République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Centre Universitaire Belhadj Bouchaib d'Aïn-Témouchent

Institut des Sciences
Département des Mathématiques et de l'Informatique

Mémoire

Pour l'obtention du Diplôme de Master en Informatique

Option : Réseaux et Ingénierie des Données (RID)

Présenté par :
Mr. DAHMANI Rayene

Reconnaissance de caractères manuscrits en utilisant les méthodes du Deep Learning.

Encadrant :
Dr. BENDIABDALLAH Mohammed Hakim
Maitre Assistant "B" à C.U.B.B.A.T.

Soutenu en 2018

Devant le jury composé de :

Président :	Mme. BELGRANA Fatima Zohra (M.C.B)	C.U.B.B.A.T.
Examinateur :	Mr. BOUAFIA ZOUHEYR (M.A.B)	C.U.B.B.A.T.
Encadrant :	Mr. BENDIABDALLAH Mohammed Hakim (M.A.B)	C.U.B.B.A.T.

Remerciement

Remerciement

Je remercie notre créateur ALLAH le tout-puissant, de m'avoir donné les forces, la santé, la volonté et le courage afin d'entamer et de terminer ce modeste travail.

Je tien à saisir cette occasion et adresser mes profonds remerciements et mes profondes reconnaissances à Mr BENDIABDALLAH pour la qualité de son encadrement exceptionnel, pour sa patience, sa rigueur et sa disponibilité durant notre préparation de ce mémoire. ce travail ne serait pas aussi riche et n'aurait pas pu voir le jour sans l'aide de son encadrement.

Mme BELGRANA, En étant conscient de l'honneur que vous m'avez fait en acceptant d'être la présidente du jury je vous remercie Veuillez accepter ce travail Madame, en gage de notre grand respect et notre profonde reconnaissance.

Je souhaite exprimer ma gratitude à Mr BOUAFIA pour l'honneur que vous me faite en examinant et en évaluant ce mémoire.

Mes remerciement s'adresse également à tous nos professeurs pour leurs générosités et la grande patience dont ils ont sues faire preuve malgré leurs charges académiques et professionnelles.

DAHMANI

Dédicaces

Dédicaces

Tous les mots ne sauraient exprimer la gratitude, l'amour, le respect, la reconnaissance, c'est tous simplement que :

Je dédie ce travail à :

À Ma tendre Mère et à mon cher père : vous qui porté envers moi un amour sans failles, vous qui surgissiez telle une oasis dans le désert, dir que je vous ai causé tant de souci, rien au monde ne vaut les efforts que vous avez fournis jour et nuit pour mon éducation et mon bien-être. Ce travail et le fruit des sacrifices que vous avez consentis pour mon éducation et ma formation le long de ces années.

À ma sœur et à mon frère : vous mes ainés qui sont toujours là pour moi, dans le malheur comme dans le bonheur, vous qui m'avez toujours soutenu aimer et chéri comme un petit frère, je vous porterais à jamais dans mon cœur.

À toute ma famille pour leur soutien tout au long de mon parcours universitaire.

À tous mes collègues et en particulier à mes deux acolytes mokhtari et moulkhaloua.

Mr. DAHMANI Rayene

Table des matières

Table des figures

Liste des tableaux

Introduction générale

Introduction générale

L'informatique s'est tellement développé ces derniers temps que le monde dans lequel nous vivons est devenu numérique ou la manipulation des informations stockées traitées indexés et recherché, par des systèmes informatiques ce qui rend leur récupération une tâche rapide et pas chère.

Des progrès considérables ont été réalisé dans le domaine de l'intelligence artificielle qu'il est même devenue le sujet le plus en vue du monde informatique, Il est même difficile d'y échapper, cette dernière est devenu la nouvelle tendance non seulement dans le monde scientifique mais aussi dans la vie quotidienne de la majorité des êtres humains.

Revenant un peu en arrière avec un petit aperçu sur le très vaste domaine d'intelligence artificielle qui nous amène à parler de l'évolution de l'informatique au fil du temps, l'IA englobe divers sous-domaines. tel que l'apprentissage profond, plus fréquemment désigné par l'expression Deep Learning qui est une branche du machine learning ou apprentissage automatique, après avoir passé beaucoup de temps dans l'oubli l'apprentissage profond a refait sa réapparition grâce à l'amélioration de la puissance de calcul des ordinateurs, de l'apparition de nouvelles bases de données plus grandes et plus riches.

L'objectif de notre travail consiste à développer un système de reconnaissance des caractères manuscrits plus précisément les chiffres en utilisant les techniques de l'apprentissage profond .

Notre mémoire se subdivise donc comme suit :

Dans le premier chapitre, nous avons invoqué quelques notions sur l'imagerie et des définitions sur l'intelligence artificielle ainsi que ses types, mais nous nous sommes surtout intéressé sur l'apprentissage automatique de ces types jusqu' aux algorithmes utilisés pour l'apprentissage supervisé et non supervisé.

Le deuxième chapitre a été consacré à l'apprentissage profond en détaillant plusieurs de ces algorithmes de classification tout en se basant sur les réseaux de neurones à convolution.

Dans le troisième et dernier chapitre nous parlerons sur l'approche proposée avec deux différents langages de programmation tout en exposant les résultats obtenus.

Chapitre 1

l'intelligence artificielle

1.1 Introduction

Dans ce chapitre, nous nous intéressons aux algorithmes de classification automatique des images .cette classification est obtenue selon deux types d'apprentissage : l'apprentissage supervisé ou chaque image est étiqueté et l'apprentissage non supervisé où les images ne possèdent pas d'étiquette. Nous étudions notamment les notions de base sur l'image.

1.2 Notion de base

Nous détaillerons dans ce qui suit quelque notion du terme image.

1.2.1 Représentation des images numériques

Une image numérique est une image dont la surface est divisée en éléments de tailles fixes appelées cellules ou pixels, donc nous pourrons appeler cette image une matrice de pixel. S'il s'agit d'une image en niveau de gris, chaque pixel est codé par 1 composante comprise entre 0 et 255 et qui représentent la luminosité du pixel, par contre si c'est une image en couleur alors chaque pixel ne sera plus codé par 1 composante mais 3 dus aux 3 couleurs fondamentales (RVB), chaque composante est comprise entre 0 et 255 et qui représentant respectivement les "doses" de rouge, vert et bleu qui caractérisent la couleur du pixel.

1.2.2 Caractéristiques d'une image numérique[Boukhlouf 2005]

L'image est un ensemble structuré d'informations caractérisé par les paramètres suivants :

1. **Pixel :**
   Le pixel (abréviation venant de l'anglais : Pictures element) est l'élément de base d'une image ou d'un écran, c'est-à-dire un point.
   L'ensemble de ces pixels est contenu dans un tableau à deux dimensions (largeur et hauteur) constituant l'image.

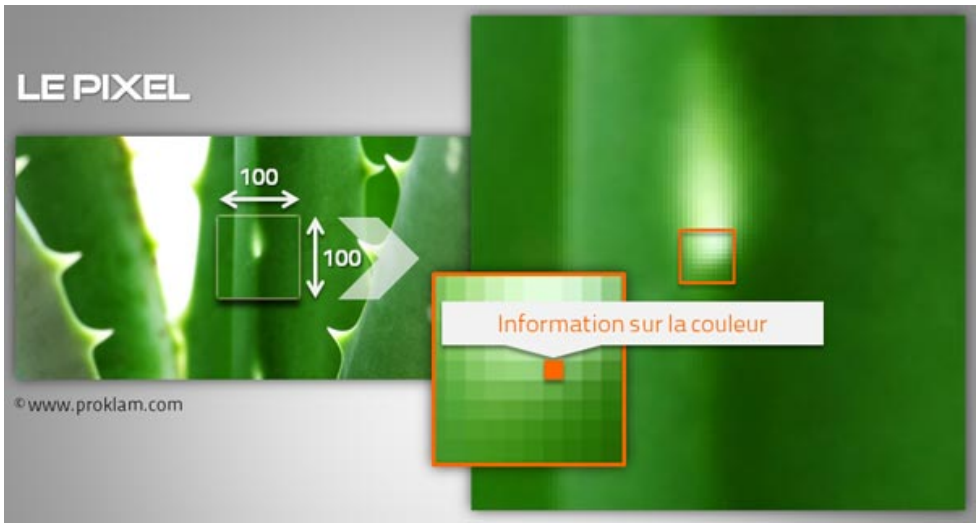


FIGURE 1.1 – Pixel [Laperrière 2012]

2. **Dimension :**
   Correspond à la taille de l'image, si on multiplie le nombre de colonnes de la matrice qui

représente l'image par le nombre de lignes le résultat sera le nombre total de pixel dans cette image.

3. **Résolution :**
   C'est la clarté ou la finesse de détails atteinte par un moniteur ou une imprimante dans la production d'images. La résolution est exprimée en nombre de pixels par unité de mesure (pouce ou centimètre). On utilise aussi le mot résolution pour désigner le nombre total de pixels affichables horizontalement ou verticalement sur un moniteur, plus grand est ce nombre, meilleure est la résolution.
4. **Bruit :**
   Le bruit d'image est la présence d'informations parasites dans cette dernière, il provient de l'éclairage des dispositifs optiques et électroniques du capteur.
5. **Luminance :**
   C'est le degré de luminosité des points de l'image. Elle est définie aussi comme étant le quotient de l'intensité lumineuse d'une surface par l'aire apparente de cette surface, pour un observateur lointain, le mot luminance est substitué au mot brillance, qui correspond à l'éclat d'un objet.
6. **Contraste :**
   C'est l'opposition marquée entre deux régions d'une image, plus précisément entre les régions sombres et les régions claires de cette image. Le contraste est défini en fonction des luminances de deux zones d'images. Si L1 et L2 sont les degrés de luminosité respectivement de deux zones voisines A1 et A2 d'une image, le contraste C est défini par le rapport.

1.3 Définition de l'intelligence artificielle

L'usage de l'expression intelligence artificielle noté IA est apparus dans les années 1950 et ceux suite aux progrès technique de l'informatique et de sa propagation dans l'activité humaine et depuis ce temps l'IA est devenue en quelque sorte une partie de notre culture. [Boisard]
Bien qu'il n'y ait pas de définition générale il existe deux grands axes à suivre, le premier concerne les processus de pensée et du raisonnement, tandis que le second se repose sur le comportement, de ce fait plusieurs définitions ont vu le jour :

1. **Des systèmes qui pensent comme les humains :**
   « La tentative nouvelle et passionnante d'amener les ordinateurs à penser ... d'en faire des machine dotées d'un esprit au sens plus le plus littéral » [Haugeland 1985]
   « L'automatisation d'activités que nous associons a la pensée humaines, des activité telle que la prise de décisions, la résolution de problème, l'apprentissage.. »[Bellman 1978]
2. **Des systèmes qui pensent rationnellement :**
   « L'étude de facultés mentales grâce des modèles informatique »[Charniak et al 1985]
   « L'étude de moyens informatique qui rendent possible la perception le raisonnement et l'action » [Winston 1992]
3. **Des systèmes qui agissent comme des humains :**
   « L'art de créer des machines capables de prendre en charge des fonctions exigeant de l'intelligence quand elle sont réalisés par des gens » [Kurzweil 1990]
   « L'étude de moyens à mettre en œuvre pour faire en sorte que des ordinateurs accomplissent des choses pour lesquelles il est préférable de recourir à des personnes pour le moment » [Rich et al 1991]
4. **Des systèmes qui agissent rationnellement :**
   L'intelligence artificielle (computational intelligence) est l'étude de la conception d'agents

intelligents [pool et al 1998]
« L'IA ..... Étudie le comportements intelligent dans des artefacts » [Nillson 1998]

1.3.1 Type d'intelligence artificielle

Les scientifiques ne sont pas sur les mêmes longueurs d'onde concernant la notion d'intelligence artificielle ils ont même redéfini les deux grands en IA faible et IA forte :

— **Intelligence artificielle faible :**
C'est des systèmes autonomes qui simulent l'intelligence mais qui ne sont pas du tout intelligents.
— **Intelligence artificielle forte :**

On parle de machine intelligente qui pense en d'autres termes des machines qui éprouve une sensation de conscience, qui seront capables de raisonner de prendre des décisions et de juger.

L'apprentissage est un aspect fondamental de l'IA, c'est grâce à lui qu'un système intelligent améliore ces performances avec de l'expérience, contrairement à l'approche manuelle ou on a connu quelques limites puisqu'on écrivait des programmes à la main pour la reconnaissance de caractères imprimés, jouez aux échecs etc. [Nicolle 1996]

1.4 Apprentissage automatique :

L'apprentissage automatique notée (AA ou ML pour machine learning) est un sous-ensemble de l'intelligence artificielle (IA) de ce point on peut dire que tout ML est IA mais non pas le contraire, la machine Learning ou apprentissage automatique nous permet d'aborder des problèmes qui par le passé étaient trop complexes pour les humains c'est-à-dire qu'au lieu d'expliquer à un ordinateur comment gérer et résoudre un problème, le ML permet à l'ordinateur d'apprendre à le résoudre par lui-même.[Chickh et al 2016]

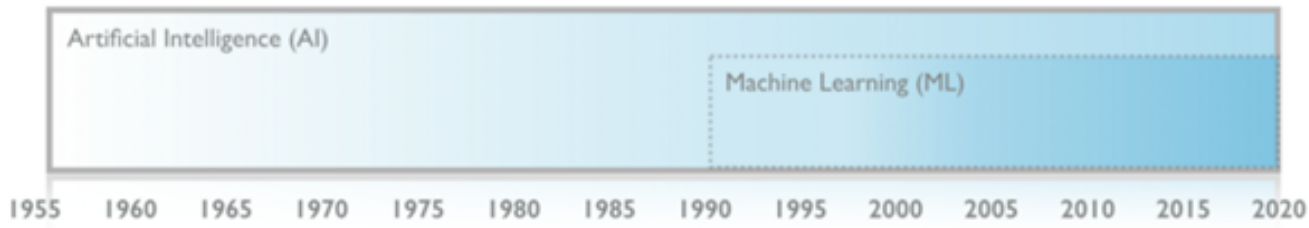


FIGURE 1.2 – L'évolution de l'intelligence artificielle [Ludovic 2016]

1.4.1 Domaines de l'apprentissage automatique

Les fouilles de données et les domaines de l'intelligence artificielle sont les domaines d'application majeur du machine Learning (apprentissage automatique).

1. **La fouille de données(data mining) :** est un ensemble d'algorithmes et de méthode qui vont explorer et analyser d'énormes quantités de donnés dans le but de rechercher

et d'extraire des règles, des associations des informations encore inconnues qui nous seront utiles pour prédire le comportement futur de ces donnes ainsi prendre des bonnes décisions.[Tafffar 2012],[Atif 2014]

2. **L'intelligence artificielle :** vu que l'apprentissage automatique est un sous ensemble de L'IA, plusieurs domaines de cette dernière seront ainsi des domaines d'application de l'AA comme la vision artificielle, la robotique, l'analyse et la compréhension des images, la reconnaissance de formes, la reconnaissance de caractères.[Haton 1989],[Tafffar 2012]

1.5 Classification et apprentissage :

La classification est la création d'une règle de décision pour regrouper des ensembles d'exemples en classes. [Chikh et al 2016]

Les algorithmes intelligents passent par une phase d'apprentissage (Learning) appelle aussi entrainement en leur donnant des exemples avec leur résultat .Plus on leur donne de données pour s'entrainer plus ils sont efficaces, le but de ces algorithmes est de faire des prédictions de comportements sur des données nouvelles dont on ne connaît pas les résultats. [Naffakhi et al 2005]

Généralement Les données d'apprentissage sont réparties en 2 catégories :

— L'ensemble d'apprentissage : c'est l'ensemble des exemples utilisé pour produire le modèle d'apprentissage.
— l'ensemble de Test ce compose des exemples sur lesquels s'exécuteras le modèle d'apprentissage pour vérifier le bon fonctionnement de l'algorithme.

1.6 Types d'apprentissage

Il existe deux types d'apprentissage : **supervisé** et **non supervisé**

1.6.1 Apprentissage supervisé

Le principe de cette méthode c'est d'avoir une base d'apprentissage qui contient un ensemble de données étiqueté ou bien des exemples à qui en leur attribué des classes par un expert (superviseur), le but des méthodes d'apprentissage supervisé est de produire automatiquement des règles ou des fonctions de classement à partir de la base d'apprentissage, ces règles vont permettre à classifier un objet. [Naffakhi et al 2005]

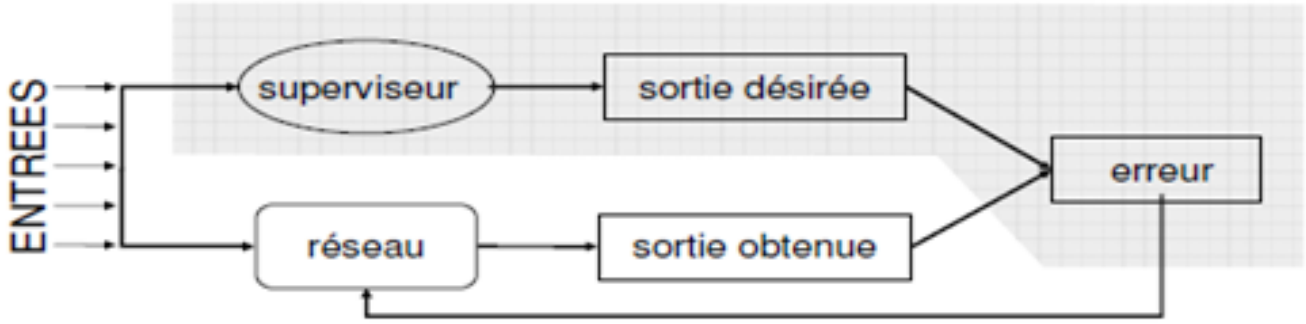


FIGURE 1.3 – Apprentissage supervisé[Benmammar et al 2009]

1. **Arbre de décisions :**

Etant donné n attributs {A1....An} e l'espace de description X et le produit cartésien du domaine Xi de chaque attribut Ai :

$$X = \lceil \rceil XiouXi = domaines(Ai) \tag{1.1}$$

Les attributs peuvent êtres : binaires, n-aires, réel.
Les arbres de décision sont des règles de classification qui basent leurs décisions sur une suite de tests associe aux attributs, le test étant organisé de manière arborescente.

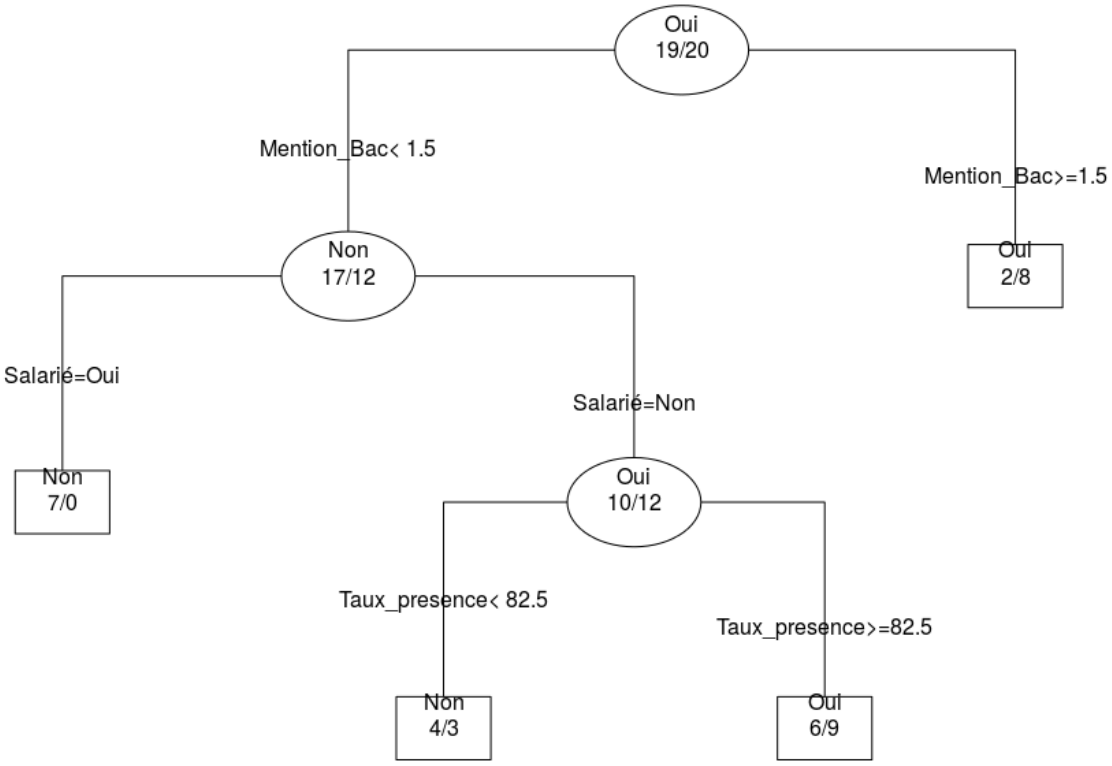


FIGURE 1.4 – Arbre de décisions[Santos 2015].

Dans un arbre de décision les nœuds internes sont notés nœuds de décisions qui sont étiqueté par des tests applicables à toute description d'un individu de la population, généralement chaque nœud interne est égal à un test sur un unique attribut, les arcs issus d'un nœud interne sont les réponses possibles au test du nœud, les feuilles de l'arbre sont étiqueté par une classe.

On associe à chaque description complète une et une seule feuille de l'arbre de décision, on d'autre terme on associe une procédure de classification a tout arbre de décision.

Toute association est définie en commençant par la racine de l'arbre et nous parcourons ce dernier depuis la racine vers le bas selon les réponses aux tests qui étiquettent les nœuds internes.

Des décisions facilement interprétables, et une classification très rapide sont les deux qualités de l'arbre de décision. [Capponie]

(a) **Avantages[Rakotomalala 2005]**

— La simplicité de compréhension et d'interprétation.
— Peu de traitements sur les données
— Contrairement aux autres techniques qui sont souvent spécialisé sur un seul type de variable qui est soit numérique ou catégorique les arbres de décision peuvent gérer les deux types de valeurs à la fois
— Une classification très rapide

(b) **Inconvénients[Rakotomalala 2005]**

En revanche, elle présente certains inconvénients :

— On peut se trouver avec des arbres de décision finale très complexe
— Certains concepts sont difficiles à exprimer à l'aide d'arbres de décision (comme XOR ou la parité).

2. **Méthode k-plus proche voisins :**

Le classifieur des k plus proche voisin notés k-ppv (k-Nearest Neighbor ou k-NN), est l'un des classifieurs les plus simples, il prend un point de test sous forme vectorielle et trouve la distance euclidienne entre celle-ci et la représentation vectorielle de chaque d'exemple d'apprentissage.[Nemouchi, et al 2010]

L'exemple d'entrainement le plus poche du point de test est appelé son voisin le plus proche, comme cet exemple et en quelque sorte le plus proche de notre point de test, il

est logique d'attribuer son label de classe au point de test, cela exploite l'hypothèse de « régularité » selon laquelle les points proches les uns des autres sont susceptibles d'avoir la même classe. [Burrow et al 2004]

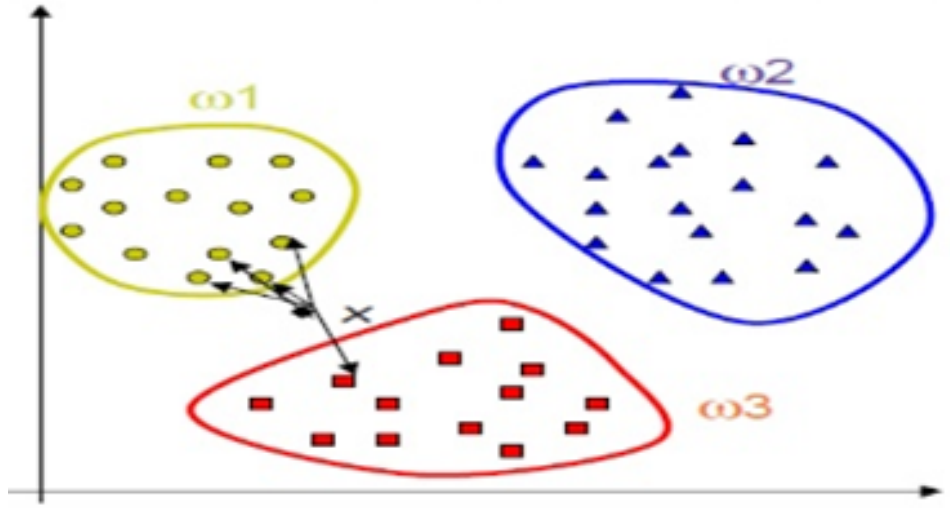


FIGURE 1.5 – illustration méthode Kppv[Rifqi 2002].

(a) **Avantages** [Wassim 2012]

— Fournit de bonne performance
— Apprentissage rapide
— Une facilité dans la conception et la compréhension de la méthode

(b) **Inconvénients** [Rakotomalala 2001]

— Faible vitesse de classification due au nombre important de distances a calculé
— Nécessite de garder la base de donné sous la main ce qui induit à la lenteur de la prédiction méthode gourmande en place mémoire
— Sensibilité de la dimensionnalité et au attribut non pertinent

3. **SVM (séparateur a vaste marge)**[Mareef 2013]

Cette technique est une méthode de classification à deux classes qui tente de séparer linéairement les exemples positifs des exemples négatifs dans l'ensemble des exemples.

En voulant que la marge entre le plus proche des positifs et des négatifs soit maximale, le séparateur d'exemples positif et négatif appelé l'hyperplan est cherché par cette méthode.

En tenant compte que de nouveaux exemples ne seront forcément pas similaires à ceux utilisés pour trouver l'hyperplan mais seront surement situés soit du côté négative ou positive ce qui garantirait une généralisation du principe.

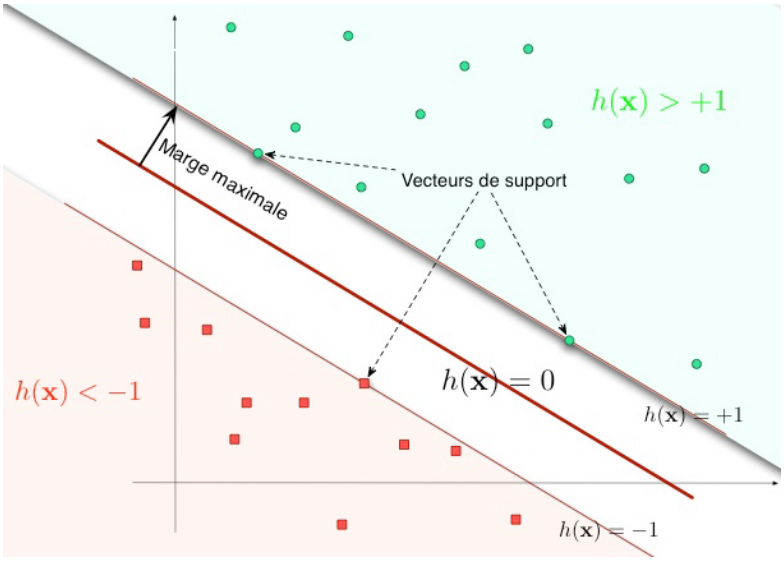


FIGURE 1.6 – illustration SVM cas lineares[CORNUÉJOLS].

Les exemples utilisé lors de la recherche de l'hyperlan ne seront plus utile et pour classer un nouveaux cas seul les vecteurs de support qui se représentent comme les vecteurs discriminant grâce auquel l'hyperlan est déterminé seront utilisé .

(a) **Avantages : [Cherif 2011]**

— les exemples de test ne sont pas comparés avec tous les exemples d'apprentissage mais juste avec juste avec les supports vecteurs.
— Traitement des problèmes non linéaires avec le choix des noyaux.
— la capacité a traité des données de grandes dimensions (variable élève).
— Plus performant que la plupart des autres méthodes.

(b) **Inconvénients :[Rakotomalala 2013]**

— Difficulté à identifier les bonnes valeurs des paramètres (et sensibilité aux paramètres.
— Utilisation des points de support pour les noyaux non linéaire.
— Les classe bruites posent des problèmes.
— Un calcul matricielle important dus aux grandes quantités. d'exemple en entrée.

4. **Réseaux de neurones artificielles**

Inspiré des réseaux de neurones du cerveau humain, un neurone effectue la somme pondérée de ses entrées (valeurs d'entrée binaire Xi provenant d'autres neurones et la valeur de leurs synapses respectives Wi) à laquelle il ajoute une valeur appelée biais, la sortie binaire Yi dépend d'une fonction d'activation, selon que la somme précédente dépasse ou non un seuil j,On peut formaliser ce comportement comme suit : [Megar et al 2016][Vincent 2017]

$$y_j = \mathrm{h}\left(-\vartheta_j + \sum_{i=0}^{N-1} w_i x_i\right)$$

Ou h est la fonction d'activation.
Quelque fonctions d'activations.

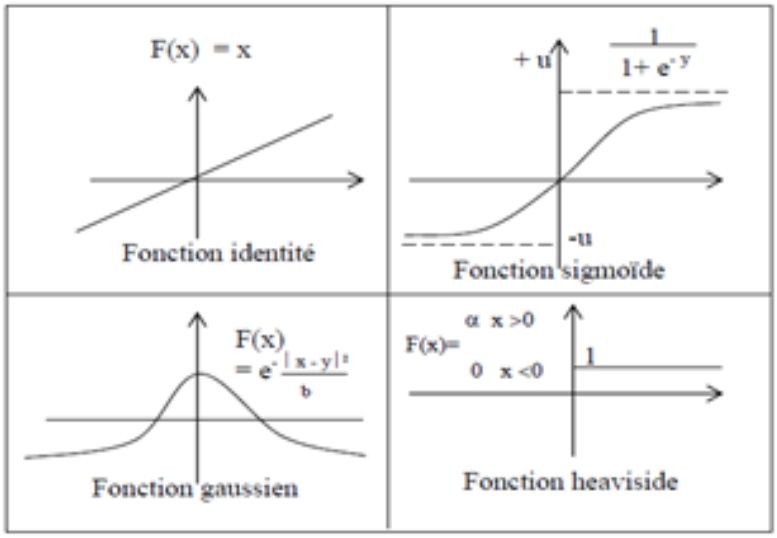


FIGURE 1.7 – fonctions d'activations.[Vincent 2017]

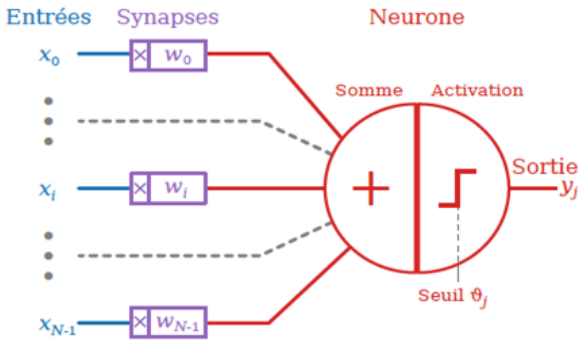


FIGURE 1.8 – illustration d'un neurone.[Vincent 2017]

(a) **Perceptron :**

Le premier modèle des réseaux de neurones qui a vu le jour est le perceptron son architecture est assez simple, il se compose de deux couches de neurones, la couche d'entrée et la couche de sortie. Ces modèles de réseaux de neurones sont capables d'apprendre à classifier des problèmes linéairement séparables, la sortie du neurone i est de la forme :

$$y_i = f\left(\sum_{j=1}^{N} x_i w_{ij}\right)$$

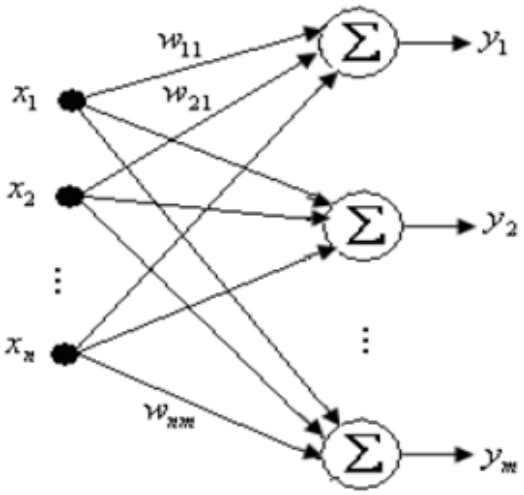


FIGURE 1.9 – Le perceptron.[MEGAR et al 2016]

(b) **Perceptron multicouches :**

Le perceptron a montré ses limites lorsqu'il s'agissait d'apprendre à classifier des problèmes non linéairement séparables c'est pourquoi le perceptron multicouche a été inventé, il est le fruit d'association de plusieurs couches successives Le perceptron multicouche se compose d'une couche d'entrée, d'une ou plusieurs couches intermédiaires appelées aussi couches cachées, et d'une couche de sortie.

Avec le perceptron multicouche plusieurs fonctions d'activation sont apparus avec, ainsi que le recours à des entrées et sorties analogiques.

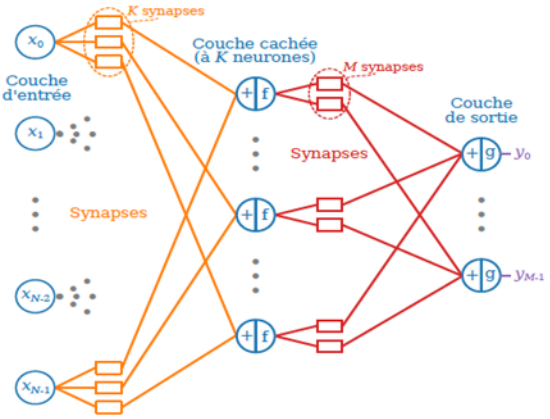


FIGURE 1.10 – Exemple d'un réseau de neurones artificiels à N entrées et M sorties.[Vincent 2017]

Durant la phase d'entrainement (apprentissage) le réseau ajuste les valeurs des poids et des biais afin de minimiser le critère d'erreur entre la sortie observée et la sortie désirée. Après cette phase d'apprentissage le réseau sera capable de classifier

de nouveaux exemples.[Parizeau 2004]

i. **Retro propagation du gradient** [Parizeau 2004]

L'objectif de l'algorithme de rétro-propagation de l'erreur est de minimiser l'erreur totale E de t qui est définie comme la somme des erreurs partielles obtenues pour chaque exemple du vecteur d'entrée.

$$E_r = \sum_p E_p \tag{1.2}$$

Avec E de p l'erreur de la pme entrée de l'ensemble d'apprentissage qui est défini sous la forme

$$E_p = \frac{1}{2}\sum_{j=1}^{M}(d_pj - z_pj)^2 \tag{1.3}$$

Où M représente le nombre de sortie du réseau, d de pj la sortie désirée et z de pj la sortie calculée.

Par conséquent, pour que la fonction coût E de p soit petite il est raisonnable de modifier ou faire varier les paramètres dans le sens négatif du gradient de E de p. Le principe de l'entraînement est de présenter le vecteur d'entrée de chaque exemple d'apprentissage et de calculer la réponse du réseau après l'avoir initialiser au hasard (poids). On calcule le gradient de l'erreur par rapport à tous les poids, puis les paramètres seront ajustés dans la direction opposée à celle du gradient de l'erreur. Cette opération est cyclique jusqu'à ce que l'erreur tant vers zéro.

ii. **Convergence et adaptation des poids[Parizeau 2004]**

La mise à jour des poids synaptiques w de ij, est faite par la minimisation de l'erreur E de p. Le développement de l'expression de la dérivée partielle de E de p par rapport à w de ij nous donne :

— **Pour les neurones de sortie :**

$$\frac{\delta E_p}{\delta w_{ij}^l} = (d_Pj - z_Pj)f^l(Y_{Pj}^l)X_{Pi}^l \tag{1.4}$$

— **Pour les neurones de couche caché**

$$\frac{\delta E_p}{\delta W_{ij}^l} = [\sum_K \frac{\delta E_p}{\delta Z_{PK}^{l+1}} f^(l+1)(Y_{PK}^{l+1})W_{kj}^{l+1}]f^l(Y_{Pj}^l)X_{Pj}^l \tag{1.5}$$

La formule itérative pour mesurer les paramètres d'un réseau est la suivante :

$$W_{ij}^{l}(t+1) = W_{ij}^{l}(t) + \Delta p W_{ij}^{l}(t+1) Avec : \Delta p W_{ij}^{l}(t+1) = -\mu \frac{\delta E_p}{\delta W_{ij}^{l}(t)} \quad (1.6)$$

mu : est facteur d'apprentissage ; leurs valeurs entre 0 et 1. La formule (2.22) s'écrit encore sous la forme :

$$W_{ij}^{l}(t+1) = W_{ij}^{l}(t) + \mu \delta_{Pj}^{l} X_{Pi}^{l} \quad (1.7)$$

Où delta de Pil est l'erreur équivalent à la sortie de neurone j de la couche l telle que :

$$\delta_{Pi}^{l} = -\frac{\partial E_p}{\partial Y_{Pj}^{l}} \quad (1.8)$$

Pour minimiser l'erreur totale E de t, on remplace l'équation (2.23) par :

$$\Delta W_{ij}^{l}(t+1) = -\mu \frac{\partial E_T}{\partial W_{ij}^{l}(t)} \quad (1.9)$$

La vitesse de convergence dépend de mu, si mu est trop petit la convergence de l'algorithme demande un large nombre d'itération, sinon, soit la convergence est rapide, soit il ne converge pas à cause de la présence d'un phénomène d'oscillation qui intervient dès qu'on se rapproche de minimum. On ajoute un moment alpha inclue [0,1] à l'algorithme de rétro-propagation pour éviter leur problème d'être très lente à la convergence, et la loi d'adaptation devient :

$$\Delta W_{ij}^{l}(t+1) = \Delta W_{ij}^{l}(t) + \mu \delta_{Pj}^{l} X_P + \alpha (W_{ij}^{l}(t) - W_{ij}^{l}(t-1)) P \quad (1.10)$$

iii. **Algorithme de retro-propagation :[Parizeau 2004]**

Les étapes de l'algorithme de retro propagation sont résumées dans ce qui suit :

A. initialiser les poids avec des petites valeurs aléatoires (wij entre 0 et 1)
B. Présenter le vecteur d'entrée Xp de l'exemple p avec la sortie désirée Dp
C. Calculer le vecteur de sortie actuelle Ep et l'erreur correspondante Ep en utilisant les équations (2.18) et (2.19)
D. calculer les dérivées partielles de l'erreur par rapport à chaque poids à l'aide des équations suivantes :
E. si j est le neurone de la couche de sortie :

$$\delta_{Pj}^{l} = f^{l}(Y_{Pj}^{l})(d_P j - z_P j) \quad (1.11)$$

F. si j est le neurone de la couche caché :

$$\delta_{Pj}^{l} = f^{l}(Y_{Pj}^{l}) \sum_{k=l}^{N_L+1} \delta_{pk}^{l+1} W_{kj}^{l+1} \quad (1.12)$$

G. Ajuster les poids wij en utilisant l'équation (2.24) ou l'équation (2.27)
H. Répéter les étapes 2 à 6 pour l'ensemble des observations de la base d'apprentissage tant que le critère d'arrêt n'a pas été atteint. [Megar et al 2016]

iv. **Critère d'arrêt [Bendjeddou 2007]**

Il y a plusieurs méthodes soit :
— on fixe le nombre d'itérations
— on fixe l'erreur minimale
— si les poids ne modifient pas où l'erreur reste constante

Avantage :[Djeffal 2015]

— Architecture simple
— Accepte les données bruitées et la classification non-linéaire
— Un classifieur très précis
— Une forte résistance aux pannes (si un neurone tombe en panne il n'affectera pas le fonctionnement du réseau)

Inconvénient :[Cherif 2011]

— N'accepte pas la classification non linéaire (perceptron)
— Une difficulté pour paramétrer (type, nombre de nœuds, organisation, connexions,...etc.) les réseaux.

1.6.2 Apprentissage non supervisé

Appelé aussi apprentissage à partir d'observations, il s'agit d'une construction automatique des classes sans avoir recours a l'expert c'est à dire qu'il n'aurait pas de phase d'entrainement, on dispose dans ce cas d'une masse de données importante non étiqueté (sans indiquer leurs classes) qui sont introduites par l'utilisateur donc l'algorithme doit découvrir par lui-même la structure des donnés.[Chikh et al 2016][Naffakhi et al 2005]

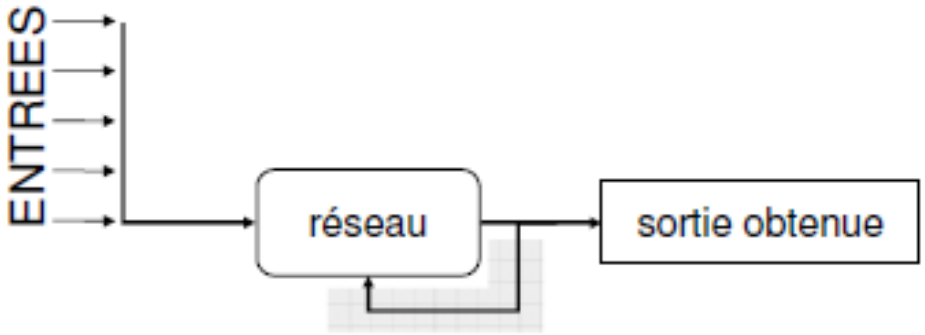


FIGURE 1.11 – Apprentissage non supervisé[Benmammar et al 2009]

1. **Clustering hiérarchique :**

   On commence par calculer un tableaux de distance ou de dissemblance entre les individus qu'on veut classer, tout en sachant que chaque individu constitue une classe (singleton) a chaque étape on cherche à constituer des classes en agrégeant les deux éléments les plus proches de l'étape précédente, on s'arrête lors de l'obtention d'une seule classe, les résultats seront affiché sous la forme d'un arbre binaire ou dendrogramme.

   On commence par calculer un tableaux de distance ou de dissemblances entre les individus qu' on veut classer, il est fondamentale de mettre a jour ce tableaux a chaque étape de l'algorithme, où on cherche aussi a constituer des classes en agrégeant les deux éléments les plus proches qui peuvent être deux individus, deux classes ou un individu avec une classe, les distances entre ce nouvel objet et les autres sont calculer et on remplace dans la matrices les distances des objets qui viennent d'être agréger , l'algorithme s'arrête lors de l'obtention d'une seule classe, les résultats seront affiché sous la forme d'un arbre binaire ou dendrogramme.

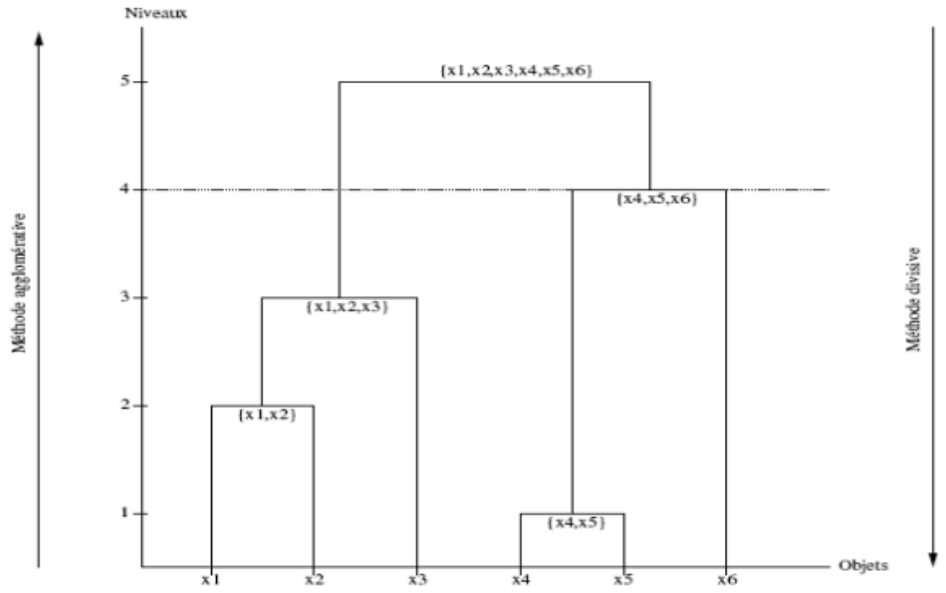


FIGURE 1.12 – Exemple dendrogramme [Khatir 2012]

(a) **Critère d'agrégation :[Chesneau 2016][khatir 2012]**

Soit A et B deux classe on individus avec les pondérations Wa et Wb et la distance entre deux individus i et j noté dij.

— **Cas d'une dissemblance :**

Les critères suivant s'accommodent d'un simple indice de dissemblance défini entre les individus.

(a) **Critère du saut minimal : (single link)**

Pour calculer la distance entre deux cluster, on prend le minimum des distances entre toutes les paires d'objets appartenant a des clusters différents.

$$d(A,B) = min(d_ij)ieAjeB \tag{1.13}$$

(b) **le critère du saut maximale (completelink)**

pour ce critère on prend le maximum des distance *

$$d(A,B) = min(d_ij)ieAjeB \tag{1.14}$$

(c) **Le critère de la moyenne (average link)**

on calcule la distance moyenne entre tous les éléments de c1 et tous les éléments de c2.

$$d(A,B) = \frac{1}{card(A)card(B)} \sum_{I \in A, i \in B} dd_ij \tag{1.15}$$

— **Cas d'une distance euclidienne :**

(a) **Le critère de ward :**

A chaque étape on choisi le regroupement de classe de tel sorte que l'augmentation de l'inertie intra classe soit minimale.

$$d(A,B) = \frac{w_A + w_B}{w_A w_B} d(g_A - g_B) \tag{1.16}$$

(b) **Le critère de centre de gravité :**

La distance entre les deux classes A et B est défini par la distance entre leur centre de gravité.

$$d(A,B) = d(g_A, g_B) \tag{1.17}$$

Avantage :

— Applicable sur différents types d'attribut.
— La lecture de l'arbre permet de déterminer le nombre optimal de classes.

Inconvénients :

— Face a de large base de donnés l'utilisation de cette méthode est compromis.
— Coûteux en temps de calcul.

2. **Le clustering par partitionnement :**

Plutôt que de proposer en sortie une structure organisationnelle du type dendrogramme telle que l'approche hiérarchique vus dans la section précédente, les algorithmes de partitionnement construisent directement en sortie une partition unique des objets a classer, c'est-à-dire que chaque classe doit contenir au moins un objet et que chaque objet ne doit appartenir qu'a une classe unique.[Khatir 2012]

De par cette définition d'une partition et le nombre de classe k requise ces algorithmes cherche a améliorer la partition initiale générer plutôt en réattribuant les objets d'une classe a une autre. Pour des raisons de complexité, il est impossible de générer toute les partitions du clustering, on cherche alors une bonne partition correspondant a l'optimum, qui sera obtenue de façon itérative en améliorant la partition initiale choisie plus au moins aléatoirement, par réallocation des objets autour des centres mobiles. [Cleuziou 2004]

(a) **Algorithmes des centres mobiles :**

L'auteur de cette méthode est forgy(1965),son principe est de tiré au hasard de l'ensemble des objets, k objets qui seront considère comme centre des classes et qu'a partir de cela on construit une partition en k classe , âpres cette sélection on calcule les distances de chaque objets a chacun des centres et on affecte chaque objet au centre le plus proche ainsi les centres de gravité constitueront les nouveaux centres

mobiles qui fourniront une nouvelle partition et ainsi de suite , on stoppe l'algorithme quand deux itérations successives fourniront la même partition .[khatir 2012]

(b) **Algorithme moyennes :[khatir 2012]**

C'est la méthode de partitionnement la plus utilisé grâce sa simplicité et a son rapport cout/efficacité avantageux, le principe de cet algorithme et de tirer au hasard kobjet de l'ensemble des objets qui vont représenter les centroides de départ, un objet est assigné au centre le plus proche, les centroides seront recalculer et on passe a l'itération suivante, il ne peut être utilise que sur des donnes décrite par des attributs numérique.

i. **Avantage :**

— Simple et compréhensible.
— S'adapte à la dimensionnalité.

ii. **Inconvénient :**

— Le nombre de classe doit être donné au départ.
— Le résultat dépend du tirage initial des centres des classes.

Algorithme k-medoides :[khatir 2012]

La différence entre cette méthode et celle des k moyenne est que dans cette dernière on utilise les centroides qui sont une valeur moyenne pour représenter une classe, dans le cas des k-medoides la classe est représenté par un de ces objets prédominant le medoide, elle couvre n'importe qu'elle type de variable.

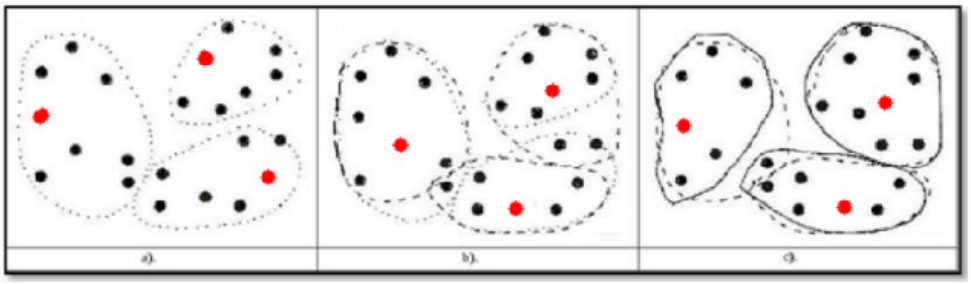


FIGURE 1.13 – Partitionnement basées sur k-medoides [khatir 2012]

i. **Avantage :**

— S'applique sur n'importe qu'elle type de variables.
— Insensible aux objets isolés.

ii. **Inconvénient :**

— Le nombre de cluster doit être fixé au départ.
— La recherche des medoides est couteuse en temps de calcul.

1.7 Conclusion :

Ce chapitre a été consacré à la notion de l'intelligence artificielle ainsi qu'à l'apprentissage automatique ,ou on a vus des notions de base de l'IA ainsi que ces diffèrent types, on a aussi détailler une multitude de méthode de l'AA qui sont soit supervisé comme les réseaux de neurones artificielles, les SVM ainsi que les arbre de décisions pour l'apprentissage non supervisé plusieurs méthodes concernant le clustering hiérarchique ainsi que le clustering par partitionnement ont étai détailler.

Dans le chapitre suivant nous étudions l'apprentissage profond ainsi que la méthode que nous avons utilisé appartenant à cette famille d'apprentissage.

Chapitre 2

Apprentissage profond

2.1 Introduction

Ce chapitre a pour objectif de présenter les réseaux de neurones utilisés dans l'apprentissage profond et en particulier les réseaux de neurones convolutifs profonds (DeepConvolutional Neural Networks CNN). De manière générale, les réseaux de neurones encodent une fonction mathématique à appliquer sur un signal d'entrée (dans notre cas, une image) et permettant de prédire un signal de sortie. Cette fonction est une composition de plusieurs fonctions non linéaires.

2.2 Apprentissage profond

Appelé aussi Deep learning, c'est un sous-ensemble de l'apprentissage automatique (AA) donc on peut dire que tout Apprentissage profond est apprentissage automatique mais pas le contraire.

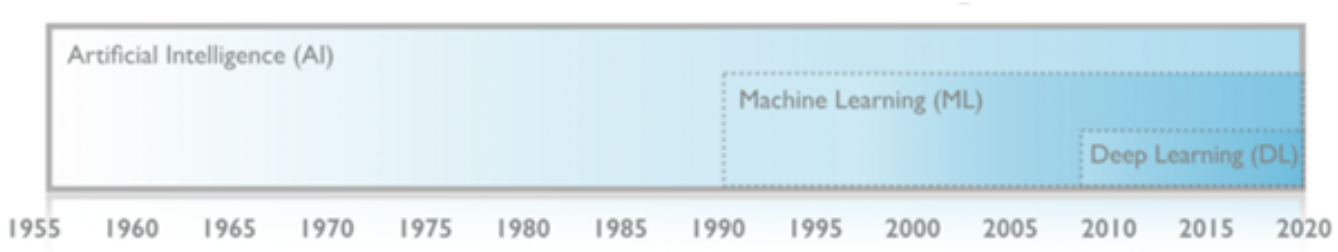


FIGURE 2.1 – L'évolution de l'intelligence artificielle [Ludovic 2016]

2.2.1 Définition :[Chafik 2017]

L'apprentissage profond limite le fonctionnement du cerveau humain dans le traitement des données. Il se réfère à des algorithmes d'apprentissage automatique (ML) permettant un traitement hiérarchique. Ce sont des techniques de traitement à plusieurs niveaux avec plusieurs couches non linéaires d'où la notion de profondeur.

La première couche présente les données d'entrée, elle est appelée couche visible, quant aux couches suivantes elles sont notées couchées cachées.

Une couche se compose de plusieurs neurones où chacun de ces derniers est connecté aux autres neurones des couches qui précèdent et qui suivent avec des connexions appelées poids, en utilisant une fonction non linéaire appelé fonction d'activation le neurone va traiter les poids d'entrées et chaque neurone caché produit une sortie qui sera ensuite traitée par d'autres neurones.

L'objectif principal de l'apprentissage profond est d'optimiser les poids des neurones dans chaque couche, c'est-à-dire que quand il reçoit une entrée, cette dernière sera analysé alors il prendra une décision à ce sujet, si la prédiction est fausse l'algorithme ajuste les connexions entre les neurones ce qui changera ces prédictions futures, même si le réseau donnera des réponses fausses à plusieurs fois, mais comme le réseau s'entraine sur des milliers d'exemples, les connexions entre les neurones seront ajusté pour que le réseau neuronal rend les bonnes décisions à chaque occasion.

2.2.2 Prétraitement des donnes :

Avant d'entrer les données dans les modèles d'apprentissage profond, ces données dites données d'entrée passent généralement par une phase de prétraitement dans le but de rendre plus facile et plus efficace les procédures d'apprentissage.

Pour traiter les données d'entrée on s'appuie sur différentes techniques de prétraitement telles que les méthodes de débruitage et de transformation :

Le débruitage consiste à effectuer une suppression ou une réduction stochastique des données d'entrée tandis que le traitement de transformation consiste à mapper l'espace de données d'entrée a un nouvel espace plus approprié.

Il existe plusieurs techniques de transformation de données, les plus utilisés sont :

1. **La soustraction moyenne :**

   Permette de centrer l'espace de caractéristique des données autour d'une origine le long de chaque dimension on soustrait la moyennes sur chaque caractéristique individuelle des données.

2. **La normalisation :**

   Une normalisation des données en entrés est effectuer pour que toute les données aient approximativement la même échelle, ainsi chaque dimension de données est normalisé pour que les vecteurs des données finaux se situent dans l'intervalle [-1,1] ou [0,1] en utilisant la formule suivante :

$$X_norm = \frac{X - X_min}{X_max - X_min} \tag{2.1}$$

   Ou x min est la valeurs minimales des donnes xet x max la valeurs maximales

3. **La standardisation :**
   Cette technique consiste à paramétrer chaque dimensions des données d'entré pour avoir une moyenne nulle et une variance unitaire en utilisant l'équation ci-dessous :

$$X_std = \frac{X - \mu}{\theta} \tag{2.2}$$

   Ou mu est la moyenne des données d'entré x et theta est l'ecart-type.

2.3 Apprentissages profonds supervisés[Chafik 2017]

L'apprentissage profond supervisé consiste à réaliser des prédictions et des classifications sur les données d'entrée en fonction de la sortie souhaitée. Dans la phase de classification les valeurs cible fournie (sortie souhaitée) sont des valeurs discrètes qui représentent des étiquettent de données.

Le modèle d'apprentissage désire alors à classer les données d'entrer dans les catégories correspondantes.

En revanche dans la phase de prédiction les valeurs cibles sont continues et le modèle d'apprentissage consiste à modéliser la relation entre la cible et les variables d'entrées.

Les taches de classification et de prédictions sont exécutées de manière hiérarchique, en essayant de minimiser un objectif prédéfini, ou une fonction de perte, en utilisant l'algorithme de rétro propagation.

2.3.1 Réseau de neurones à propagation avant profond (Deepfeed-forward neural network)

Noté feed-forward puisqu'il n'y a pas de connexion de retour dans les sorties, il est également connu sous le nom de perceptron multicouche profond, c'est une séquence de couche chacune traitant la sortie des couches précédentes.

En effet, chaque nœud de la couche x est connecté à chaque nœud de la couche x+1 par un arc, tout en sachant qu'un nœud de la couche x+1 est parent de chaque nœud de la couche x+2 et de ce fait un nœud de la couche x+1 est l'enfant de chaque nœud de la couche x, il n'existe pas de connexions entre les nœuds d'une même couche.

Les couches sont alors appelées couche entièrement connecté donc un réseau feed-forward profond est un graphe k-partite dirigé acyclique, ou les nœuds dans chaque couche forment une partition du graphe et où k est égal au nombre de couche dans les réseaux .le but d'un réseau feed-forward est d'approximer certaines fonctions F*, par exemple pour un classificateur y=F*(x)(mappe une entré x à la catégorie y) donc le réseau définit une application y=f(x, ô)et apprend la valeur des paramètres o qui donne les meilleures approximations de la fonction.

Ce type de réseaux de neurones est entrainé en utilisant l'algorithme de rétro propagation.[Goodfellow et al 2016][Graesser 2016]

2.3.2 Les réseaux de neurone convolutif

Les réseaux de neurone convolutif noté aussi CNN pour Convolutional Neural Network sont des réseaux feed-forward multicouche qui a été spécialement conçu pour la reconnaissance de caractéristique dans les images bidimensionnelles.

Puisque les CNN sont utilisés pour la reconnaissance d'image 2 D, l'image est composée de plusieurs pixels. Chaque pixel porte des informations sur la couleur, la couleur peut être représenté par les canaux RVB ou bien par un seul canal de niveaux de gris.

Dans les réseaux convolutif les neurones fonctionnent en prenant compte une petite partie de l'image appelé un sou-image, dans le but de rechercher des caractéristiques qui peuvent être reconnues par le réseau, les sous images sont examinés, ses caractéristiques sont capturées par les cartes de caractéristique (featuremaps) respective des réseaux, ces cartes de caractéristique sont la sortie d'un filtre qui est applique sur la couche précédente.

Il existe deux types de couche, la couche convolutionelle(convolution layer) et la couche de sous-échantillonnage (pooling ou subsampling layer).

L'identification des caractéristiques de l'image en entrée est l'objectif de la couche convolutionnelle, cette couche se compose de plusieurs cartes de caractéristique, pour la couche de sous échantillonnage elle suit toujours la couche convolutionelle et elle se compose du même nombre de cartes de caractéristique ou chaque carte de caractéristique de la couche convolutionnelle est utilisé comme entré pour la carte de caractéristique correspondante à la couche de sous-échantillonnage.

Selon la profondeur du réseau, on alterne entre les couches de convolution et de sous-échantillonnage jusqu'à ce que la dernière couche de cette dernière soit atteinte, il peut y avoir un certain nombre de couches entièrement connecté comme décrit dans la section précédente et enfin ont dernier lieux on a la couche de sortie.[Chafik 2017] [Vojt 2016]

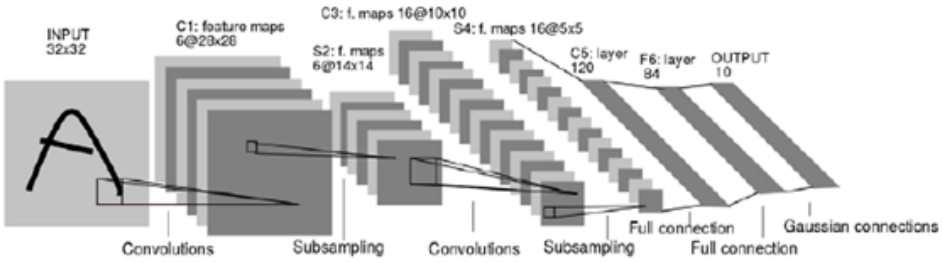


FIGURE 2.2 – Architecture CNN [Vojt 2016]

1. **Architecture du CNN :**

   Un réseau de neurones convolutif se compose de deux parties essentielles, ou chaque partie à un rôle à jouer et un objectif à remplir, la première partie (feature extraction) se charge de l'extraction des caractéristiques, des couches de convolution et des couches de sous-échantillonnage y sont alterné dedans tandis que la seconde partie effectue la classification en fonction des caractéristiques extraite dans la partie précédente, généralement un perceptron multicouche (MLP) entièrement connecté est utilisé dans cette partie.

   Le processus d'extraction des caractéristiques se fait ainsi : dans l'image qui est en entrée chaque pixel est vus comme une entité pour les neurones groupés dans les cartes de caractéristique de la première couche de convolution qui sont organisé en une grille de deux dimensions, afin d'optimiser l'implémentation en exigeant moins de mémoire et on offrant de meilleures performances, tous les neurones qui sont dans la même carte de caractéristique se partage leur poids s, et qui garantissent que le même filtre a était utilisé pour chaque pixel.

   Dans chaque carte de caractéristique donnée, chaque neurone devrait reconnaitre la même caractéristique.
   À noter que la couche d'entrée peut être vus comme une couche de sous-échantillonnage avec seulement une seule carte de caractéristique, le but des couches de sous-échantillonnage et de réduire les cartes de caractéristique.

   La seconde partie du réseau commence le processus de classification à l'aide d'un réseau MLP entièrement connecté.
2. **Les différents modules d'un réseau de neurones convolutif :**

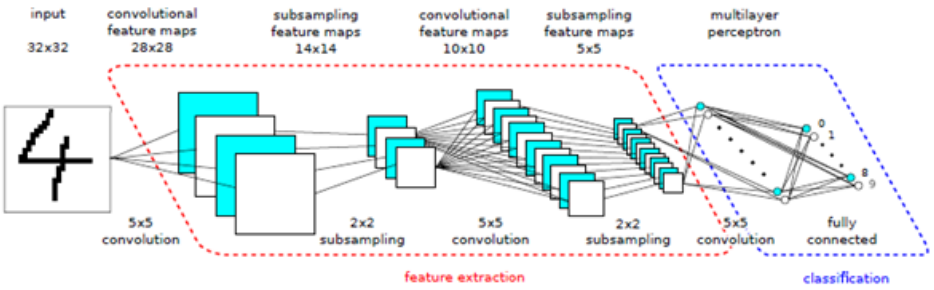


FIGURE 2.3 – Architecture CNN 2 [Vojt 2016]

Dans cette section nous allons présenter les différents modules utilisés dans les CNN :

(a) **La convolution :**

La pierre maitresse d'un CNN est la couche convolutive, son but est de détecter les caractéristiques des données en entrée, donc cette opération se réside à appliquer un noyau ou un filtre de convolution sur une matrice d'entrée (une image ou carte de caractéristique précédente) a plusieurs emplacements de la donnée en entrée et tout en extrayant les caractéristiques abstraites de chaque partie de l'entrée.

Les filtres utilisés sont appliqués sur toute l'entrée, des caractéristiques plus complexes peuvent être extraite en utilisant plusieurs couches ainsi on aura des cartes de caractéristique qui seront prêtes à être utilisé par les couches de convolution suivantes.

Le nombre de cartes de caractéristique dans la couche convolutionelle est une décision importante qui est prise en compte lors de la conception d'un CNN, il n'existe pas de valeur optimale, la meilleure façon de déterminer le nombre des cartes de caractéristiques est d'expérimenter, mais généralement un grand nombre de carte de caractéristique pour reconnaitre des images très complexes tandis qu'un petit nombre serviraient pour les taches dites simples.[Chafik 2017] [Vojt 2016]

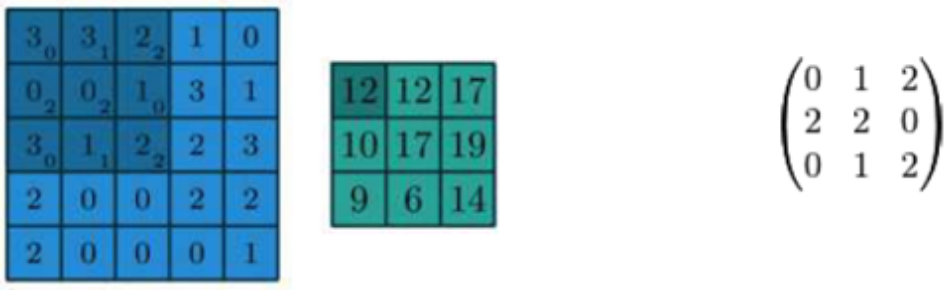


FIGURE 2.4 – Illustration de la convolution

Trois paramètres permettent de dimensionner le volume de la couche de convolution **la profondeur, le pas et la marge.**

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

— **Le pas :**

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

— **La marge (à 0) ou zeropadding :**

parfois, il est commode de mettre des zéros à la frontière du volume d'entrée. La taille de ce 'zero-padding' est le troisième hyper paramètre. Cette marge permet de contrôler la dimension spatiale du volume de sortie. En particulier, il est parfois souhaitable de conserver la même surface que celle du volume d'entrée.

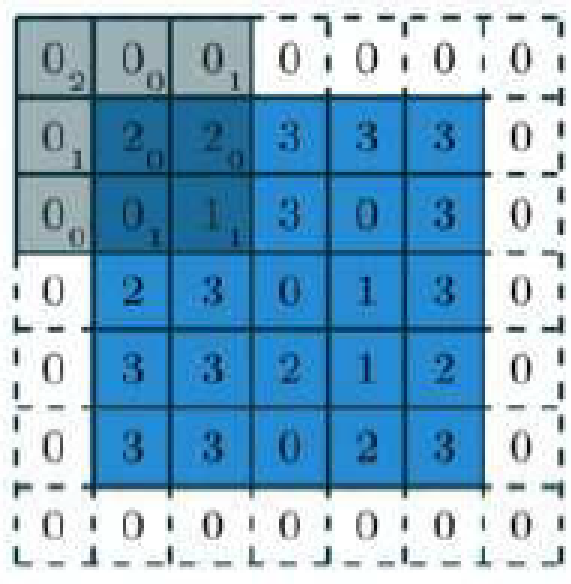


FIGURE 2.5 – Illustration du zeropadding

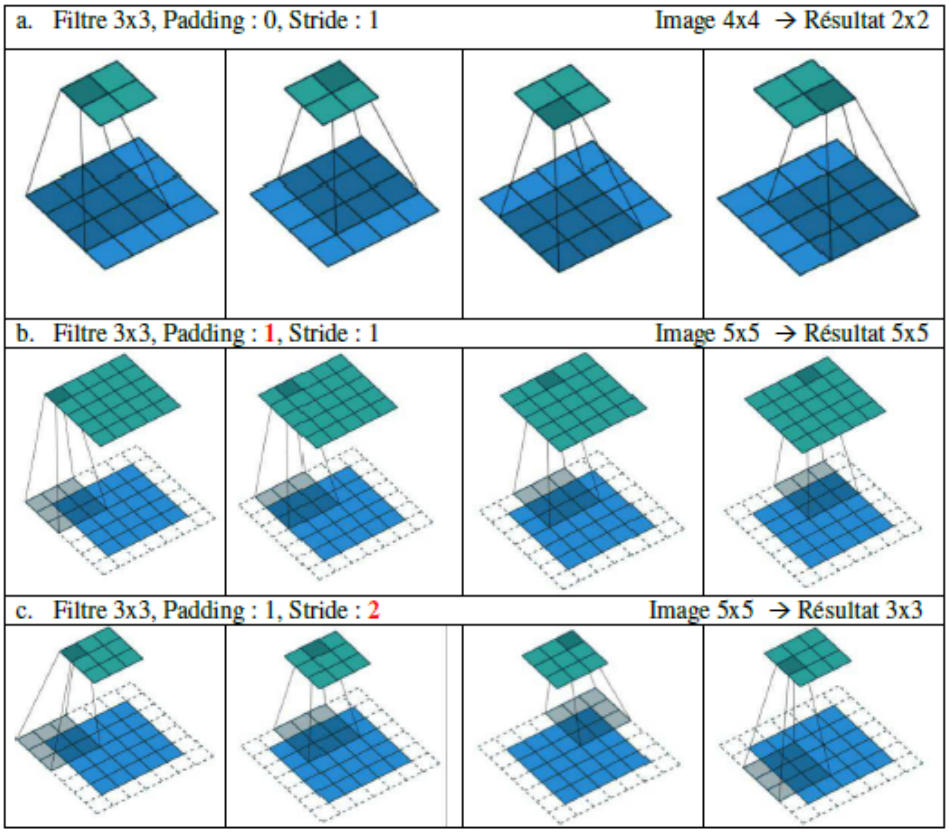


FIGURE 2.6 – illustration de la convolution avec le padding et le stride

(b) **Le pooling :**

La couche d'agglomération (subsampling ou pooling) peut-être la couche d'entrée du réseau mais généralement elle vient après la couche de convolution, le but de cette couche est de réduire la dimension de ces entrées pour simplifier et généraliser les caractéristiques extraites, il existe différents types de pooling mais les plus utilisés sont le max pooling qui renvoie l'élément maximum sur une fenêtre de calcul et l'averge pooling qui renvoie la moyenne des éléments sur une fenêtre de calcul.[Chabot 2017]

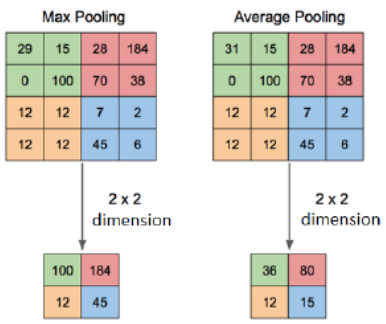


FIGURE 2.7 – illustration du pooling

(c) **Les fonctions d'activations :**

Le choix de la fonction d'activation joue un rôle très important sur les comportements des réseaux neuronal, les fonctions d'activations les plus connus et qui permette la non-linéarité dans les couches des CNN sont la sigmoïde, la tangente hyperbolique et la RELU

Sigmoïde :

C'est une fonction d'activation non linéaire simple à calculer et à différencier. Elle se définie en :

$$sigmoid(x) = \frac{1}{1 - e^{-x}} \tag{2.3}$$

Hyperbolique tangente :

C'est une autre fonction non-linéaire, mais qui donne une convergence plus que la sigmoïde puisque la sortie de la fonction sigmoïde est centrée sur 0, ce qui favorise la fonction tanh pour l'entrainement des modèles profond.

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.4}$$

RELU :

Abréviation de Unités Rectifié linéairesC'est la fonction la plus populaire et la plus utilisé dans les CNN profond c'est une fonction tres simple définit par :

$$relu(x) = max(0, x) \tag{2.5}$$

Fournir des réponses parcimonieuse (sparse) est sont point fort elle force les neurones à retourner des valeurs positives.

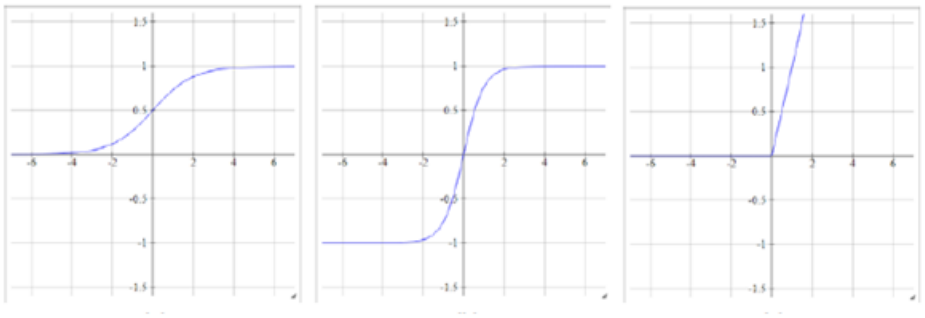


FIGURE 2.8 – Quelque fonction d'activations[Chabot 2017]

(d) **Le dropout**

C'est une couche qui est utilisée pendant l'apprentissage, son but est de désactiver au hasard plusieurs neurones pendant les différentes itérations de l'apprentissage, l'objectif étant d'éviter le sur-apprentissage (overfitting).

En d'autres termes, le dropout forme de nombreuses architectures de réseaux de neurones différents contenants moins de paramètres.

Dans la phase de test, tous les neurones seront réactivés.

(e) **La batch normalisation**

Cette technique est né d'une observation pendant l'apprentissage ou dans chaque itération, il y avait un changement sur la distribution des entrées des différentes couches du réseau.

L'idée de cette technique est de normaliser les entrées de chaque couche afin que les distributions de celles-ci soient de moyenne nulle et de variance unitaire.

Durant l'apprentissage, les couches de batch normalisation apprennent des paramètres (un facteur d'échelle et un biais) permettant d'ajuster cette normalisation : ces paramètres permettent d'appliquer une transformation sur la distribution normalisée.

(f) **La fonction de perte Softmax**

Généralement utilisée pour l'optimisation de réseau de classification d'images. Elle permet la maximisation de la probabilité qu'a une entrée d'appartenir à une classe plutôt qu'à une autre. Soit un vecteur de sortie prédit par le CNN

$$y* = [y*1, .., y*K]^T$$

et la classe désirée i. La perte Softmax est définie par :

$$E(y^*, i) = -\log(\frac{e^{y_i^*}}{\sum_j e^{y_j^*}}) \tag{2.6}$$

2.4 Apprentissages profonds non supervisé

L'apprentissage profond non supervisé (unsuperviseddeeplearning) est la classe d'apprentissage la plus adaptée pour l'entrainement sur des bases de données à grandes échelles, pendant cet entrainement les informations sur les étiquettes ne sont pas utilisées en raison de leur indisponibilité et pour le cout de leur acquisition, la méthode la plus connue pour cet apprentissage est le deepbelief network.[Chafik 2017]

2.4.1 Deepbelief network

Un DBN est un réseau de neurones profond qui est obtenu grâce à l'empilement de plusieurs modèles de machine de Boltzmann restreint, ou la sortie du RBM à la couche l-1 est l'entrée du RBM de la couche l .Un RBM est un modèle de réseaux de neurones stochastique génératif

qui relie les unités visibles aux unités cachées.[Chafik 2017]

Ces modèles génératifs sont souvent utilisés pour initialiser les poids des modèles de réseaux neuronaux afin d'aider à la généralisation de l'apprentissage, en particulier pour les modèles d'apprentissage très profond.

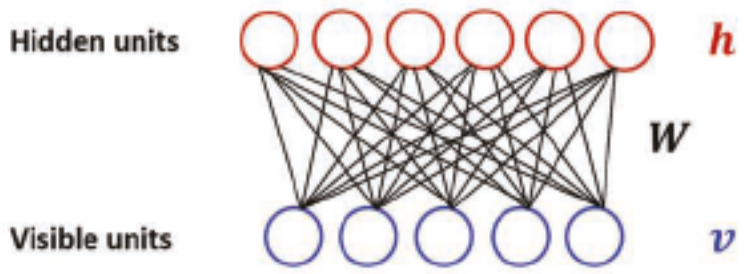


FIGURE 2.9 – Machine de Boltzmann restreint[Chafik 2017]

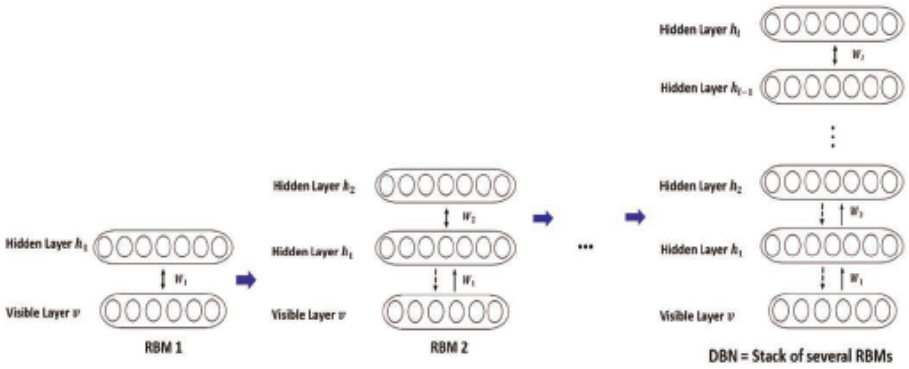


FIGURE 2.10 – Deepbelief network[Chafik 2017]

2.5 Conclusion

Dans ce chapitre nous avons présenté quelques modèles réseaux de neurones de l'apprentissage profond, tout on se basant sur les réseaux de neurones a convolution.
Ces réseaux ont la possibilité d'extraire les caractéristiques de l'image en entrée et de les classifier, les réseaux de neurones a convolution présentent cependant un certain nombre de limitations, ainsi le nombre de couches, les nombres de neurones par couches ou encore les différentes connexions entre couches sont des éléments cruciaux qu'on ne peut déterminer qu'après une multitude de tests/calcule d'erreurs (ce qui est coûteux en temps).

Chapitre 3

Expérimentation et Résultats

3.1 Introduction

L'objectif de notre travail consiste à la reconnaissance des chiffres manuscrits et que notre réseau de neurones à convolution saura faire la différence entre les différents chiffres à partir des caractéristiques extraits auparavant.

Notre choix s'est porté vers le réseau de neurone a convolution pour sa réputation, ces résultat et a la compréhensibilité de son algorithme

3.2 Critère d'évaluation :[Mokhtari et al 2018]

On va prendre un exemple pour le chiffre 0 et sa sera de même pour les autres chiffres :

3.2.1 La précision :

Cette métrique, également relative à chaque catégorie, renseigne sur la probabilité qu'une prédiction d'une catégorie donnée soit correcte.

$$prcision = \frac{VP}{VP + FP} \times 100\% \tag{3.1}$$

3.2.2 La sensibilité :

Elle représente le taux de classification ou la classe est un zéro (détecté Correctement par le classifieur) par rapport au nombre total des classes zéros.

$$Sensibilite = \frac{VP}{VP + FN} \times 100\% \tag{3.2}$$

3.2.3 La spécificité :

Elle représente le taux de classification des exemples ou la classe n'est pas un zéro est prédit non-zéro. (Détectés correctement par le classifieur) par rapport au nombre total des classes non zéros.

$$Specificite = \frac{VN}{VN + FP} \times 100\% \tag{3.3}$$

VP, VN, FP et FN désignent respectivement :

— Vrai positif : un exemple ou la classe est un zéro est prédit correctement.
— Vrai négatif : un exemple ou la classe n'est pas un zéro est prédit non zéro.
— Faux positif : un exemple ou la classe n'est pas un zéro est prédit zéro.
— Faux négatif : un exemple ou la classe est un zéro est prédit non zéro.

3.3 Mnist :[Lecun et al]

La base de données MNIST des chiffres manuscrits, disponible à partir de le site officiel présent dans la bibliographie, a un ensemble d'apprentissage de 60 000 exemples et un ensemble de tests de 10 000 exemples. C'est un sous-ensemble d'un plus grand ensemble disponible du NIST. Les chiffres ont été normalisés en taille et centrés dans une image de taille fixe.

C'est une bonne base de données pour les personnes qui veulent essayer des techniques d'apprentissage et des méthodes de reconnaissance de formes sur des données réelles tout en dépensant un minimum d'efforts pour le prétraitement et le formatage.

Les images originales en noir et blanc (Bilevel) du NIST ont été normalisées pour s'adapter à une boîte de 20 x 20 pixels tout en préservant leur format. Les images résultantes contiennent des niveaux de gris résultant de la technique d'anti-aliasing utilisée par l'algorithme de normalisation. Les images ont été centrées dans une image 28x28 en calculant le centre de masse des pixels, et en traduisant l'image de manière à positionner ce point au centre du champ 28x28.

La base de données MNIST a été construite à partir de la base de données spéciale 3 du NIST et de la base de données spéciale 1 qui contiennent des images binaires de chiffres manuscrits. À l'origine, le NIST désignait SD-3 comme ensemble d'apprentissage et SD-1 comme ensemble de test. Cependant, SD-3 est beaucoup plus propre et plus facile à reconnaître que SD-1. La raison en est que SD-3 a été recueilli parmi les employés du Census Bureau, tandis que SD-1 a été recueilli parmi les élèves du secondaire. Tirer des conclusions sensées des expériences d'apprentissage exige que le résultat soit indépendant du choix de l'ensemble d'apprentissage et du test parmi l'ensemble complet des échantillons. Il était donc nécessaire de construire une nouvelle base de données en mélangeant les jeux de données du NIST.

L'ensemble de formation MNIST est composé de 30 000 modèles de SD-3 et 30 000 modèles de SD-1. Notre ensemble de test était composé de 5000 modèles de SD-3 et de 5 000 modèles de SD-1. L'ensemble de formation de 60 000 modèles contenait des exemples d'environ 250 écrivains. Nous nous sommes assuré que les groupes d'auteurs de l'ensemble d'apprentissage et de test étaient disjoints.SD-1 contient 58 527 images numériques écrites par 500 auteurs différents. Contrairement à SD-3, où les blocs de données de chaque auteur apparaissaient en séquence, les données dans SD-1 sont brouillées. Les identités d'auteur pour SD-1 sont disponibles et nous avons utilisé cette information pour déchiffrer les auteurs. Nous avons ensuite divisé SD-1 en deux : les caractères écrits par les 250 premiers auteurs ont été intégrés dans notre nouvel ensemble d'entraînement. Les 250 auteurs restants ont été placés dans notre ensemble de test. Nous avions donc deux séries avec près de 30 000 exemplaires chacune. Le nouvel ensemble d'entraînement a été complété avec suffisamment d'exemples de SD-3, à partir du modèle n 0, pour créer un ensemble complet de 60 000 schémas d'entraînement. De même, le nouvel ensemble de test a été complété avec des exemples SD-3 commençant au modèle 35000 pour faire un ensemble complet avec 60 000 modèles de test. Seul un sous-ensemble de 10 000 images de test (5 000 de SD-1 et 5 000 de SD-3) est disponible sur ce site. L'ensemble complet de 60 000 échantillons est disponible.

L'ensemble d'apprentissage contient 60000 exemples et l'ensemble d'essai 10000 exemples.

Les 5000 premiers exemples de l'ensemble de test sont tirés de l'ensemble d'apprentissage NIST d'origine. Les 5000 derniers sont tirés de l'ensemble de test NIST d'origine. Les premiers 5000 sont plus propres et plus faciles que les 5000 derniers.

3.4 Les expérimentations

Nous avons opté pour deux grandes expérimentations, la première est implémentée en python et la deuxième en java.

3.4.1 Première expérimentation

Cette partie a étais développé avec le langage de programmation python.

1. **Architecture de notre CNN :** Nous avons créé trois modèles, où chacun de ce dernier a une architecture différente des autres et qui sont tous déployé sur la même base mnist.
   (a) **Premier modèle :**

Le 1er modèle se compose de 3 couches de convolution, deux couches de pooling et trois couches cachées pour le perceptron multicouche, la figure 3.1 détaille bien ce schéma.

La taille de l'image en entrée et de taille 28*28, elle passe directement sur la première couche de convolution qui se compose de 32 filtres de taille 3*3 ce qui fait que 32 cartes de caractéristiques (features maps) de taille (26*26) seront créées en sortie, notons que chacune de nos couches de convolution est suivie d'une fonction d'activation Relu qui va forcer les neurones à retourner des valeurs positives.

La sortie obtenue auparavant (les 32 features map) se présentera comme l'entrée de la couche pooling pour réduire la taille de l'image ainsi la quantité de paramètres et de calcul, on aurait donc en sortie 32 features maps de taille (13*13) qui sera redirigée vers la deuxième couche de convolution composée de 64 filtres qui seront ainsi de taille (11*11) suivi de la fonction d'activation Relu.

Une troisième couche de convolution (64 features maps) est applique sur la sortie de la couche précédente qui sera toujours suivie par la fonction d'activation Relu, après un dernier couche de pooling qui est suivi de la fonction dropout, on aura ainsi 64 features maps de taille 4*4, le vecteur de caractéristiques issues des convolutions a une dimension de 1024.

Nous utilisons un réseau de neurones composé de trois couches caché, les deux premières sont chacune 1024 neurones et sont tous les deux suivis de la fonction dropout, la dernière couche est un soft max composé de 10 neurones qui permet de calculer la distribution de probabilité des 10 classes (nombre de classe dans la base d'image Mnist).

Layer (type)	Output Shape	Param #
conv2d_6 (Conv2D)	(None, 26, 26, 32)	320
max_pooling2d_5 (MaxPooling2	(None, 13, 13, 32)	0
conv2d_7 (Conv2D)	(None, 11, 11, 64)	18496
conv2d_8 (Conv2D)	(None, 9, 9, 64)	36928
max_pooling2d_6 (MaxPooling2	(None, 4, 4, 64)	0
dropout_7 (Dropout)	(None, 4, 4, 64)	0
flatten_3 (Flatten)	(None, 1024)	0
dense_7 (Dense)	(None, 1024)	1049600
dropout_8 (Dropout)	(None, 1024)	0
dense_8 (Dense)	(None, 1024)	1049600
dropout_9 (Dropout)	(None, 1024)	0
dense_9 (Dense)	(None, 10)	10250

Total params: 2,165,194
Trainable params: 2,165,194
Non-trainable params: 0

None

FIGURE 3.1 – Paramètres modèle 1

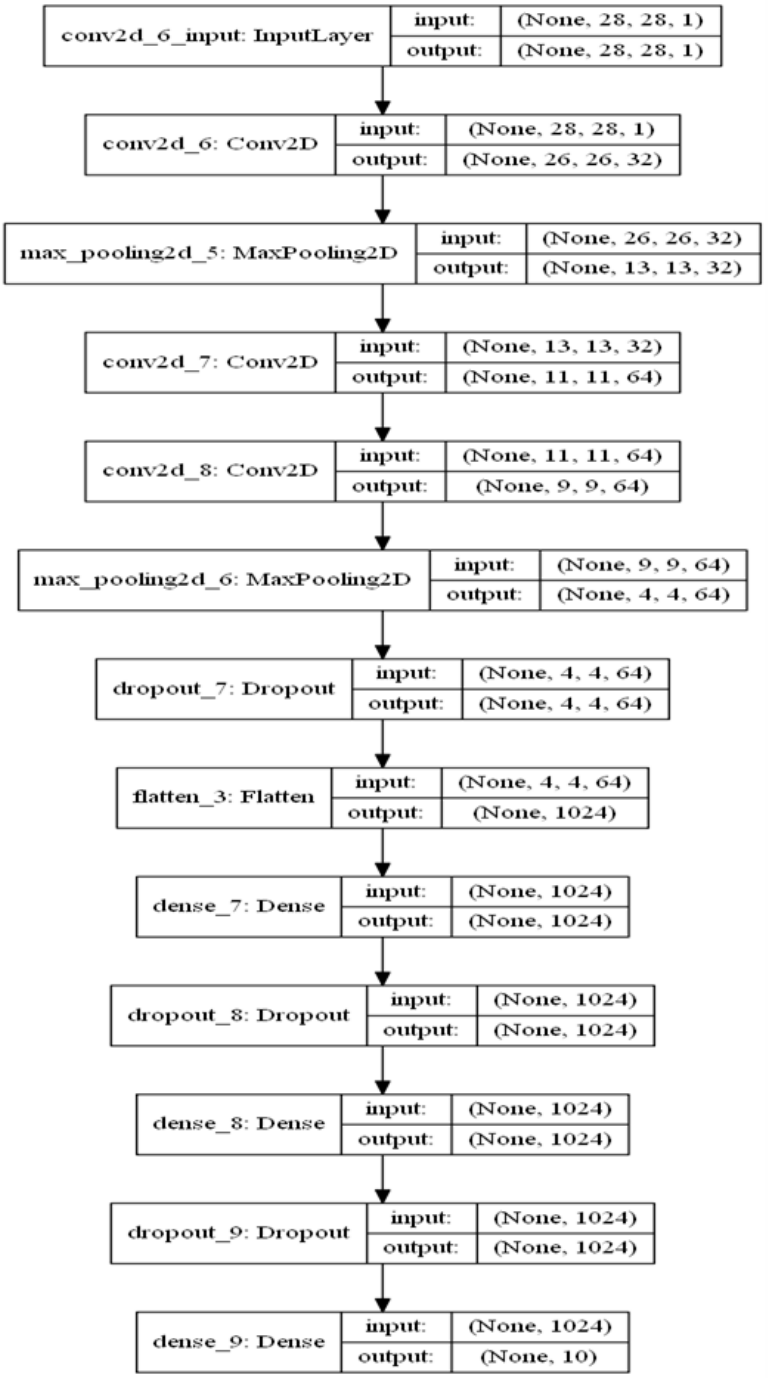


FIGURE 3.2 – Architecture modèle 1

(b) **Deuxième modèle :**

Pour le 2eme modèle, il se compose de 3 couches de convolution, deux couches de pooling et trois couches cachées pour le perceptron multicouche le seul changement par apport au 1er modèle c'est dans la taille de la convolution puisqu'elle ne sera plus de 3*3 mais de 5*5, la figure 3.3 détaille bien ce schéma.

La taille de l'image en entrée et de taille 28*28, elle passe directement sur la première couche de convolution qui se compose de 32 filtres de taille 5*5 ce qui fait que 32 cartes de caractéristiques (features maps) de taille (24*24) sera créée en sortie, notons que chacune de nos couches de convolution est suivie d'une fonction

d'activation Relu qui va forcer les neurones à retourner des valeurs positives.

La sortie obtenue auparavant (les 32 features map)se présentera comme l'entrée de la couche pooling pour réduire la taille de l'image ainsi la quantité de paramètres et de calcul, on aurait donc en sortie 32 features maps de taille (12*12) qui sera redirigée vers la deuxième couche de convolution composée de 64 filtres qui seront ainsi de taille (8*8) suivi de la fonction d'activation Relu. Une troisième couche de convolution (64 features maps) est applique sur la sortie de la couche précédente qui sera toujours suivie par la fonction d'activation Relu, après un dernier couche de pooling qui est suivi de la fonction dropout, on aura ainsi 64 features maps de taille 2*2, le vecteur de caractéristiques issues des convolutions a une dimension de 256.

Nous utilisons un réseau de neurones composé de trois couches caché, les deux premières ont chacune 1024 neurones et sont tous les deux suivis de la fonction dropout, la dernière couche est une softmax composé de 10 neurones qui permet de calculer la distribution de probabilité des 10 classes (nombre de classe dans la base d'image Mnist).

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 24, 24, 32)        832
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 12, 12, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 8, 8, 64)          51264
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 4, 4, 64)          102464
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 2, 2, 64)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 2, 2, 64)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 256)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1024)              263168
_________________________________________________________________
dropout_2 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_2 (Dense)              (None, 1024)              1049600
_________________________________________________________________
dropout_3 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_3 (Dense)              (None, 10)                10250
=================================================================
Total params: 1,477,578
Trainable params: 1,477,578
Non-trainable params: 0
_________________________________________________________________
None
```

FIGURE 3.3 – Paramètres modèle 2

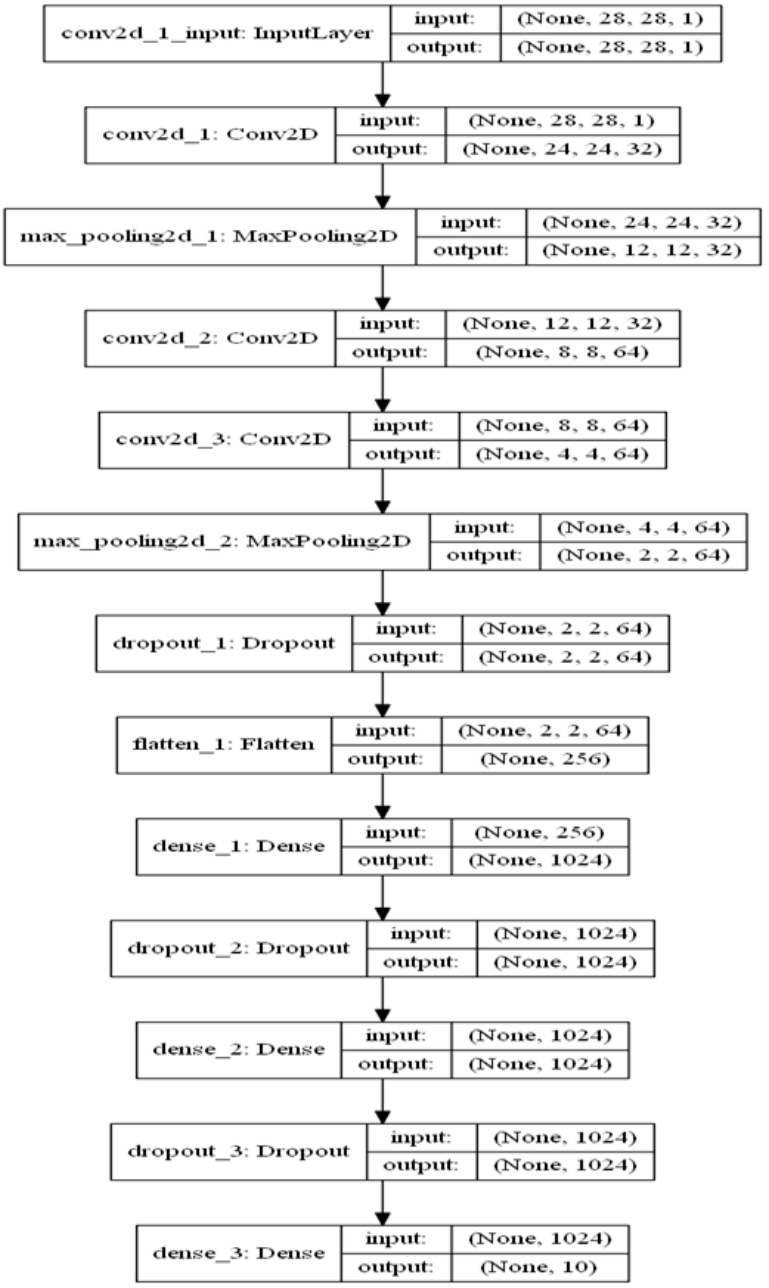


FIGURE 3.4 – Architecture modèle 2

(c) **Troisième modèle :**

Pour Le 3eme est dernier modèle, nous avons opté pour six couches de convolution, une couche de pooling et quatre couches cachées pour le perceptron multicouche, la figure 3.5 détaille bien ce schéma.

La taille de l'image en entrée et de taille 28*28, elle passe directement sur la première couche de convolution qui se compose de 32 filtres de taille 3*3 ce qui fait que 32 cartes de caractéristiques (features maps) de taille (26*26) sera créée en sortie, notons que chacune de nos couches de convolution est suivie d'une fonction d'activation Relu qui va forcer les neurones à retourner des valeurs positives.

La sortie obtenue auparavant (les 32 features map) se présentera comme l'entrée de deuxième couche de convolution composée de 32 filtres qui seront ainsi de taille (24*24) suivi de la fonction d'activation Relu. Deux autres couches de convolution se succéderont qui seront composés de 64 features maps.

Une couche de pooling suivi de la fonction dropout qui réduira la taille de l'image de 20*20 à 10*10, Deux autres couches de convolution se succéderont qui seront composé de 128 features maps. On aura ainsi 128 features maps de taille 6*6, le vecteur de caractéristiques issues des convolutions à une dimension de 4608.

Nous utilisons un réseau de neurones composé de quatre couches caché, la première et la troisième ont chacune 1024 neurones et sont tous les deux suivis de la fonction dropout, la deuxième couche est composée de 2048 neurones, la dernière couche est une softmax composé de 10 neurones qui permet de calculer la distribution de probabilité des 10 classes (nombre de classe dans la base d'image Mnist).

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 26, 26, 32)	320
conv2d_2 (Conv2D)	(None, 24, 24, 32)	9248
conv2d_3 (Conv2D)	(None, 22, 22, 64)	18496
conv2d_4 (Conv2D)	(None, 20, 20, 64)	36928
max_pooling2d_1 (MaxPooling2	(None, 10, 10, 64)	0
conv2d_5 (Conv2D)	(None, 8, 8, 128)	73856
conv2d_6 (Conv2D)	(None, 6, 6, 128)	147584
dropout_1 (Dropout)	(None, 6, 6, 128)	0
flatten_1 (Flatten)	(None, 4608)	0
dense_1 (Dense)	(None, 1024)	4719616
dropout_2 (Dropout)	(None, 1024)	0
dense_2 (Dense)	(None, 2048)	2099200
dropout_3 (Dropout)	(None, 2048)	0
dense_3 (Dense)	(None, 1024)	2098176
dropout_4 (Dropout)	(None, 1024)	0
dense_4 (Dense)	(None, 10)	10250

Total params: 9,213,674
Trainable params: 9,213,674
Non-trainable params: 0

FIGURE 3.5 – Paramètres modèle 3

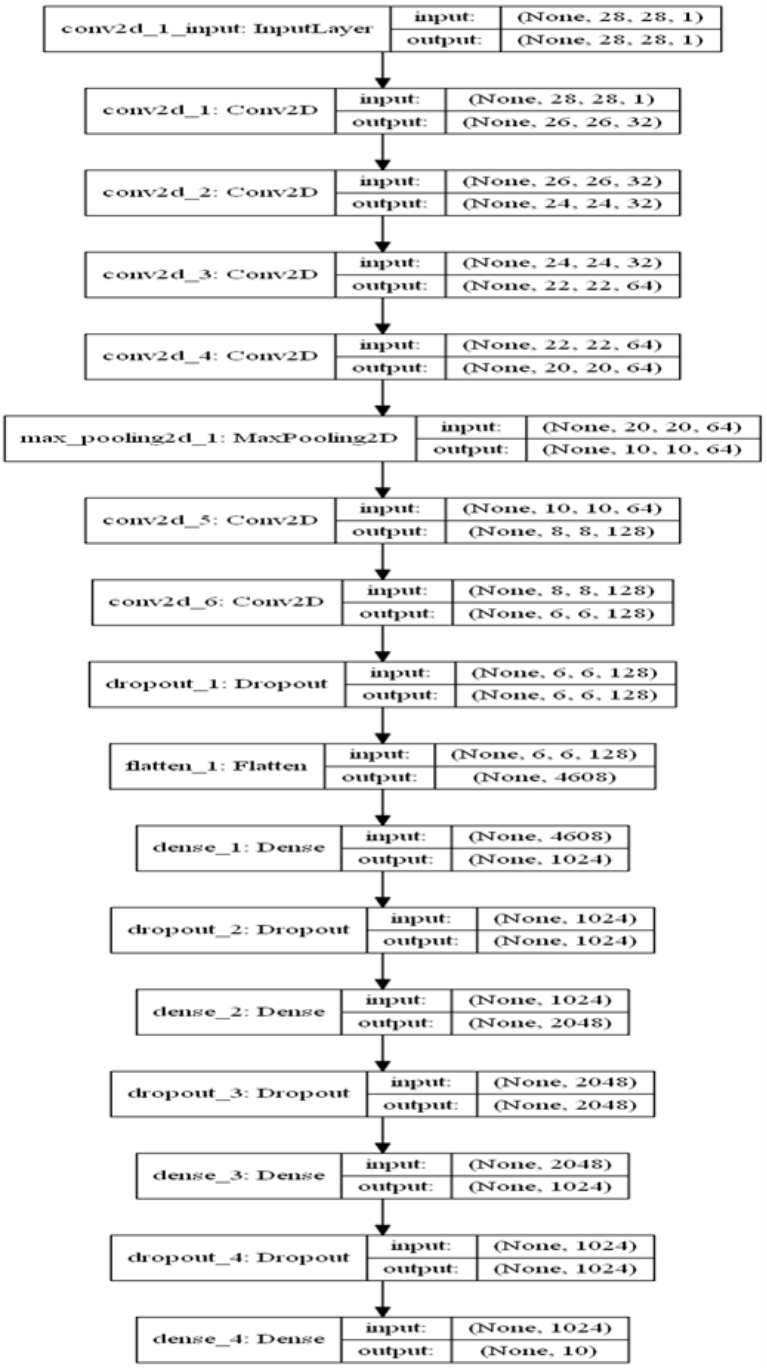


FIGURE 3.6 – Architecture modèle 3

2. **Résultats obtenus et discussion :** Afin de montrer les résultats obtenus pour les trois modèles, on illustre dans ce qui suit les résultats en matière de précision et d'erreur ainsi que matrice de confusion pour chacun des trois modèles.

(a) **Résultats obtenus pour le modèle 1 :**

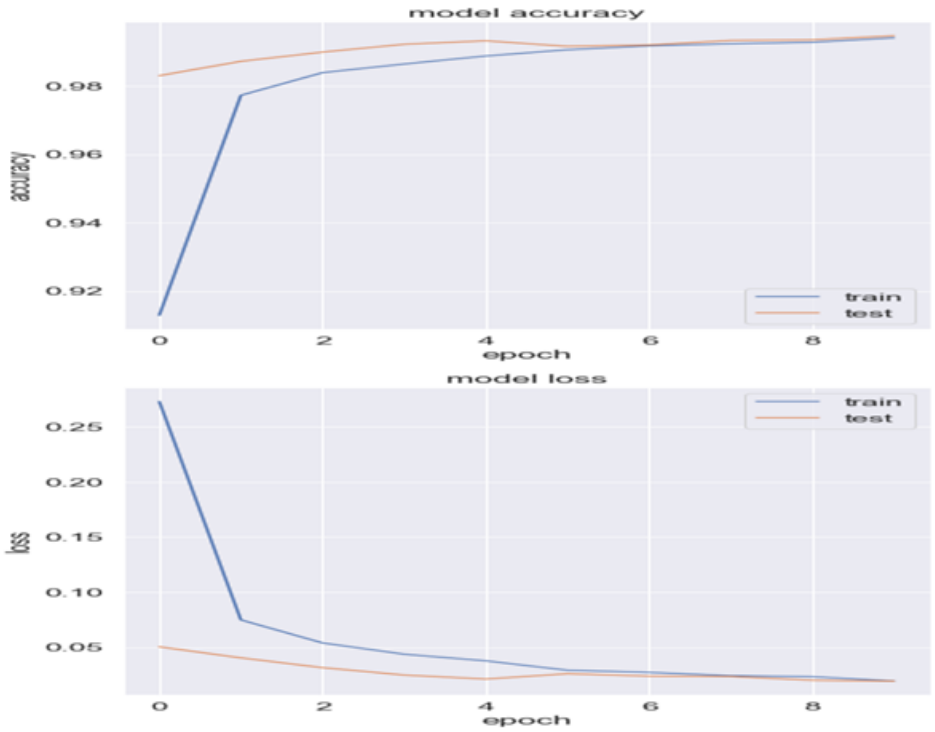


FIGURE 3.7 – Précision et Erreur pour le Modèle 1

Après l'analyse des résultats obtenus, On constate les remarques suivantes :

D'après la Figure 3.7 la précision de l'apprentissage et de test augmente avec le nombre d'époque,ceci reflète qu'à chaque époque le modèle apprend plus d'informations.

De même, l'erreur d'apprentissage et de la validation diminue avec le nombre d'époque.

Nous remarquons aussi que la totalité des images mal classées est de 52 images, un taux d'erreur de 1.93 et la totalité des images bien classées est de 9948 un taux de précision de 99.48.

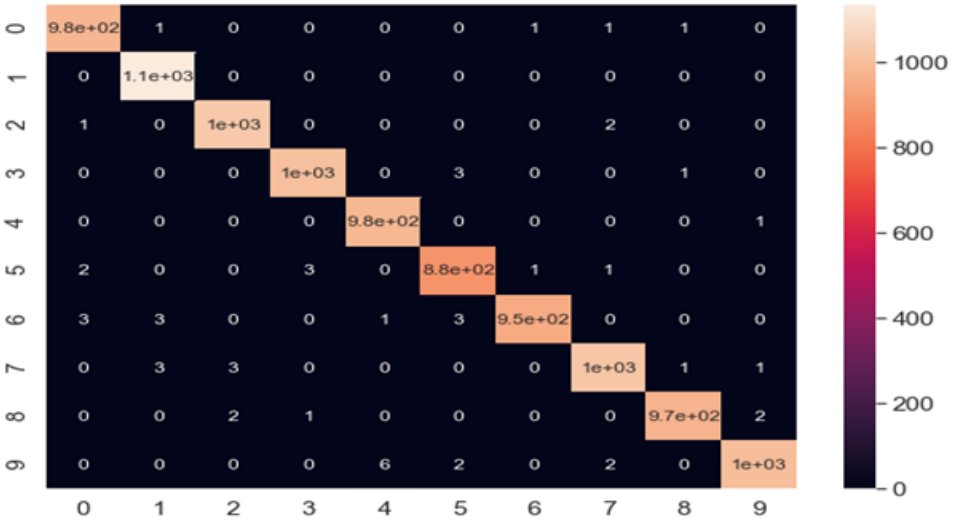


FIGURE 3.8 – Matrice de confusion pour le Modèle 1

(b) **Résultats obtenus pour le modèle 2 :**

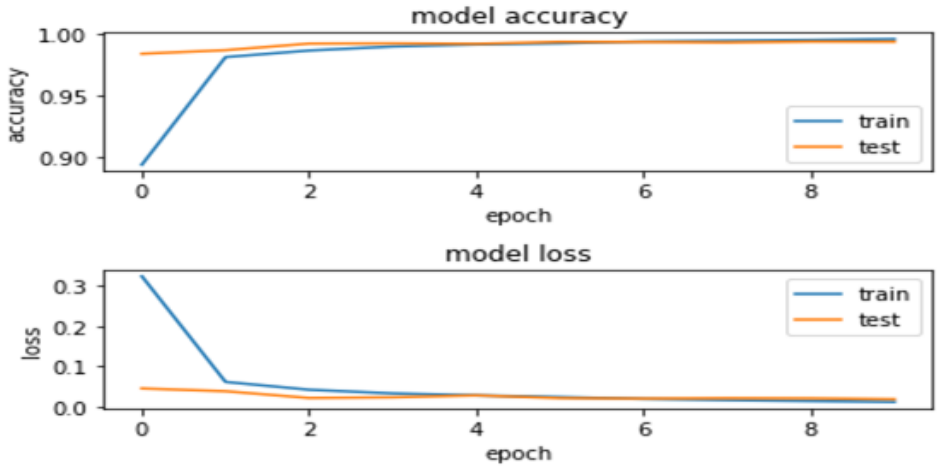


FIGURE 3.9 – Précision et Erreur pour le Modèle 2

Après l'analyse des résultats obtenus, On constate les remarques suivantes :

D'après la Figure 3.9 la précision de l'apprentissage et de test augmente avec le nombre d'époque,ceci reflète qu'à chaque époque le modèle apprend plus d'informations.

De même, l'erreur d'apprentissage et de la validation diminue avec le nombre d'époque.

Nous remarquons aussi que la totalité des images mal classées est de 59 images, un taux d'erreur de 1.92 et la totalité des images bien classées est de 9941 un taux de précision de 99.41.

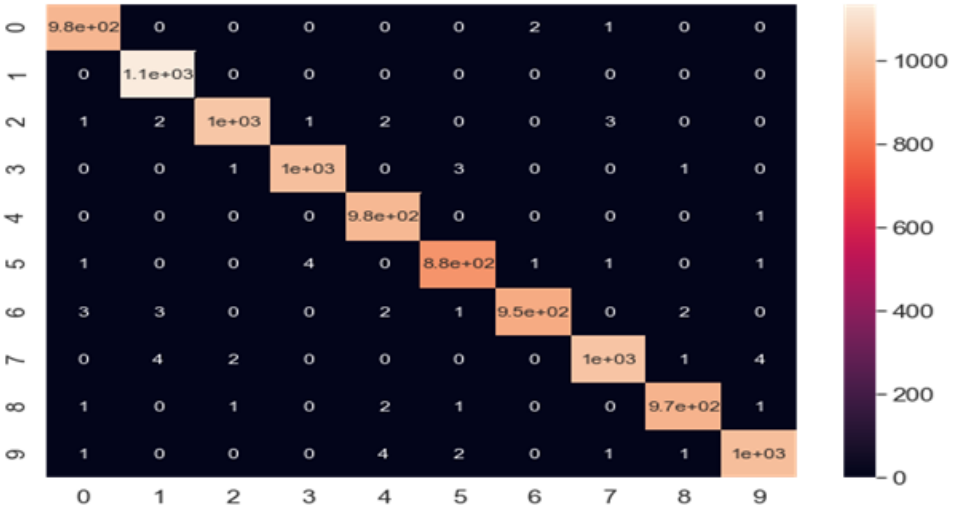


FIGURE 3.10 – Matrice de confusion pour le Modèle 2

(c) **Résultats obtenus pour le modèle 3 :**

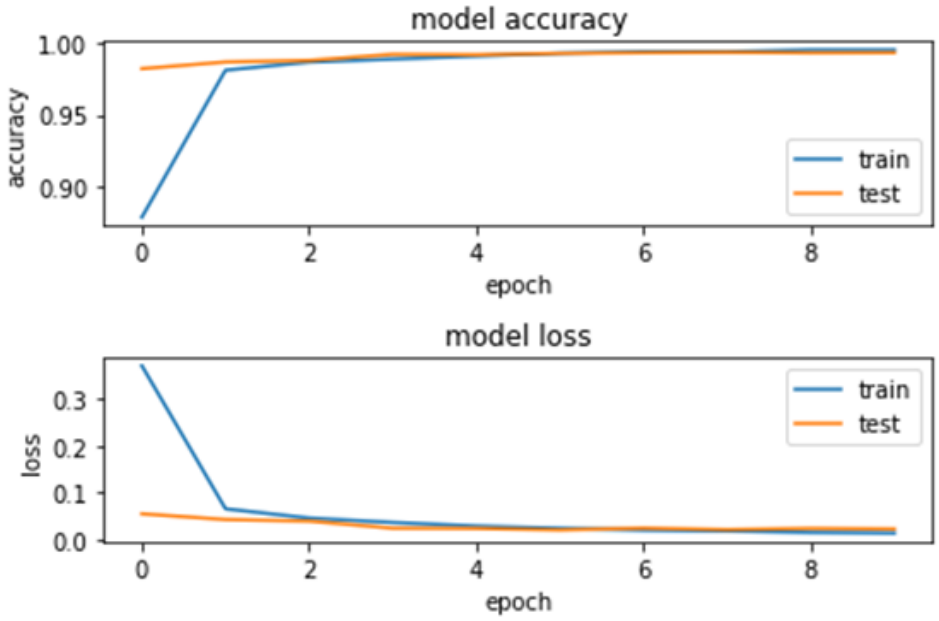


FIGURE 3.11 – Précision et Erreur pour le Modèle 3

Après l'analyse des résultats obtenus, On constate les remarques suivantes :

D'après la Figure 3.10 la précision de l'apprentissage et de test augmente avec le nombre d'époque,ceci reflète qu'à chaque époque le modèle apprend plus d'informations.

De même, l'erreur d'apprentissage et de la validation diminue avec le nombre d'époque.

Nous remarquons aussi que la totalité des images mal classées est de 62 images,

un taux d'erreur de 2.25 et la totalité des images bien classées est de 9938 un taux de précision de 99.38.

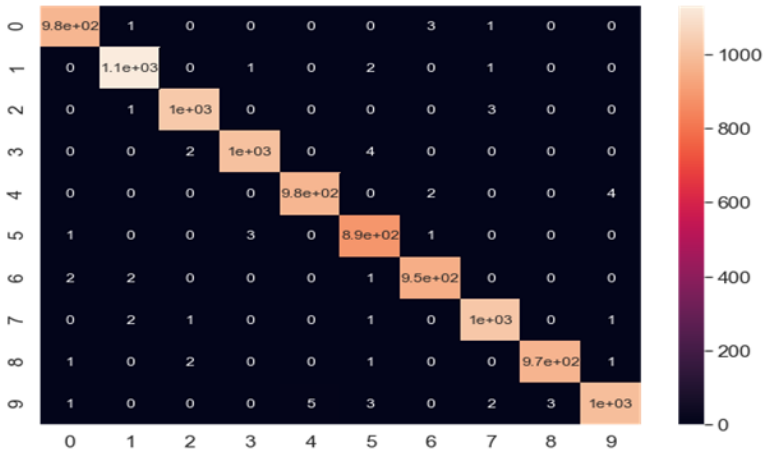


FIGURE 3.12 – Matrice de confusion pour le Modèle 3

3. **Tableau de comparaison des résultats :**

Le tableau ci-dessous montre les différents résultats obtenus sur les trois modèles :

	Architecture Utilisé				Nombre époque	Précision obtenu Sur la base d'apprentissage	Précision obtenu Sur la base de test	Erreur	Temps d'exécution
	Couche de Convolution	Taille convolution	Couche de pooling	Couche caché MLP					
1	3	3*3	2	3	10	99.42%	99.48%	1.93%	1486 secondes (25minutes)
2	3	5*5	2	3	10	99.57%	99.37%	2.45%	1280 Secondes (22 minute)
3	6	3*3	1	4	10	99.58%	99.38%	2.25%	8493 secondes (2heure)

FIGURE 3.13 – Tableau de comparaison des résultats

Le tableau montre l'architecture utilisée dans chaque modèle ainsi que le nombre d'époque. Les résultats obtenus sont exprimés en matière de précision d'apprentissage, de test et erreur et enfin de temps d'exécution.

Le temps d'exécution un peu couteux, celui-ci démunirait si on utilisait un GPU au lieu d'un CPU. Les 3 modèles présents d'excellents résultats.

Chaque fois qu'on approfondi un peu plus notre réseau donc on augmente le nombre de paramètres la précision obtenu sur la base d'apprentissage augmente ainsi que le taux

d'erreur du test et ça va de même pour le temps d'exécution, pour ce qui est de la précision obtenue Sur la base test elle a diminué.

3.4.2 Deuxième expérimentation :

Cette partie a été implémentée avec le langage de programmation java.

1. **Approche proposé :**

   Dans notre approche proposée nous avons utilisé la nouvelle tendance de la recherche informatique l'apprentissage profond.

   Cette technique d'apprentissage, basée sur des réseaux de neurones artificiels, a complètement bouleversé le domaine de l'intelligence artificielle.

   Cette technique est composée de deux blocs, la première partie est l'extraction des caractéristiques de chaque image en entrée, via le Réseau de Neurones a Convolution «RNC».

   Cette étape est le prétraitement des données de la base utilisée afin de les utiliser dans la deuxième partie de cette technique fait appel au processus de classification à l'aide d'un Perceptron Multi Couche «PMC».

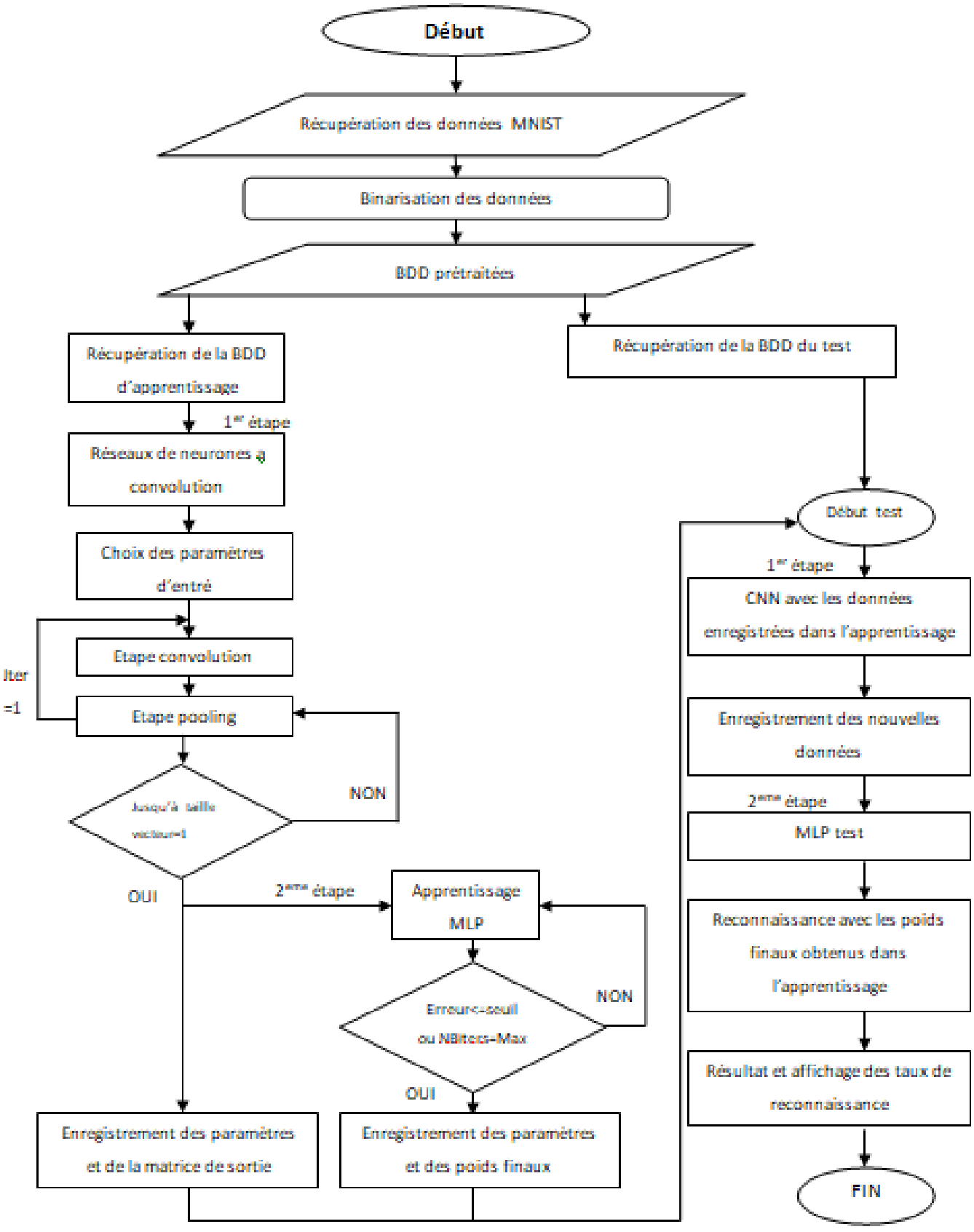


FIGURE 3.14 – Organigramme de notre approche proposée

2. **Récupération des données MNIST :**

Lors du chargement de la base de données nous avons profité de la notion de pa-

rallélisme vu que la base utilisée est assez importante (60000 images) sachant que la base de données a était divisée en 10 groupes un pour chaque classe et chaque classe est récupérée par un thread afin de gagner en temps d'exécution.

3. **Binarisation par seuillage :**

La première stratégie consiste à binariser l'image par seuillage et à garder les pixels dont l'intensité est supérieure au seuil fixé. On regroupe donc au sein d'une même classe, des pixels de valeurs comprises dans un intervalle donné. Le seuil est généralement défini par l'utilisateur en tenant compte des caractéristiques de l'image.

Cette stratégie de binarisation présente l'avantage d'être simple et rapide à mettre en œuvre. Cependant, le choix du seuil est une opération délicate. Des parties d'objets d'intérêts peuvent être affectées au fond de l'image si les pixels ont une intensité légèrement inférieure au seuil fixé.[Sheeren et al 2007]

(a) **Binarisation proposé**

L'appelle de cette méthode est due au résultat infini des valeurs obtenues de la 1er partie (CNN), donc on a opté pour un seuillage tel que suit :

- On attribue la valeur 0 pour le pixel inférieur ou égal a 128.
- On attribue la valeur 1 pour les pixels supérieurs à 128.

Après le prétraitement des bases données l'approche proposée se constitue de deux principales étapes : l'étape d'apprentissage et l'étape de test.

4. **L'étape d'apprentissage :**

Après la récupération de la base de données prétraitée dédié à l'apprentissage contenant 60000 images on procède aux étapes suivantes :

(a) **Le réseau de neurones a convolution :**

Juste après avoir chargé la base de données on choisit les paramètres d'entrée, notre CNN passe en premier lieu sur l'étape de convolution où nous avons droit à 3 choix (une convolution de taille 3*3,5*5, 7*7).

Pour la convolution a 3*3 plusieurs morceaux (selon le nombre des features maps choisi) de taille 3*3 sont pris de l'image (28*28) ces morceaux sont appelé filtre et on a le choix entre 2 types de stride (un pas, deux pas).

Pour le stride a un seule pas les filtres choisis auparavant se déplacent sur l'image d'un seul pixel de gauche à droite et de haut en bas, On voulant appliquer la partie convolution à l'image les filtres choisis se déplacent sur celle-ci en sommant le produit des valeurs de chaque pixel qui ont les mêmes positions ainsi juste après cette étape on aura une nouvelle image qui démunirait de deux pixels sur la largeur ainsi que sur la hauteur (26*26) vient ensuite l'étape de pooling on prend un morceau de taille fixe de 2*2 de l'image et on fait soit un average pooling qui consiste à faire la moyenne de ces quatre valeurs ou bien un max pooling qui consiste à prendre le maximum des quatre valeurs, dans les deux cas une autre nouvelle image apparaîtras (13*13) une autre convolution sera applique à cette dernière et l'image aura une taille de 11*11 après cela on utilisera trois couches de pooling pour faire diminuer la taille de l'image successivement à 5*5,2*2, et on dernier lieux ont une seule valeur.

Pour le stride a deux pas les filtres se déplacent sur l'image de deux pixels, une fois la première convolution la taille de la nouvelle image sera de 13*13 qui passe par une étape de pooling qui sera alors de 6*6 une autre couche de convolution est appliqué ceux qui donnerais une image de taille 2*2 ensuite une dernière couche de pooling dans la sortie sera une seule valeur.

Pour la convolution 5*5 avec le stride d'un seul pas, on appliquant cette dernière sur l'image en entrée (28*28) la taille de l'image obtenue sera de 24*24, une couche de pooling sur celle-ci démunirait la taille à 12*12, une autre ouche de convolution et on aura 8*8 Vien ensuite une succession de couches de pooling jusqu'à ce qu'on aura une seule valeur.

Pour le stride à deux pas et après la première convolution l'image sera de taille 12*12 vien ensuite une étape de polling et on aura 6*6 comme taille de l'image une dernière étape de convolution et le résultat seront une seule valeur.

Pour la convolution 7*7 avec le stride d'un seul pas, ont appliquant cette dernière sur l'image en entrée (28*28) la taille de l'image obtenue sera de 22*22, une couche de pooling sur celle-ci diminuerait la taille à 11*11, une autre ouche de convolution et on aura 5*5 vienne ensuite une succession de couches de pooling jusqu'à ce qu'on aura une seule valeur.

Pour le stride à deux pas et après la première convolution l'image sera de taille 11*11 vienne ensuite une multitude de couches de pooling jusqu'à avoir une seule valeur en sortie.

Maintenant que les réseaux de neurones à convolution terminé, on enregistre les paramètres d'entrée ainsi que le vecteur de sortie dans une base de données.

(b) **Perceptron multicouche :**

Notre MLP est constitué de trois couches, une couche d'entrée, une cachée et une de sortie décrite comme suit :

- Couche d'entrée : les valeurs d'entrés représentent des vecteurs variables selon le nombre des features maps choisie.

- Couche cachée : nous avons opté que pour une seule couche cachée avec un nombre de neurones fixé à 128.

- Couche de sortie : notre MLP contient dix neurones de sortie une pour chaque classe (0,1...,8,9)

- Fonction d'activation : nous avons utilisé la fonction sigmoïde définit par :

$$f(x) = 1/(1 + e^{(-x)}) \tag{3.4}$$

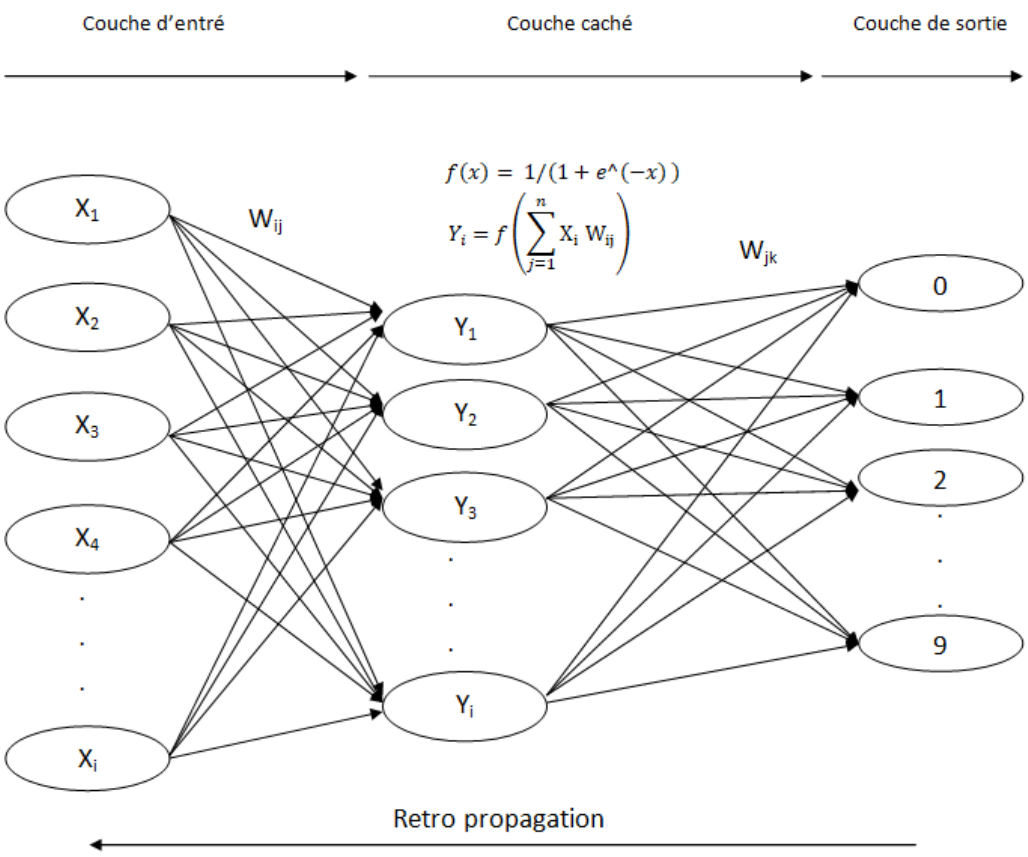


FIGURE 3.15 – MLP

Cet apprentissage est effectué dans le but de calculer les matrices de poids Wij et Wjk les plus optimaux.

Au début de l'apprentissage les matrices de poids sont initialisées aléatoirement.

Cet apprentissage se fera sur le vecteur de sortie de notre réseau de neurones à convolution.

Maintenant que l'apprentissage du perceptron multicouche terminé, on enregistre les paramètres d'entrée ainsi que les matrices de poids Wij et Wjk dans le but de les réutiliser dans la phase de test.

5. **L'étape de test :**

Après la récupération de la base de données prétraitée dédié aux tests contenant 10000 images on procède aux étapes suivantes :

les données sont passé aux réseaux de neurones a convolution avec les paramètres d'entrée enregistrée dans la phase d'apprentissage et on enregistre les nouvelles données dans la base de données.

Ces mêmes données sont repassées sur le perceptron multicouche du test avec les mêmes paramètres enregistrés auparavant, la reconnaissance se fait en appliquant les poids finaux obtenus dans l'apprentissage sur les vecteurs en entrée.

On dernier lieu on affiche les résultats et on affiche les taux de reconnaissance.

6. **Présentation du logiciel et exécution :**
   Après avoir chargé la base de données on choisit les paramètres de notre CNN et qui se résument sous : le stride, pooling, taille de la convolution, taille du pooling et des features maps (carte de caractéristique).

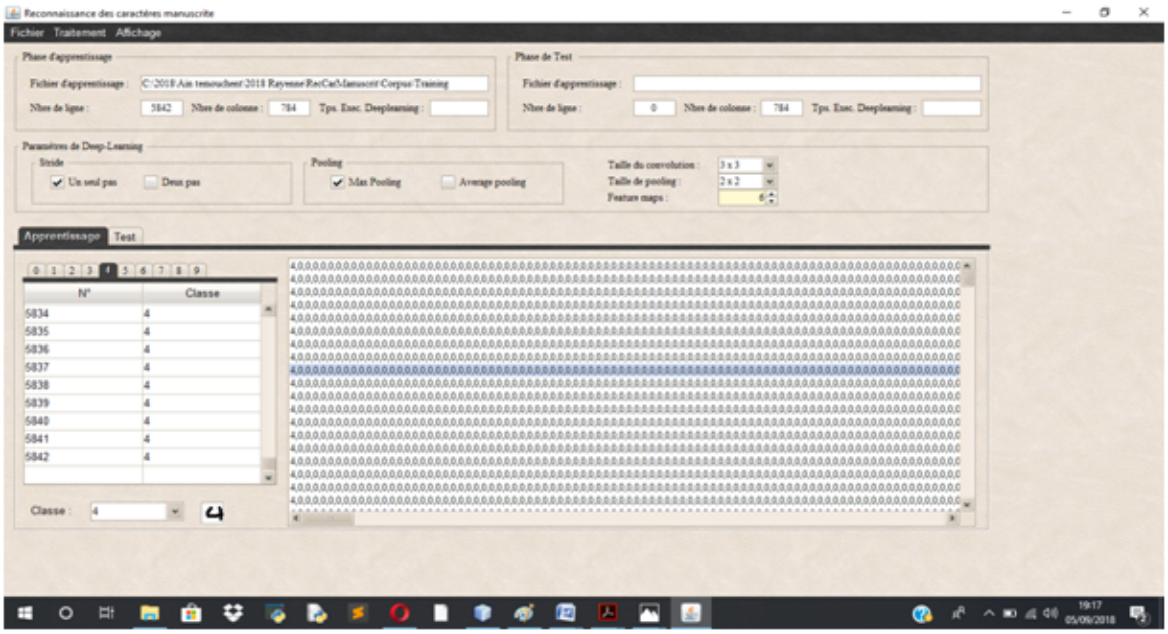


FIGURE 3.16 – Interface apprentissage (choix des paramètres)

On lance la partie CNN qui va permettre d'extraire les caractéristiques de chaque image et de nous préparer le vecteur d'entrée pour la partie MLP.

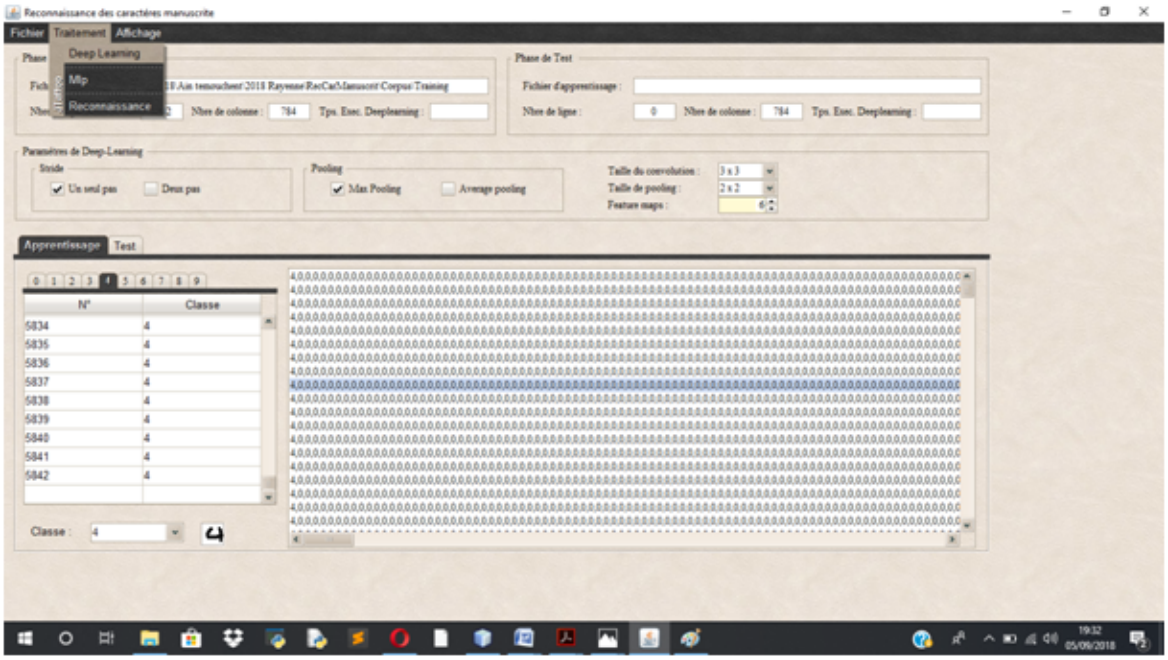


FIGURE 3.17 – Interface apprentissage (lancement CNN)

Maintenant qu'on a comme acquis le vecteur de sortie du CNN qui est de plus le vecteur d'entrée pour le MLP on lance cette dernière partie.

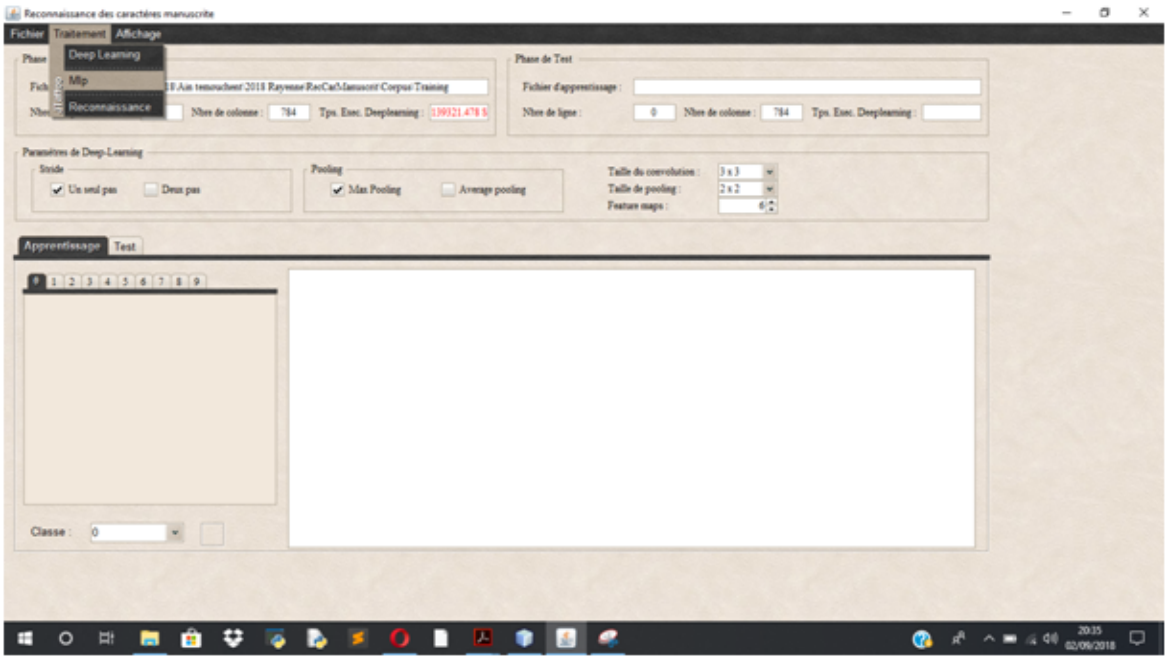


FIGURE 3.18 – Interface apprentissage (lancement MLP)

Une fois l'apprentissage terminé le temps d'exécution de notre MLP s'afficheras en haut droite de l'interface un graphe apparaîtra en bas qui nous illustreras la convergence de l'erreur vers epsilon.

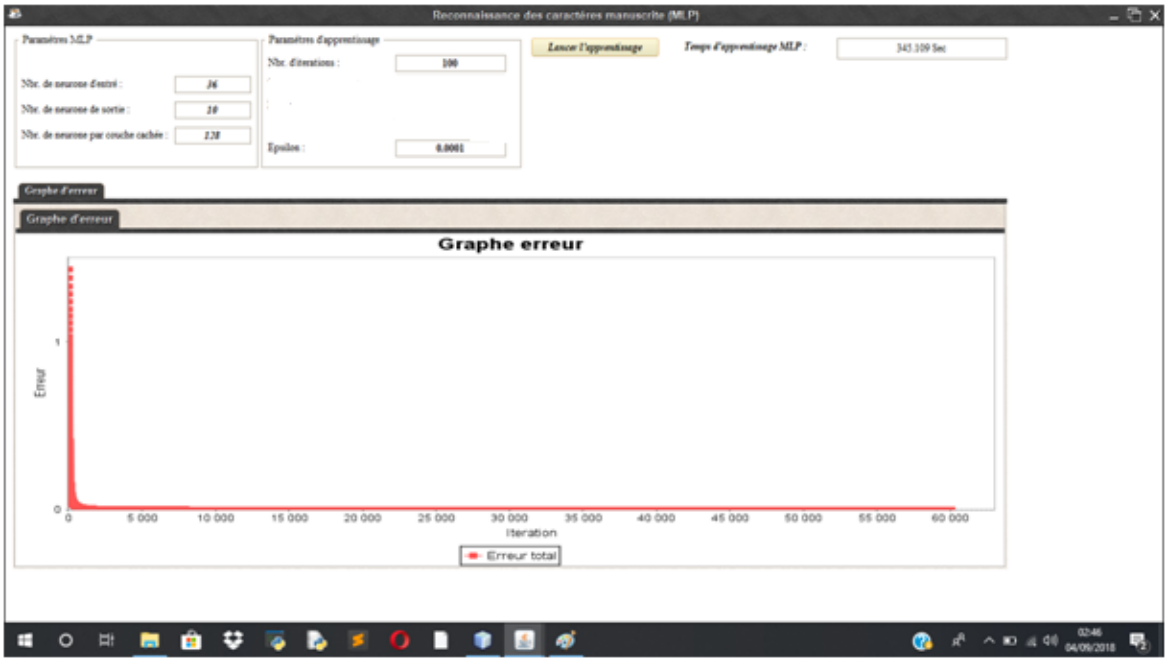


FIGURE 3.19 – Interface test (graphe erreur)

Une fois l'apprentissage terminé on s'attaque à la phase de test en chargeant d'abord la base de données dédiée, et on relance la partie CNN sur cette base avec les mêmes paramètres que ceux configuré précédemment.

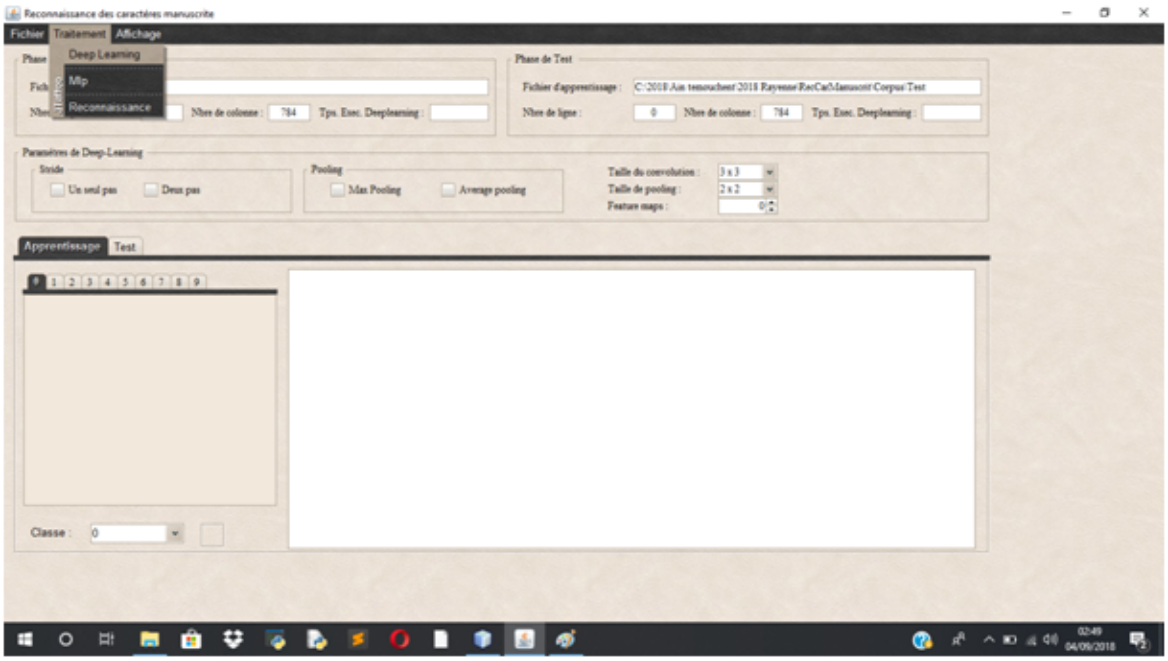


FIGURE 3.20 – Interface apprentissage (lancement CNN test)

Une fois la phase de reconnaissance terminée on aura ainsi le temps et le taux de reconnaissance totale ainsi que pour chaque chiffre un histogramme nous permettra de bien lire ces taux de reconnaissance.

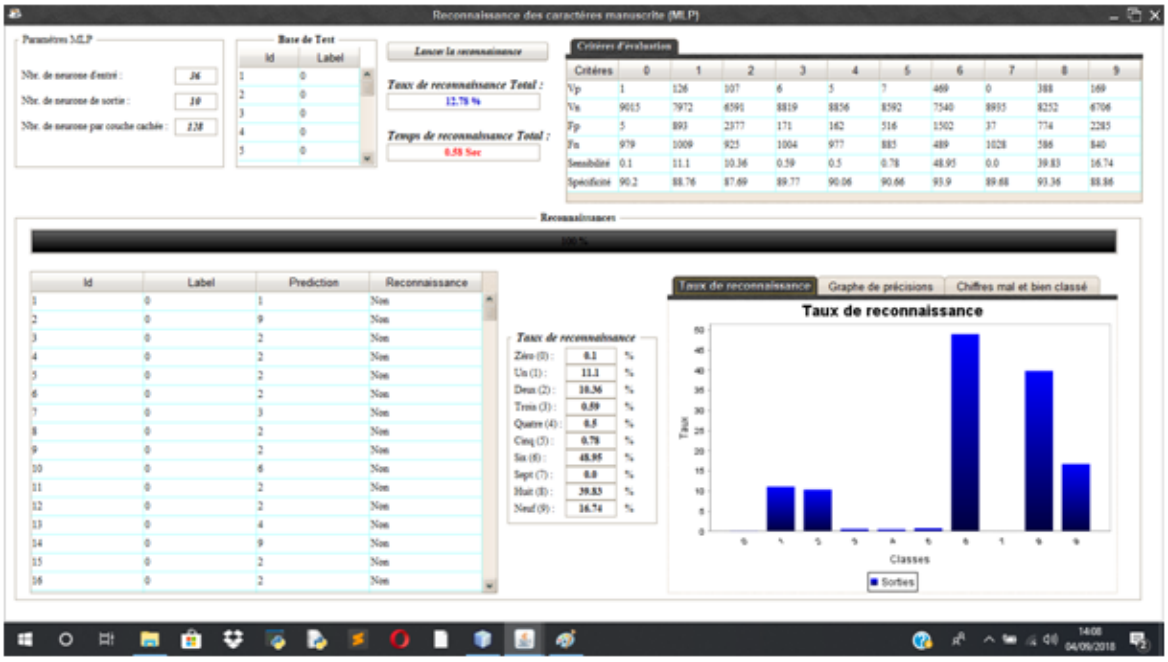


FIGURE 3.21 – Interface test (résultat)

7. **Résultats et discussions :**

Dans cette expérimentation nous effectuons la classification sur notre base de données MNIST qui contient 60000 images. Les paramètres de la partie CNN sont comme suit :

Paramètres	**Valeurs**
Stride.	Un pas
Pooling.	Max pooling
Taille de la convolution.	3*3
Taille du pooling.	2*2
Features maps.	6

TABLE 3.1 – Les paramètres du CNN

Pour ce qui est des paramètres du MLP :

Paramètres	**Valeurs**
Nombre neurone d'entrées.	36
Nombre neurone sortie.	10
Nombre Couche cachée.	1
Nombre de neurones par Couche cachée.	128
Erreur.	0.00001
Nombre d'itérations	100

TABLE 3.2 – Les paramètres du MLP

Le temps d'exécution totale est de 163008.063 s ce qui est égal à 42.28h et qui est un temps largement conséquent dû à la grandeur de la base de données et aux opérations très complexes nécessaires à la réalisation de notre programme.

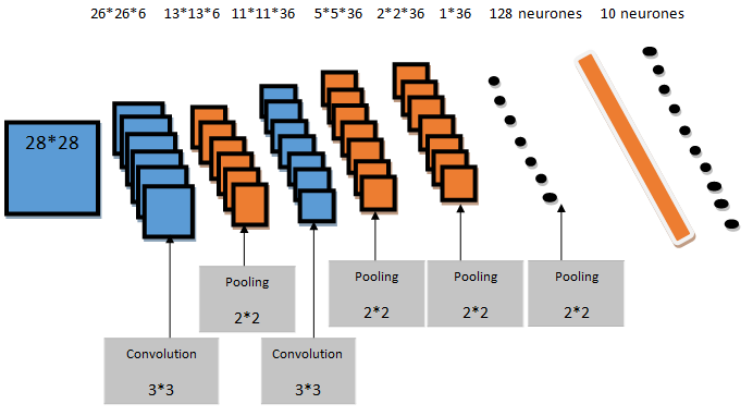


FIGURE 3.22 – Architecture modèle proposé

Quant à la figure ci-dessous qui représente le taux de classification de chaque chiffre, on a un pourcentage qui est assez bas mais très encourageant pour notre première expérience dans le domaine de l'intelligence artificielle et plus précisément dans l'apprentissage profond, encourageant puisqu'il faut noter que la configuration de notre matériel qui ne nous permet pas de viser des taux d'apprentissage élevés, puisque par exemple notre ordinateur ne supporter pas qu'on est plusieurs couches cachées avec des milliers de neurones, notons que 128 neurones dans une seule couche cachée ne favorisent pas une bonne classification .

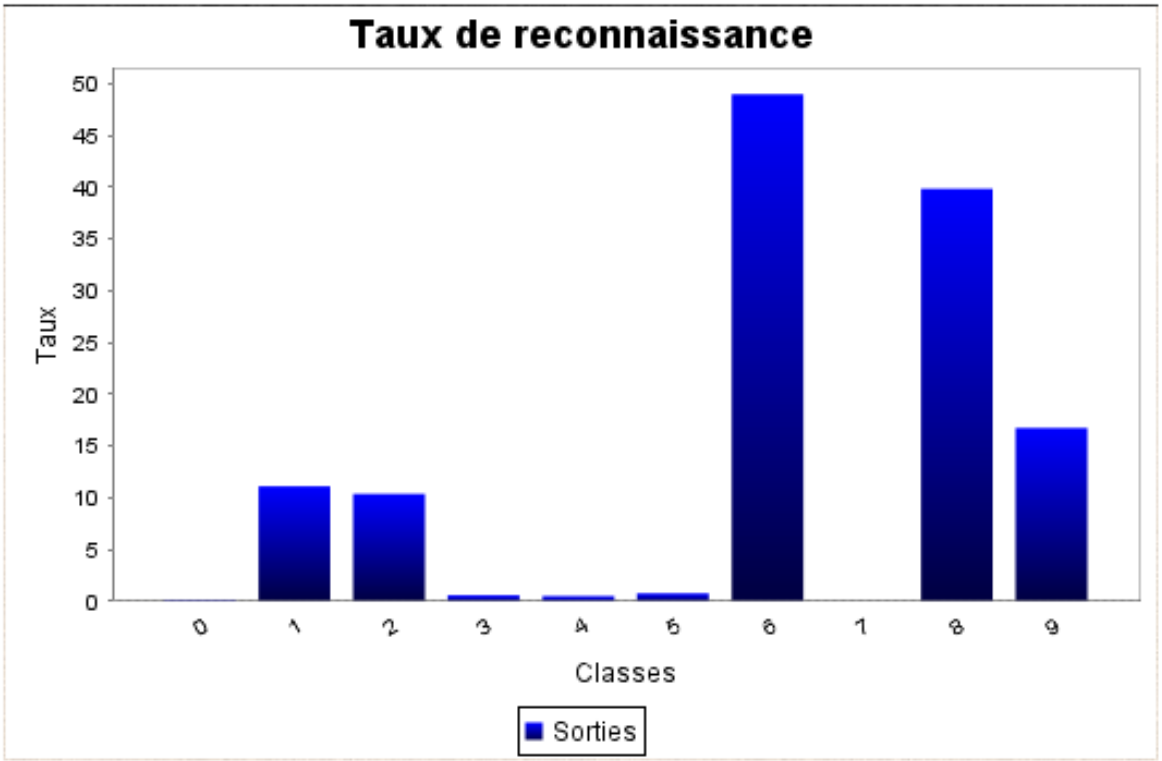


FIGURE 3.23 – Taux de reconnaissance

3.5 Conclusion

Dans ce dernier chapitre nous avons présenté une approche de classification basée sur les réseaux de neurones a convolution, pour cela deux grandes expérimentations on était mené, la première sous python où on a comparé les résultats de trois modèles ou les résultats était excellents, et la deuxième sous java où on a détaillé notre approche, de l'apprentissage profond qui est se compose en deux parties une pour l'extraction des caractéristiques (réseaux de neurones à convolution) et la deuxième pour la classification (perceptron multicouche). Notre approche était testée est valide sur la base de données Mnist qui ont été utilisées pour l'apprentissage (60000) et le test(10000).

En premiers lieux toutes les étapes nécessaires à la conception de notre application et à notre approche ont été détaillées, et par la suite nous avons donné les divers résultats obtenus.

C'est surement un petit pas dans le domaine de l'apprentissage profond mais c'est un très grand pas pour nous, amateur de l'apprentissage profond.

Perspectives de recherche

Nous avons réussi à reconnaitre des caractères manuscrits à l'aide de notre logiciel, mais néanmoins le taux d'apprentissage reste très bas et ce n'est qu'une question de temps et de matérielle et du fait que les informations sur la conception de ces algorithmes sont rares ou introuvables, nous avions quand même célébré ce taux de classification comme une grande victoire et avec une joie immense, puisque l'apprentissage profond est un domaine nouveau, très pointu et très complexe.

Parmi les points qu'on devrait améliorer dans notre logiciel, l'un a un rapport avec la matérielle et l'utilisation des GPU au lieu d'un CPU, et c'est ce point qui a un lien avec tous les autres points, parmi les autres points on devrait augmenter le nombre des couches caché et le nombre de neurones par couches caché, l'utilisation de plusieurs techniques liés à l'apprentissage profond qui sont cité au deuxième chapitre.

Nous espérons aussi que dans une future proche nous pourrions augmenter assez largement le taux de reconnaissance et faire de la reconnaissance des caractères manuscrits en tous genres (en mélangeant, les chiffres l'alphabet arabe ainsi que celle du français) et pour quoi par utiliser la reconnaissance de la voie.

.

Conclusion générale

Conclusion générale

Dans ce projet, nous avons fait un bref tour du domaine de l'intelligence artificielle, où nous avons réussi à expliquer et discuter des notions fondamentales de plusieurs algorithmes d'apprentissage automatique, comme ceux de l'apprentissage profond, tout en se basant sur les réseaux de neurones en général et des réseaux de neurones à convolution en particulier.

Nous avons présenté toutes les étapes essentielles à la réalisation de ces réseaux de neurones, qui est à la pointe des algorithmes pour l'apprentissage profond : la couche de convolution, la couche de pooling et la couche fullyconnected. Nous avons parlé aussi sur les méthodes de régularisation (dropout) utilisées pour éviter le problème de sur-apprentissage.

Les paramètres du réseau sont difficiles à définir a priori, mais compte tenu du temps d'exécution extrêmement long nous n'avons pas pu faire plusieurs expériences, c'était l'un des problèmes rencontrés pendant l'implémentation de notre logiciel, même si ce problème est conséquent il n'en est rien comparé aux autres problèmes qu'on a eu tout au long de la conception.

Bibliographie

Bibliographie :

[Atif 2014] Jamal atif, "Analyse et Fouille de Données",2014

[Bellman 1978] Bellman, Richard. An introduction to artificial intelligence : Can computers think ?. Thomson Course Technology, 1978.

[Bendjeddou 2008] BENDJEDDOU Toufik, "Le choix de paramètre pour la reconnaissance de chiffres manuscrits",2008

[Benmammar et al 2009] Benmammar, Badr. "Intelligence Artificielle et Systèmes Multi-Agents." (2009).

[Boisard] Olivier Boisard,"intelligence artificielle"

[Boukhlouf 2005]Boukhlouf Djemaa,"Résolution de problèmes par écosystèmes : Application au traitement d'images",2005

[Burrow et al 2004]Burrow, Peter. "Arabic handwriting recognition." Report of Master of Science School of Informatics, University of Edinburgh (2004).

[Chabot 2017] Chabot, Florian. Analyse fine 2D/3D de véhicules par réseaux de neurones profonds. Diss. Université Clermont Auvergne, 2017.

[Charniak et al 1985] Charniak, Eugene, and Drew McDermott. "Introduction to AI." Reading (Mass.) : Addison (1985).

[Chafik 2017] Chafik, Sanaa. Machine learning techniques for content-based information retrieval. Diss. Université Paris-Saclay, 2017.

[Cherif 2011] HAMZA CHERIF, Ikram. "Classification des tracés TocoGraphiques (CTG) d'un foetus à l'aide de classifieurs multiples."

[Chesneau 2016] Chesneau, Christophe. "Eléments de classification." (2016).

[Chikh et al 2016] ABA, Safa, and Maroua CHIKH. "Reconnaissance Des Mots Arabes Manuscrites."

[Cleuziou 2004] Cleuziou, Guillaume. Une méthode de classification non-supervisée pour l'apprentissage de règles et la recherche d'information. Diss. Université d'Orléans, 2004.

[Cornuéjols] Antoine. "Apprentissage Supervisé."

[Djeffal 2015]Djeffal, Abdelhamid. "Cours Fouille de données avancée." Université Mohamed Khider Biskra (2015).

[Haton 1989] Haton, Jean-Paul Haton, Jean-Paul, and Marie-Christine Haton. L'intelligence artificielle. Presses universitaires de France, 1989.

[Haugeland 1985] Haugeland, John. "Artificial Intelligence : The Very Idea. 1985." Cited on (1985) : 26.

[Graesser 2016] L. Graesser,"Introduction to Neural Networks", July 26,2016

[Khatir 2012] Nadjia khatir, "Clustering dans les bases de données", 2012

[Kurzweil 1990] Kurzweil, Ray, et al. The age of intelligent machines. Vol. 579. Cambridge, MA : MIT press, 1990.

[Mareef 2013] Marref 2013 Marref, Nadia. Apprentissage Incrémental Machines à Vecteurs Supports. Diss. Université Mustapha Ben Boulaid Batna 2, 2013.

[Megar et al 2016] MEGAR, Ishak, and Adel GEUDDA. "Contribution à la reconnaissance d'écriture arabe manuscrite en utilisant la classification neuronale."

[Mokhtari et al 2018] Système DE Détection D'intrusions Informatiques par Système MultiAgents.

[Naffakhi et al 2005]Faiz, Rim, Najeh Naffakhi, and Khaled Mellouli. "Apprentissage supervisé pour la classification des images basé sur la structure P-tree." EGC. 2005.

[Nemouchi et al 2010] Nemouchi, S., and N. Farah. "Reconnaissance de l'Ecriture Arabe par Systèmes Flous." Département Informatique, Université Badji Mokhtar Laboratoire de Gestion Electronique du Document (LABGED), Algérie (2010).

[Nicolle 1996] Nicolle, Anne. L'expérimentation et l'intelligence artificielle. Intellectica, 1996.

[Nillson 1998] Nils J. NELSON, Proncipales of Artificial Intelligence, Maurgan KAUFMANN , 1980, 471p.

[Parizeau 2004]Parizeau, Marc. "Réseaux de neurones." GIF-21140 et GIF-64326 124 (2004).

[Rakotomalala 2001] RAKOTOMALALA, Ricco. "Quelques approches pour rendre calculable P (Y/X)."

[Rakotomalala 2005] RAKOTOMALALA, Ricco. "Arbres de Décision"(2005).

[Rakotomalala 2013] Ricco Rakotomalala,"Support Vector Machine",2013

[Rich et al] Rich, Elaine, and Kevin Knight. "Articial intelligence." McGraw-Hill, New (1991).

[Rifqi 2002] rifqi, "k plus proche voisins", 2002

[Santos 2015] Santos, Frédéric. "Arbres de décision." (2015).

[Sheeren et al 2007] Sheeren, David, Sébastien Lefèvre, and Jonathan Weber. "La morphologie mathématique binaire pour l'extraction automatique des bâtiments dans les images THRS." Revue internationale de Géomatique 17.3–4 (2007).

[Taffar 2012] http ://elearning.univ-jijel.dz/elearning/pluginfile.php/4333/mod$_r$*esource/content*/1/*Su*

[Vincent 2017] Vincent, Adrien F. Vers une utilisation synaptique de composants mémoires innovants pour l'électronique neuro-inspirée. Diss. Université Paris-Saclay, 2017.

[Vojt 2016] Bc. Ján Vojt, "Deep neural networks and their implementation",2016

[Winston 1992] Winston, P. "Learning by building identification trees." Artificial intelligence (1992) : 423-442.

Webographie

Webographie :

[Capponie] Consulté le 27/03/2018 http ://docplayer.fr/21798218-Arbres-de-decision-m2-mass-cecile-capponi-cecile-capponi-lif-univ-mrs-fr-universite-aix-marseille.html.

[Goodfellow et al 2016] Consulté le 20/03/2018 http ://www.deeplearningbook.org/

[Keras] Consulté le 20/08/2018 https ://en.wikipedia.org/wiki/Keras

[Laperrière 2012] Consulté le 23/03/2018 http ://proklam.com/2012/09/28/de-bonnes-resolutions-pour-les-images-de-vos-presentations/

[Lecun et al] Consulté le 12/02/2018 http ://yann.lecun.com/exdb/mnist/

[Ludovic 2016] Consulté le 02/04/2018 https ://siecledigital.fr/2016/12/22/machine-learning-deep-learning-ca-marche/.

[Python] Consulté le 20/08/2018 https ://fr.wikipedia.org/wiki/Python$_{(langage)}$

[TensorFlow] Consulté le 20/08/2018 https ://www.techopedia.com/definition/32862/tensorflow]

[Wassim 2012] Consulté le 27/03/2018 https ://fr.slideshare.net/wassimlahbibi/algorithme-knn

Annexe 1

Définitions :

TensorFlow : [TensorFlow]

TensorFlow est est un framework de programmation axée sur l'apprentissage automatique créée par Google, TensorFlow a été initialement développé par des ingénieurs et des chercheurs de l'équipe Google Brain, principalement pour un usage interne, il a été rendu Open Source par Google en Novembre 2015. Depuis ce temps il s'est imposé comme etant le frameworks le plus populaire et le plus utilisé pour le deep learning.

TensorFlow est considéré comme le successeur de l'application fermée DistBelief et est actuellement utilisé par Google à des fins de recherche et de production. TensorFlow est considéré comme la première mise en œuvre sérieuse d'un cadre axé sur l'apprentissage en profondeur. TensorFlow est également connu sous le nom de Google TensorFlow.

TensorFlow tire son nom des tableaux multidimensionnels connus sous le nom de tenseurs, qui sont utilisés par les réseaux de neurones pour différentes opérations, Un Tensor à deux dimensions est l'équivalent d'une matrice. Selon Google, par rapport à DistBelief, TensorFlow est plus rapide, plus intelligent et plus flexible et peut facilement être adapté à de nouveaux domaines et produits. Il a été principalement créé pour la recherche sur les réseaux neuronaux profonds et pour faciliter l'apprentissage automatique, bien que TensorFlow ait été utilisé dans un large éventail d'autres domaines.

TensorFlow est disponible sur différents systèmes d'exploitation tels que Linux, Windows, MacOS et également sur des plateformes d'exploitation mobiles comme iOS et Android. L'une des principales caractéristiques de TensorFlow est qu'il est capable de fonctionner sur plusieurs CPU et GPU. Les calculs dans TensorFlow sont signalés en tant que graphes de flux de données avec état. Actuellement, TensorFlow est utilisé dans plus de six mille dépôts en ligne gratuits.

KERAS : [keras]

Keras est une bibliothèque de réseau neuronal open source écrite en Python. Il est capable de fonctionner sur TensorFlow, Microsoft Cognitive Toolkit, Theano ou MXNet. Il a été développé dans le but de permettre une expérimentation rapide avec les réseaux neuronaux profonds, Pouvoir passer de l'idée au résultat avec le moins de retard possible est la clé pour faire de bonnes recherches.

IL se concentre sur l'ergonomie, la modularité et l'extensibilité. Il a été développé dans le cadre de l'effort de recherche du projet ONEIROS (Système d'Exploitation de Robots Intelligents Neuroélectroniques Ouverts), et son principal auteur et mainteneur est François Chollet, un ingénieur de Google.

En 2017, l'équipe TensorFlow de Google a décidé de soutenir Keras dans la bibliothèque principale de TensorFlow. Chollet a expliqué que Keras a été conçu pour être une interface plutôt qu'un cadre autonome d'apprentissage automatique. Il offre un ensemble d'abstractions de niveau supérieur et plus intuitif qui facilite le développement de modèles d'apprentissage en profondeur, quel que soit le système de calcul utilisé.

Python : [Python]

Python est un langage de programmation objet, multi-paradigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions ; il est ainsi similaire à Perl, Ruby, Scheme, Smalltalk et Tcl.

Le langage Python est placé sous une licence libre proche de la licence BSD4 et fonctionne sur la plupart des plates-formes informatiques, des supercalculateurs aux ordinateurs centraux5, de Windows à Unix avec notamment GNU/Linux en passant par macOS, ou encore Android, iOS, et aussi avec Java ou encore .NET. Il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser.

Il est également apprécié par certains pédagogues qui y trouvent un langage où la syntaxe, clairement séparée des mécanismes de bas niveau, permet une initiation aisée aux concepts de base de la programmation6. Python est un langage simple, facile à apprendre et permet une bonne réduction du cout de la maintenance des codes. Les bibliothèques (packages) python encouragent la modularité et la réutilisabilité des codes. Python et ses bibliothèques sont disponibles sans charges pour la majorité des plateformes et peuvent être redistribués gratuitement.

NetBeans :

NetBeans est un environnement de développement intégré (IDE) open-source pour le développement avec Java, PHP, C ++ et d'autres langages de programmation. NetBeans est également appelé une plate-forme de composants modulaires utilisés pour développer des applications de bureau Java.

WampServer :

WampServer fait référence à une pile logicielle pour le système d'exploitation Microsoft Windows, créée par Romain Bourdon et composée du serveur web Apache, du support OpenSSL pour SSL, de la base de données MySQL et du langage de programmation PHP.

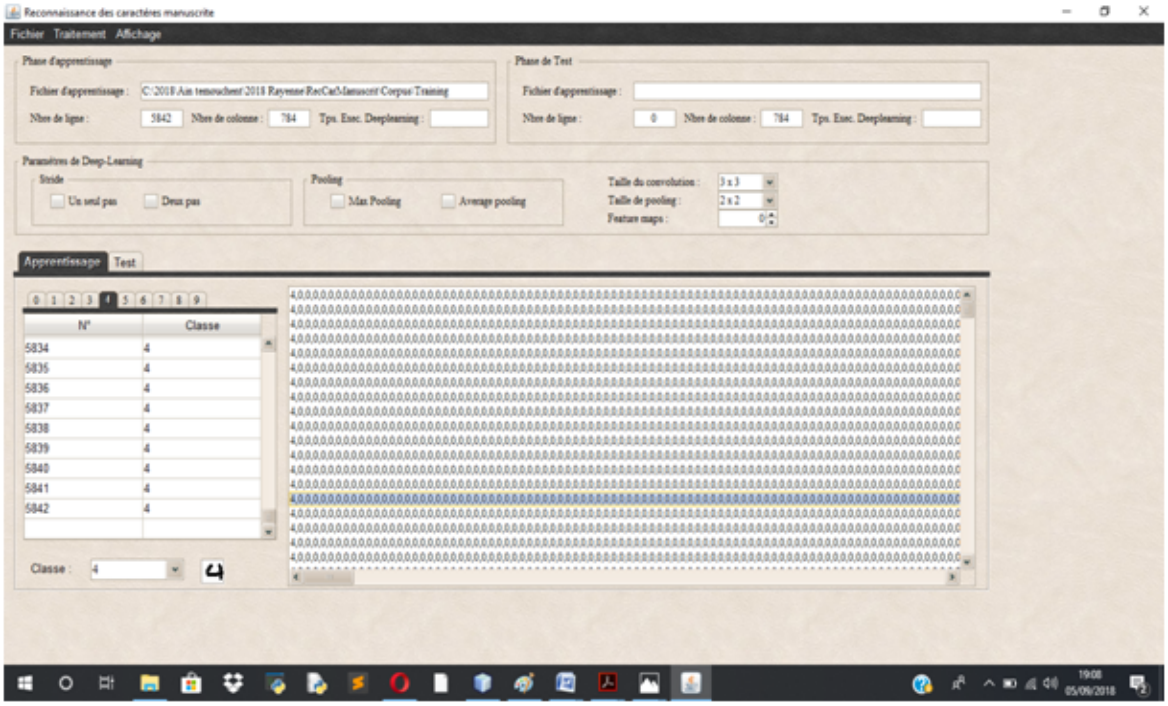
Materiel :

La configuration du matériel utilisé dans notre implémentation est :
— Un PC portable HP Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.70GHz
— Carte graphique AMD Radeon(TM) R5 M430 , Intel(R) HD Graphics 620
— RAM de taille 4 GO
— Disque dur de taille 1 TO
— Système d'exploitation Windows 10 64 bits

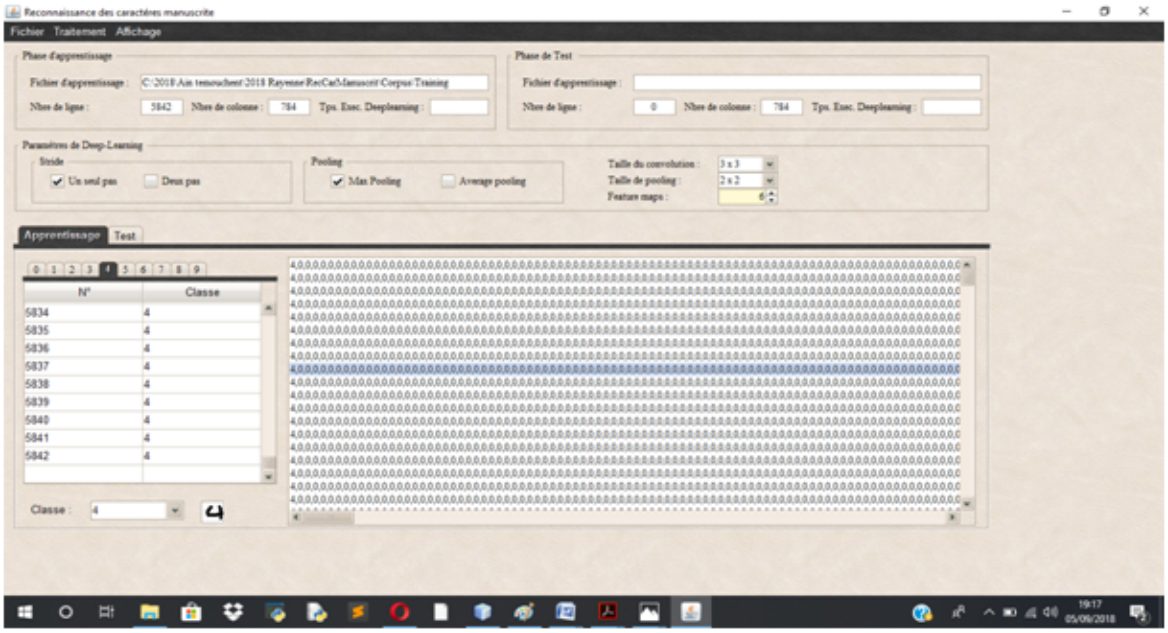
Annexe 2

Dans cette partie nous allons présenter les différentes étapes du logiciel conçu via le langage java sous l'environnement de développement NetBeans.

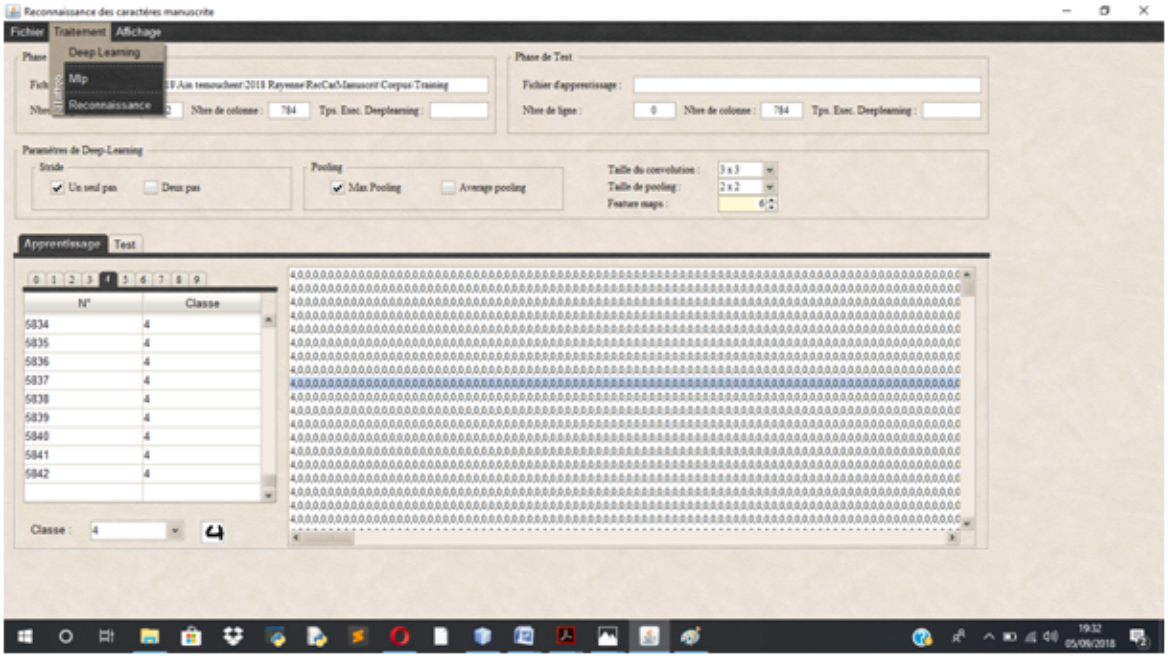
Pour l'exécution de l'application conçue il faut tout d'abord charger la base de données de notre apprentissage qui contient 60000 images, ou chaque image est représentée par un vecteur de taille 784.



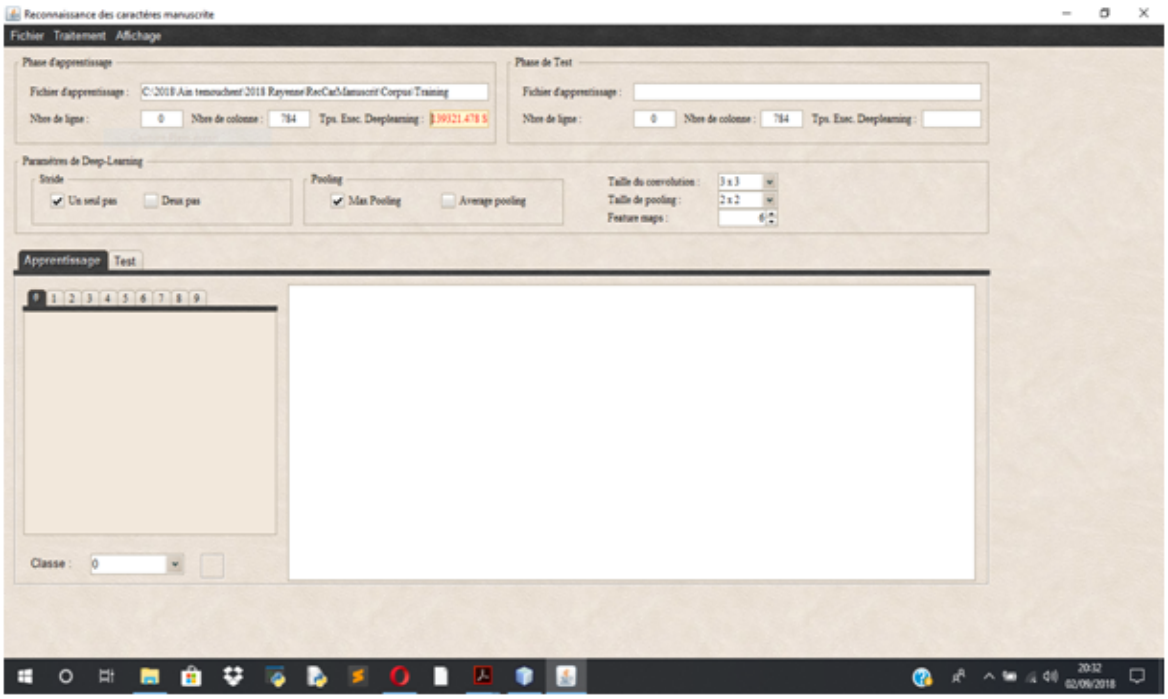
Après avoir chargé la base de données on choisit les paramètres de notre CNN et qui se résument sous : le stride, pooling, taille de la convolution, taille du pooling et des features maps (carte de caractéristique).



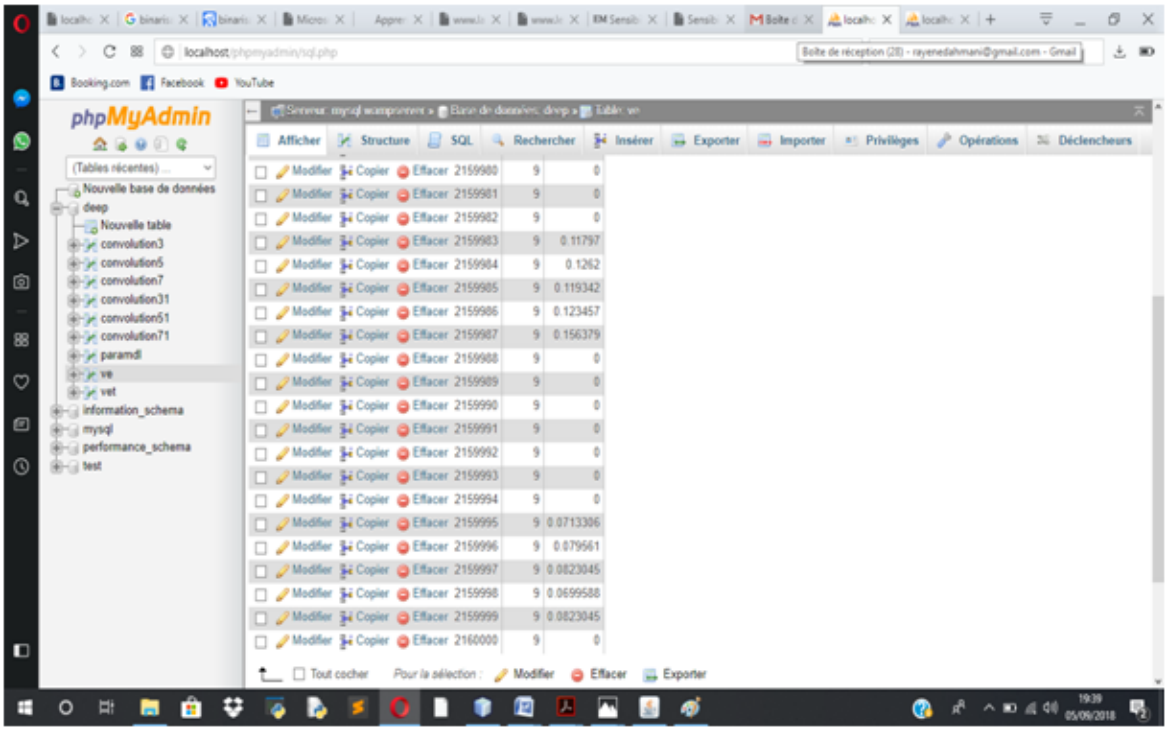
On lance la partie CNN qui va permettre d'extraire les caractéristiques de chaque image et de nous préparer le vecteur d'entrée pour la partie MLP.



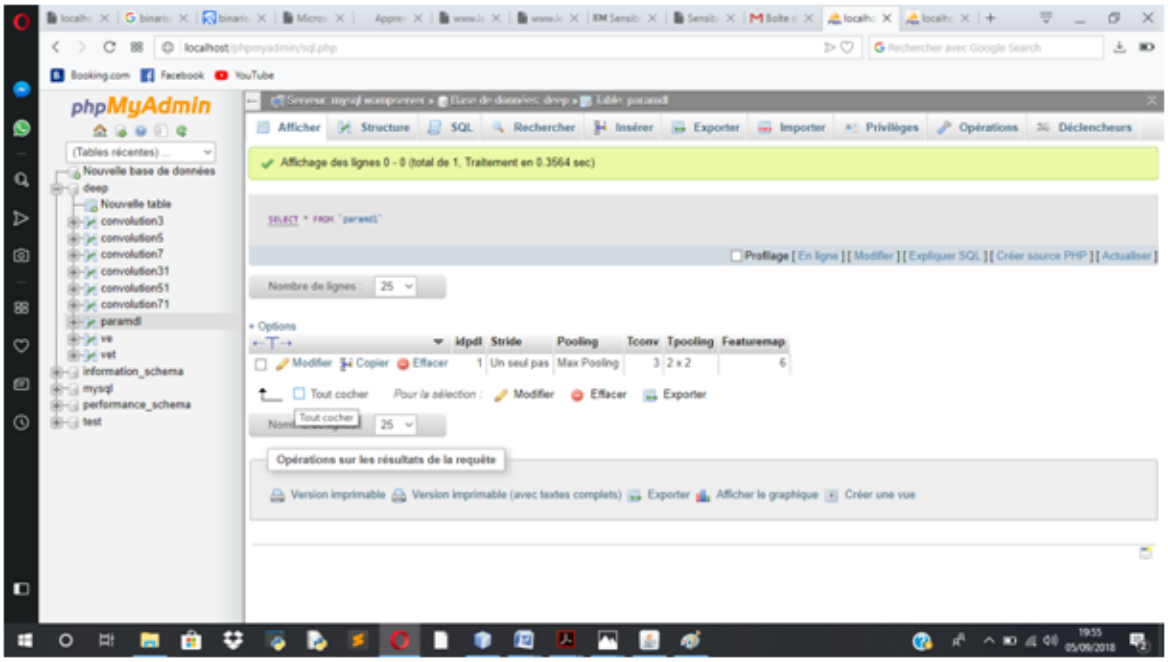
Les figures suivantes nous donnent le temps d'exécution en seconde de la partie CNN ainsi qu'un aperçue de la base de données contenant le vecteur de sortie.



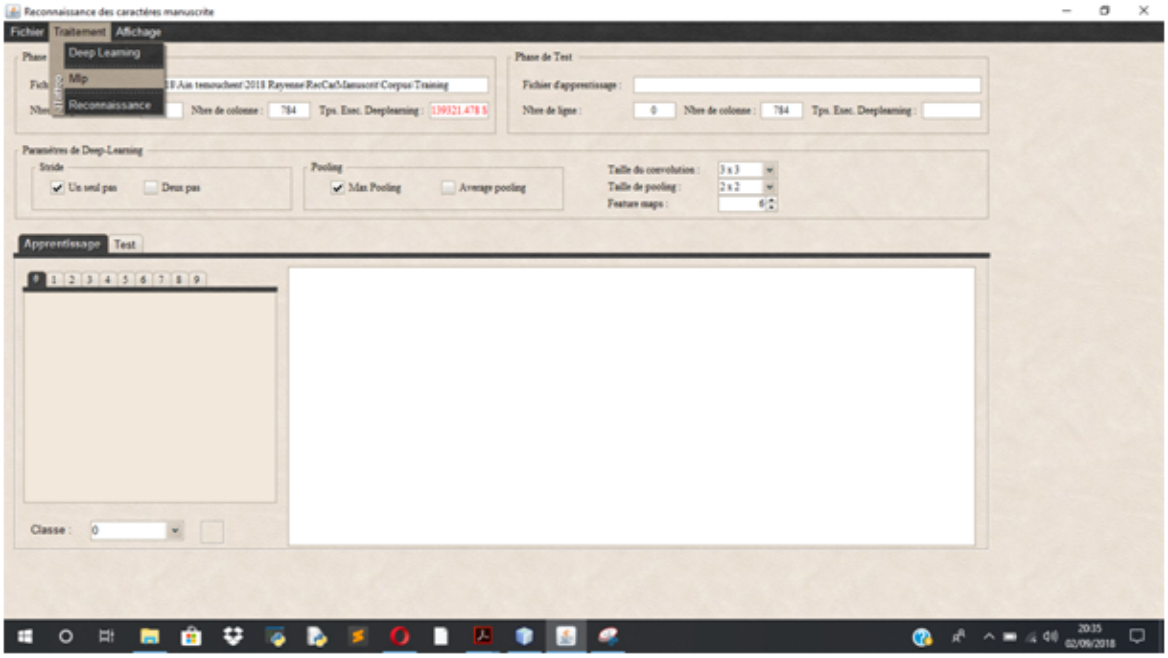
Le vecteur de sortie du CNN fait un totale de 2160000 valeurs.



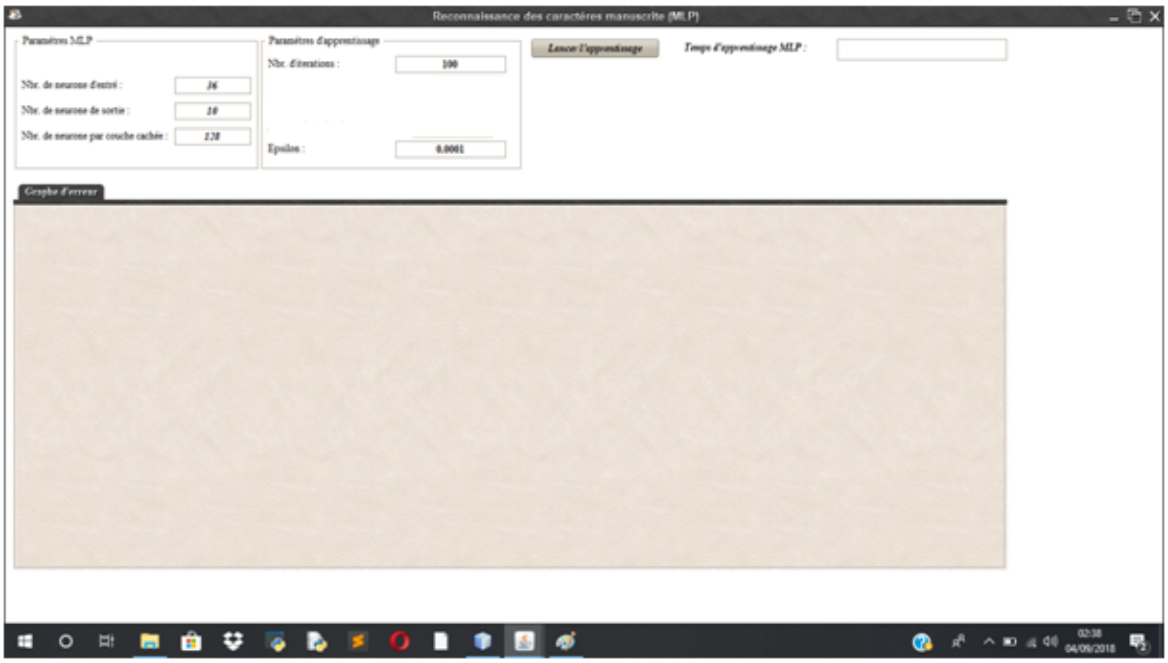
Un aperçue de la base de données où sont stockés les paramètres de notre CNN pour les réutiliser dans la phase de test.



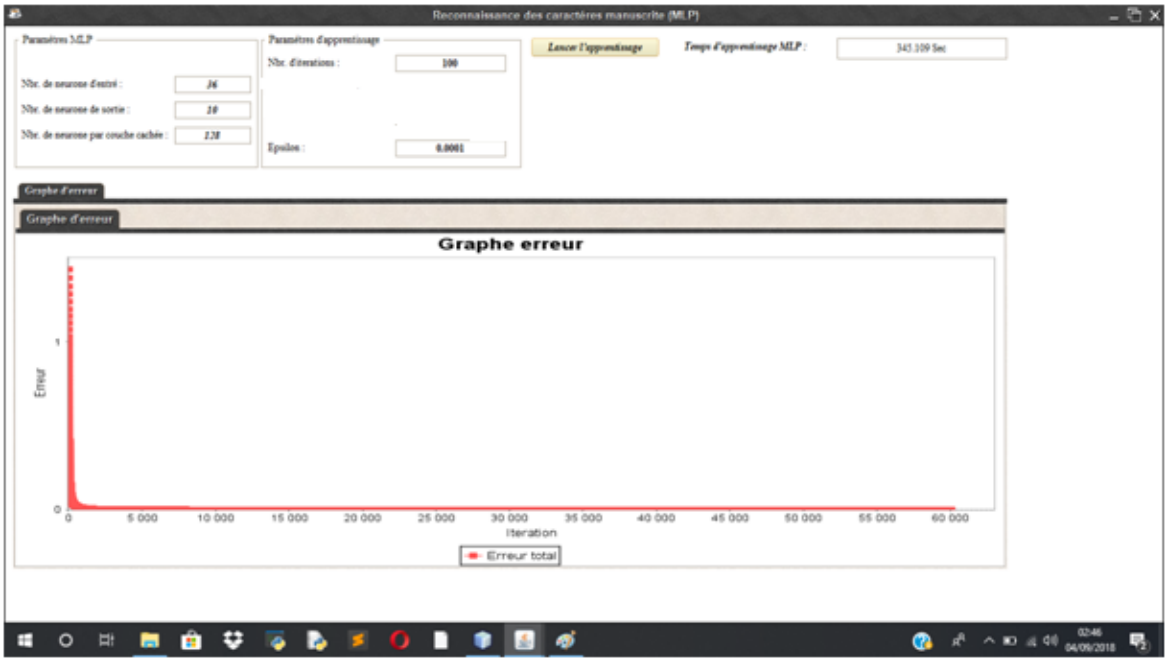
Maintenant qu'on a comme acquis le vecteur de sortie du CNN qui est de plus le vecteur d'entrée pour le MLP on lance cette dernière partie.



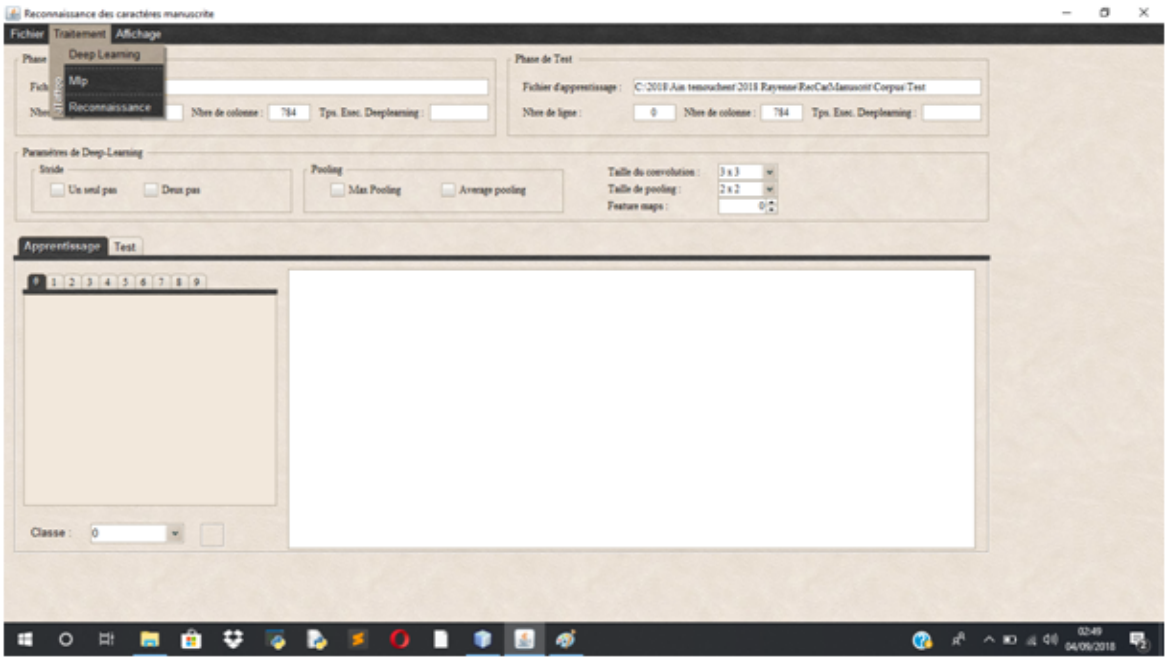
Notre écran basculera sur une nouvelle fenêtre, une interface spécialement conçue pour notre MLP où on peut configurer le nombre d'itération et le seuil epsilon.



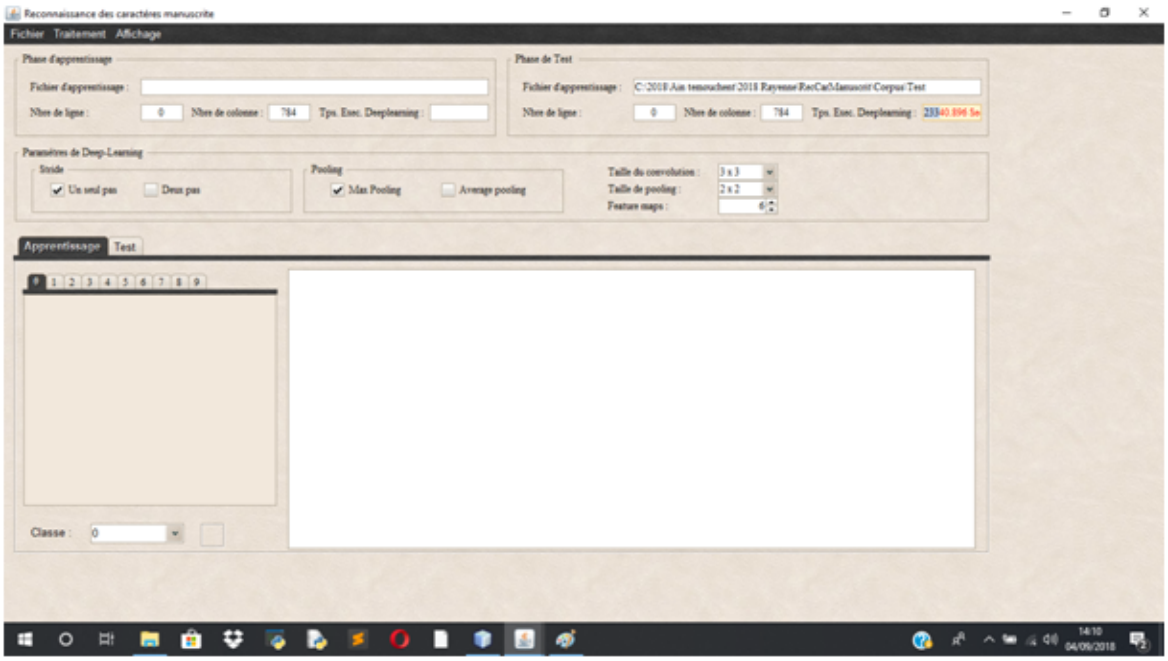
Une fois l'apprentissage terminé le temps d'exécution de notre MLP s'afficheras en haut droite de l'interface un graphe apparaîtra en bas qui nous illustreras la convergence de l'erreur vers epsilon.



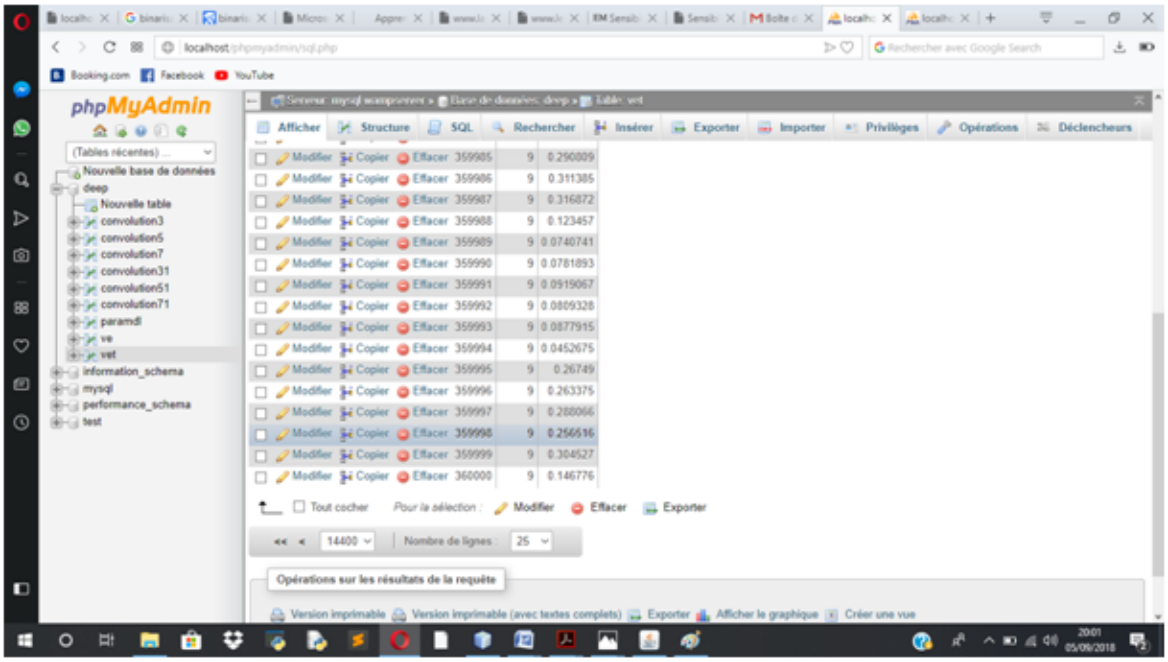
Une fois l'apprentissage terminé on s'attaque à la phase de test en chargeant d'abord la base de données dédiée, et on relance la partie CNN sur cette base avec les mêmes paramètres que ceux configuré précédemment.



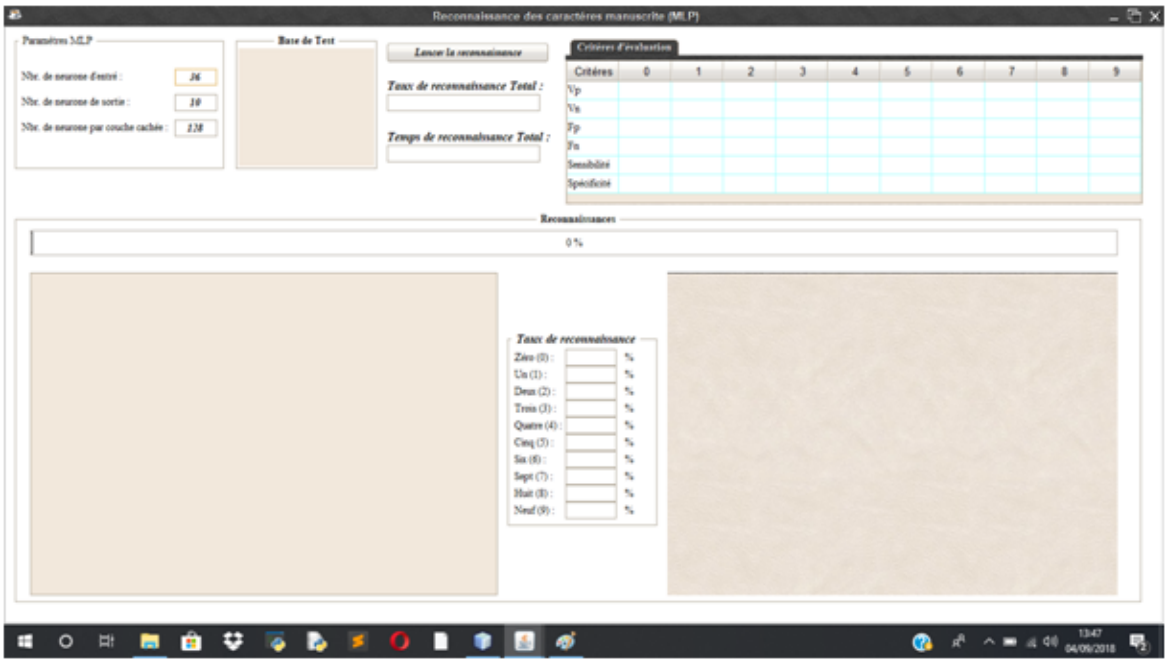
La partie CNN du test terminé le temps d'exécutions apparaîtra en rouge en haut.



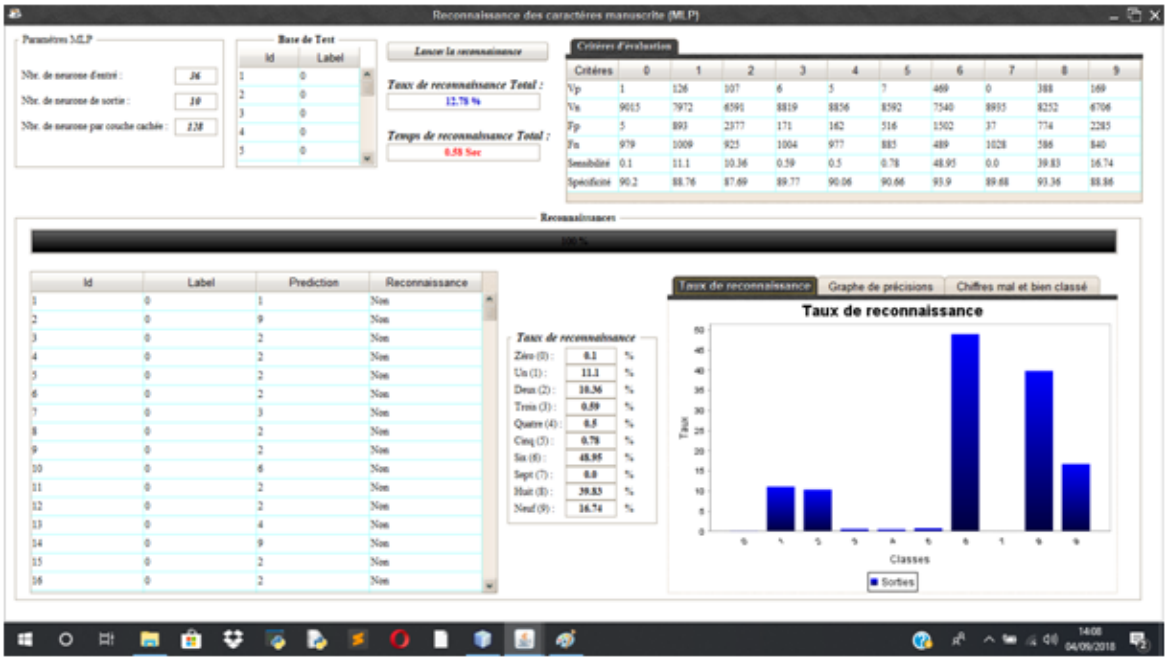
Un aperçu de la base de données où on a enregistré le vecteur de sortie (360000 valeurs) du CNN et qui est le vecteur d'entrée de notre MLP du test.



On basculera sur une autre interface dé qu'on aura lancé la partie reconnaissance.



Une fois la phase de reconnaissance terminée on aura ainsi le temps et le taux de reconnaissance totale ainsi que pour chaque chiffre un histogramme nous permettra de bien lire ces taux de reconnaissance.



Le graphe de précision.