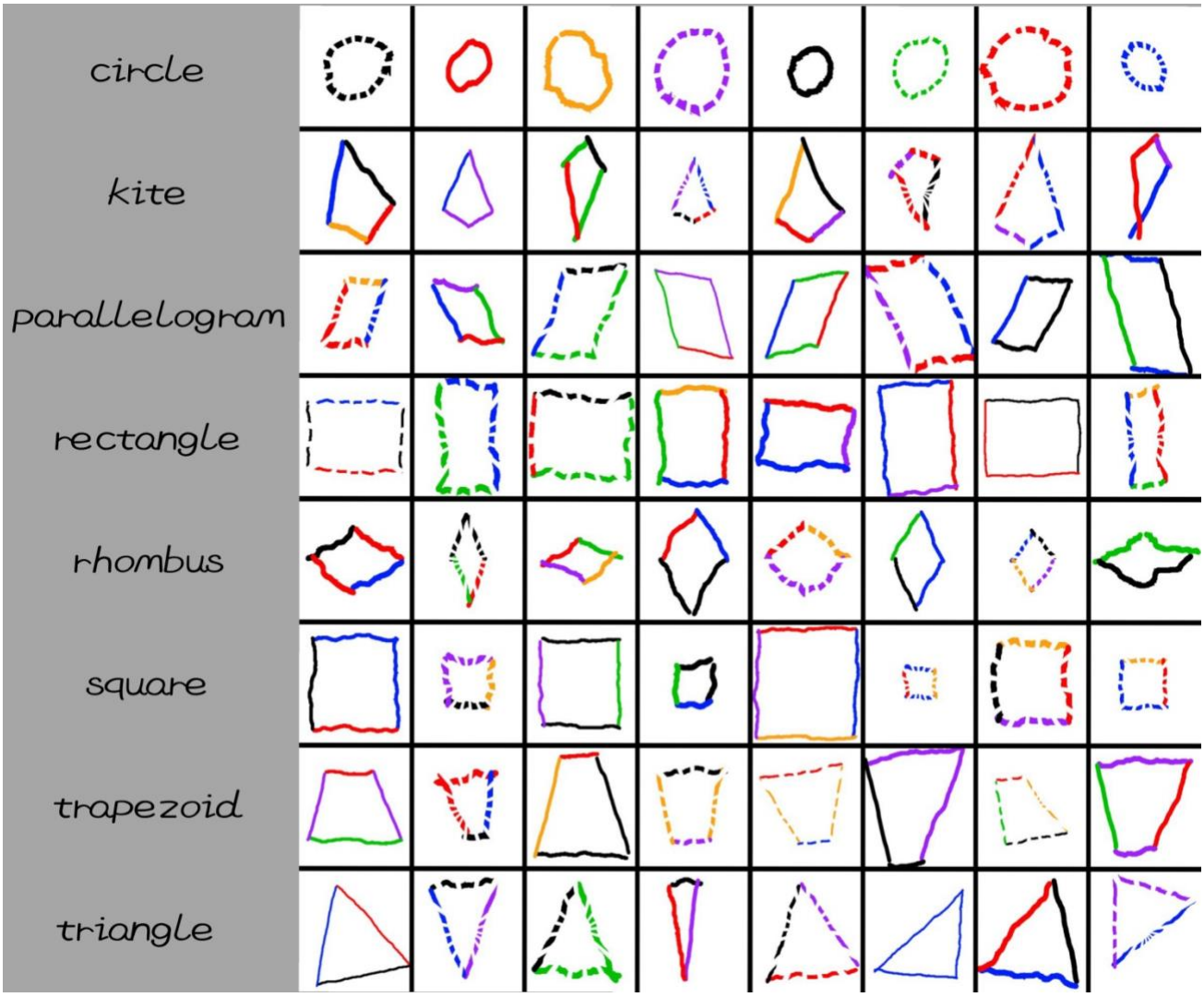


Figure III. 7 : la structure générale de notre base de données des formes géométriques

III.6.1 Définition

Une base de données est une collection organisée d'informations structurées, généralement stockées électroniquement dans un système informatique. Une base de données est généralement contrôlée par un système de gestion de base de données (DBMS).

La base de données utilisée comme montrer dans la figure III.7 durant notre travail est bien une base de données de forme géométrique qui contiens 8 formes différentes (circle, kite, triangle…) qu'on appelle classe. Cette dataset contient deux fichiers importants pour notre apprentissage, le 1er fichier est bien le fichier d'entrainement avec les 8 classes et dans chaque classe on a 1500 images ce qui veut dire 12000 images dans l'ensemble, concernant le

deuxième fichier est bien le fichier test toujours pour 8 classes avec 500 images par classe c'est-à-dire 4000 images en tous.

III.7 Architecture du réseau CNN

Dans nos expérimentations, nous avons criée trois modèles avec des architectures déférentes (2 couches, 4 couches et 6 couches) qu'on a appliquées à la base de données des formes géométriques on applique deux modèles différents : avec la binarisation et marquage et sans binarisation et marquage.

Pour chaque modèle, on a fait une évaluation sur le nombre d'époques. Après nous décrirons la structure des trois modèles à deux basses et les résultats obtenus dans chaque cas.

III.7.1 Réseau neuronal convolutif à 2 couches

L'architecture du réseau neuronal convolutif à 2 couches pour la reconnaissance des formes géométriques qui est basée sur le réseau de neurones convolutionnel comme illustré dans la Figure III.8 si dessous :

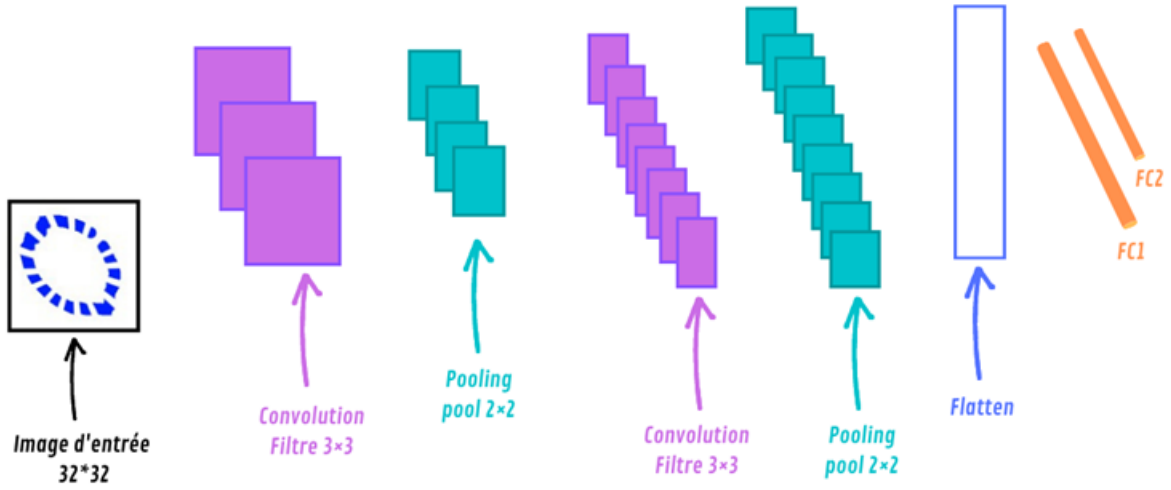
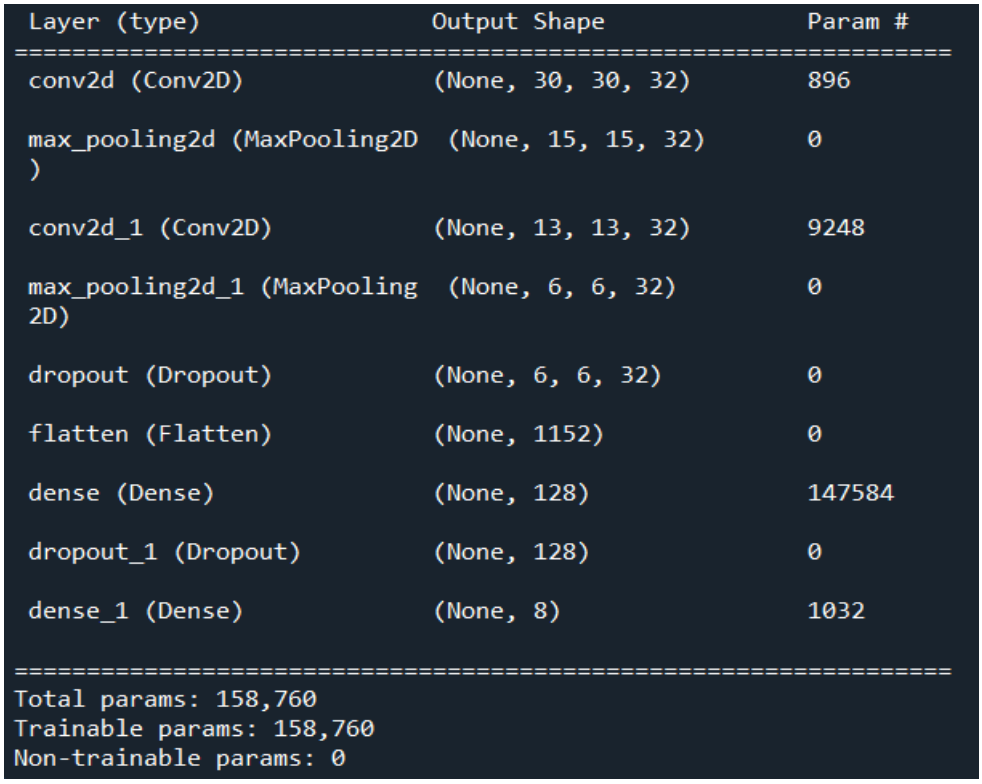


Figure III. 8 : Architecture de modèle à 2 couches

Le modèle est composé de deux couches de convolution, deux couches de maxpooling et deux couches de fullyconnected. L'image en entrée est de taille 32*32, elle passe d'abord à la première couche de convolution. Cette couche est composée de 32 filtres de taille 3*3, Chacune de nos couches de convolution est suivie d'une fonction d'activation ReLU cette

fonction force les neurones à retourner des valeurs positives, Le Maxpooling est appliqué après pour réduire la taille de l'image 32 featuresmaps (images caractéristiques) de taille 28*28 seront créés. Ensuite on applique Maxpooling pour réduire la taille de l'image ainsi la quantité de paramètres et de calcul. À la sortie de cette couche, nous aurons 32 featuremaps de taille 15*15. On répète la même chose avec la deuxième couche de convolution cette couche est composée de 64 filtres, la fonction d'activation ReLU est appliquée toujours sur chaque convolution. Une couche de Maxpooling est appliquée après la deuxième couche de convolution. À la sortie de cette couche, nous aurons 32 featuremaps de taille 6*6. Pour ne pas tomber dans le problème de la surinterprétation il faut utiliser l'instruction Dropout elle est très efficace pour les réseaux de neurones, elle permet de désactiver un nombre de neurones selon notre configuration, cette dernière sera utilisée aussi à la sortie de la première couche de fully-connected. Le vecteur de caractéristiques issu des convolutions a une dimension de 2304. Après ces couches de convolution, nous utilisons un réseau de neurones composé de deux couches fullyconnected, la première couche possède 128 neurones où la fonction d'activation utilisée est le ReLU, pour la deuxième couche sa fonction d'activation est softmax qui permet de calculer la distribution de probabilité des 8 classes (nombre de classe dans la base donnée des formes géométriques utilisées).



Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_1 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_1 (MaxPooling2D)	(None, 6, 6, 32)	0
dropout (Dropout)	(None, 6, 6, 32)	0
flatten (Flatten)	(None, 1152)	0
dense (Dense)	(None, 128)	147584
dropout_1 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 8)	1032

Total params: 158,760
Trainable params: 158,760
Non-trainable params: 0

Figure III. 9 : configuration du modèle à 2 couches

III.7.2 Réseau neuronal convolutif à 4 couches

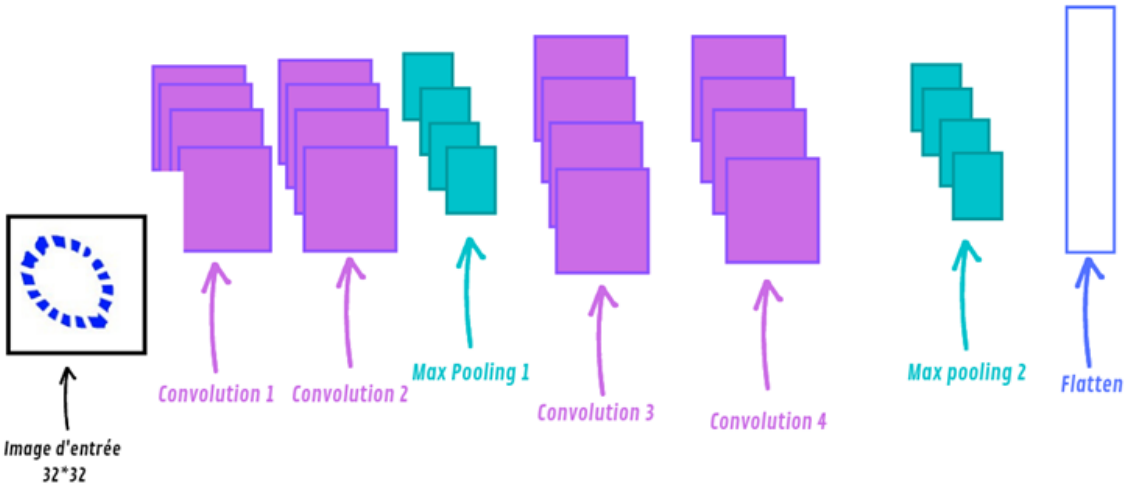


Figure III. 10 : Architecture de modèle à 4 couches

Le modèle que nous présentons dans la figure III.10 est composé de quatre couches de convolution, deux couches de maxpooling et de trois couches de fullyconnected.

L'image en entrée est de taille 32*32, chaque couche de convolution composée de plusieurs filtres (32 pour les 2 première couche), où la taille de chaque filtre est de 5*5. La fonction d'activation ReLU est utilisée à chaque fois qu'on passe par une couche de convolution, après cette convolution, 32 featuresmaps de taille 28*28 seront créés. Le Maxpooling est appliqué à la fin de la deuxième couche de convolution Dans la troisième et quatrième couche, on change quelques paramètres comme le nombre de filtres qui devient 64 au-lieu de 32, la fonction ReLU reste la même tout comme maxpooling, nous aurons 64 featuremaps de taille 5*5 puis on utilise Dropout. Après ces quatre couches nous utilisons un réseau de neurones composé de trois couches fully- connected. Pour convertir nos données 3D en 1D, nous utiliserons la fonction Flatten, Les deux premières couches ont chacune 256 neurones où la fonction d'activation utilisée est bien ReLU, tant dis que pour la troisième couche, c'est un softmax, illustré par la figure III.11

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 32, 32, 32)	896
conv2d_1 (Conv2D)	(None, 30, 30, 32)	9248
max_pooling2d (MaxPooling2D)	(None, 15, 15, 32)	0
dropout (Dropout)	(None, 15, 15, 32)	0
conv2d_2 (Conv2D)	(None, 13, 13, 64)	18496
conv2d_3 (Conv2D)	(None, 11, 11, 64)	36928
max_pooling2d_1 (MaxPooling2D)	(None, 6, 6, 64)	0
dropout_1 (Dropout)	(None, 6, 6, 64)	0
flatten (Flatten)	(None, 2304)	0
dense (Dense)	(None, 256)	590080
dense_1 (Dense)	(None, 128)	32896

Figure III. 11 : configuration du modèle à 4 couches

III.7.3 Réseau neuronal convolutif à 6 couches

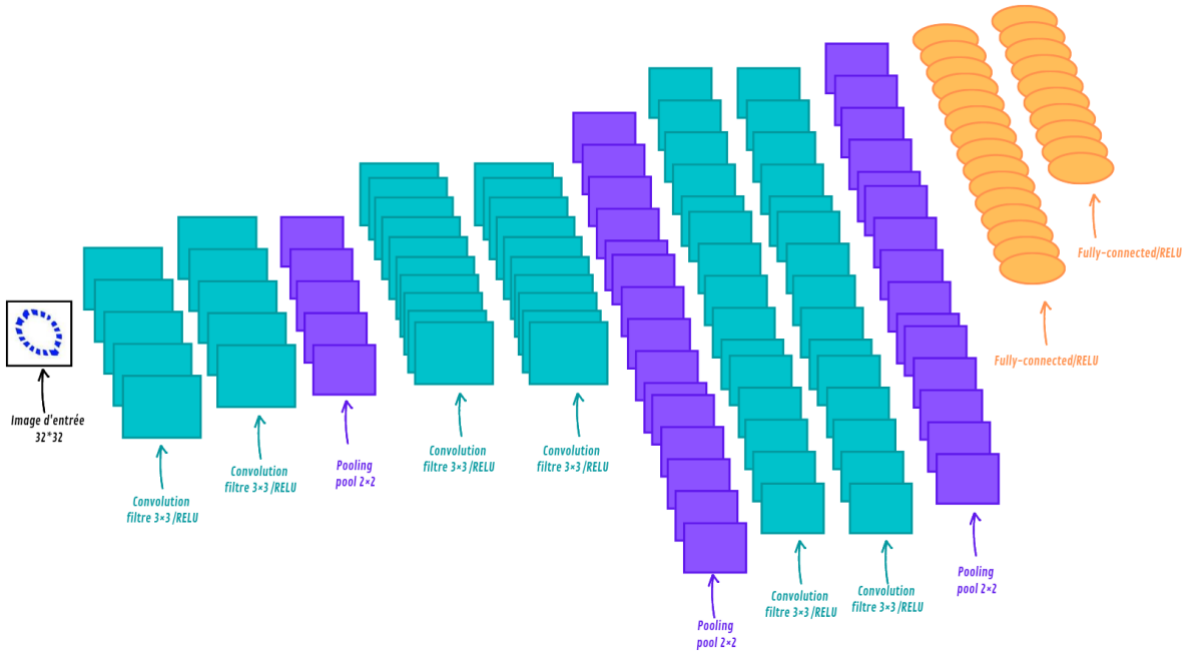


Figure III.12 : Architecture de modèle à 6 couches

Sur ce troisième modèle, on a modifié le nombre de couches pour la convolution et le maxpooling, maintenant notre architecture est composée de six couches convolution, trois couches de maxpooling et de trois dernières couches de fullyconnected. Comme pour les deux autres modèles, l'image en entrée a une taille de 32*32, chaque couche de convolution composée de plusieurs filtres (32 pour les deux premières couches, 64 pour les deux autres couches et 128 filtres pour les deux dernières couches), la taille de chaque filtre est inchangeable pour les 6 couches de convolution (elle est de 3*3), la fonction d'activation utilisée pour toutes ces couches est ReLU, à la fin de chaque deux couches on utilise le Maxpooling. On à utiliser Dropout avec un pourcentage de 20% après chaque couche de convolution et de regroupement, Après ces six couches nous utilisons un réseau de neurones composé de trois couches de fullyconnected en passant par la fonction flatten qui convertira les données à une dimension.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 32, 32, 32)	896
conv2d_1 (Conv2D)	(None, 30, 30, 32)	9248
max_pooling2d (MaxPooling2D)	(None, 15, 15, 32)	0
dropout (Dropout)	(None, 15, 15, 32)	0
conv2d_2 (Conv2D)	(None, 13, 13, 64)	18496
conv2d_3 (Conv2D)	(None, 11, 11, 64)	36928
max_pooling2d_1 (MaxPooling2D)	(None, 6, 6, 64)	0
dropout_1 (Dropout)	(None, 6, 6, 64)	0
conv2d_4 (Conv2D)	(None, 4, 4, 128)	73856
conv2d_5 (Conv2D)	(None, 2, 2, 128)	147584
max_pooling2d_2 (MaxPooling2D)	(None, 1, 1, 128)	0

Figure III. 12 : configuration du modèle à 6 couches

III.8 Code source de notre architecture

```
train_datagen=ImageDataGenerator(rescale=(1./255),shear_range=0.2,zoom_range=0.2,horizontal_flip=(True))
test_datagen=ImageDataGenerator(rescale=(1./255),shear_range=0.2,zoom_range=0.2,horizontal_flip=(True))
training_set= train_datagen.flow_from_directory(path_train,target_size=(32,32),class_mode='categorical')
test_set= test_datagen.flow_from_directory(path_test,target_size=(32,32),class_mode='categorical')
```

- Charger et ajuster la base de données d'image

- 12000 images d'apprentissage

- 4000 images de test

- Compiler le modèle

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',input_shape=(32,32,3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.2))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
```

- Construction de notre modèle CNN à 2 couches

```
model.compile(loss='categorical_crossentropy', optimizer='adam',metrics =['accuracy'])
```

- Compilation et optimisation du modèle

```
history=model.fit(training_set,epochs=4,validation_data=test_set,validation_steps=105)
```

- Lancement d'apprentissage

```
score= model.evaluate(test_set)
```

- Phase d'évaluation

III.9 Résultats et discussion

III.9.1 Sans binarisation et marquage

Dans le Tableau III.3 ci-dessous qui représente la comparaison des 3 modèles sans binarisation et sans marquage (2 couches, 4 couches et 6 couches) en voit que pour le 1er modèle de 2 couches de convolution on aobtenu un pourcentage de 93% pour epoch 4 et 94% pour l'epoch 8 et enfin 95% pour l'epoch 12.

Concernant le 2ème modèle de 4 couches de convolution on obtient un résultat de 94% pour epoch 4 et 8 ainsi qu'un pourcentage de 96% pour l'epoch 12.

Pour finir on a le 3ème et dernier modèle de 6 couches de convolution avec un résultat de 94% pour l'epoch 4 et 95% pour epoch 8 et 12.

	Couche convolution	**Couche Pooling**	**Couche Fullyconnected**	**Nombre Epoques**	**Précision**	**Erreur**	**Temps D'exécution**
1er Modèle 2 couches	2	2	2	4	**93%**	7%	7 min
				8	**94%**	6%	15 min
				12	**95%**	5%	23 min
2èmeModèle 4 couches	4	2	3	4	**94%**	6%	11 min
				8	**95%**	5%	23 min
				12	**96%**	4%	34 min
3ème Modèle 6 couches	6	3	3	4	**94%**	6%	16 min
				8	**95%**	5%	25 min
				12	**95%**	5%	35 min

Tableau III. 3 : comparaison des 3 modèles sans binarisation et marquage

Commentaire :

Après la comparaison des résultats obtenue pour les 3 modèles de 2, 4 et 6 couches de convolution sans binarisation et sans marquage on constate que le meilleur modèle avec le meilleur résultat obtenu est bien le modèle de 4 couches de convolution avec 2 couches de pooling et finir avec les 3 dernières couches de fullyconnected ayant un pourcentage de 96% et un taux d'erreur de 4% pour un epoch de 12 avec un temps d'exécution de 34min.

III.9.1.1 Résultats obtenue pour le modèle à 2 couches

- **Nombre d'époques =4**

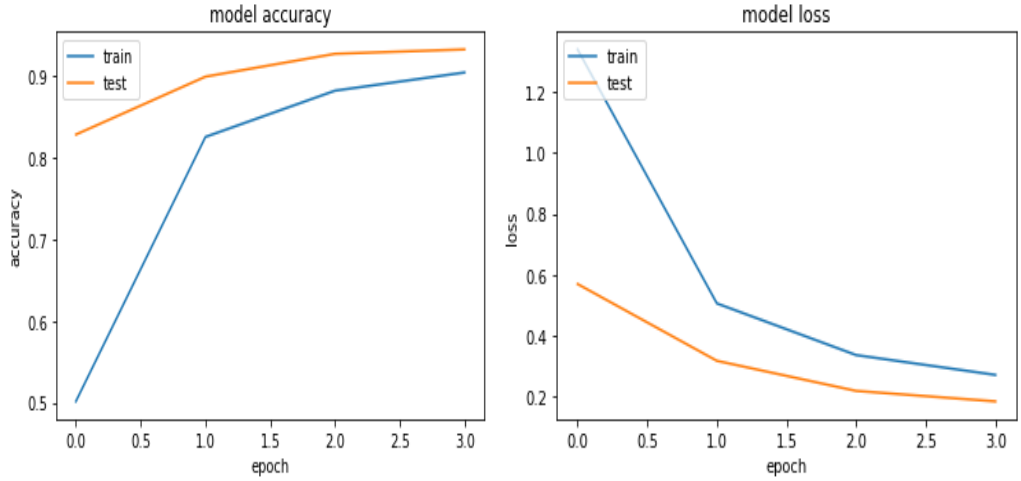


Figure III. 13 : Précision et erreur du modèle à 2 couches (4 époques)

D'après la figure III.14 ci-dessus du modèle à 2 couches sans binarisation et sans marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 4, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 93% ce qui signifie la reconnaissance de 3720 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 7% c'est-à-dire 280 formes mal reconnus du fichier test.

- **Nombre d'époques =8**

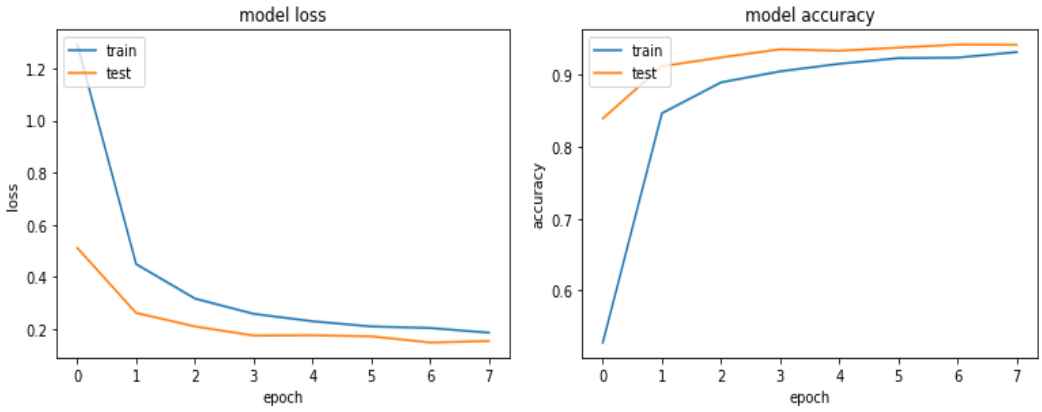


Figure III. 14 : Précision et erreur du modèle à 2 couches (8 époques)

D'après la figure III.15 ci-dessus du modèle à 2 couches sans binarisation et sans marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 8, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 94% ce qui signifie la reconnaissance de 3760 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 6% c'est-à-dire 240 formes mal reconnus du fichier test.

- **Nombre d'époques =12**

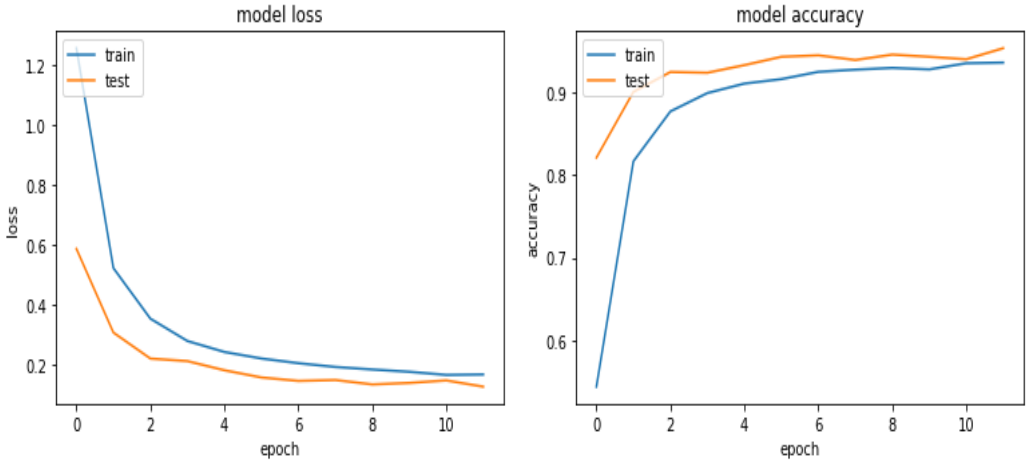


Figure III. 15 : Précision et erreur du modèle à 2 couches (12 époques)

D'après la figure III.16 ci-dessus du modèle à 2 couches sans binarisation et sans marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 12, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 95% ce qui signifie la reconnaissance de 3800 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 5% c'est-à-dire 200 formes mal reconnus du fichier test.

III.9.1.2 Résultats obtenue pour le modèle à 4 couches

- **Nombre d'époques =4**

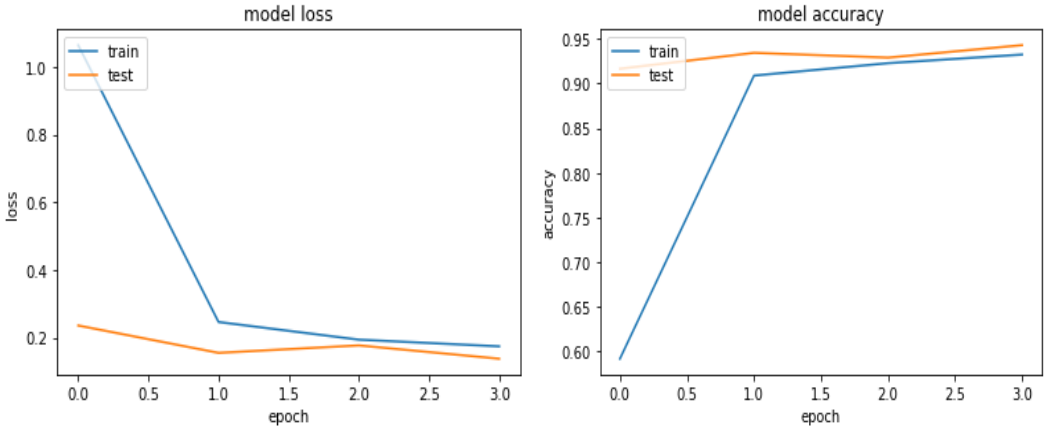


Figure III. 16 :Précision et erreur du modèle à 4 couches (4 époques)

D'après la figure III.17 ci-dessus du modèle à 4 couches sans binarisation et sans marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 4, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 94% ce qui signifie la reconnaissance de 3760 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 6% c'est-à-dire 240 formes mal reconnus du fichier test.

```
Attention

Figures now render in the Plots pane by default. To make them also appear inline in the Console, uncheck "Mute Inline Plotting"
under the Plots pane options menu.

125/125 [==============================] - 14s 116ms/step - loss: 0.1435 - accuracy: 0.9450
Test loss: 0.14353549480438232
Test accuracy: 94.0 %
125/125 [==============================] - 9s 70ms/step
```

➢ **Nombre d'époques =8**

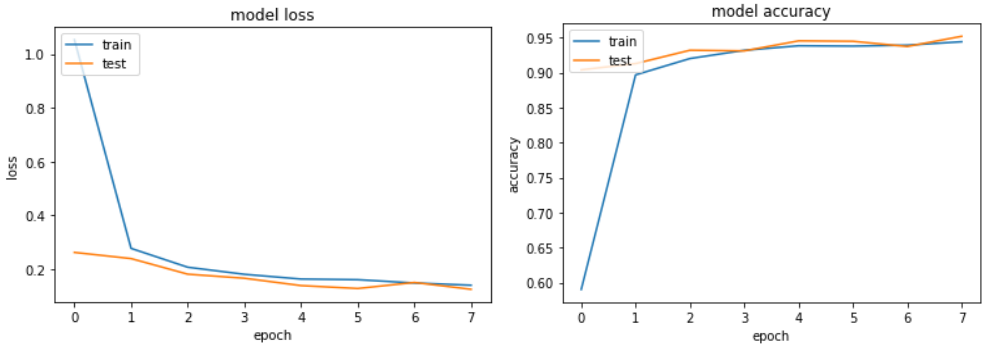


Figure III. 17 : Précision et erreur du modèle à 4 couches (8 époques)

D'après la figure III.18 ci-dessus du modèle à 4 couches sans binarisation et sans marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 8, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 95% ce qui signifie la reconnaissance de 3800 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 5% c'est-à-dire 200 formes mal reconnus du fichier test.

Attention

```
Figures now render in the Plots pane by default. To make them also appear inline in the Console, uncheck "Mute Inline Plotting"
under the Plots pane options menu.
125/125 [==============================] - 9s 70ms/step - loss: 0.1155 - accuracy: 0.9528
Test loss: 0.1155286580324173
Test accuracy: 95.0 %
125/125 [==============================] - 9s 75ms/step
```

➢ **Nombre d'époques =12**

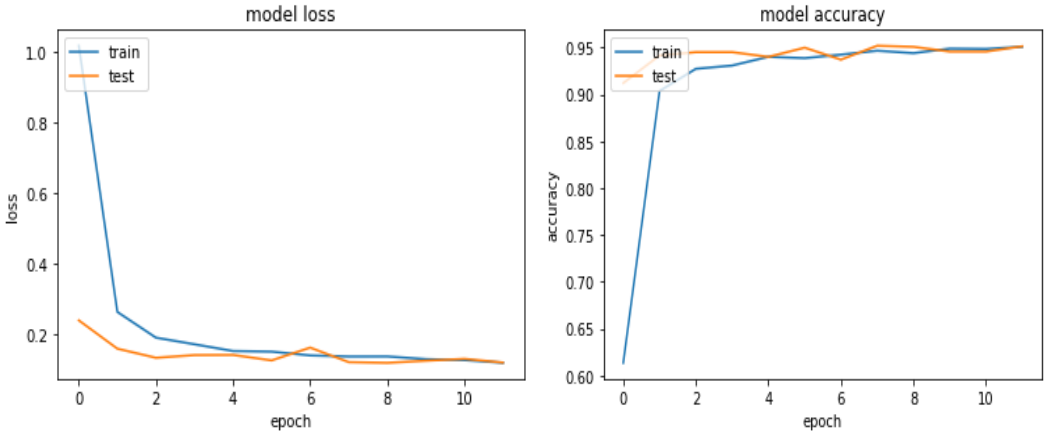


Figure III. 18 : Précision et erreur du modèle à 4 couches (12 époques)

D'après la figure III.19 ci-dessus du modèle à 4 couches sans binarisation et sans marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 12, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 96% ce qui signifie la reconnaissance de 3840 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 4% c'est-à-dire 160 formes mal reconnus du fichier test.

```
Attention

Figures now render in the Plots pane by default. To make them also appear inline in the Console, uncheck "Mute Inline Plotting"
under the Plots pane options menu.

125/125 [==============================] - 10s 82ms/step - loss: 0.1104 - accuracy: 0.9555
Test loss: 0.11036521941423416
Test accuracy: 96.0 %
125/125 [==============================] - 9s 68ms/step
```

III.9.1.3 Résultats obtenue pour le modèle à 6 couches

- **Nombre d'époques =4**

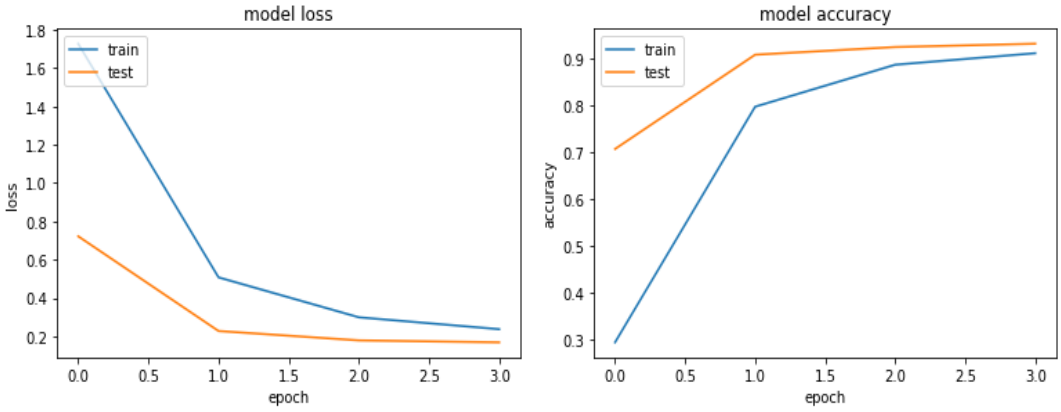


Figure III. 19 : Précision et erreur du modèle à 6 couches (4 époques)

D'après la figure III.20 ci-dessus du modèle à 6 couches sans binarisation et sans marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 4, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 94% ce qui signifie la reconnaissance de 3760 formes parmi les 4000 existante dans notre base de données

des formes géométrique, et un taux d'erreur de 6% c'est-à-dire 240 formes mal reconnus du fichier test.

- **Nombre d'époques =8**

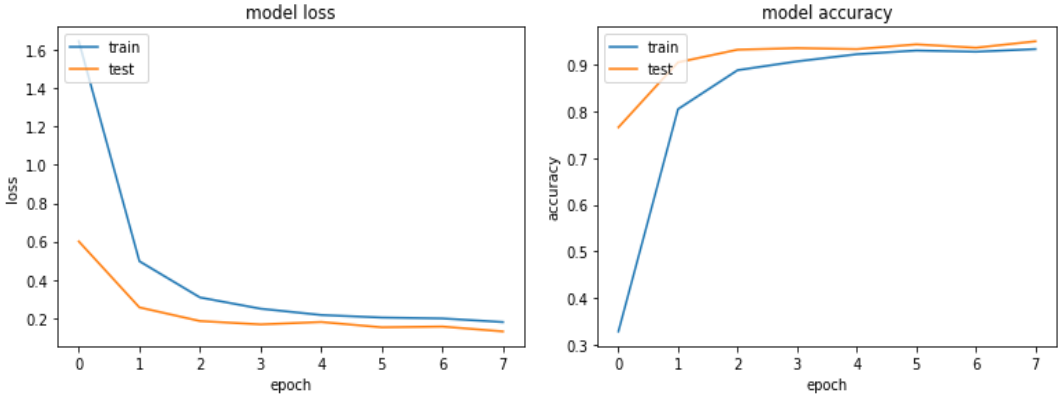


Figure III. 20 : Précision et erreur du modèle à 6 couches (8 époques)

D'après la figure III.21 ci-dessus du modèle à 6 couches sans binarisation et sans marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 8, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 95% ce qui signifie la reconnaissance de 3800 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 5% c'est-à-dire 200 formes mal reconnus du fichier test.

- **Nombre d'époques =12**

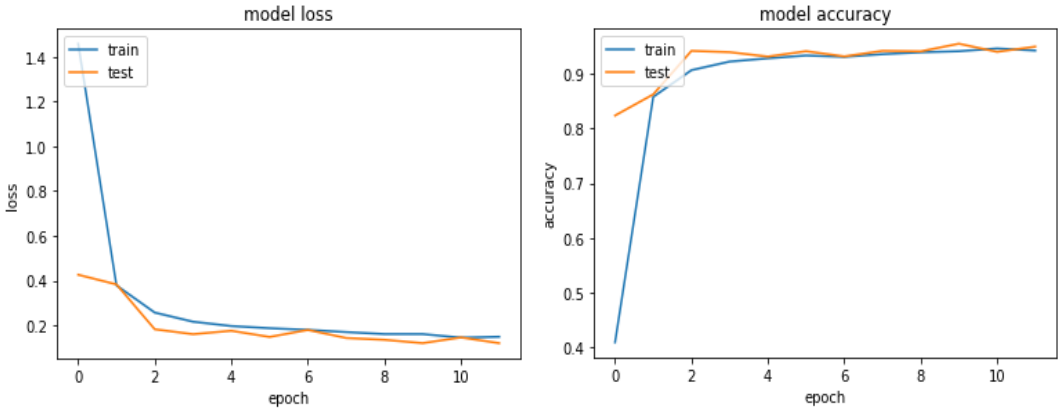


Figure III. 21 : Précision et erreur du modèle à 6 couches (12 époques)

D'après la figure III.22 ci-dessus du modèle à 6 couches sans binarisation et sans marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 12, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 95% ce qui signifie la reconnaissance de 3800 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 5% c'est-à-dire 200 formes mal reconnus du fichier test.

III.9.2 Avec binarisation et marquage

Dans le Tableau III.4 ci-dessous qui représente la comparaison des 3 modèles avec la binarisation et avec marquage (2 couches, 4 couches et 6 couches) en voit que pour le 1er modèle de 2 couches de convolution on à obtenue un pourcentage de 89% pour epoch 4 et 92% pour l'epoch 8 et12.

Concernant le 2ème modèle de 4 couches de convolution on obtient un résultat de 93% pour epoch 4 et 8 ainsi qu'un pourcentage de 94% pour l'epoch 12.

Pour finir on a le 3ème et dernier modèle de 6 couches de convolution avec un résultat de 89% pour l'epoch 4 et 93% pour epoch 8 et 12.

	Couche convolution	**Couche Pooling**	**Couche Fullyconnected**	**Nombre Epoques**	**Précision**	**Erreur**	**Temps D'exécution**
1er Modèle 2 couches	2	2	2	4	**89%**	11%	5 min
				8	**92%**	8%	10 min
				12	**92%**	8%	20 min
				4	**93%**	7%	9 min

2ème Modèle 4 couches	4	2	3	8	**93%**	7%	19 min
				12	**94%**	6%	30 min
3ème Modèle 6 couches	6	3	3	4	**89%**	11%	12 min
				8	**93%**	7%	22 min
				12	**93%**	7%	30 min

Tableau III. 4 : comparaison des 3 modèles avec binarisation et marquage

Commentaire :

Après la comparaison des résultats obtenue pour les 3 modèles de 2, 4 et 6 couches de convolution avec binarisation et marquage on constate que le meilleur modèle avec le meilleur résultat obtenu est bien le modèle de 4 couches de convolution avec 2 couches de

pooling et finir avec les 3 dernières couches de fullyconnected ayant un pourcentage de 94% et un taux d'erreur de 6% pour un epoch de 12 avec un temps d'exécution de 32min.

III.9.2.1 Résultats obtenue pour le modèle à 2 couches

- **Nombre d'époques =4**

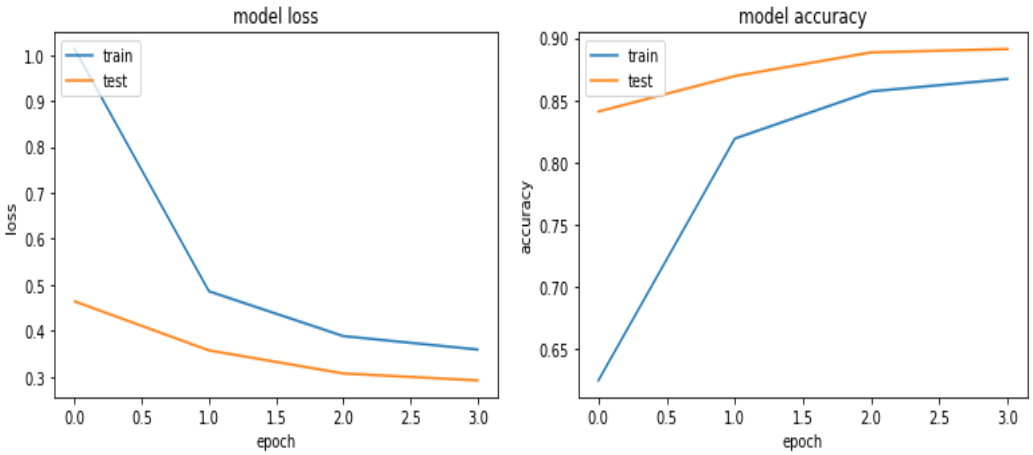


Figure III. 22 : Précision et erreur du modèle à 2 couches (4 époques)

D'après la figure III.21 ci-dessus du modèle à 2 couches avec binarisation et avec marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test

augmente avec l'augmentation du nombre d'époque qui est 4, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 89% ce qui signifie la reconnaissance de 3560 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 11% c'est-à-dire 440 formes mal reconnus du fichier test.

- **Nombre d'époques =8**

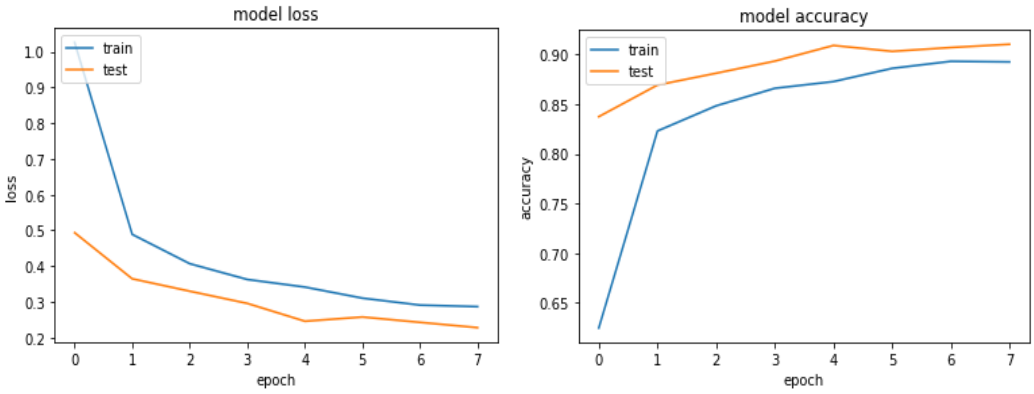


Figure III. 23 : Précision et erreur du modèle à 2 couches (8 époques)

D'après la figure III.22 ci-dessus du modèle à 2 couches avec binarisation et avec marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 8, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 92% ce qui signifie la reconnaissance de 3680 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 8% c'est-à-dire 320 formes mal reconnus du fichier test.

- **Nombre d'époques =12**

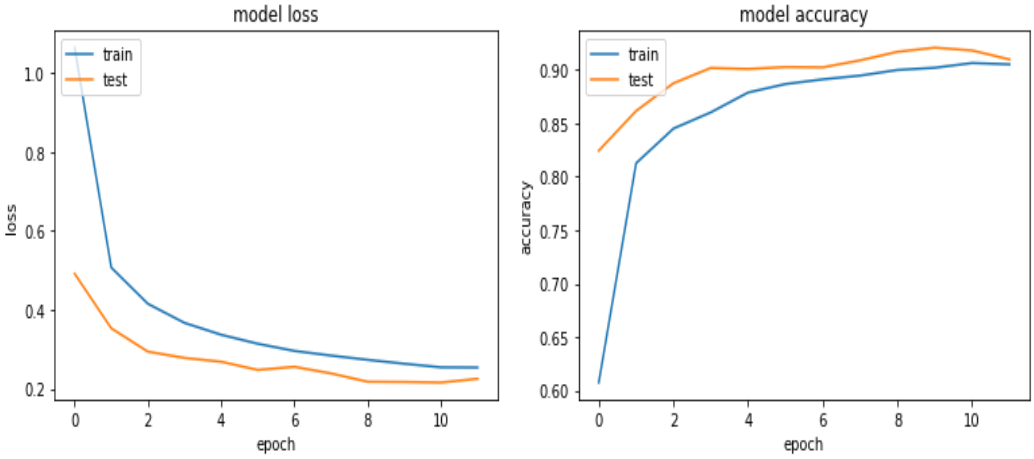


Figure III. 24 : Précision et erreur du modèle à 2 couches (12 époques)

D'après la figure III.23 ci-dessus du modèle à 2 couches avec binarisation et avec marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 12, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 92% ce qui signifie la reconnaissance de 3680 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 8% c'est-à-dire 320 formes mal reconnus du fichier test.

III.9.2.2 Résultats obtenue pour le modèle à 4 couches

- **Nombre d'époques =4**

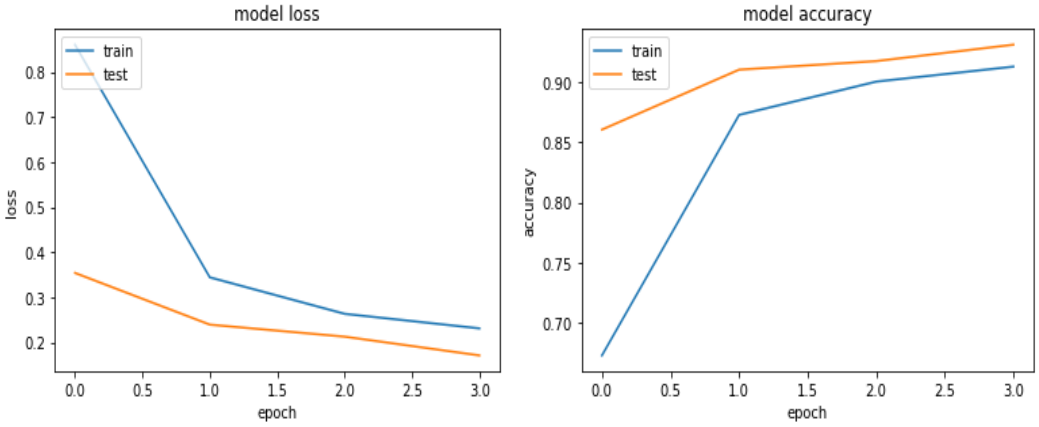


Figure III. 25 : Précision et erreur du modèle à 4 couches (4 époques)

D'après la figure III.24 ci-dessus du modèle à 4 couches avec binarisation et avec marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 4, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 93% ce qui signifie la reconnaissance de 3720 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 7% c'est-à-dire 280 formes mal reconnus du fichier test.

```
Warning
Figures now render in the Plots pane by default. To make them also appear inline in the Console, uncheck "Mute Inline Plotting"
under the Plots pane options menu.
125/125 [==============================] - 36s 290ms/step - loss: 0.1981 - accuracy: 0.9250
Test loss: 0.19807422161102295
Test accuracy: 93.0 %
125/125 [==============================] - 18s 143ms/step
```

- **Nombre d'époques =8**

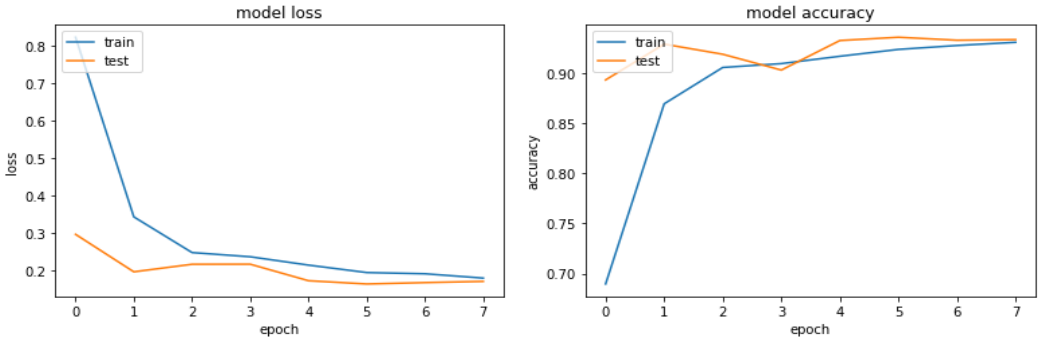


Figure III. 26: Précision et erreur du modèle à 4 couches (8 époques)

D'après la figure III.25 ci-dessus du modèle à 4 couches avec binarisation et avec marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 8, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 93% ce qui signifie la reconnaissance de 3720 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 7% c'est-à-dire 280 formes mal reconnus du fichier test.

```
Warning
Figures now render in the Plots pane by default. To make them also appear inline in the Console, uncheck "Mute Inline Plotting"
under the Plots pane options menu.
125/125 [==============================] - 23s 185ms/step - loss: 0.1701 - accuracy: 0.9327
Test loss: 0.17011964321136475
Test accuracy: 93.0 %
125/125 [==============================] - 21s 163ms/step
```

- **Nombre d'époques =12**

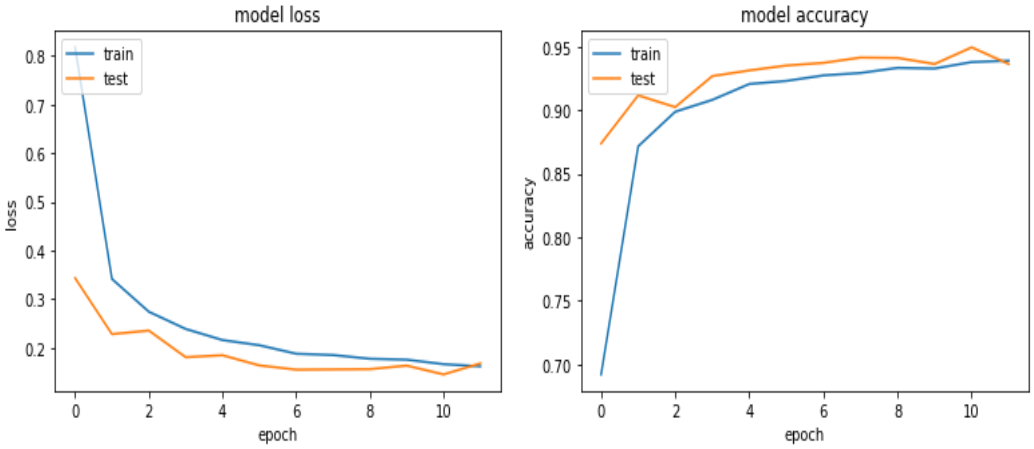


Figure III. 27 : Précision et erreur du modèle à 4 couches (12 époques)

D'après la figure III.26 ci-dessus du modèle à 4 couches avec binarisation et avec marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 12, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 94% ce qui signifie la reconnaissance de 3760 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 6% c'est-à-dire 240 formes mal reconnues du fichier test.

```
Warning
Figures now render in the Plots pane by default. To make them also appear inline in the Console, uncheck "Mute Inline Plotting"
under the Plots pane options menu.
125/125 [==============================] - 27s 219ms/step - loss: 0.1664 - accuracy: 0.9400
Test loss: 0.16635297238826752
Test accuracy: 94.0 %
125/125 [==============================] - 24s 144ms/step
```

III.9.2.3 Résultats obtenue pour le modèle à 6 couches

- **Nombre d'époques =4**

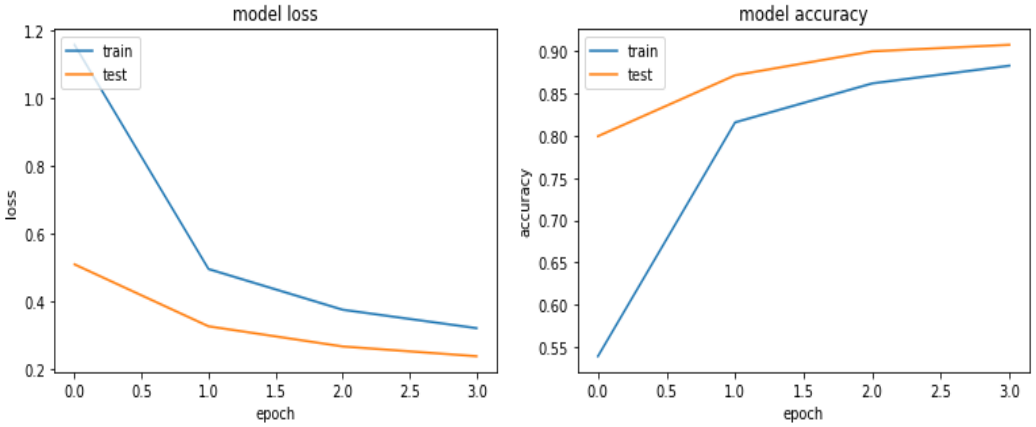


Figure III. 28 : Précision et erreur du modèle à 6 couches (4 époques)

D'après la figure III.27 ci-dessus du modèle à 6 couches avec binarisation et avec marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 4, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 89% ce qui signifie la reconnaissance de 3560 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 11% c'est-à-dire 440 formes mal reconnus du fichier test.

- **Nombre d'époques =8**

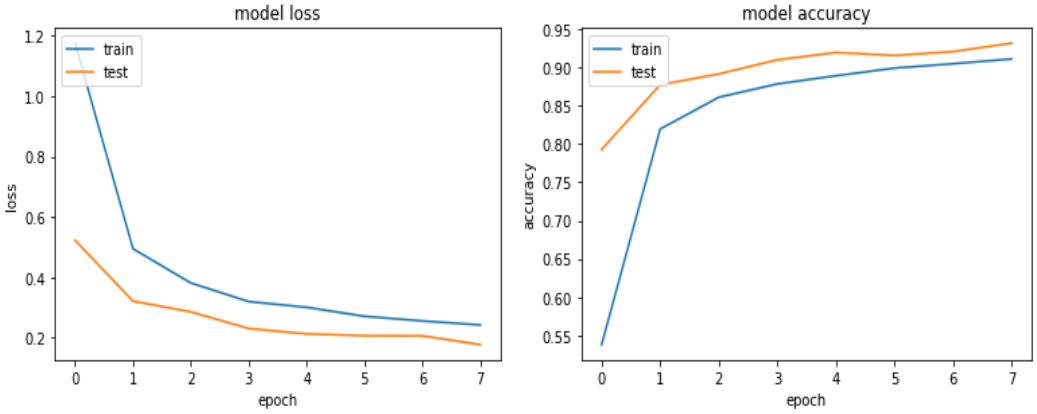


Figure III. 29 : Précision et erreur du modèle à 6 couches (8 époques)

D'après la figure III.28 ci-dessus du modèle à 6 couches avec binarisation et avec marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 8, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 93% ce qui signifie la reconnaissance de 3720 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 7% c'est-à-dire 280 formes mal reconnus du fichier test.

- **Nombre d'époques =12**

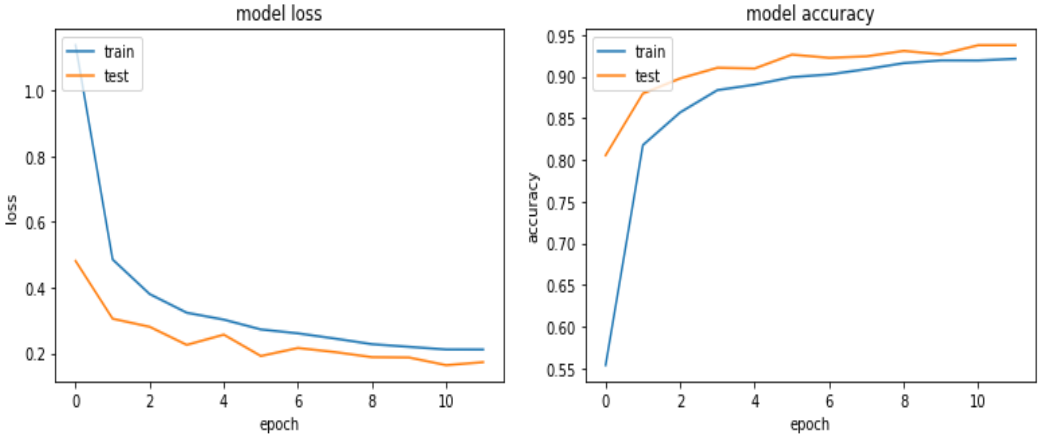


Figure III. 30: **Précision et erreur du modèle à 6 couches (12 époques)**

D'après la figure III.29 ci-dessus du modèle à 6 couches avec binarisation et avec marquage, la précision d'entrainement ou d'apprentissage (fichier train) et celle du test augmente avec l'augmentation du nombre d'époque qui est 12, Contrairement à l'erreur qui diminue avec l'augmentation du nombre d'époques.

On a obtenu un taux de précision durant notre apprentissage qui égale à 93% ce qui signifie la reconnaissance de 3720 formes parmi les 4000 existante dans notre base de données des formes géométrique, et un taux d'erreur de 7% c'est-à-dire 280 formes mal reconnus du fichier test.

III.10 Conclusion

Dans ce chapitre, nous avons élaboré une implémentation pour la reconnaissance des formes géométriques en utilisant une approche de classification basée sur le réseau de neurone convolutionnel combiner avec l'attribut HMB (Histogram of marquedbackground) ainsi que les différentes couches et fonctions d'activation.

Après l'exécution de notre algorithme nous avons obtenues ceci : pour l'algorithme du modèle sans binarisation et sans marquage on remarque bien que le meilleur modèle avec un meilleur pourcentage est bien le modèle de 4 couches (4 couches de convolution, 2 couches de pooling et enfin 3 couches de fully-connected) avec l'époque = 12 pour un taux de précision de 96% pour une exécution qui à durée 34min.

Par contre concernant l'algorithme du modèle avec binarisation et marquage on remarque bien que le meilleur modèle avec un meilleur pourcentage est toujours le modèle de 4 couches (4 couches de convolution, 2 couches de pooling et enfin 3 couches de fully-connected) avec l'époque = 12 mais pour un taux de précision de 94% pour une exécution qui à durée aussi 30min; donc on remarque bien que le temps d'exécution à déminué dans le cas de la binarisation et marquage ce qui est un avantage.

On remarque bien que l'attribut HMB fonctionne bien sur les lettres arabe et amazigh mais sa marche pas sur les formes qui sont complexes comparant au reste.

Enfin en conclue que plus en augmente le nombre d'époque plus on aura un pourcentage élevé et pour limiter le sur apprentissage ou ce qu'on appelle la surinterprétation on doit limiter le nombre de couches du réseau et de libérer les paramètres libres (connexions) du réseau. Ceci réduit directement la puissance et le potentiel prédictif du réseau. C'est équivalent à avoir une "norme zéro".

Conclusion générale

Dans ce projet de master en Automatique effectué au sein de l'université de Béjaia, nous avons rappelé les différentes techniques de reconnaissance deformes. On a pu constater que cette dernière a connu des progrès très importants durant ces dernières années, permettant désormais de faire face à la variabilité desmotifs existants.

Ce travail de recherche avait pour but de concevoir et implémenter un modèle de reconnaissance de formes par les réseaux de neurones convolutifs, et de discuté des notions fondamentales basés sur l'approche de deep Learning afin d'améliorer la précision. Pour cela nous avons proposé une nouvelle architecture avec 3 modèles différents (2couches, 4couches et 6couches de convolution), tous en introduisent les différents types de couches utilisées dans la classification : la couche convolutionnelle, la couche de rectification, la couche de pooling et la couche fullyconnected. Nous avons parlé aussi sur les méthodes de régularisation (dropout) utilisées pour éviter le problème de la surinterprétation, enfin ont fini par un apprentissage d'histogram of marqued background (HMB).

On a pu implémentés les trois modèles de réseaux neurones convolutifs avec différentes architecturespour la base de données des formes géométriquepour ensuite appliquer pour chacun de ces modèles des tests qui consistent à changer le nombre d'époques, et l'utilisation de l'instruction dropout pour surveiller et éviter la surinterprétation.

L'implémentation a été faite avec le langage de programmation python, on a utilisé des bibliothèques pour faciliter la tâche de création de nos modèles et pour l'accélération du training.

Enfin on a terminé avec un tableau récapitulatif et comparatif des résultats obtenus qui s'avèrent être très satisfaisants et encourageants, alors on a pu accomplir notre objectif avec un minimum d'erreur et un taux de précision très élevé.

Liste bibliographique

Bibliographie

[1] ‘’Mise au Point d’une Application de Reconnaissance de Formes’’, Mémoire de fin d’études pour l’obtention du diplôme de Master en Informatique, DJABEUR DJEZZAR Mohammed Rafik, BENKADA Feth-a.

[2] Theodoridis et Koutroumbas, S. Theodoridis, K. Koutroumbas, “Pattern Recognition, Second Edition”, Academic Press, Elsevier, 2003.

[3] https://www.interstices.info/jcms/c_5952/histoire-du-traitement-d-images

[4] https://interstices.info/jcms/p_88807/marvin-minsky-un-des-cerveaux-de-l-intelligenceartificiell

[5] Chung et al, K.W. Cheung, J.T. Kwok, M.H. Law, K.C. Tsui,"Mining customer product ratings for personalized marketing", Decision Support Systems, vol. 35, issue 2, pp. 231-243, 2003

[6] Kaastra et Boyd, I. Kaastra, M. Boyd, "Designing a neural network for forecasting financial and economic time series", Neurocomputing. Vol. 10, no. 3, pp. 215-236. 1996.

[7] Gupta et al, S.K. Gupta, W.C. Regli, D.S. Nau, "Manufacturing Feature Instances: Which Ones to Recognize?", Symposium on Solid Modeling and Applications, pp. 141152, 1995.

[8] Hippert et al, H.S. Hippert, C.E. Pedreira, R.C. Souza, "Neural networks for shortterm load forecasting: a review and evaluation", IEEE Transactions on Power Systems, vol. 16, Part 1, pp. 44-55, 2001.

[9] Munich et Perona, M.E. Munich, P. Perona, "Visual Input for Pen-Based Computers", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, issue 3, 2002.

[10]Yang et al, M.H. Yang, D.J. Kriegman, N. Ahuja, "Detecting Faces in Images: A 51 Survey", IEEE Transactions On Pattern Analysis And Machine Intelligence PAMI, vol.24; part 1, pp. 34-58, 2002.

[11]Theodoridis et Koutroumbas, S. Theodoridis, K. Koutroumbas, “Pattern Recognition, Second Edition”, Academic Press, Elsevier, 2003.

[12] ZACCONE Giancarlo, MD REZAUL Karim, MENSHAWY Ahmed : « Deep learning with tensorflow ». 2017

[13]https://atcold.github.io/pytorch-Deep-Learning/fr/week01/01-1/

[14] Livre capteurs intelligents, cours détaillé : « Conception de capteurs intelligents par Deeplearning ».

[15]https://www.racinely.com/post/les-fonctions-d-activation-part/

[16] K. Fukushima : « A neural network model for selective attention in visual pattern recognition». Biological Cybernetics, vol., 55, (1986) pp.5–15.

[17] K. Fukushima. Neocognitron : « A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position ». BiologicalCybernetics, vol. 36, (1980) pp. 193–202.

[18] TOUZET Claude : « Les réseaux de neurones artificiels, introduction à la connexion nisme » 1992.

[19] D. H. Hubel and T. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. Physiol, 160 :106–154, 1962

[20] « Classification des images avec les réseaux de neurones convolutionnels », Mémoire de fin d'études Pour l'obtention du diplôme de Masteren informatique, Mr Mokri Mohammed Zakaria.

[21]MALKI NAR IMENE, Classification automatique des textes par Les réseaux de neurones à convolution, thèse MASTER ACADEMIQUE, Domaine : Mathématique et Informatique, Filière : Informatique, Spécialité : Vision Artificielle, Faculté sciences exactes et sciences de la nature et de la vie, Département de mathématiques et d'informatique, Université LARBI ben M'HIDI O.E.B, 2018/2019.

[22] WANG Peng, XU Jiaming, XU Bo, LIU Chenglin, ZHANG Heng, WANG Fangyuan HAO Hongwei : « Semantic clustering and convolutional neural networks for short text categorization ». 2015.

[23] O'SHEA Keiron, NASH Ryan : « An introduction to convolutional neural networks » 2015.

[24]Houacine Noura, Khelifa Nadia, Classification des textures par les réseaux de neurones convolutif, thèse MASTER ACADEMIQUE, Domaine : Sciences et Technologies, Filière : Génie électrique, Spécialité : Commande des systèmes, FACULTE DU GENIE ELECTRIQUE ET D'INFORMATIQUE DEPARTEMENT D'AUTOMATIQUE, UNIVERSITE MOULOUD MAMMERI DE TIZI-OUZOU, 27/09/2018.

[25] GAGAOUA, Meriem, GHILAS, Hamza, TARI, Abdelkamel, *et al.* Histogram of marked background (HMB) feature extraction method for Arabic handwriting recognition. *International Journal of Image and Graphics*, 2022, vol. 22, no 02, p. 2250015.

[26] Lamtougui, H., et al. Offline Arabic Handwriting Recognition Using Deep Learning: Comparative Study. in 2020 International Conference on Intelligent Systems and Computer Vision (ISCV). 2020. IEEE.

[27] Altwaijry, N. and I. Al-Turaiki, Arabic handwriting recognition system using convolutional neural network. Neural Computing and Applications, 2020: p. 1-13.

[28] https://www.techniques-ingenieur.fr/base-documentaire/technologies-de-l-information-th9/programmation-42304210/langage-python-h3110/historique-h3110niv10001.html#:~:text=Python%20a%20été%20créé%20au,d%27un%20langage%20de%20script/

[29] https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445304-python-definition-et-utilisation-de-ce-langage-informatique/

[30] https://deepai.org/machine-learning-glossary-and-terms/keras/

[31] https://www.kaggle.com/datasets/benaddym/amazigh-handwritten-character-database-amhcd

[32] https://www.lemondeinformatique.fr/actualites/lire-tout-savoir-sur-le-framework-machine-learning-tensorflow-75776.html#:~:text=Fonctionnement%20de%20TensorFlow&text=Chaque%20nœud%20du%20graphique%20représente,programmeur%20via%20le%20langage%20Python.

[33] https://fr.acervolima.com/difference-entre-tensorflow-et-keras/#:~:text=TensorFlow%20est%20utilisé%20pour%20les,les%20petits%20ensembles%20de%20données/

Résumé

Dans notre travail, nous utilisons le deeplearning, plus précisément, les réseaux de neurones convoltionnel (CNN) pour la reconnaissance des formes.

Ce projet consiste à utiliser et créer un classificateur avec trois modèles de réseaux neurones convolutionnels différents et à évaluer le meilleur d'entre eux en modifiant des hyper paramètres tous on combinons notre apprentissage du deeplearning avec l'attribut HMB (histogram of marqued background).

Les résultats obtenus ont montré que le choix du nombre d'époque, la taille de la base de données ainsi que la profondeur du réseau ont une grande influence pour avoir de meilleurs résultats.

Mots clés : Reconnaissance de Formes, apprentissage profond, réseau de neurone convolutionnel (CNN)

Abstract

In our work, we use deep learning, more precisely, convolutional neural networks (CNN) for pattern recognition.

This project consists in using and creating a classifier with three different models of convolutional neural networks and to evaluate the best of them by modifying hyper parameters all we combine our deep learning with the attribute HMB (histogram of marked background).

The results obtained showed that the choice of the number of epochs, the size of the database as well as the depth of the network have a great influence to have better results.

Keywords: Shape Recognition, deep learning,convolutional neural networks (CNN)