الجمهورية الجزائرية الديمقراطية الشعبية
République algérienne démocratique et populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique
جامعة عين تموشنت بلحاج بوشعيب
Université –Ain Temouchent- Belhadj Bouchaib
Faculté des Sciences et de Technologie
Département Mathématiques et Informatique

Projet de Fin d'Etudes
Pour l'obtention du diplôme de Master en Informatique
Domaine : Mathématique et Informatique
Filière : Informatique
Spécialité : Réseaux et Ingénierie des Données

Thème

Développement d'un système de reconnaissance automatique des chiffres manuscrits isolé

Présenté par:

1) **Melle Zahira MAROC**.

Devant le jury composé de :

Dr Ali BENZERBADJ	UAT.B.B (Ain Temouchent)	Président
Dr Imad BENARIBI	UAT.B.B (Ain Temouchent)	Examinateur
Dr Fatima Zahra BERRAKEM	UAT.B.B (Ain Temouchent)	Encadreur

Année Universitaire 2022/2023


بسم الله الرحمن الرحيم
١٤٢٠

Dédicaces

Avant tout, Je remercie *ALLAH* qui m'aide

Je dédie ce modeste travail à mes très chers parents *MOHAMED* et *NADJIA* qui n'ont pas cessé de m'encourager durant toutes mes études, *de me soutenir et de m'épauler pour que je puisse atteindre mes objectifs,*

Et que dieu me les garde ;

A ma chère grand-mère SALIMA,

A mon cher grand-père MOKHTAR,

Qui n'ont jamais cessé, de formuler des prières à mon égard

Je vous souhaite une bonne santé.

À mes très chers frères *TAYEB, KACEM Ceux qui ont partagé avec moi tous les moments d'émotion lors de la réalisation de ce travail. Ils m'ont chaleureusement supporté et encouragé tout au long de mon parcours ;*

***A** ma famille, mes proches et à ceux qui me donnent de l'amour et de la vivacité.*

***A** tous mes amis qui m'ont toujours encouragé, et à qui je souhaite plus de succès.*

***A** tous ceux que j'aime.*

Zahira Maroc

Remerciement

Grand merci à notre encadrante Mme Fatima Zahra BERRAKEM, pour l'orientation,

La patience qui a constitué un apport considérable.

Nos vifs remerciements vont également aux membres du jury

Dr Ali BENZERBADJ *et* **Dr Imad BENARIBI**

Pour l'intérêt qu'ils ont porté à notre travail en acceptant d'examiner ce mémoire.

A tous les enseignants du département d'informatique.

Résumé

La reconnaissance des chiffres manuscrits est un problème clé dans le domaine de la reconnaissance d'écriture et de l'apprentissage automatique depuis plus de trois décennies.

L'utilisation des techniques de l'intelligence artificielle et l'apprentissage profond permet de développer des systèmes de reconnaissances des chiffres manuscrits capables de classer et d'identifier avec précision les chiffres écrits par différentes personnes.

L'objectif de notre travail est de mettre en place un système de reconnaissance automatiques des chiffes manuscrits isolés en utilisant l'apprentissage profond basé sur le modèle de réseau neuronal convolutif (CNN) afin de reconnaître les images de la base de données MNIST.

Après avoir entraîné et validé le modèle CNN sur l'ensemble de données prétraité, nous avons évalué les performances et les capacités de notre système en le testant sur un échantillon de données de tests. Nous avons obtenu de meilleurs résultats, atteignant environ 99,39% de précision avec un taux d'erreur négligeable.

Mots clés : Intelligence Artificielle (IA), Reconnaissance automatique des chiffres manuscrits, Réseaux de Neurones Convolutives (CNNs), Apprentissage en profondeur, apprentissage automatique, rembourrage (Padding), Tensorflow, Flask.

Abstract

The recognition of handwritten digits has been a key problem in the field of handwriting recognition and machine learning for more than three decades.

The use of artificial intelligence techniques and deep learning makes it possible to develop handwritten digit recognition systems capable of classifying and accurately identifying digits written by different people.

The objective of our work is to set up an automatic recognition system for isolated handwritten ciphers using deep learning based on the convolutional neural network (CNN) model in order to recognize images from the MNIST database.

After having trained and validated the CNN model on the preprocessed dataset, we evaluated the performance and capabilities of our system by testing it on a sample of test data. We achieved better results, achieving approximately 99.39% accuracy with a negligible error rate.

Key words: Artificial Intelligence (AI), Automatic recognition of handwritten digits, Convolutional Neural Networks (CNNs), Deep learning, Machine learning, Padding, Tensorflow, Flask.

ملخص

كان التعرف على الأرقام المكتوبة بخط اليد مشكلة رئيسية في مجال التعرف على خط اليد والتعلم الآلي لأكثر من ثلاثة عقود

إن استخدام تقنيات الذكاء الاصطناعي والتعلم العميق يجعل من الممكن تطوير أنظمة التعرف على الأرقام المكتوبة بخط اليد القادرة على تصنيف الأرقام التي كتبها أشخاص مختلفون وتحديدها بدقة.

الهدف من عملنا هو إعداد نظام التعرف التلقائي للأصفار المكتوبة بخط اليد المعزولة باستخدام التعلم العميق القائم على نموذج الشبكة العصبية التلافيفية (سي إن إن) من أجل التعرف على الصور من قاعدة بيانات .

بعد تدريب نموذج سي إن إن والتحقق من صحته على مجموعة البيانات المعالجة مسبقا ، قمنا بتقييم أداء وقدرات نظامنا من خلال اختباره على عينة من بيانات الاختبار. لقد حققنا نتائج أفضل ، وحققنا دقة 99.39 ٪ تقريبا مع معدل خطأ ضئيل

الكلمات الرئيسية: الذكاء الاصطناعي(IA)، التعرف التلقائي على الأرقام المكتوبة بخط اليد ، الشبكات العصبية التلافيفية (CNNs) ، التعلم العميق ، التعلم الآلي ، الحشو ، تنسورفلو ، فلاسك.

Table de matières

Liste des abréviations

IA :	Intelligence Artificiel.
DP :	Deep learning.
GPU :	Graphical Processing Unit.
SRCM :	Systéme de reconnaissance des chiffres manuscrits.
TPU :	Tensor Processing Unit.
VSC :	Visual Studio Code.
TF :	TensorFlow.
PIL :	Python Imaging Library.
ANN :	Artificial Neural Network.
CNN :	Convolutional Neural Network.
MLP :	Multi-Layer Perceptron.
AG :	Algorithme génétique.
OCR :	Optical Character Recognition.
RBP :	Fonction de base radiale.
SVM :	Support Vector Machine.
HMM :	Hidden Markov Models.
LSTM :	Long-Short Term Memory.
GRU :	Gated Recurrent Unit.
RNN :	Recurrent Neural Network.
CART :	Classification And Regression Trees.
MNIST :	Modified National Institute of Standards and Technology.

Liste de tableaux

Liste des figures

Introduction Générale

La technique d'interprétation et d'identification des chiffres manuscrits est connu sous le nom de reconnaissance des chiffres manuscrits isolé. Dans les domaines de l'intelligence artificiel et l'apprentissage automatique, la reconnaissance des chiffres manuscrits est un défi courant et l'une des tâches les plus étudiées qui implique le processus d'identification des chiffres écrits à la main et de les transformer en format lisible par la machine.

Le terme "chiffres manuscrits" décrit les caractères numériques manuscrits (0-9). Les gens écrivent généralement ces chiffres avec un crayon, un stylo ou tout autre outil d'écriture. Les chiffres manuscrits peuvent prendre une variété des styles d'écritures, des différentes tailles et formes ce qui rend difficile pour les machines de les identifier.

Les systèmes de reconnaissance des chiffres manuscrits isolé (SRCMI) sont utilisés dans divers domaines telle que le secteur bancaire et financier pour le traitement des chèques, l'éducation, la lecture des numéros de compte, la vérification des signatures et la reconnaissance des codes postaux nécessitent une reconnaissance fiable des chiffres manuscrits.

L'une des approches les plus appliquées à l'apprentissage automatique pour la reconnaissance est l'apprentissage supervisé. Un modèle d'apprentissage automatique est entrainé à l'aide d'un ensemble de données étiquetées. Les systèmes de reconnaissance des chiffres manuscrits basé sur l'apprentissage profond sont un traitement avancé de données qui permet à une machine d'apprendre de manière autonome à effectuer des tâches complexes, ils sont basés sur les réseaux de neurones, qui sont un type d'algorithme inspiré de la fonction et la structure du cerveau humain.

Les réseaux de neurones convolutifs (CNNs) sont un type de réseau de neurones artificiel, nécessite plusieurs couches telle que les couches convolutives, les couches de regroupements et des couches entièrement connectées. Chaque couche applique des méthodes et des techniques pour l'extraction des caractéristiques et l'amélioration des performance du modèle, ils sont capables d'apprendre automatiquement les caractéristiques et les représentations hiérarchiques et faire des prédictions précises à partir des valeurs de pixels des images d'entrée. Ils ont montré des performances exceptionnelles dans les tâches de reconnaissance et la classification des images ces dernières années car ils ont été créés en particulier pour interpréter l'entrée visuelle.

La reconnaissance des chiffres manuscrits est une partie du problème de reconnaissance de l'écriture, elle est une tâche difficile car elle nécessite l'analyse et le traitement des images, la correction d'erreurs et la classification.

Dans ce projet, nous nous intéresserons d'étudier et de mettre en œuvre le système de reconnaissance automatique hors ligne des chiffres manuscrits isolé, en utilisant des techniques de traitement d'images et l'apprentissage profond pour identifier et classifier les images en dix classes de (0 à 9).

Nous avons proposé une stratégie de modélisation et l'implémentation d'un système de reconnaissance basé sur le modèle « Convolutional Neural Networks- CNNs », notre modèle CNNs a de trois couches de convolutions, avec quatre couches de batch normalisation, trois couches de regroupement (Max pooling), une couche Flatten et deux couches entièrement

connecté (dense), chaque couche a des filtres de tailles différent à l'autres qui servent à augmenter et donne meilleurs taux de reconnaissance. Nous avons choisi les meilleures architectures du CNNs pour mettre en œuvre et nous évaluerons la performance sur des jeux de données publiques largement utilisés MNIST.

L'objet de ce mémoire est de développer un système de reconnaissance des chiffres manuscrits isolé plus efficace, plus fiable et robuste pour une utilisation dans des applications réelles. Le travail est organisé en trois chapitres structurés comme suit :

Dans Le premier chapitre, nous présentons dans ce chapitre une description générale des différents aspects de la reconnaissance de l'écriture manuscrite et un schéma standard pour la réalisation d'un système d'identification.

Dans le deuxième chapitre, nous parlerons de l'apprentissage profond et les types de réseaux de neurones artificiels, qui ont été utilisées pour la reconnaissance de l'écriture manuscrite. Nous détaillerons les réseaux neurones convolutifs (CNNs) et ces architectures comme AlexNet, VGGnet…. et comment ils ont aidé à améliorer la conception d'un réseau neuronal convolutif.

Le dernier chapitre, présente la conception du notre modèle et les autres architectures, il exposera les résultats obtenus de chaque modèle pour la validation et l'évaluation, et nous allons réaliser le système de reconnaissance des chiffres manuscrits sous forme d'une application WEB et nous allons présenter et discuter les résultats obtenus.

Le travail se termine par une conclusion générale et des perspectives qui montreront quelques travaux futurs.

Chapitre 01

Généralités sur la reconnaissance de l'écriture

1.1 Introduction

L'une des tâches les plus populaires et les plus étudiées dans le domaine de l'apprentissage automatique est la reconnaissance des chiffres manuscrits. Cette tâche implique l'apprentissage d'un modèle pour reconnaître les chiffres qui ont été écrits à la main.

Nous présentons dans ce chapitre une description générale des différents aspects de la reconnaissance de l'écriture manuscrite. En fait, dans un premier temps, nous présentons un ensemble de concepts préliminaires qui introduisent le domaine. Ensuite, nous décrivons le système de reconnaissance des chiffres manuscrite à travers un schéma qui explique les phases du système. Ainsi que, on parle sur les différents techniques et les méthodes pour la reconnaissance. Nous donnons ensuite des exemples de domaines d'application pour la reconnaissance de l'écriture manuscrite. Enfin, ce chapitre en présentant quelques travaux connexes.

1.2 La reconnaissance automatique de l'écriture

L'écriture a reçu beaucoup d'attention à partir du moment où les êtres humains ont commencé à écrire [42]. Elle est devenue un point focal pour les chercheurs du thème de la reconnaissance automatique de l'écriture.

Le terme "écriture" peut se référer aux mouvements complexes effectués par la main au cours de la production d'un texte [37], ou aux résultats de ce processus. En tant que processus, l'écriture est une tâche perceptivo-motrice complexe, une compétence qui s'acquiert généralement à l'école. [27]

L'interaction entre l'homme et la machine (Human Computer Interaction ou HCI) par l'écrit ou un geste graphique joue un rôle prépondérant dans les nouvelles technologies associées à l'informatique. La qualité des systèmes de reconnaissance d'écriture est un élément clé dans l'ergonomie associée à ce type d'interactions. [79]

L'objectif de la reconnaissance de l'écriture manuscrite est développé des systèmes intelligents et incisifs capables de déchiffrer et d'identifier indépendamment les entrées (qu'elles soient imprimées ou manuscrites) provenant de sources telles que des écrans tactiles, des documents papier, des photographies, etc. Cette tâche n'est pas triviale, car l'écriture possède une infinité de représentations dues au fait que chaque personne possède une écriture qui lui est propre avec de nombreux styles.

Ce qui rend la tâche difficile pour la reconnaissance, c'est la variabilité du type de l'écriture (manuscrite ou imprimée) [90], c'est ce qui a poussé les chercheurs à améliorer et développer les techniques d'extraction des caractères et /ou de classifications.

1.3 La reconnaissance automatique du caractère isolé

C'est la tâche la plus basique d'un système de reconnaissance de l'écriture. L'effort d'analyse est concentré sur un seul élément à la fois du vocabulaire (vue comme une forme globale).[21]

Le processus de reconnaissance automatique des caractères implique plusieurs étapes, notamment le prétraitement des images, l'extraction des caractéristiques et la classification. Le prétraitement de l'image consiste à nettoyer l'image numérisée pour supprimer le bruit, améliorer le contraste et ajuster les variations d'éclairage et d'orientation de l'image. Le processus d'extraction des caractéristiques consiste à déterminer les éléments essentiels des caractères, tels que l'épaisseur et le contour du trait, et à les représenter numériquement. La classification implique l'utilisation d'algorithmes d'apprentissage automatique pour faire correspondre les caractéristiques extraites avec des modèles de caractères connus et les affecter à la classe de caractères appropriée.

1.4 Les différents aspects de la reconnaissance d'écriture

Le problème de la reconnaissance de l'écriture peut être abordé par deux approches différentes, approche hors-ligne et approches-en-ligne, ces deux approches se diffèrent par la façon d'acquérir les tracés de l'écriture (en ligne ou hors-ligne), et par la présence ou non des informations décrivant l'ordre de construction de chaque tracé ; donc les méthodes de reconnaissance se diffèrent d'une approche à une autre.[64]

1.4.1 Le mode d'acquisition de l'écriture

1.4.1.1 La reconnaissance en ligne

La reconnaissance en ligne est effectuée au fur et à mesure que le caractère est tracé en temps réel ce qui permet, d'avoir moins de bruit et de faire des modifications suivant la réponse donnée à la phase de reconnaissance. Les moyens utilisés comme la tablette graphique avec un stylo électronique et l'écran tactile, sont couramment utilisés en reconnaissance en ligne.[59]

L'écriture en ligne (dynamique) est obtenue par une saisie en continue et se présente sous la forme d'une séquence de points ordonnée dans le temps. Dans ce cas, la donnée est de type signal et l'approche doit tirer profit de la représentation temporelle.[21]

1.4.1.2 La reconnaissance hors ligne

L'écriture hors-ligne (statique), elle convient aux documents imprimés et les manuscrits déjà rédigés. Les données en entrée sont acquises via un scanner. Elle a comme tâche de déterminer quelles lettres ou mots sont présents dans l'image du texte scanné.[14]

Ces systèmes disposent seulement des images des caractères ou des mots qui sont indépendantes de l'ordre de leur génération. Ce qui rende ce cas plus difficile. En effet, ces systèmes nécessitent davantage une étape des prétraitements : Seuillage, squelettisation, élimination de bruit, normalisation et segmentation qui peut être beaucoup plus complexe. Cependant, l'utilisation large de ces systèmes nécessite un traitement rapide avec un taux de reconnaissance élevé. [16]

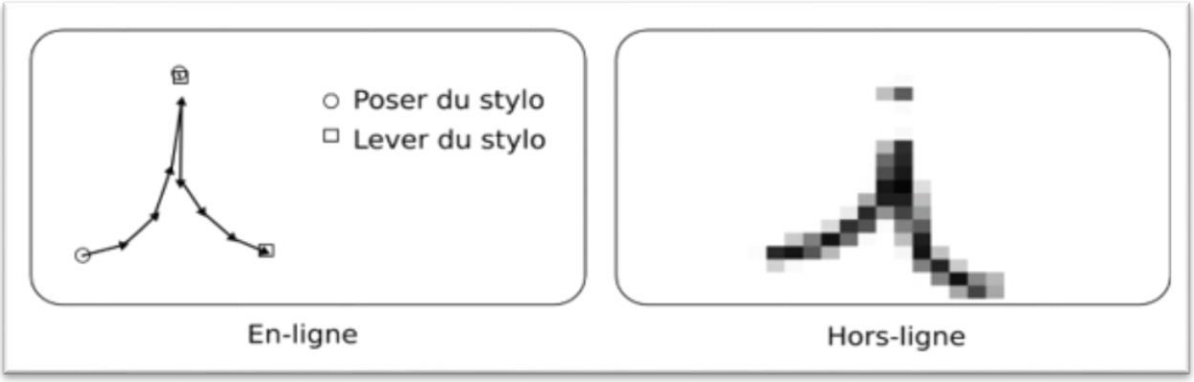


Figure 1: Reconnaissance en ligne et hors-ligne.[78]

1.4.2 Types d'écriture

1.4.2.1 Les systèmes de reconnaissance de l'écriture imprimée

Les systèmes de reconnaissance pour l'écriture imprimée, également connus sous le nom de systèmes de reconnaissance optique de caractères (OCR), sont des logiciels conçus pour identifier et convertir un texte imprimé numérisé en texte numérique modifiable. Ces systèmes utilisent des algorithmes complexes pour analyser l'image numérisée du texte imprimé, en identifiant les caractères individuels et en les faisant correspondre à une base de données de caractères connus. Les systèmes OCR ont un large éventail d'applications, de la numérisation d'anciens documents imprimés à la reconnaissance de texte dans les fichiers PDF à base d'images. Ils sont couramment utilisés dans les systèmes de gestion de documents, la saisie de données et les systèmes de recherche d'informations. La technologie OCR est également utilisée dans la traduction automatique, où le texte imprimé est numérisé et traduit automatiquement dans une autre langue. Les systèmes OCR peuvent être utilisés avec différents types de documents imprimés, tels que des livres, des journaux et des formulaires. Cependant, la précision des systèmes OCR peut être affectée par des facteurs tels que la qualité de l'image numérisée, la police de caractères utilisée et la disposition du matériel imprimé. Dans l'ensemble, les systèmes OCR ont révolutionné la façon dont le texte imprimé est numérisé et rendu consultable, permettant aux utilisateurs d'accéder et de manipuler de gros volumes de documents imprimés de manière plus efficace.

1.4.2.2 Les systèmes de reconnaissance de l'écriture manuscrites

L'écriture manuscrite fait référence à l'acte d'écrire ou de créer un texte à la main, par opposition à l'utilisation d'une machine à écrire ou d'un clavier d'ordinateur. L'écriture manuscrite peut prendre de nombreuses formes, des notes informelles et des gribouillis. Cela peut être fait avec une variété d'instruments d'écriture, tels que des stylos, des crayons, des marqueurs et des plumes, sur une large gamme de surfaces, elle peut également être difficile à lire ou à déchiffrer.

Les systèmes de reconnaissance manuscrite interpréter et traiter du texte ou des symboles manuscrits, Il existe deux approches principales pour les systèmes de reconnaissance manuscrite : Reconnaissance hors ligne, Reconnaissance en ligne.

Les systèmes de reconnaissance manuscrite se sont considérablement améliorés ces dernières années, grâce aux progrès des algorithmes d'apprentissage automatique et de la puissance de calcul. Ces systèmes sont désormais capables de reconnaître l'écriture manuscrite avec un haut degré de précision, ce qui les rend utiles dans une large gamme d'applications.

Ces systèmes de reconnaissance dépendent de plusieurs facteurs, à savoir, le nombre de scripteurs, le type ou le style d'écriture, la taille et le type de vocabulaire.[16]

1.5 Un système de reconnaissance des chiffres manuscrites

La reconnaissance des chiffres manuscrits a été la première recherche problème de la communauté d'analyse et de reconnaissance de documents depuis plus de trois décennies. [11]

La reconnaissance manuscrite des chiffres est la capacité d'un ordinateur de reconnaître les chiffres manuscrits humains de différentes sources telles que des images, des papiers, des écrans tactiles, etc., et classer les répartir en 10 classes prédéfinies (0-9). Cela a été un sujet de recherche illimitée dans le domaine de l'apprentissage en profondeur. La reconnaissance des chiffres a de nombreuses applications comme la plaque d'immatriculation reconnaissance, tri du courrier postal, traitement des chèques bancaires, etc. Dans la reconnaissance des chiffres manuscrits, nous sommes confrontés à de nombreux défis en raison des différents styles d'écriture des différents peuples car il n'est pas une reconnaissance optique de caractères. Cette recherche donne une comparaison complète entre différentes machines algorithmes d'apprentissage et d'apprentissage en profondeur dans le but de reconnaissance des chiffres manuscrits. Pour cela, nous avons utilisé Support Machine vectorielle, Perceptron multicouche et convolutif Réseau neuronal. La comparaison entre ces algorithmes est effectuée sur la base de leur exactitude, de leurs erreurs et de temps de test-formation corroboré par des graphiques et des graphiques qui ont été construits en utilisant Matplotlib pour la visualisation. [8]

De nombreux systèmes de reconnaissance des chiffres manuscrits ont été proposés pour des applications pratiques exigeant une précision et une fiabilité élevées. La présence de caractères touchés et superposés, de différents modèles d'écriture et de styles d'écriture propre à chaque individu ajoute à la complexité du système de reconnaissance. Dans le domaine de la reconnaissance de l'écriture manuscrite. [82]

Pour entraîner un modèle de reconnaissance de chiffres manuscrits, il est nécessaire de disposer d'un ensemble de données étiquetées. Cela signifie que chaque image de chiffre manuscrit doit être associée à l'étiquette correcte correspondant au chiffre représenté dans l'image. Une fois que le modèle est entraîné, il peut être utilisé pour prédire les chiffres dans de nouvelles images. Pour ce faire, l'image doit être prétraitée pour s'assurer qu'elle est au format et à la taille appropriés pour être donnée en entrée au modèle. Ensuite, le modèle peut prédire le chiffre dans l'image en utilisant une classification basée sur les probabilités.

La plupart de ces systèmes ont été évalués sur la base du jeu de données MNIST, largement utilisée.[11]

1.6 Architecture d'un système de reconnaissance d'écriture manuscrite

Un système de reconnaissance des chiffres manuscrits (SRCM) contient des phases importantes, Nous aborderons les étapes suivantes : acquisition, prétraitement, segmentation, extraction des caractères et classification.

La figure (2) illustre le schéma général d'un système de reconnaissance des chiffres.

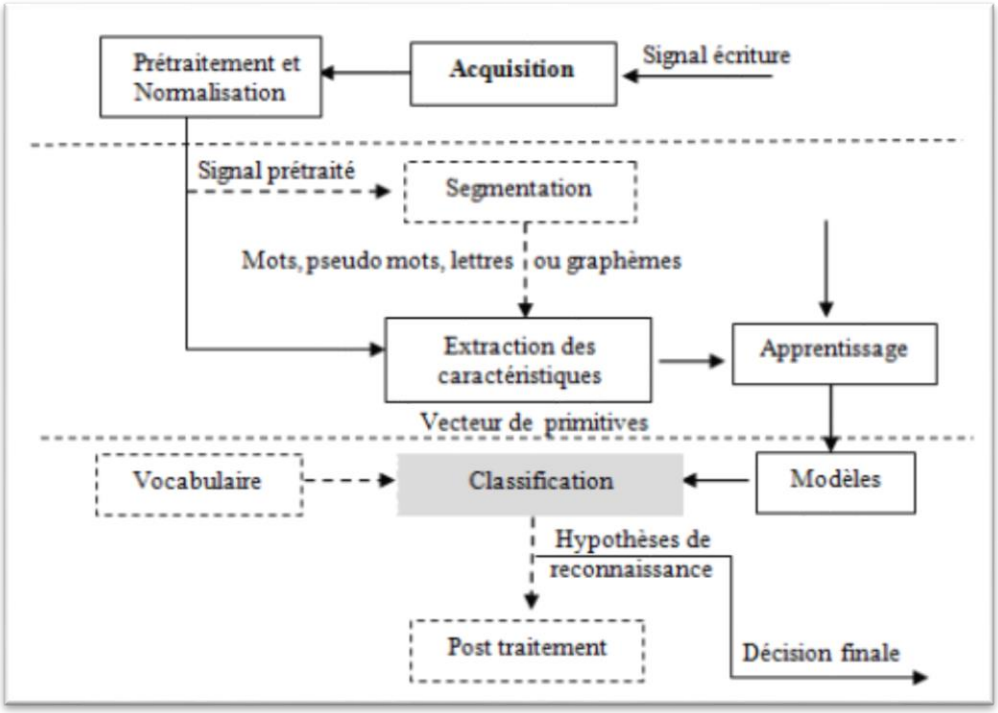


Figure 2: Diagramme de référence d'un système de reconnaissance des chiffres.[16]

1.6.1 Acquisition de l'image

Le processus de reconnaissance est précédé de la phase d'acquisition qui consiste à acquérir les images numériques de documents. L'opération s'effectue généralement à l'aide d'un scanner apparu dans les années 80. Les scanners actuels travaillent avec des résolutions de 300 dpi, une résolution qui permet de conserver un trait d'écriture d'une épaisseur de quelques pixels sous forme binaire ou en niveaux gris ou même en couleur. Il faut noter que la résolution de l'image numérisée influence les étapes ultérieures d'un système de reconnaissance.[59]

1.6.2 Prétraitement

Les objectifs visés par les étapes de prétraitement sont nombreux. Il est principalement utilisé pour réduire le bruit sur les données manuscrites, parfois pour corriger les défauts, et essayez de ne conserver que les informations importantes du formulaire de représentant.[3]

Ils comprennent principalement plusieurs taches qui facilite la reconnaissance d'image, on peut les citer ci-dessous :

1.6.2.1 Réalignement (de-skew)

Si le document n'a pas été correctement aligné lorsqu'il est numérisé, il peut avoir besoin d'être tourné de quelques degrés dans le sens horaire ou antihoraire pour s'assurer que les lignes de texte soient parfaitement horizontales ou verticales.[51]

1.6.2.2 Éliminer le bruit

Élimination du bruit comprend principalement le décrochage, le lissage et la correction du point sauvage, caractérisé par de grandes variations angulaires. Il s'agit d'éliminer le maximum de bruit pour augmenter la discrimination. [3] Voir « la figure 3 » :

Figure 3: Exemple de Suppression du bruit.[3]

1.6.2.3 Binarisation

C'est la façon dont l'image est convertie en image binaire, à savoir une image en noir et blanc. L'objectif est de mettre en valeur l'image [55] et affecter le niveau 1 aux pixels dont la valeur est supérieure à un seuil S et le niveau 0 aux autres « Inferieur » en utilisant le seuillage.[3]

Les pixels sont calculés en noir ou blanc en fonction de la valeur de seuil fournie, en fonction du type d'approche par seuil utilisée.[55]

Selon la méthode de calcul du seuil de binarisation, on distingue deux types de binarisation : le seuillage global et le seuillage adaptatif.[44]

- Seuillage global : ne sont pas trop consommatrices en termes de temps de calcul, par contre, elles ne donnent pas de bons résultats que si le document est uniformément éclairé.[3]
- Seuillage adaptatif (locale) : il s'adaptant à la différence de distribution des niveaux de gris on peut trouver une bonne synthèse des méthodes de binarisation,[21]

On calcule un seuil de binarisation pour chaque pixel de l'image, en fonction de son voisinage.[44]

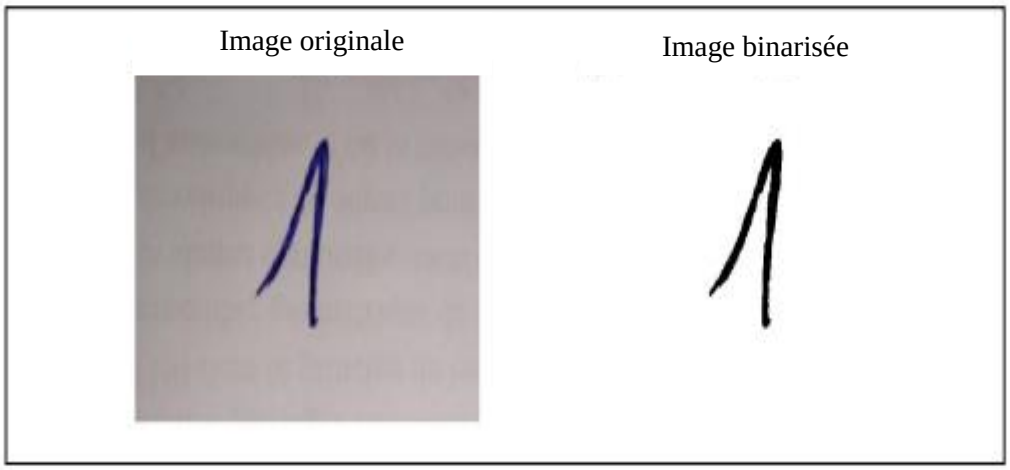


Figure 4: Exemple de la binarisation.[3]

1.6.2.4 Squelettisation

Cette première étape est souvent utilisée pour simplifier les formes à reconnaître. Elle permet notamment de s'affranchir du problème de l'épaisseur du trait qui est extrêmement variable au niveau de l'écriture manuscrite hors ligne. En amont de cette étape, nous effectuons un lissage de l'image brute opérant sur un voisinage de 8 points. Cette étape permet de limiter l'apparition de trous à l'intérieur des chiffres et de lisser leurs contours extérieurs. L'algorithme de squelettisation présenté est une amélioration de l'algorithme de Marthon. Contrairement à ce dernier, il permet d 'obtenir un squelette d'épaisseur 1 pixel. Il est fondé sur le calcul, en chaque point p (i, j) de l'image, d'une fonction simple F (i, j) opérant sur un voisinage de 8 points. F (i, j) est la longueur de la somme des vecteurs joignant le point p (i, j) à tous ses voisins. [71]

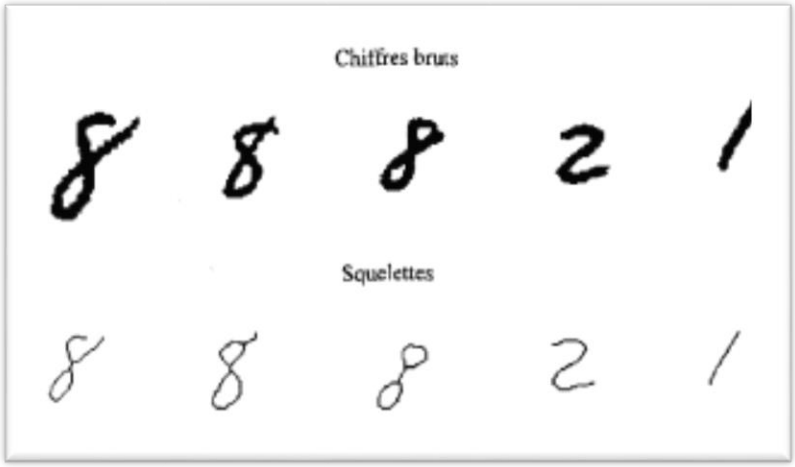


Figure 5: Squelettisation.[71]

La représentation squelettique possède les avantages suivants :

- La squelettisation est une étape essentielle de la reconnaissance de forme. Elle a pour but de décrire chaque objet par un ensemble de lignes infiniment fines.[99] Une bonne manière pour représenter au mieux les relations structurelles entre les composants de

modèle très utilisée dans les systèmes de reconnaissance de caractères, mot, signature et empreinte.[3]

- La méthode offre une faible sensibilité au bruit et permet d'obtenir une représentation structurelle robuste des chiffres manuscrits.
- Le squelette conserve les propriétés métriques de la forme comme le nombre de parties, le nombres de trous et la connexité.[71]

1.6.2.5 Normalisation de l'image

La normalisation tend à réduire ou d'éliminer autant que possible les variabilités liées aux styles, tailles et orientations d'écriture pour rendre celle-ci la plus indépendante possible du scripteur. L'importance de cette phase et de faire en sorte que l'étape d'extraction de primitives soit la moins influencée par la variation des styles d'écriture.[16]

❖ **Normalisation de taille :**

Cette étape permet de changer les images des chiffres manuscrits à des tailles standard.[35]

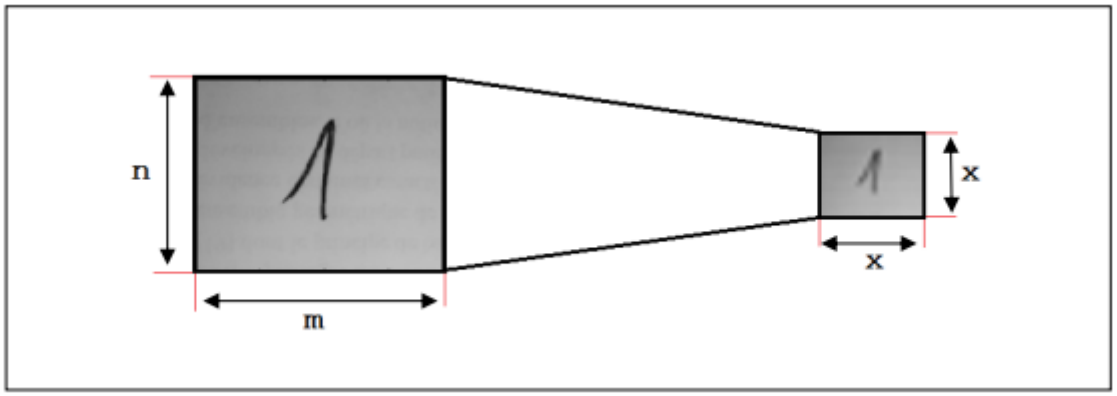


Figure 6 : Exemple d'un chiffre manuscrit normalisé.[3]

❖ **Normalisation de l'inclinaison (Redressement) :**

Le redressement est une opération fréquente en analyse de documents, souvent due à un mauvais positionnement du document sur le scanner, conduisant à une inclinaison de l'image. Plusieurs méthodes de redressement ont été proposées cette dernière décennie, et la plupart des OCR s'en trouvent pourvus actuellement tellement ces opérations sont courantes.[21]

Le redressement se fait selon deux niveaux :

- Le redressement de la ligne de base : L'idée est de rendre horizontaux les chiffres à l'aide d'une rotation isométrique des points de l'image. [35]
- Le redressement des écritures penchées : Ce redressement facilite la segmentation préalable des chiffres.[35]

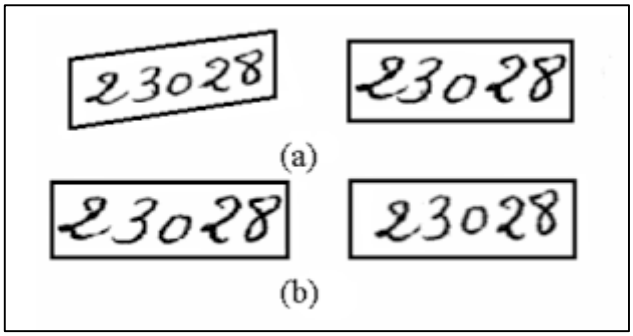


Figure 7: Exemple de redressement.[35]

1.6.2.6 Zonage

Le zonage ou segmentation physique permet de localiser les blocs d'information d'homogènes et de les classer en fonction de leur contenu : texte ou non texte. Cette séparation permet d'écarter les zones de graphique et de photographie du processus de reconnaissance de texte.[21]

*Figure 8: Exemple d'un chiffre zoné de taille (3 * 3) [3]*

1.6.3 Segmentation

La segmentation dans le traitement d'image désigne le processus de division d'une image en plusieurs segments ou régions en fonction de certaines caractéristiques comme la couleur, la texture ou l'intensité. Le but de la segmentation est de simplifier et/ou de changer la représentation d'une image en quelque chose de plus significatif et plus facile à analyser.la segmentation des images est une étape importante dans diverses applications telles que la reconnaissance d'objets, la compression d'image et la vision par ordinateur.

Plus la segmentation est meilleure plus l'étape de reconnaissance est réussite.[66]

Il existe deux types de segmentation :

1.6.3.1 Segmentation explicite

La segmentation explicite fait référence au processus de séparation d'une entité ou d'un objet en parties ou segments distincts sur la base de critères clairs et spécifiques. Dans le contexte des chiffres, la segmentation explicite fait référence au processus de séparation des chiffres individuels dans un nombre. Par exemple, le nombre « 12345 » peut être explicitement

segmenté en cinq chiffres distincts : « 1 », « 2 », « 3 », « 4 » et « 5 ». La segmentation explicite des chiffres est souvent utilisée dans divers domaines, tels que la vision par ordinateur et le traitement d'images, où il est nécessaire d'extraire et d'analyser des chiffres individuels à partir d'une image d'un nombre. Cela peut être fait en utilisant diverses techniques, telles que le seuillage et la détection des contours. Une fois les chiffres segmentés, ils peuvent être traités et analysés plus avant pour extraire des informations utiles, telles que la valeur du nombre ou sa représentation dans un système numérique spécifique.

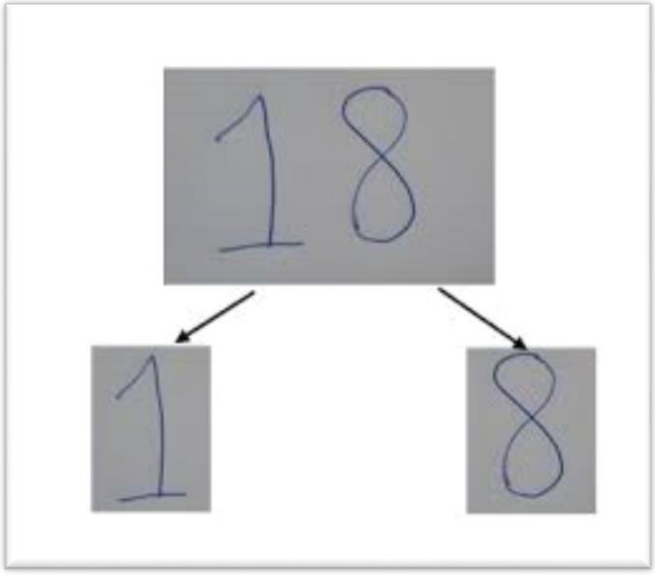


Figure 9: La segmentions explicites des chiffres manuscrits.[35]

1.6.3.2 Segmentation implicite

La segmentation implicite, contrairement à la segmentation explicite, fait référence au processus de séparation d'une entité ou d'un objet sans l'utilisation de critères clairs et spécifiques. Dans le contexte des chiffres, la segmentation implicite fait référence au processus de séparation des chiffres individuels dans un nombre basé sur des indices implicites, tels que l'espacement entre les chiffres ou les modèles visuels qui les distinguent. Par exemple, le nombre "12345" peut être implicitement segmenté en cinq chiffres distincts en fonction des repères visuels qui distinguent chaque chiffre, même s'il n'y a pas de frontière claire entre eux. La segmentation implicite des chiffres peut être plus difficile que la segmentation explicite, car elle nécessite la capacité de reconnaître et d'interpréter des indices visuels qui ne sont pas toujours cohérents ou fiables. Cela peut être particulièrement difficile dans les cas où les chiffres sont étroitement espacés ou lorsqu'il y a du bruit ou de la distorsion dans l'image. Néanmoins, la segmentation implicite est une compétence importante dans divers domaines, tels que le traitement du langage naturel et l'apprentissage automatique, où il est nécessaire d'extraire et d'analyser des chiffres à partir de données non structurées, telles que du texte ou de la parole. Cela peut être fait en utilisant diverses techniques, telles que la reconnaissance de formes, les algorithmes d'apprentissage automatique et les réseaux de neurones.

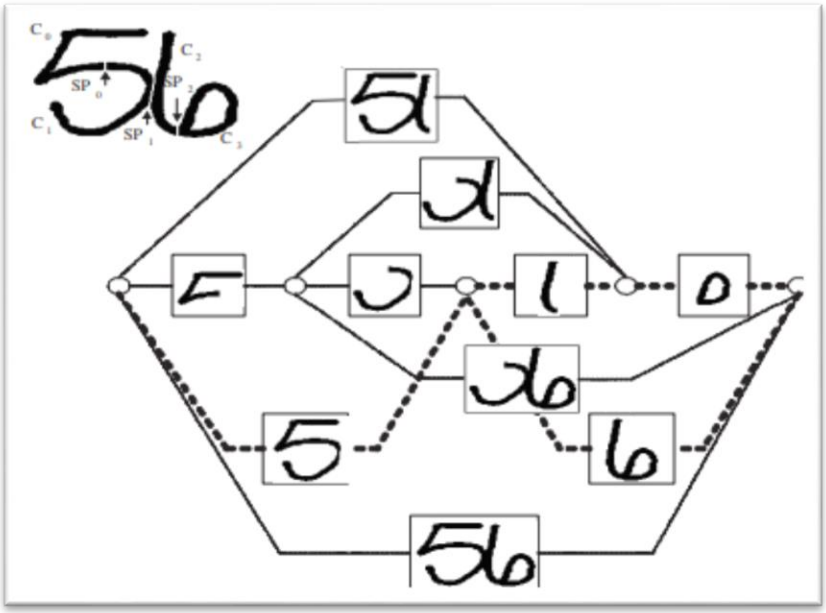


Figure 10: La segmentions implicites des chiffres manuscrits.[10]

1.6.4 Extraction des caractéristiques

L'extraction de caractéristiques est l'étape la plus critique de tout modèle tâche de classement. L'idée clé de l'extraction de caractéristiques est de trouver une représentation efficace et discriminatoire des modèles sous étude (des chiffres dans notre cas) qui minimise la variabilité intra-classe et qui maximise la variabilité interclasse.[12]

L'extraction de caractéristiques vise à exprimer les données d'entrée à l'aide d'une représentation numérique ou un ensemble de symboles (codage) pour sélectionner le meilleur ensemble de caractéristiques (vecteur de caractéristiques) pour un problème particulier, reconnaissance des chiffres dans notre cas. [13]

La difficulté ici est de trouver de bonnes caractéristiques. De « bonnes » caractéristiques permettent aux classifieurs de reconnaître facilement les différentes classes d'objets ; on dit alors qu'elles sont discriminantes. Elles doivent aussi être invariantes à certaines transformations « le chiffre 1 appartient à la même classe quelle que soit sa taille et la structure ».[3]

Il existe différentes méthodes d'extraction de primitives citées dans la littérature. Cette extraction se fait sur des images en niveaux de gris, en binaire, en contour binaire et selon une représentation vectorielle. Il existe quatre types des caractéristiques : les primitives structurelles, les primitives statistiques, les primitives globales et les primitives morphologiques.[23]

1.6.4.1 Caractéristiques structurelles

Les caractéristiques structurelles basées sur une représentation linéaire du caractère (décomposition du caractère en segments de droites et courbes, contours du caractère, squelette).[23] Des exemples de caractéristiques structurelles comprennent virages, extrémités, intersections, boucles et mesures de concavité etc. Les propriétés structurelles peuvent parfois aussi être exprimée à l'aide de mesures statistiques. [13]

Les primitives structurelles ont une grande capacité discriminative très forte, cette prétention influe sur la rapidité de la décision de l'appartenance d'une classe lors de la classification. De ce fait, la mauvaise gestion de ces primitives (mauvaise détection de leur présence) génère automatiquement des résultats insatisfaisants pendant le processus de reconnaissance.[16]

1.6.4.2 Caractéristiques statistiques

Les caractéristiques statistiques représentent des classes de modèles par des mesures statistiques,[13] Ils sont établis sur la planification de la manière dont les données sont sélectionnées. Il utilise les informations de la distribution statistique de pixels dans l'image.[83]

Elles sont générées généralement par l'estimation de la densité de pixels appartenant à l'image entière (caractères ou mots) ou dans certaines parties uniquement, en utilisant des mesures statistiques (entropie, moyenne, variance, etc.).[16]

Les caractéristiques utilisées pour la reconnaissance des chiffres manuscrits sont :

• Le zonage (zoning) : consiste à répartir l'image en région ou zones. L'image source est une image binaire.

• L'histogramme : c'est un moyen de représentation du nombre de pixels sur chaque ligne ou colonne de l'image.[35]

Ces caractéristiques sont considérées comme moins discriminantes par rapport aux autres primitives car elles portent une information faible sur le signal d'écriture.[16]

1.6.4.3 Caractéristiques globales

La caractéristique d'une primitive globale est de dépendre de la totalité des pixels d'une image et d'une transformation globale de l'image. Ces primitives sont donc dérivées de la distribution des pixels, elles produisent trois types de caractéristiques telles que : moments invariants, projections et profils.[35]

Les techniques les plus utilisées dans la reconnaissance de l'écriture sont :

- La transformation de Hough : cette méthode est largement explorée dans le domaine de la reconnaissance d'écriture vue sa robustesse et son efficacité dans l'extraction des primitives de haut niveau. [16]
- La transformation de Karhunen-Loéve (KL) : qu'est largement utilisée récemment pour la reconnaissance de l'écriture. Le défaut majeur de cette transformation est qu'elle est couteuse en termes de calcul, par rapport à d'autres.[16]

1.6.4.4 Caractéristiques morphologiques

La morphologie est une théorie fondamentale du traitement non linéaire d'images. Les opérateurs qu'elle propose permettent de fournir des outils pour toute la chaîne de traitement d'images, des prétraitements à la segmentation et à l'interprétation. Une des caractéristiques importantes de ces opérateurs est qu'ils sont non linéaires. Ils permettent de transformer les images, d'en extraire des caractéristiques, des objets ou encore des mesures par une analyse associant propriétés des objets eux-mêmes (forme, taille, apparence...) et propriétés du contexte (voisinage local ou relations avec d'autres objets).[99]

Les deux principales opérations morphologiques sont :

- La dilatation : consiste à élargir la figure, la hauteur et largeur de la figure dilatée seront les sommes respectivement des hauteurs et largeurs de la figure originelle et de l'élément structurant. [52]
- L'érosion : consiste à rétrécir la figure, la hauteur et largeur de la figure érodée seront les différences respectivement des hauteurs et largeurs de la figure originelle et de l'élément structurant (en particulier si l'élément structurant est plus large ou plus haut que la figure, l'érosion de celle-ci sera vide).[52]

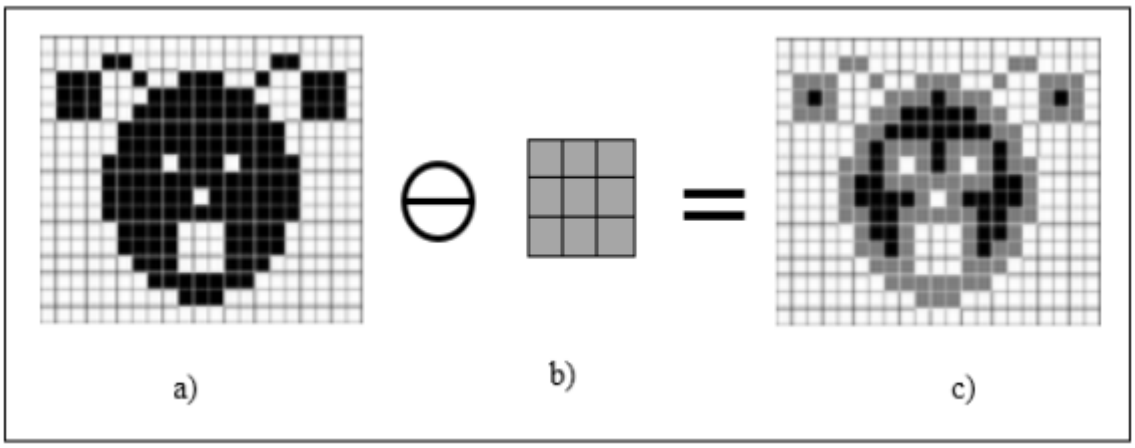


Figure 11: Erosion d'image.

Image originale ; b) élément structurant 3* 3 ; c) image érodée ⊖ : Operateur d'érosion.[52]

1.6.5 Classification

Dans la classification d'images, l'objectif est d'apprendre à un algorithme informatique à reconnaître et à catégoriser différents objets ou motifs au sein d'une image. Cette phase comprend généralement les étapes suivantes :

- Collecte de données : la première étape consiste à rassembler un ensemble de données d'images qui comprend les objets ou les modèles d'intérêt. L'ensemble de données doit être suffisamment volumineux et diversifié pour former l'algorithme avec précision.
- Prétraitement des données : L'ensemble de données collectées peut nécessiter un prétraitement pour assurer la cohérence et la normalisation. Cela peut inclure le redimensionnement des images, la suppression du bruit ou de la distorsion et le réglage des niveaux de couleur.
- Extraction de caractéristiques : l'étape suivante consiste à extraire les caractéristiques des images, qui seront utilisées pour former l'algorithme. Cela implique d'identifier les caractéristiques ou les motifs clés de l'image, tels que les bords ou la texture, et de les convertir en valeurs numériques.
- Sélection et entraînement du modèle : Suite à l'extraction des fonctionnalités, l'ensemble de données est utilisé pour choisir et entraîner un modèle d'apprentissage automatique. Les modèles courants utilisés pour la classification des images comprennent les réseaux de neurones convolutionnels (CNN) et les machines à vecteurs de support (SVM).

- Validation et test : une fois le modèle formé, il est testé sur un ensemble de données de validation distinct pour s'assurer qu'il classe les images avec précision. Si le modèle fonctionne bien, il peut alors être utilisé pour classer de nouvelles images.
- Déploiement : Enfin, le modèle peut être déployé dans un environnement de production, où il peut être utilisé pour classer des images en temps réel.

1.7 La reconnaissance d'écriture et l'apprentissage automatique

1.7.1 L'apprentissage automatique

La Machine Learning est couramment traduit en français par apprentissage automatique, apprentissage machine ou encore apprentissage artificiel ou statistique. Il s'agit de l'une des grandes technologies de l'intelligence artificielle.[26]

L'apprentissage automatique signifie que les ordinateurs apprennent par elle-même à partir de données à l'aide d'algorithmes pour effectuer une tâche sans qu'un humain doive lui programmer toutes les étapes du processus. L'apprentissage profond (Deep-Learning) utilise une structure complexe d'algorithmes modélisés sur le cerveau humain. Cela permet le traitement de données non structurées telles que des documents, des images et du texte**.**[98] C'est donc un domaine de l'intelligence artificielle qui permet de résoudre des problèmes complexes en utilisant des modèles statistiques et des techniques d'apprentissage.

1.7.2 Le *processus d'apprentissage*

Le *processus d'apprentissage* repose sur les étapes suivantes [40] :

- Alimenter un algorithme en données (au cours de cette étape, on fournit des informations supplémentaires au modèle, par exemple, en effectuant l'extraction de caractéristiques).
- Utilisez ces données pour entraîner un modèle.
- Tester et déployer le modèle.
- Utiliser le modèle déployé pour effectuer une tâche prédictive automatisée (en d'autres termes, appeler et utiliser le modèle déployé pour recevoir les prédictions retournées par le modèle).[40]

1.7.3 Les types d'apprentissage automatique

Il y a plusieurs types d'apprentissage automatique notamment l'apprentissage supervisé, l'apprentissage non supervisé, et l'apprentissage par renforcement. Ces différentes techniques permettent de capturer les modèles et les relations dans les données afin de pouvoir les utiliser pour des prédictions ou des décisions.

1.7.3.1 L'apprentissage supervisé (supervised Learning)

L'apprentissage supervisé consiste à utiliser des ensembles de données étiquetées pour former un algorithme de machine Learning. Ladite supervision tient au fait que les étiquettes sur les données aident le modèle à prédire des résultats avec précision. Autrement dit, les données de formation contiennent déjà les réponses correspondantes aux sorties attendues.[95]

L'apprentissage supervisé peut être séparé en deux types de problèmes lors de l'exploration de données :

a. Apprentissage supervisé : La classification

Les problèmes de classification consistent à utiliser un algorithme pour attribuer avec précision des données de test à des catégories spécifiques. Par exemple, les algorithmes d'apprentissage supervisé peuvent être utilisés pour classer les spams dans un dossier distinct de la boîte de réception. Les classifieurs linéaires, les machines à vecteurs de support, les arbres de décision et les forêts d'arbres décisionnels sont tous des types courants d'algorithmes de classification. [65]

b. Apprentissage supervisé : La régression

La régression est un autre type de méthode d'apprentissage supervisé qui utilise un algorithme pour comprendre la relation entre les variables dépendantes et indépendantes. Les modèles de régression sont utiles pour prédire des valeurs numériques sur la base de différents points de données. Les algorithmes de régression sont par exemple la régression linéaire, la régression logistique et la régression polynomiale.[65]

1.7.3.2 L'apprentissage non supervisé (unsupervised Learning)

L'apprentissage non supervisé consiste à entraîner des modèles, sans réaliser d'étiquetage manuel ou automatique des données au préalable. Les algorithmes regroupent les données en fonction de leur similitude, sans aucune intervention humaine.[49]

Les modèles d'apprentissage non supervisé sont utilisés pour trois tâches principales :

a. Apprentissage non supervisé : Le clustering ou partitionnement de données

Le clustering est une technique d'exploration de données permettant de regrouper des données non étiquetées en fonction de leurs similitudes ou de leurs différences. Cette technique est utile pour la segmentation de marché, la compression d'images, etc. [65]

b. Apprentissage non supervisé : L'association

L'association est un autre type de méthode d'apprentissage non supervisé qui utilise différentes règles pour trouver des relations entre les variables d'un ensemble de données donné. [65]

c. Apprentissage non supervisé : La réduction de la dimensionnalité

Tâches de dimensionnalité où l'algorithme cherche à réduire le nombre de variables, de caractéristiques ou de fonctionnalités dans un ensemble de données. Étant donné que certaines de ces dimensions sont corrélées, des informations redondantes ou répétées peuvent augmenter le bruit de l'ensemble de données et avoir un impact sur la formation et performant de modèle. Cette technique est souvent utilisée dans l'étape de prétraitement des données, par exemple lorsque le bruit est supprimé des données visuelles pour améliorer la qualité de l'image.[41]

1.7.3.3 L'apprentissage par renforcement

L'apprentissage par renforcement ou Reinforcement Learning est une méthode de Machine Learning de plus en plus utilisée. Elle consiste à laisser les ordinateurs apprendre de leurs expériences grâce à un système de récompense ou de pénalité. Il pourrait même s'agir de la clé permettant l'avènement d'une intelligence artificielle générale comparable à celle de l'humain.[19]

Les exemples d'algorithmes de renforcement incluent les méthodes Q-Learning et SARSA.

1.7.4 Des principales différences entre l'apprentissage supervisé et non supervisé

- Complexité informatique : l'apprentissage supervisé est une méthode plus simple qui ne nécessite qu'un programme comme Python ou R. L'approche non supervisée est plus complexe et nécessite donc des outils plus puissants. [41]
- Précision et classes : l'apprentissage supervisé est plus précis, fiable et le nombre de classes est connu. Le nombre de classes n'est pas connu dans l'apprentissage non supervisé, et il a tendance à être moins précis et moins fiable. [41]
- Objectif : l'objectif de l'apprentissage supervisé est de prédire les résultats de nouvelles données. En revanche, l'objectif de l'apprentissage non supervisé est d'extraire des informations et des structures à partir des données. Il *se base* sur ce que le modèle détermine comme étant intéressant ou différent. [41]
- Inconvénients potentiels : l'apprentissage supervisé est une approche de formation qui prend du temps et de l'expertise humaine. D'un autre côté, l'apprentissage non supervisé peut donner des résultats inexacts ou sans valeur à moins qu'il n'y ait un humain pour valider les variables de sortie.[41]

1.7.5 Les différentes techniques pour la reconnaissance de l'écriture

Il existe différents types d'algorithmes pour l'apprentissage automatique. Ces algorithmes peuvent être utilisés pour diverses tâches, telles que la classification, la prédiction et la reconnaissance de motifs. Cependant, ils ont leurs avantages et leurs limites en fonction du type de données et de la tâche à accomplir.

1.7.5.1 Naïve Bayes

Naïve Bayes est un algorithme d'apprentissage automatique probabiliste basé sur le théorème de Bayes, utilisé dans une grande variété de tâches de classification. [69]

Le théorème de Bayes est une formule mathématique simple utilisée pour calculer les probabilités conditionnelles. La probabilité conditionnelle est une mesure de la probabilité qu'un événement se produise étant donné qu'un autre événement s'est produit (par hypothèse, présomption, affirmation ou preuve).[69]

Cet algorithme fonctionne en calculant la probabilité d'appartenir à chaque classe pour chaque nouvelle donnée et en le classant selon la valeur de probabilité la plus élevée. Des résultats très efficaces peuvent être obtenus même avec un faible temps de calcul. Cela peut bien fonctionner sur des ensembles de données déséquilibrés. L'équation du théorème de Bayes est donnée ci-dessous. [55]

$$(A/B) = (P(B/A) \times P(A)) / P(B).$$

Les algorithmes naïfs de Bayes sont souvent utilisés dans l'analyse des sentiments, le filtrage des spams, les systèmes de recommandation, etc. Ils sont rapides et faciles à mettre en œuvre, mais leur plus grand inconvénient est l'exigence d'indépendance des prédicteurs.[69] Et ces hypothèses ne sont pas toujours valables dans la vie réelle.[84]

1.7.5.2 L'arbre de décision

L'arbre de décision est le classificateur le plus puissant. Un arbre de décision est un organigramme semblable à une structure arborescente, où chaque nœud interne désigne un test sur un attribut (une condition), chaque branche représente un résultat du test (vrai ou faux) et chaque nœud feuille (node terminal) contient une étiquette de classe. Sur la base de cet arbre, des divisions

sont effectuées pour différencier les classes dans l'ensemble de données d'origine donné. Le classificateur prédit à laquelle des classes appartient un nouveau point de données en fonction de l'arbre de décision. Les limites de prédiction sont des lignes horizontales et verticales. [84]

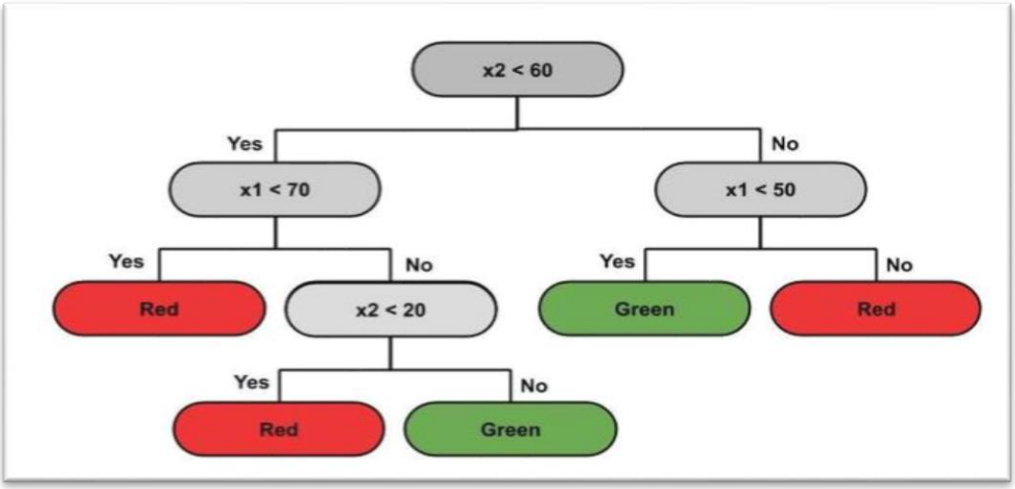


Figure 12: Arbre de décision.[84]

Il existe plusieurs algorithmes de construction d'arbres de décision, qui différent dans leur méthode de sélection des attributs et dans leur critère de division des nœuds de l'arbre notamment l'algorithme ID3, l'algorithme C4.5 et l'algorithme CART.

1.7.5.3 La machine à vecteurs de support

La machine à vecteurs de support (ou SVM pour Support Vector Machine en anglais) est un algorithme d'apprentissage automatique supervisé. L'objectif principal de la SVM est de créer une frontière de décision qui sépare les données en deux classes différentes de manière optimale. Cette frontière est appelée hyperplan, qui est déterminé en maximisant la marge entre les deux classes, c'est-à-dire la distance entre les points les plus proches de chaque classe. [8]

Les SVM sont utilisées dans de nombreux domaines, tels que la classification de textes, la reconnaissance des caractères, la détection de fraudes, etc.

Les machines à vecteurs de support sont divisées en deux selon la séparabilité et la non-séparabilité linéaires de l'ensemble de données. L'un des points les plus importants des machines à vecteurs de support est les fonctions du noyau. La fonction noyau appropriée doit être sélectionné selon que l'ensemble de données peut être séparé linéairement ou non [26]. Ces fonctions du noyau affectent grandement succès. Les fonctions du noyau les plus couramment utilisées sont "linéaire", "rbf", "poly" et "sigmoïde". Alors que le "linéaire" la fonction noyau est utilisée pour les ensembles de données séparables linéairement, "rbf" est utilisé pour les ensembles de données non linéairement séparables. [55]

La figure 13 montre un diagramme schématique du processus de classification utilisant la méthode SVM, qui comprend un processus de formation et de test. Les noyaux SVM utilisent des polynômes et une fonction de base radiale (RBF). Ces choix sont considérés comme un mécanisme de classification efficace puisque ces noyaux montrent une séparation non linéaire entre les classes. Pour garantir une grande précision de prédiction, le processus de validation croisée est effectué. Cela permet d'obtenir les meilleurs paramètres du noyau.[80]

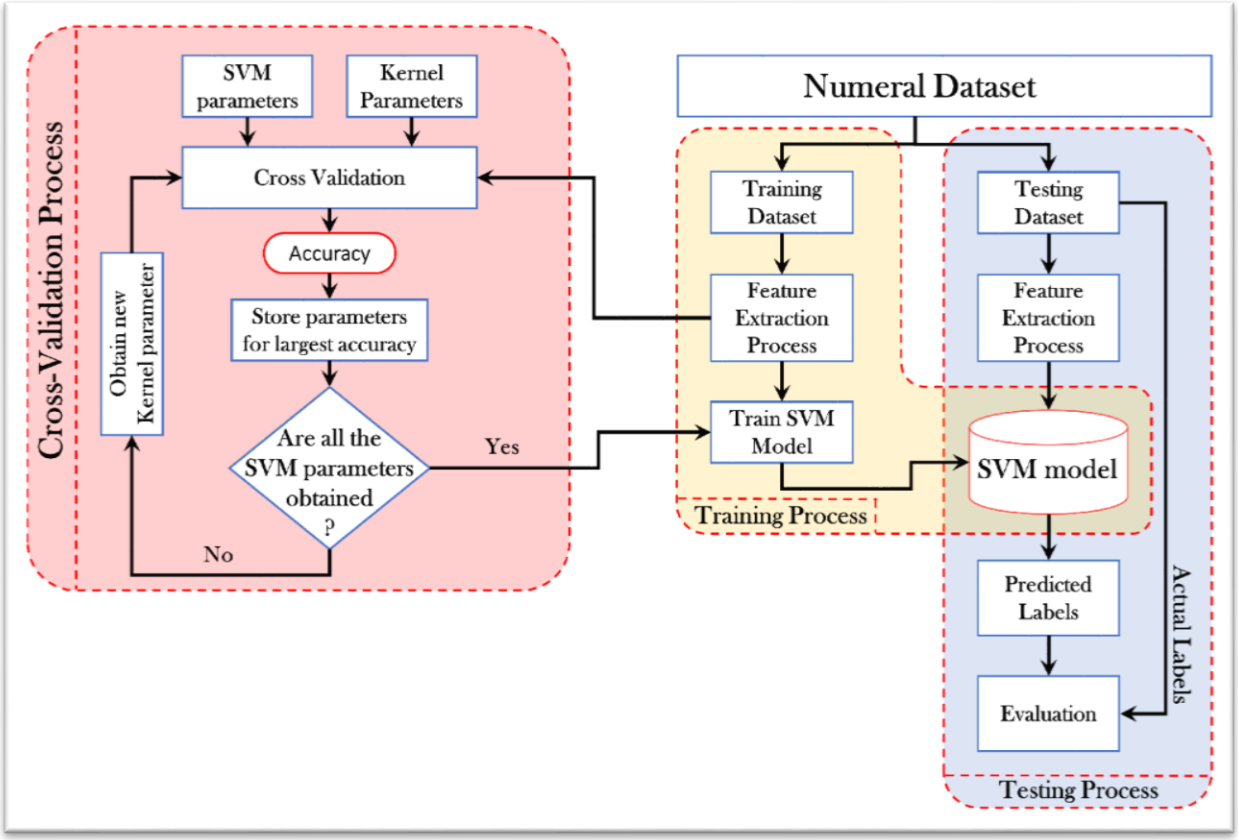


Figure 13: Diagramme schématique du processus de classification des SVM.[80]

L'algorithme SVM n'est pas adapté aux grands ensembles de données et ne fonctionne pas très bien lorsque l'ensemble de données a plus de bruit, c'est-à-dire que les classes cibles se chevauchent. Comme le classificateur de vecteur de support fonctionne en plaçant des points de données, au-dessus et en dessous de l'hyperplan de classification, il n'y a pas d'explication probabiliste pour la classification.[14]

1.7.5.4 Modèle de Markov caché (HMM)

Les algorithmes HMM (hidden marcov model) sont des algorithmes d'apprentissage automatique utilisés pour résoudre le problème de la reconnaissance automatique de l'écriture manuscrite et pour modélisés les données séquentielles. Le modèle de markov caché est le modèle utilisé pour cette tâche, dans lequel un système dynamique est modélisé sous forme de processus de Markov, où chaque état est représenté par une variable cachée.

Le HMM est un type de chaîne de Markov. Son état ne peut pas être observé directement mais peut être identifié en observant la série de vecteurs. Depuis les années 1980, HMM a été utilisé avec succès pour la reconnaissance vocale, la reconnaissance de caractères et les techniques de communication mobile. Il a également été rapidement adopté dans des domaines tels que la bio-informatique et le diagnostic de pannes. Le principe de base du HMM est que les événements observés n'ont pas de correspondance univoque avec les états, mais sont liés aux états par le biais de la distribution de probabilité.[14]

Le processus de Markov peut être une approximation appropriée pour résoudre des problèmes complexes d'apprentissage automatique et par renforcement. De plus, la probabilité de transition d'un état à un autre peut être emballée dans une matrice de transition comme celle ci-dessous:[48]

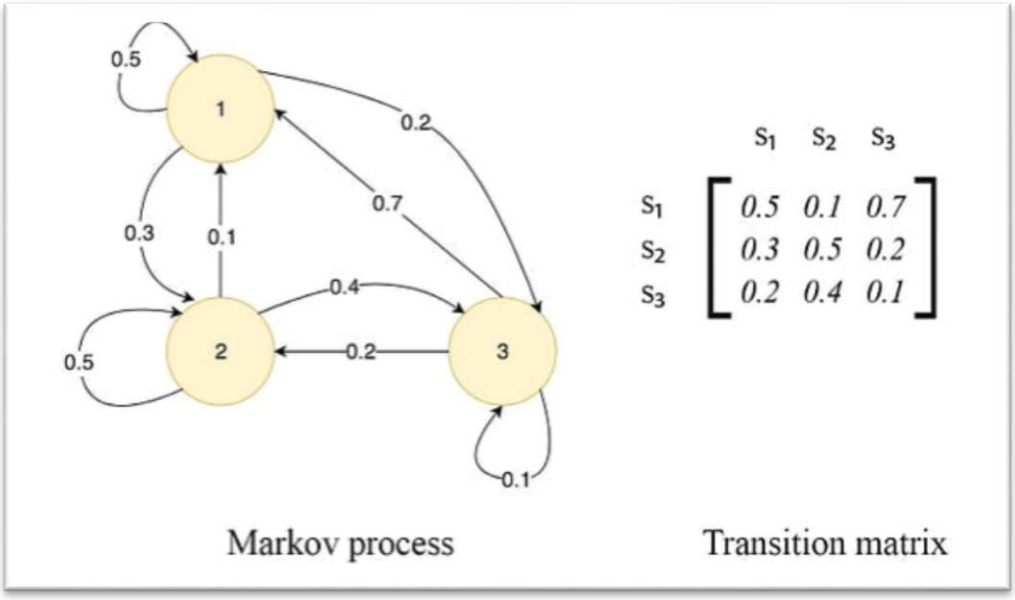


Figure 14:Processus de Markov et matrice de transition.[39]

Le modèle HMM est formé à partir de deux processus :

- Processus d'état : il s'agit du processus de Markov caché, où les variables cachées représentent l'état du système. [59]
- Processus d'observation : il s'agit du processus qui génère les données observées à partir de l'état courant du système. [59]

❖ **Les types des HMM :**

Le type d'un HMM peut être spécifié suivant des contraintes [59] :

- Par rapport à la matrice de transition A (Ergodique ou Gauche-Droite).
- Par rapport à la densité de probabilité d'observation (Discrète DHMM, Continue CHMM ou Semi-Continue SCHMM).
- Par rapport à la durée de séjour dans un état (Durée variable Discrète VDHMM ou Durée variable Continue CVDHMM).
- Par rapport à l'ordre de la chaine de Markov (HMM d'ordre r).[59]

Les algorithmes HMM suivants peuvent être utilisés :

- Algorithme de Baume-Welch :il s'agit d'un algorithme qui utilise la méthode de maximisation de l'espérance pour estimer les paramètres du modèle HMM. Cet algorithme est utilisé pour l'apprentissage non-supervisé.
- Algorithme de viterbi : il s'agit d'un algorithme qui utilise la programmation dynamique pour trouver le chemin le plus probable dans le modèle HMM. Cet algorithme est utilisé pour la prédiction avec des données d'entrée.

- Algorithme de forward-backward : il s'agit d'un algorithme qui utilise la méthode d'estimation de l'espérance pour calculer la probabilité d'observation pour chaque état. Cet algorithme utilisé pour la classification.

Les algorithmes HMM sont utilisés dans plusieurs domaine, notamment la reconnaissance des caractères, la biométrie et la prédiction de séries chronologique.

L'implémentation de ces algorithmes s'appuie sur des techniques bien maitrisées de la programmation dynamique. En plus, un certain nombre de librairies HTK, GHMM et ESMERALDA et de modules sont également publiquement accessibles pour l'apprentissage et le décodage de Modèles de Markov Cachés. En reconnaissance de l'écriture, les graphèmes, les caractères ou les mots sont souvent modélisés par des états cachés dont leur succession est donnée par une chaîne de Markov.[14]

Les techniques à base de HMM sont à l'heure actuelle des plus utilisées, le modèle HMM présente différents avantages : clarté, rigueur et généralité. Des modèles hybrides ont été proposés afin de combiner les HMM avec d'autres modèles telles que les approches connexionnistes dans le cadre de l'intelligence artificielle. [77]

❖ **Les avantages et les inconvénients des HMM :**

Avantages	Inconvénients
• Base mathématique solide pour comprendre son fonctionnement. • Variabilité de la forme. • Alignement temporel incorporé systématiquement. • Prise en compte de l'ordre d'apparition dans une séquence de vecteurs. • Reconnaissance réalisée par un simple calcul de probabilité cumulée. • Décision globale sans obligation d'utiliser des seuils. • Séparation franche entre données et algorithmes.	• Ignorance complète de la durée relative des événements du signal. • Dégradation des performances si l'apprentissage n'est pas suffisant. • Le choix à priori de la typologie des modèles (nombre d'états, transitions autorisées et règles de transition) limite la souplesse des modèles. • Modèle comportemental et non fonctionnel.

Tableau 1: Les avantages et les inconvénients des HMM [77].

1.7.5.5 Les K-plus proches voisins (K-NN)

L'algorithme des k-plus proches voisins (k-NN) se base sur les données en entier. En effet, pour une observation, qui ne fait pas partie des données, qu'on souhaite prédire, l'algorithme va chercher les k instances les plus proches de notre observation et choisir pour chaque observation la classe majoritaire parmi ses k plus proches voisins.[18]

Les k-plus proches voisins est un algorithme d'apprentissage automatique non supervisé qui appartient à la catégorie des méthodes d'apprentissage par proximité.il est utilisé pour classifier des données en trouvant les k exemples les plus similaires à un nouveau point de données, et il suppose donc que les points de données les plus similaires sont ceux trouvés à proximité. Il calcule d'abord la distance entre les points de données, puis attribue une catégorie en fonction de sa fréquence ou de sa moyenne.[41]

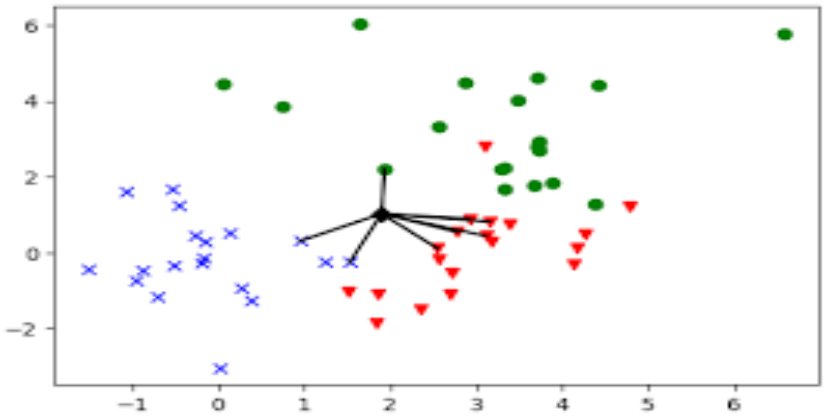


Figure 15: Classification par K-Plus Proches Voisins. [30]

Les deux paramètres les plus importants affectant le taux de réussite de l'algorithme sont le nombre de voisins (k) et le critère de distance, qui est la valeur de la distance aux voisins. Les données sont classées en examinant les voisins les plus proches autant que la valeur de k déterminée dans l'algorithme. La donnée sélectionnée comme voisine la plus proche est sélectionné en fonction du paramètre déterminé comme critère de distance. Le critère de distance le plus courant ; Les distances de Minkowski, Euclidienne et Manhattan sont utilisées.[55]

$$d(p, q) = d(q, p) = \sqrt{(q1 - p1)2 + (q2 - p2)2 + \cdots + (qn - pn)2} \quad [18]$$

L'objectif de l'algorithme est de déterminer la classe des nouvelles observations en comparant avec les observations d'entrainement et en trouvant les plus proches voisins de l'observation testée.

L'algorithme du K-plus proche voisin est un algorithme puissant car il est simple et résistant aux données d'entraînement bruyantes. Le L'inconvénient de cet algorithme est qu'il nécessite une très grande espace mémoire dans un grand ensemble de données, car il stocke toutes les données et comptes.[55]

De plus, si la plupart des voisins sont très différents de l'instance à prédire, cela peut entrainer une prédiction inexacte. Enfin, le choix de la valeur k peut avoir un impact significatif sur les résultats et peut nécessiter des ajustements pour différentes applications.

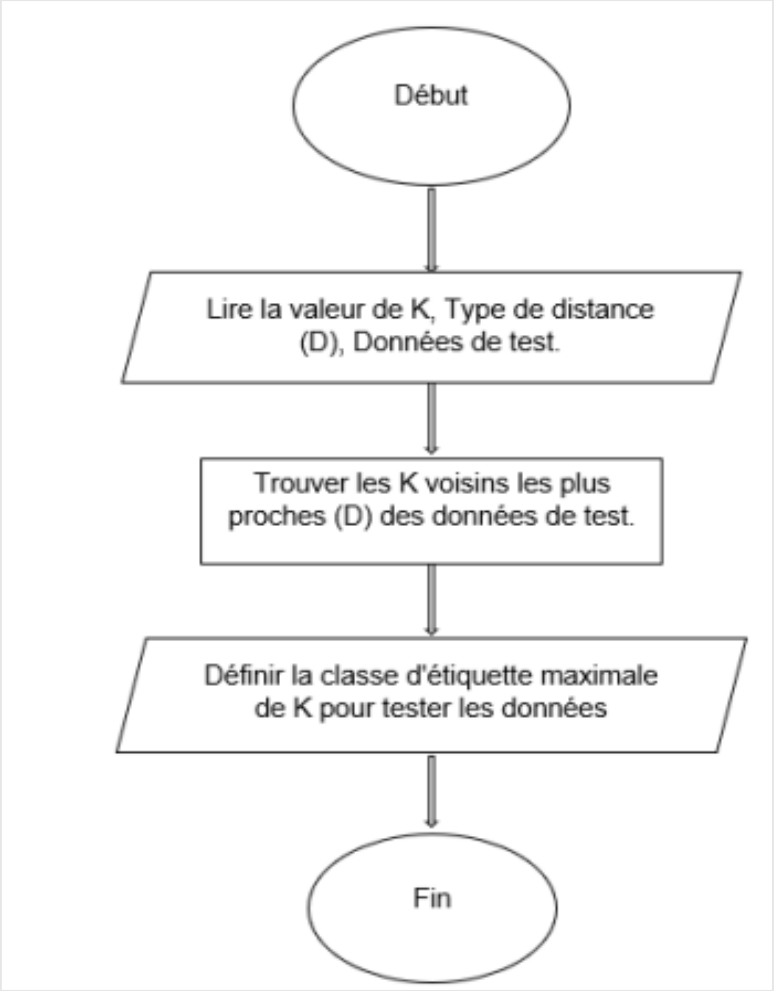


Figure 16: Organigramme de l'algorithme KNN.[18]

1.7.5.6 Algorithme génétique

Les algorithmes génétiques (AG) appartiennent à la famille des algorithmes évolutionnistes. Les principaux opérateurs de AG sont la sélection, croisement et mutation, AG s'inspire de la théorie darwinienne de l'évolution, dans laquelle la survie de créatures plus en forme et leurs gènes ont été simulés. Le but des AG est d'obtenir une solution approchée à un problème d'optimisation, lorsqu'il n'existe pas de méthode exacte (ou que la solution est inconnue) pour le résoudre en un temps raisonnable.[14]

Le processus d'optimisation se poursuit par des itérations successives, au cours desquelles les solutions de la population sont évaluées en fonction de leur aptitude à résoudre le problème, et les meilleures solutions sont sélectionnées pour la prochaine génération.

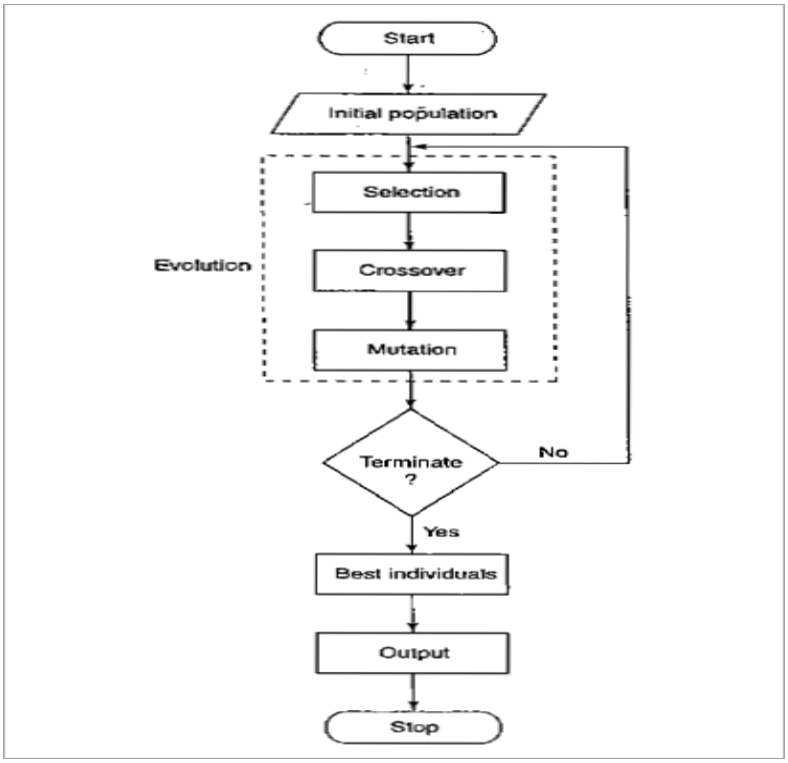


Figure 17: Un algorithme génétique simple. [38]

❖ **Certaines applications où AG est utilisé :**

- Apprentissage automatique : Les AG ont été utilisés pour résoudre des problèmes liés à la classification, à la prédiction, créer des règles d'apprentissage et de classification.
- Problème d'optimisation : L'un des meilleurs exemples de problèmes d'optimisation est le problème du vendeur de voyages qui utilise AG. D'autres problèmes d'optimisation tels que la planification des travaux, l'optimisation de la qualité sonore des AG sont largement utilisés.
- Modèle du système immunitaire : Les AG sont utilisés pour modéliser divers aspects du système immunitaire pour des familles individuelles de gènes et de gènes multiples au cours de l'évolution.[38]

❖ **Avantages :** [38]

- Il dispose d'un espace de solution plus large.
- Il est plus facile de découvrir l'optimum global.
- Parallélisme : Plusieurs AG peuvent fonctionner ensemble en utilisant le même processeur sans interférer les uns avec les autres. Ils fonctionnent parallèlement dans l'isolement.

❖ **Inconvénients :** [38]

- La convergence des algorithmes peut être trop rapide ou trop lente.
- Il existe une limitation de la sélection des paramètres tels que le croisement, la probabilité de mutation, la taille de la population, etc. [38]

1.7.5.7 Les réseaux de neurones

Les réseaux de neurones sont un type de modèle d'apprentissage automatique qui s'inspire de la structure et de la fonction du cerveau humain. Ils sont composés de nœuds interconnectés ou "neurones" qui traitent les informations et font des prédictions basées sur ces

informations. De nombreuses tâches, telles que la reconnaissance vocale, le traitement du langage naturel, la reconnaissance d'images, etc., peut être réalisée à l'aide de réseaux de neurones.

Voici quelques algorithmes couramment utilisés dans les réseaux de neurones :

Perceptron, perceptron multicouche (MLP), réseaux de neurones convolutifs (CNN), réseaux de neurones récurrents (RNN), Mémoire longue à courte terme (LSTM), réseaux antagonistes génératifs (GAN).

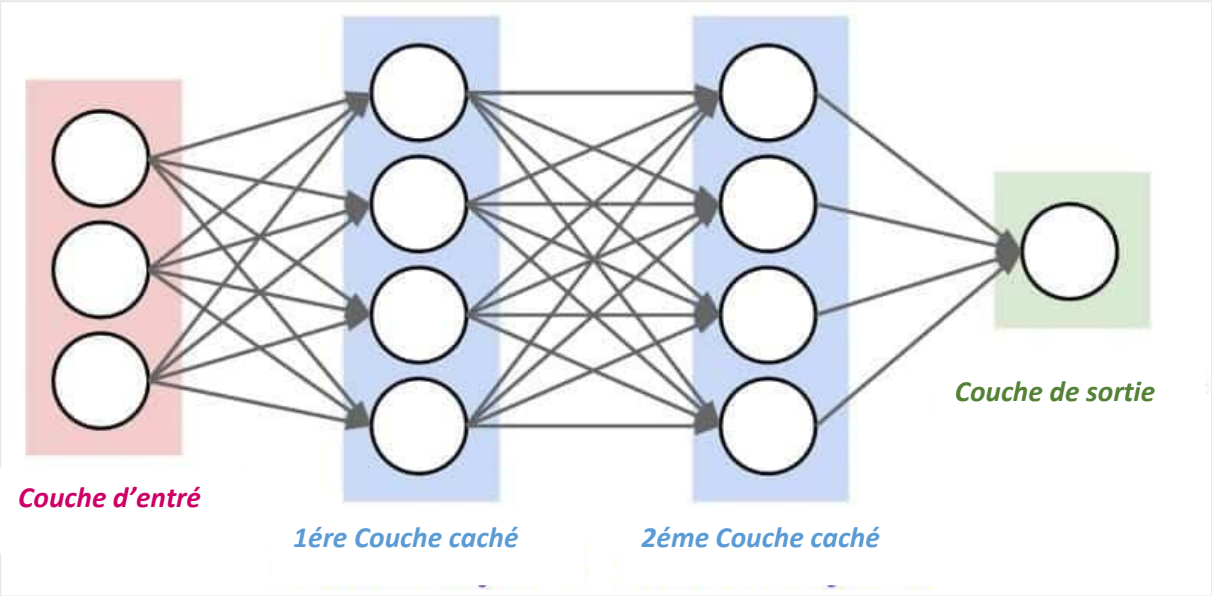


Figure 18: Exemple d'un réseau neuronal.[56]

1.8 Description de différents travaux ultérieurs

La reconnaissance d'un seul chiffre est un problème courant en vision par ordinateur et en apprentissage automatique, et de nombreux travaux ultérieurs ont tenté d'améliorer la précision et l'efficacité de cette tâche. Voici quelques exemples de travaux ultérieurs sur la reconnaissance à un seul chiffre :

1.8.1 Reconnaissance de chiffres manuscrits isolés

1.8.1.1 Le modèle de Ritiki, Rishika, et Samay (2021) [8]

Dans cette recherche, l'auteur implémente trois modèles de reconnaissance manuscrite des chiffres à l'aide d'ensembles de données MNIST (ensemble de données des chiffres manuscrits), basée sur des algorithmes d'apprentissage automatique et profond (SVM, ANN-MLP, CNN). Ils les ont comparés sur la base de leurs caractéristiques (temps d'exécution, la complexité, taux de précision, nombre d'époques et nombre de couches cachées dans le cas d'algorithmes d'apprentissage profond) pour évaluer le modèle le plus précis parmi eux.

Le modèle en entré utilise l'étape de prétraitement qui se concentre sur l'amélioration des données d'entrée en réduisant les impuretés indésirables et la redondance. Pour simplifier et décomposer les données d'entrée, les images présentes dans le jeu de données ont été remodelées en images bidimensionnelles, c'est-à-dire (28,28,1).

Modèle	Précision de formation	Précision du tests	Temps d'exécution
SVM	99,98%	94,005%	1 :35 min
MLP	99,92%	98,85%	2 :32 min
CNN	99,53%	99,29%	44 :02 min

Tableau 2: Analyse comparative de différents modèles.

Les machines à vecteurs de support sont l'un des classificateurs de base les plus rapide que la plupart des algorithmes, ils donnent le taux de précision d'entraînement maximal, mais en raison de sa simplicité, ce n'est pas possible de classer des images complexes et ambiguës aussi précisément que réalisé avec les algorithmes MLP et CNN. Ils ont découvert que CNN a donné les résultats les plus précis pour la reconnaissance des chiffres manuscrits. Donc, cela conclure que CNN est le mieux adapté à tout type de problème de prédiction incluant des données d'image en entrée. Ensuite, ils ont comparé le temps d'exécution pour mieux comprendre le fonctionnement des algorithmes. Généralement, le temps d'exécution d'un L'algorithme dépend du nombre d'opérations qu'il a effectuées. Donc, ils ont formé notre modèle d'apprentissage en profondeur jusqu'à 30 époques et SVM modèles selon les normes pour obtenir le résultat approprié. SVM a pris le temps minimum d'exécution tandis que CNN représente le temps maximal de marche.[8]

1.8.1.2 Le modèle de Kübra, et Ünal (2022) [55]

Dans cette étude, Un total de 6 algorithmes différents (ANN, CNN, K-NN, algorithme Naïve Bayes, SVM et arbres de décision) a été utilisés. Les modèles ANN et CNN ont obtenu les meilleurs résultats en termes de précision lors de la comparaison des modèles. Un taux de réussite de 98,66 % a été atteint avec le modèle ANN-Adam et 99,45% avec le modèle CNN-Adam. Les résultats expérimentaux ont montré que les réseaux de neurones étaient plus performants que les autres algorithmes étudiés. Cependant, l'architecture des réseaux de neurones et les fonctions d'activation choisies ont un impact significatif sur les performances. En ce qui concerne les algorithmes de classification traditionnels, on observe que les modèles K-NN et SVM peuvent atteindre un taux de réussite de plus de 97 %. En conséquence, presque tous les algorithmes de classification ont été examiné avec l'ensemble de données MNIST, contrairement à des études similaires dans la littérature. Des expériences ont été menées en variant les paramètres dans chacun d'eux, dans le but de trouver la combinaison la plus efficace.

Pendant ces expériences, le langage de programmation Python et par conséquent ; Bibliothèques Keras et Tensorflow pour MNIST ensemble de données, structures CNN et ANN ; La bibliothèque ScikitLearn était utilisée pour les rapports de résultats de test de K-NN, algorithme Naive Bayes, SVM, arbres de décision et modèles. Les rapports de résultats et valeur de précision obtenue à l'aide de la bibliothèque ScikitLearn mesures ont été utilisées pour comparer les taux de réussite des algorithmes. [55]

1.8.2 Reconnaissance une chaine de chiffres manuscrits

1.8.2.1 Le modèle de A.G., Hochuli, Oliveira L. S., Britto Jr A. S., et Sabourin R (2018) [10]

Dans ce travail, l'auteur démontre grâce à une série d'expériences complètes sur deux ensembles de données (synthétique et le NIST SD19), la segmentation numérique peut être

remplacée avec succès par des classificateurs CNN entraînés sur des données synthétiques pour deux tâches spécifiques, à savoir la longueur de la chaîne classification et classification des chiffres. La classification des chiffres comprend trois classificateurs chargés de classer les chiffres isolés, et chaînes à 2 chiffres et à 3 chiffres. Ces classificateurs sont utilisés en fonction de la sortie de la longueur classificateur. Pour éviter une décision difficile et surmonter une partie de la confusion causée par le classificateur de longueur, la méthode de fusion peut utiliser les sorties de classificateurs à deux chiffres pour produire la décision finale.

Des expérimentations sur la base de données mettent en évidence les avantages de la méthode proposée en réalisant un taux de reconnaissance de 97 %. Le résultat le plus proche rapporté dans la littérature atteint 93,8% de segmentation correctes. Mais avec un coût associé d'avoir à évaluer un grand nombre d'hypothèses qui sont créé par plus de 45 coupes de segmentation. [10].

1.8.2.2 Le modèle de Andre G, Hochuli, Oliveira Luiz S, Britto Jr Alceu de Souza, et Sabourin Robert (2018) [17]

Ce travail décrit un système de reconnaissance d'une chaîne des chiffres manuscrits sans contrainte, de longueurs connues, de taille 2, 3 et 4 pour former des solutions de bout en bout en utilisant une stratégie sans segmentation (segmentation-free) basée sur le classifieur Convolutional Neural Networks (CNNs) pour prendre des décisions. La technique s'est basée sur la méthode bien connu LeNet-5 pour effectuer des décisions plus efficaces. [3] Le tableau 3 présente les différentes phases (apprentissage, validation et test), ainsi que la quantité de données utilisées. Les chiffres isolés ont été extraits du NIST SD19 (voir la table 3), Les résultats rapportés sur (la table 4).

Longeur /Classes	Echantillons	Objet
1-chiffres isolés	197,784	Formation
10 Classes	23,384	Validation
	23 ,621	Tests
2-chaîne de chiffres	161,563	Formation
100 Classes	53,907	Validation
	55,091	Tests
3-chaîne de chiffres	1448,680	Formation
1000 Classes	484 ,346	Validation
	491,749	Tests
4-chaîne de chiffres	100,000	Formation
	20,000	Validation
	20,000	Tests

Tableau 3: Diffusion des données utilisées pour l'entraînement et les tests classificateurs. Les échantillons sont répartis uniformément entre les classes. [17]

Figure 19: Données synthétiques représentant des chaînes numériques touchantes composées de (a) 2 chiffres, (b) 3 chiffres et (c) 4 chiffres.

Le tableau (4) montre les résultats des trois approches sans segmentation abordées dans ce travail.

Méthode	Chiffre isolé	2-chiffres	3-chiffres
End-to-end	97.68	94.09	96.05
End-to-end+$\mathcal{L}$	98.73	96.82	95.50
Hochuli et al. (2018)	99.56	99.00	94.88

Tableau 4: Performance des approches sans segmentation sur les données synthétiques.[17]

1.9 Comparaison entre les travaux ultérieurs

Au fil des ans, de nombreux algorithmes d'apprentissage automatique différents ont été développés et testés pour la reconnaissance des chiffres manuscrits, avec plus ou moins de succès. Voici quelques-uns des algorithmes les plus populaires ont été qui utilisés, les différant bases de données et le taux obtenus par chaque méthode de classification comme illustré à la table suivante :

Auteurs	Années	Classifieur Utilisé	Base de données	Mode	Taux de précision
F. Menasri et N. Vincent	2008	CNN	MNIST	Hors-ligne	99,02%
N. VenkateswaraRao, A.Srikrishna, B. RaveendraBabu et G. Rama Mohan	2011	ANN	MNIST	Hors-ligne	95,60%
A.Gattal et al	2014	SVM	CVL	Hors-ligne	96,62%
A.Gattal et al	2016	SVM	CVL	Hors-ligne	95,21%
A.G.Hochuli et A.Robert	2018	CNN	NIST SD19	Hors-ligne	97%
Shruti R. Kulkarni, Bipin Rajendran	2018	SNN	MNIST	Hors-ligne	98 ,17%
S M Shamim et al	2018	Perceptron multicouche	Austria	Hors-ligne	90,37%
Junfei et Gongming	2018	Q-Learning	MNIST	Hors-ligne	99,18%
Savita & Amit	2020	CNN-SVM	MNIST	Hors-ligne	99,28%
Ritik&Richika	2021	CNN	MNIST	Hors-ligne	99,29%

Tableau 5: Classifieurs utilisés dans les systèmes de reconnaissance des chiffres manuscrits.

1.10 Conclusion

Dans ce chapitre, nous avons présenté un état de l'art des différents aspects et les techniques de la reconnaissance des chiffres manuscrits et que l'image doit être prétraitée pour s'assurer qu'elle est au format et à la taille appropriés pour être donnée au modèle. Ensuite, le modèle peut prédire le chiffre dans l'image en utilisant une classification. Dans le deuxième chapitre, nous aurons abordé les principaux méthodes et techniques de l`apprentissage profond « Deep Learning ».

Chapitre 2

Système de reconnaissance basé sur l'apprentissage profond

2.1 Introduction

Les systèmes de reconnaissance de l'écriture manuscrite sont disponibles depuis un certain temps, mais les nouveaux développements de l'apprentissage en profondeur ont complètement changé l'industrie. Les approches d'apprentissage en profondeur se sont révélées extrêmement prometteuses pour améliorer la précision et la fiabilité des systèmes d'identification manuscrite en raison de leur capacité à apprendre des modèles et des caractéristiques complexes à partir d'énormes volumes de données.

Les modèles d'apprentissage en profondeur tels que les réseaux de neurones convolutionnels (CNN) et les réseaux de neurones récurrents (RNN) ont atteint des performances de pointe sur de nombreuses tâches de reconnaissance manuscrite.

Dans ce chapitre, nous parlerons de l'apprentissage profond et des réseaux de neurones artificiels. Ensuite, nous discuterons sur les nombreuses méthodes d'apprentissage en profondeur, y compris les CNN, les RNN et leurs variantes, qui ont été utilisées pour la reconnaissance de l'écriture manuscrite et peuvent être appliquées pour augmenter la précision du système de reconnaissance. Nous parlerons également des difficultés et des restrictions de ces méthodes, comme la nécessité d'avoir beaucoup de données d'apprentissage et la difficulté à gérer différents styles d'écriture. Nous examinerons ensuite les spécificités de plusieurs architectures d'apprentissage en profondeur et leurs variantes, Dans la dernière partie, nous détaillerons les couches de réseaux neurones convolutifs telle que la couche conventionnelle, la couche de regroupement et la couche entiérement connecté. Les architectures comme AlexNet, VGGnet…. et comment ils ont aidé à améliorer la conception d'un neurone convolutif réseau.

2.2 L'apprentissage en profondeur

L'apprentissage en profondeur est un sous-ensemble de l'apprentissage automatique, les algorithmes d'apprentissage en profondeur sont un type spécifique d'algorithmes d'apprentissage automatique basés sur des réseaux de neurones artificiels. dans ce cas, le processus d'apprentissage repose sur les mêmes étapes que celles de l'apprentissage automatique, mais il est appelé profond car la structure des algorithmes est basée sur des réseaux de neurones artificiels constitués de plusieurs couches d'entrée, de sortie et cachées contenant des unités qui transforment l'entrée des données dans une information que la couche suivante peut utiliser pour effectuer une certaine tâche prédictive automatisée une fois le modèle d'apprentissage en profondeur déployé. [57]

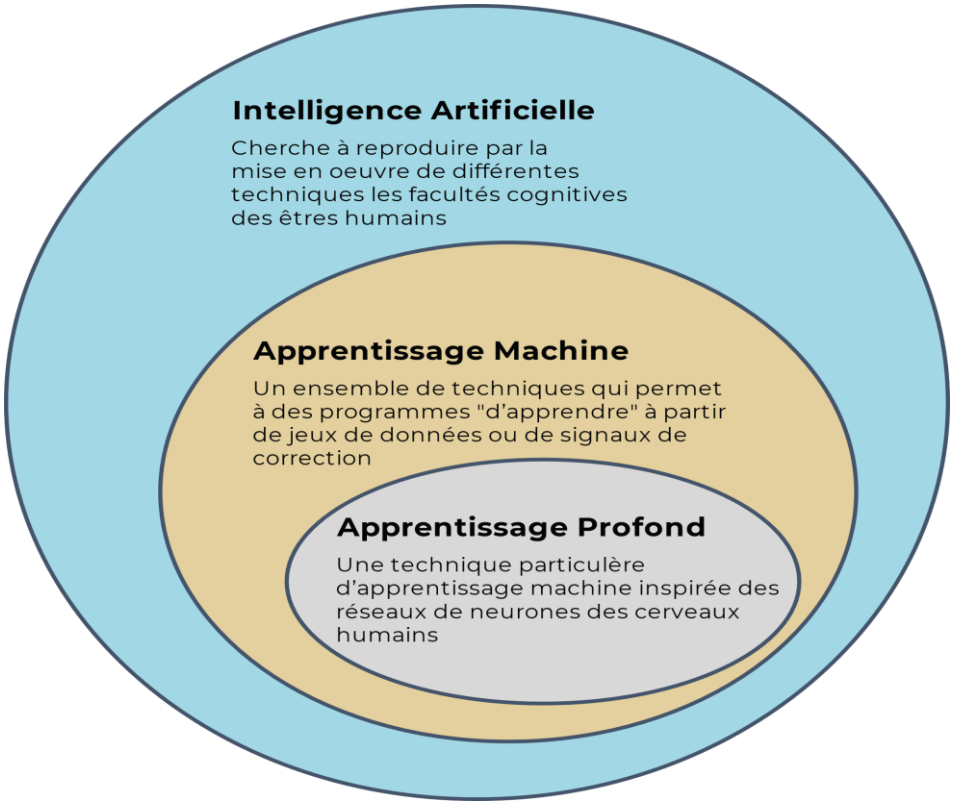


Figure 20: Intelligence artificiel, Apprentissage machine, Apprentissage profond. [98]

2.3 Les réseaux de neurones

Un réseau de neurones est un système logiciel et / ou matériel qui imite le fonctionnement des neurones biologiques. Les réseaux neuronaux, aussi appelés réseaux de neurones artificiels (RNA ou ANN en anglais), font partie des technologies d'apprentissage profond ou « deep learning », couvertes également par l'intelligence artificielle (IA). [87]

Un réseau neuronal est essentiellement un groupe de nœuds interconnectés, ou de neurones, qui coopèrent pour effectuer une tâche spécifique. Chaque couche de ces neurones traite une partie différente des données d'entrée. Ces neurones sont organisés en couches. La couche de sortie crée le résultat souhaité, tel qu'une classification ou une prédiction, tandis que la couche d'entrée accepte les données (comme une image ou un texte).

2.4 Les types de réseau de neurones

Il existe de nombreuses variétés de réseaux de neurones, notamment perceptron multicouche (MLP), les réseaux de neurones convolutionnels et les réseaux de neurones récurrents. Chaque type de réseau est le mieux adapté à des applications et des problèmes particuliers.

La reconnaissance d'images et de la parole est l'une des principales utilisations des réseaux de neurones. Étant donné qu'ils peuvent automatiquement s'entraîner à reconnaître les caractéristiques des images qui sont pertinentes pour la tâche à accomplir, les réseaux de neurones convolutifs (CNN) sont particulièrement bien adaptés aux tâches de reconnaissance d'images. Comme ils peuvent gérer des données séquentielles, comme les signaux audios, les

réseaux de neurones récurrents (RNN) sont fréquemment utilisés dans les applications de reconnaissance vocale.

2.4.1 Perceptrons multicouches (MLP)

MLP est l'algorithme d'apprentissage en profondeur le plus basique et également l'une des plus anciennes techniques. Les MLP peuvent être appelés une forme de réseaux de neurones Feedforward. [76]

Dans un réseau feedforward, les informations circulent dans une seule direction, de la couche d'entrée vers la couche de sortie. [40]

Le nombre de nœuds dans la couche d'entrée dépend du nombre d'attributs présents dans le jeu de données. Le nombre des nœuds dans la couche de sortie dépend du nombre des classes apparentes qui existent dans l'ensemble de données. Le pratique nombre de couches cachées ou le nombre pratique de nœuds dans une couche cachée pour un problème spécifique est difficile à déterminer. Mais en général, ces nombres sont sélectionnés expérimentalement. Dans le perceptron multicouche, la connexion entre deux nœuds consiste en un poids. Pendant la formation processus, il apprend essentiellement le poids précis réglage qui correspond à chaque connexion. [83]

La figure représente un exemple de MLP. Le diagramme calcule les pondérations et les biais et applique des fonctions d'activation appropriées pour classer les images des chats et des chiens. [25]

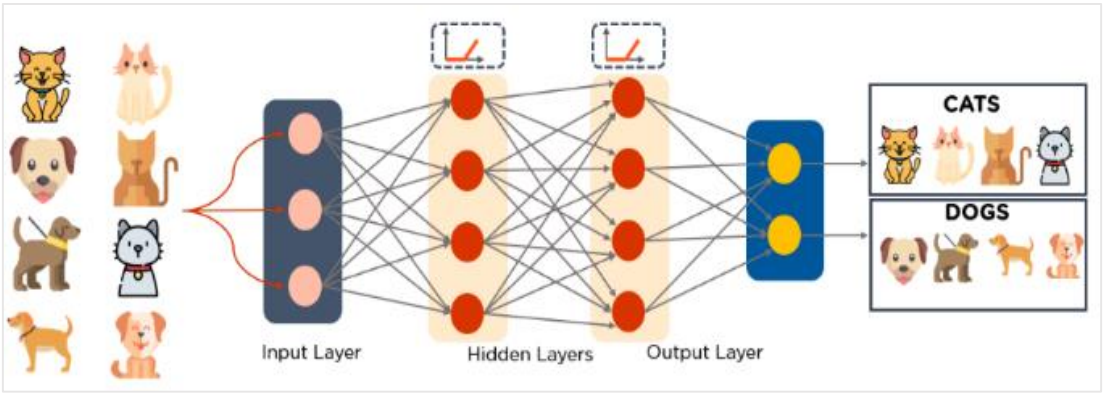


Figure 21: Un exemple de MLP. [25]

À l'exception des nœuds d'entrée, chaque nœud est un neurone utilisant une fonction d'activation non linéaire. MLP utilise une technique d'apprentissage supervisée appelée rétropropagation pour la formation. Ses multiples couches et son activation non linéaire distinguent MLP d'un perceptron linéaire. Il peut distinguer des données qui ne sont pas séparables linéairement. [3]

À chaque connexion neuronale est associé un poids W, comme pour les entrées du perceptron. Le calcul de la sortie d'un neurone se fait selon la fonction d'activation qui a été choisie. [45]

❖ **Fonction d'activation :**

La fonction d'activation d'un réseau permet d'introduire de la non-linéarité dans la fonction y = f(x). C'est elle qui permet de transformer les données afin qu'elles soient linéairement séparables. [70]

Il existe différentes fonctions d'activation possible pour un neurone. Remarquons qu'il est important que la fonction d'activation soit dérivable, car il sera impossible de calculer l'erreur individuelle de chaque neurone utilisé si la fonction utilisée n'est pas dérivable. Actuellement, la fonction la plus populaire est la fonction Relu, car elle nécessite peu de ressource processeur pour être calculée et permet d'obtenir des résultats similaires à la fonction sigmoïde qui est très précise. [45]

Lors du choix de la fonction d'activation, le programmeur regardera le succès qu'obtient le réseau de neurones dans la tâche qui lui à été attribué, mais aussi le temps d'exécution de la fonction d'activation. [45]

2.4.2 Réseaux de neurones récurrents

Un Réseau de neurone récurrent (RNN) est un algorithme couramment utilisé et familier dans la discipline de DL. RNN est principalement appliqué dans le domaine du traitement de la parole et des contextes NLP. [15]

Il aide à modéliser des données séquentielles dérivées de réseaux à anticipation. Il fonctionne de la même manière que le cerveau humain pour fournir des résultats prédictifs. [50]

Un réseau de neurones récurrent est un réseau de neurones dont le graphe de connexion contient au moins un cycle. Contrairement aux « MLP », les réseaux de neurones récurrents « Recurrent neural networks ou RNN », présentés dans « la figure 2.15 », comportent des cycles au sein du graphe de neurones [3]

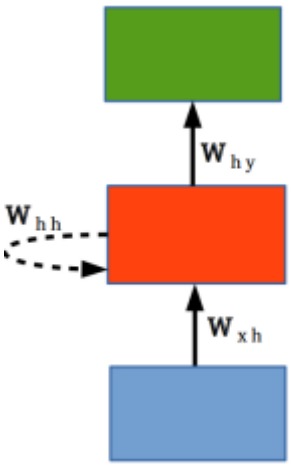


Figure 22: Représentation compacte des RNN. [3]

Toutes les flèches représentent des connexions complètes. La flèche en pointillée représente les connexions ayant un décalage temporel « t – 1 ». [3]

Récurrent signifie que la sortie au pas de temps actuel devient l'entrée au pas de temps suivant. À chaque élément de la séquence, le modèle considère non seulement l'entrée actuelle, mais ce dont il se souvient des éléments précédents. [97]

La motivation principale derrière ce type d'architectures est de pouvoir manipuler des séquences de vecteurs d'entrée, représentant chacun un événement temporel, et non pas seulement des données isolées n'ayant pas de signification temporelle. En déroulant, par rapport au temps, la modélisation compacte d'un « RNN » « voir la figure », ce type de réseaux peut ainsi être considéré comme une suite temporelle de réseaux « MLP » reliés entre eux à travers leurs couches cachées respectives. Cette liaison permet aux « RNN » d'encoder des dépendances latentes entre les événements d'une séquence de vecteurs d'entrée. [3]

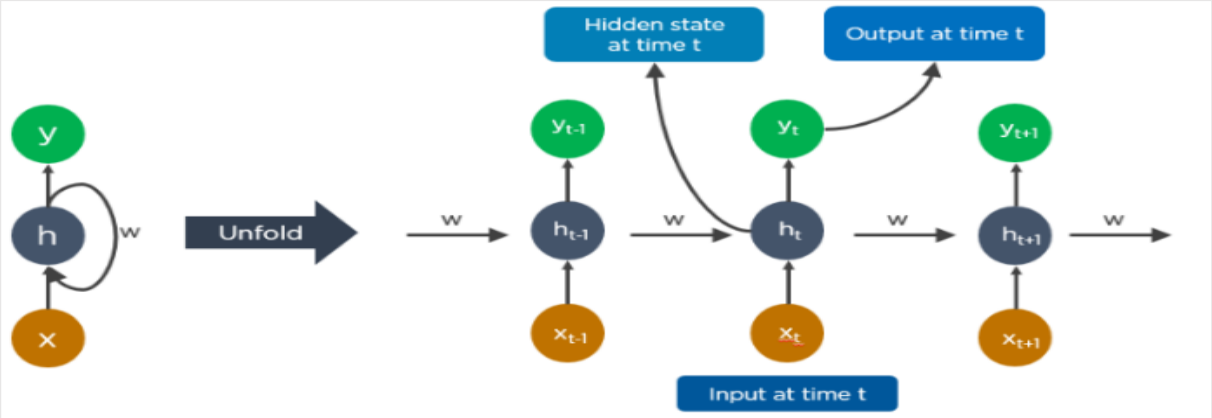


Figure 23: Représentation dépliées des RNNs.[25]

La sensibilité de RNN au gradient explosif et aux problèmes de disparition représente l'un des principaux problèmes de cette approche. Plus précisément, pendant le processus d'apprentissage, les réduplications de plusieurs grands ou petits dérivés peuvent provoquer une explosion ou une décroissance exponentielle des gradients. Avec l'entrée de nouveaux apports, le réseau cesse de penser aux premiers ; par conséquent, cette sensibilité diminue avec le temps. De plus, ce problème peut être traité à l'aide de LSTM. [15]

La capacité des modèles RNN à se souvenir des informations tout au long de la période de formation joue un rôle central dans la prédiction des séries chronologiques. Par contre il est Difficile de traiter de longues séquences dans l'ensemble de données d'apprentissage, principalement lorsque nous utilisons ReLU ou tanh comme fonctions d'activation. [76]

Les réseaux de neurones récurrents peuvent contenir plusieurs couches, ce qui leur permet de capturer davantage de non-linéarité parmi les données, mais augmente également le temps de calcul en phase d'apprentissage. [73]

Il existe trois grands types de réseaux de neurones récurrents : le RNN simple, le LSTM et enfin le GRU. [73]

2.4.2.1.RNN simple

Les RNNs « classiques » (réseaux de neurones récurrents simples ou Vanilla RNNs) sont la forme la plus basique de RNN. Ils ne sont pas capables de mémoriser que le passé dit proche, et commencent à « oublier » au bout d'une cinquantaine d'itérations environ. [31]

Bien que les RNN sont très pratiques comparé à une architecture ANN classique pour le traitement des données séquentielles, il s'avère qu'ils sont extrêmement difficiles à entraîner pour gérer la dépendance à long terme en raison du problème de la disparition du gradient. Une explosion de gradient peut également se produire mais très rarement. [91]

2.4.2.2.Longue mémoire à court terme (LMCT)

LMCT « Long Short-Term Memory – LSTM en Anglais » est une architecture de réseau de neurones récurrents « RNNs » spécifique conçue pour modéliser des séquences temporelles et il est très capables d'apprendre les dépendances à long terme. [3][76]

Pour surmonter le problème du vanishing du gradient, le **LSTM** est proposé initialement par S. Hochreiter et al, puis amélioré dans l'article de F. Gers et j. Schmidhuber. L'unité LSTM illustré dans la figure 3 est le composant de base d'une architecture LSTM. C'est une série de portes et de cellules qui coopèrent pour produire un résultat final. [91]

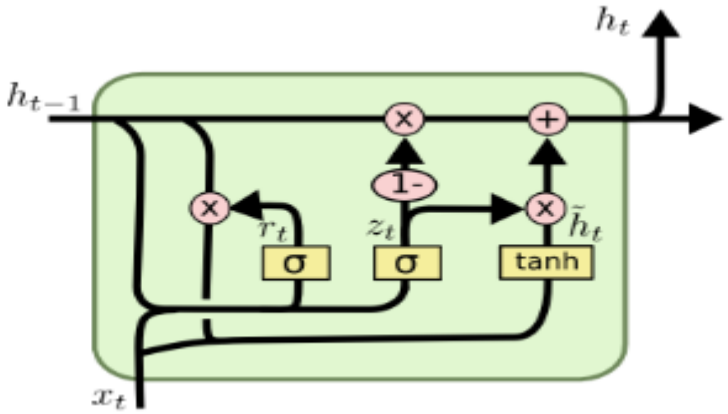


Figure 24: Unité de base LSTM. [29]

Cette approche offre des connexions récurrentes aux blocs de mémoire du réseau. Chaque bloc mémoire contient un certain nombre de cellules mémoire, qui ont la capacité de stocker les états temporels du réseau. [15]

Les LSTM ont également cette structure en forme de chaîne, mais le module répétitif a une structure différente. Au lieu d'avoir une seule couche de réseau de neurones, il y en a quatre, qui interagissent d'une manière très spéciale. [29]

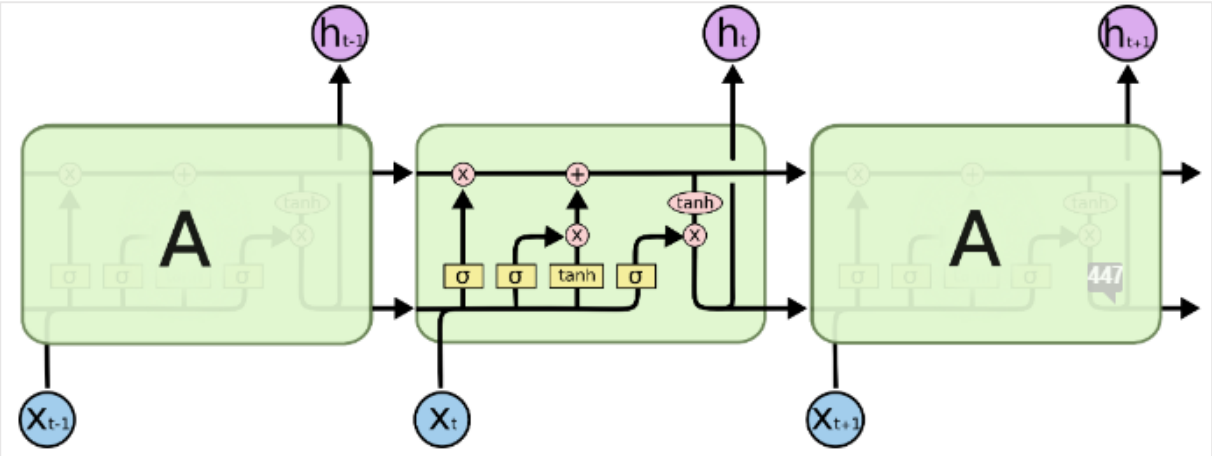


Figure 25: Architecture du modèle de mémoire à court terme et dépliée. [29]

❖ **Fonctionnement des LSTM**

Une cellule LSTM des réseaux de neurones récurrents est beaucoup plus complexe qu'une cellule de RNN traditionnelle ou qu'un neurone traditionnel. Une unité LSTM commune est composée d'une cellule, d'une porte d'oubli (Forget Gate), d'une porte d'entrée (Input Gate) et d'une porte de sortie (Output Gate) et gère une mémoire dynamique (notée C) qui évolue en fonction de la séquence de données temporelle. [73]

Les LSTM sont très pratiques pour modéliser les séquences chronologiques et les dépendances à longue portée. Par contre Des calculs et des ressources élevés sont nécessaires pour former le modèle LSTM, c'est aussi un processus qui prend beaucoup de temps et Ils sont sujets au surajustement. [76]

2.4.2.3.Unité récurrente fermée (GRU)

L'unité récurrente fermée GRU (Gated Recurrent Unit) a été introduite en 2014 par Cho et Al [8] pour résoudre le problème de disparition du gradient rencontré par les réseaux récurrents classiques mais aussi pour proposer une architecture avec moins de paramètres à entrainer par rapport à une LSTM. [91]

Il existe une couche d'entrée composée de plusieurs neurones, le nombre de neurones est déterminé par la taille de l'espace des caractéristiques. De même, le nombre de neurones dans la couche de sortie correspond à l'espace de sortie. La ou les couches cachées contenant les cellules mémoire couvrent les fonctions principales des réseaux GRU. [4]

GRU a deux portes (*porte de réinitialisation et de mise à jour)*. Il utilise moins de paramètres de formation et utilise donc moins de mémoire, s'exécute plus rapidement et s'entraîne plus rapidement que LSTM, tandis que LSTM est plus précis sur un ensemble de données utilisant une séquence plus longue. LSTM a trois valeurs en sortie (output, hidden et cell) alors que GRU a deux valeurs en sortie (output et hidden). [81]

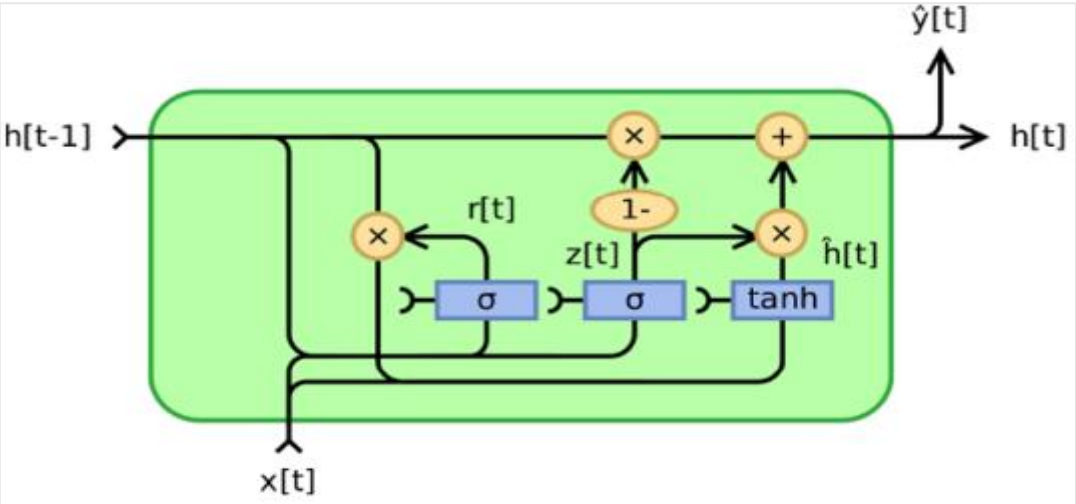


Figure 26: Unité de base GRU. [91]

2.4.3 Les réseaux de neurones convolutifs (CNN)

Les réseaux de neurones convolutifs, également connus sous le nom de CNN ou ConvNets, sont un type de réseau de neurones artificiels à anticipation dont la structure de connectivité s'inspire de l'organisation du cortex visuel humain et animal.De petits amas de cellules dans le cortex visuel sont sensibles à certaines zones du champ visuel. Les cellules neuronales individuelles du cerveau ne réagissent ou ne se déclenchent que lorsque certaines orientations des bords sont présentes. Certains neurones s'activent lorsque des bords verticaux sont affichés, tandis que d'autres se déclenchent lorsqu'ils sont affichés sur des bords horizontaux ou diagonaux. [2]

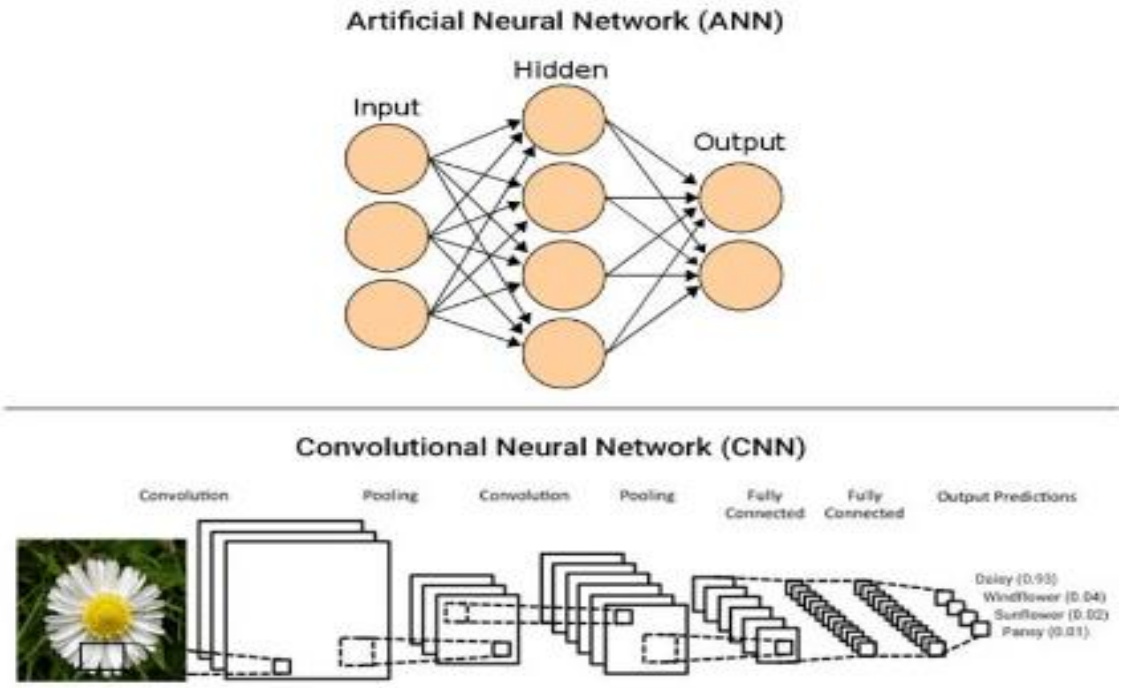


Figure 27: Réseau neuronal artificiel vs réseau neuronal convolutifs. [75]

Dans le domaine du Deep Learning, le CNN est l'algorithme le plus connu et le plus couramment utilisé. Le principal avantage de CNN par rapport à ses prédécesseurs est qu'il

identifie automatiquement les fonctionnalités pertinentes sans aucune supervision humaine.Les CNN ont été largement appliqués dans une gamme de domaines différents, y compris la vision par ordinateur, le traitement de la parole, la reconnaissance faciale, etc. [15]

Yann LeCun a développé le premier CNN en 1988 sous le nom de LeNet.Il a été utilisé pour reconnaître des caractères comme les codes postaux et les chiffres. [25]

Avant l'utilisation des GPU, les ordinateurs n'étaient pas en mesure de traiter de grandes quantités de données d'image dans un délai raisonnable, et la formation était effectuée uniquement sur des images à faible résolution. C'est pourquoi les réseaux de neurones n'ont fait leur apparition qu'en 2010. [1]

Le terme "convolution" dans CNN désigne la fonction mathématique de convolution qui est un type spécial d'opération linéaire dans laquelle deux fonctions sont multipliées pour produire une troisième fonction qui exprime comment la forme d'une fonction est modifiée par l'autre. En termes simples [5], L'image d'entrée est alambiquée avec l'application de filtres dans les CNN, ce qui donne une carte des fonctionnalités. Les filtres sont des poids et des biais qui sont des vecteurs générés aléatoirement dans le réseau. Au lieu d'avoir des poids et des biais individuels pour chaque neurone, CNN utilise les mêmes poids et biais pour tous les neurones. De nombreux filtres peuvent être créés, chacun capturant un aspect différent de l'entrée. Les noyaux sont un autre nom pour les filtres. [2]

Les CNN fonctionnent avec des images en niveaux de gris et RVB. Une image est constituée de pixels. Dans l'apprentissage en profondeur, les images sont représentées sous forme de matrice tridimensionnelle (largeur x hauteur x profondeur) de valeurs allant de 0 à 255. [9][7]

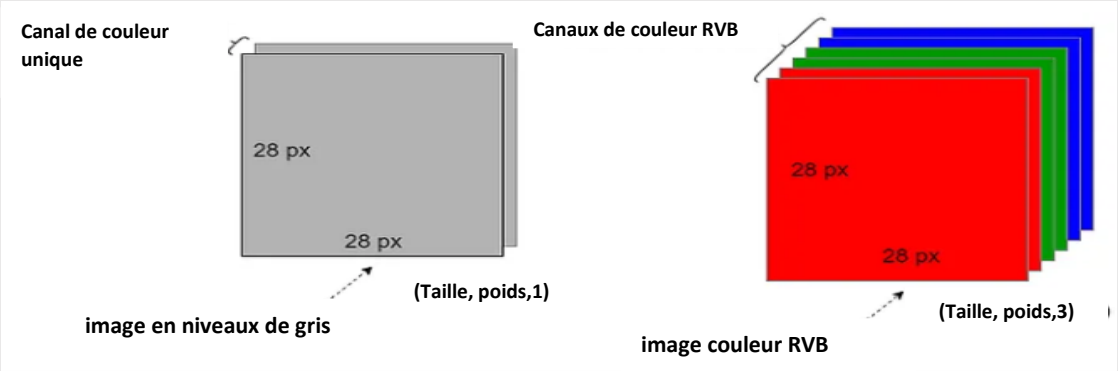


Figure 28: Représentation d'image en niveaux de gris vs RVB. [9]

2.4.3.1.Les types de couches de réseau neurones convolutionnels

2.4.3.1.1. Couches convolutives et opération de convolution

Dans les réseaux de neurones convolutifs, les principaux éléments de construction sont les couches convolutives. Cette couche contient souvent des vecteurs d'entrée, tels qu'une image, des filtres, tels qu'un détecteur d'entités, et des vecteurs de sortie, tels qu'une carte d'entités. L'image est extraite d'une carte de caractéristiques, également connue sous le nom de carte d'activation en anglais (**activation map** or **feature map**), après avoir traversé une couche convolutive. [2]

Carte d'entités = Image d'entrée x Détecteur d'entités [2]

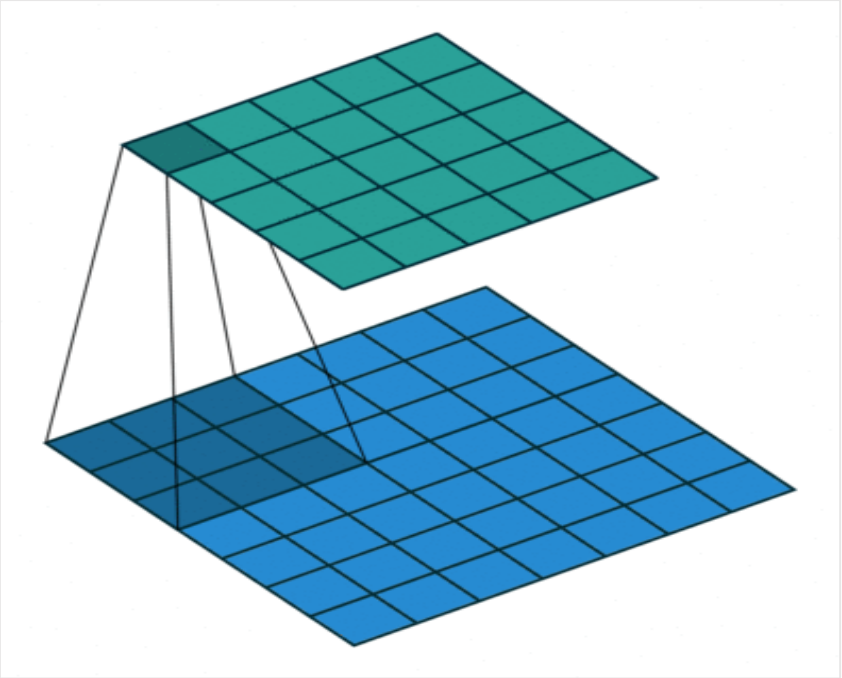


Figure 29: Couches convolutives et opération de convolution. [7]

La figure représente l'enveloppe ou l'effacement de l'image (en bleu) et le stockage des résultats du produit interne dans la carte d'activation (en vert). [7]

- **Filtre :**

Ceci est également appelé ***Kernel*** ou ***Feature Detector***. Il s'agit d'une petite matrice. Il peut y avoir plusieurs filtres dans une même couche convolutive. Les filtres de même taille sont utilisés dans une couche convolutive. Chaque filtre a une fonction spécifique. Plusieurs filtres sont utilisés pour identifier un ensemble différent de caractéristiques dans l'image. La taille du filtre et le nombre de filtres doivent être spécifiés par l'utilisateur en tant qu'hyperparamètres. La taille doit être inférieure à la taille de l'image d'entrée. Les éléments à l'intérieur du filtre définissent la *configuration du filtre*. Ces éléments sont un type de paramètres dans le CNN et sont appris au cours de la formation. [9]

Le filtre est suffisamment petit pour balayer l'intégralité de l'image et appliquer les opérations mathématiques appropriées entre les valeurs du filtre et les pixels afin d'extraire les caractéristiques. [7]

La connectivité locale fait référence aux images représentées dans une matrice de valeurs de pixels. La dimension augmente en fonction de la taille de l'image. Si tous les neurones sont connectés à tous les neurones précédents comme dans une couche entièrement connectée, le nombre de paramètres augmente considérablement. [1]

Pour résoudre ce problème, nous connectons chaque neurone à un seul patch de données d'entrée. Cette étendue spatiale (également appelée *champ récepteur du neurone*) détermine la taille du filtre. [1]

L'opération de convolution n'est rien d'autre qu'une opération *de somme multipliée par élément* entre une section d'image et le filtre. Voir le schéma suivant : [9]

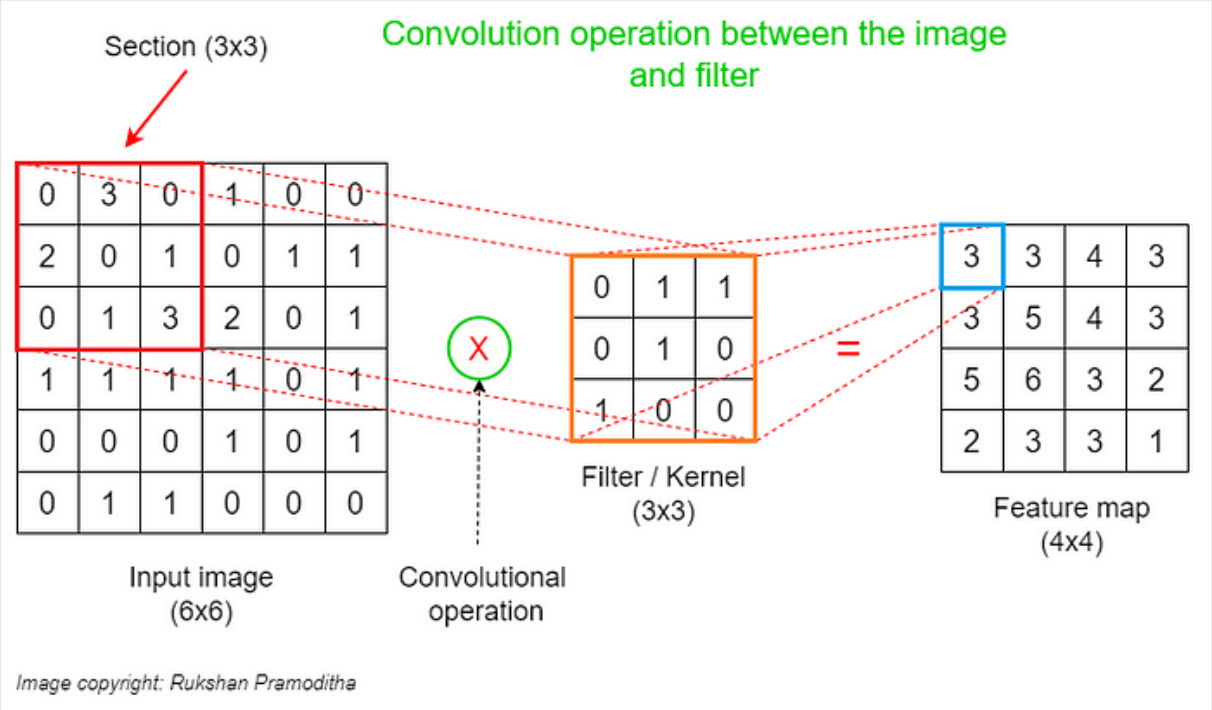


Figure 30: Opération de convolution. [9]

Le produit scalaire est calculé pour les pixels d'entrée et les poids du filtre. Ce processus, également connu sous le nom de convolution, se poursuit jusqu'à la fin de la matrice d'entrée. La sortie de chaque produit scalaire crée une matrice appelée carte d'activation ou carte de caractéristiques. La carte des caractéristiques contient les informations sur les caractéristiques qui ont été extraites de l'image à l'aide du filtre spécifique. [75]

- **Rembourrage**

Le rembourrage Ceci est également appelé padding est un hyperparamètre que nous devons configurer dans la couche convolutive. Il ajoute des pixels supplémentaires avec des valeurs nulles de chaque côté de l'image. Cela aide à obtenir la carte des fonctionnalités de la même taille que l'entrée. [9]

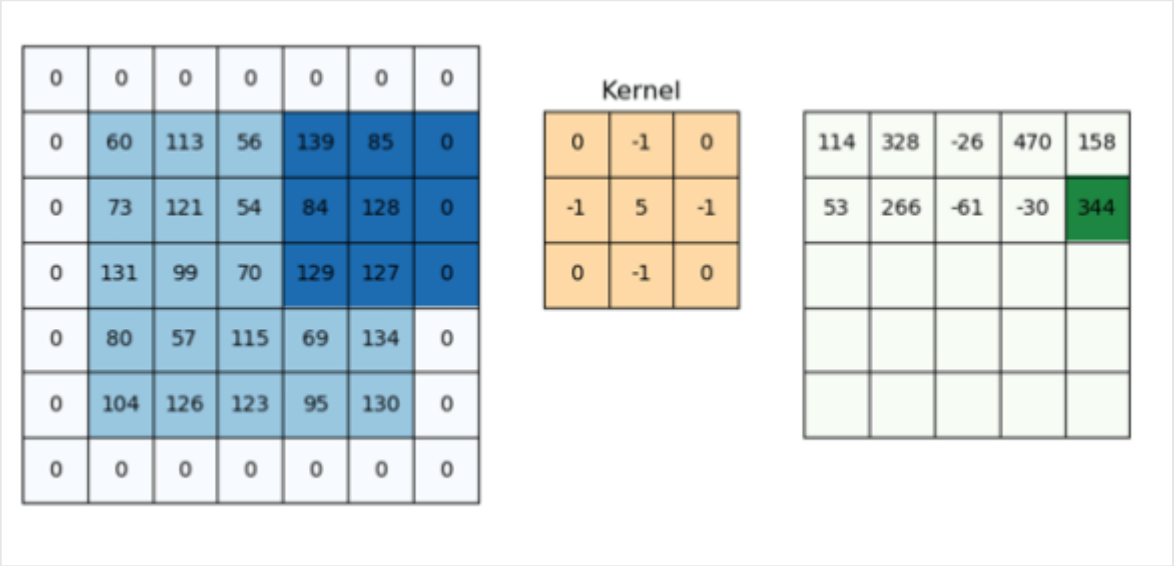


Figure 31: Rembourrage applique à l'image d'entrée. [2]

❖ Foulée

La foulée appelé stride est un hyperparamètre du filtre du réseau neuronal qui modifie la quantité de mouvement sur l'image ou la vidéo. Nous avons eu la foulée 1 donc ça va prendre un par un. Si nous donnons la foulée 2 alors elle prendra de la valeur en sautant les 2 pixels suivants. [33]

Plus la valeur de foulée est petite, plus la sortie est petite et vice versa. [2]

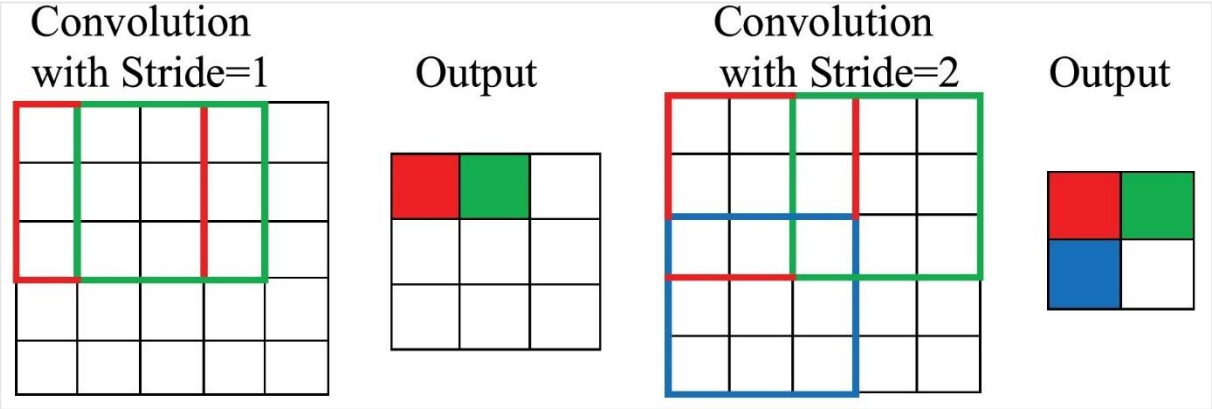


Figure 32: Une autre opération de convolution avec stride. [2]

Le partage de paramètres signifie que la même matrice de pondération agit sur tous les neurones d'une carte de caractéristiques particulière le même filtre est appliqué dans différentes régions de l'image. Les images naturelles ont des propriétés statistiques, l'une étant invariante à la traduction. Les CNN tiennent compte de cette propriété en partageant des paramètres sur plusieurs emplacements d'image. [1]

La couche de convolution dans CNN transmet le résultat (carte de caractéristiques) à la couche suivante une fois l'opération de convolution appliquée dans l'entrée, cette carte de caractéristiques est transmise pour apprendre plusieurs autres caractéristiques de l'image

d'entrée. Les couches convolutives dans CNN en bénéficient beaucoup car elles garantissent que la relation spatiale entre les pixels est intacte. [5]

2.4.3.1.2. Couche de regroupement

Le but de la couche de mutualisation est de réduire la taille des cartes d'activation et réduit la quantité de calculs nécessaires, ce qui la rend moins sujette au surajustement [7], [l'accélération des calculs et l'amélioration de l'efficacité de CNN. [75]

Il existe deux types de couches de regroupement :

a) La mise en pool max (Max Pooling)

La technique la plus populaire qui consiste à scanner la carte d'activation (ou feature array) avec une petite fenêtre [7] et le pixel d'entrée avec la valeur maximale est enregistré dans la matrice de sortie. [75]

b) La mise en commun moyenne (average pooling)

Calcule la moyenne des éléments présents dans la région de la carte d'entités couverte par le filtre. Il calcule simplement la moyenne des entités de la carte des entités. [1]

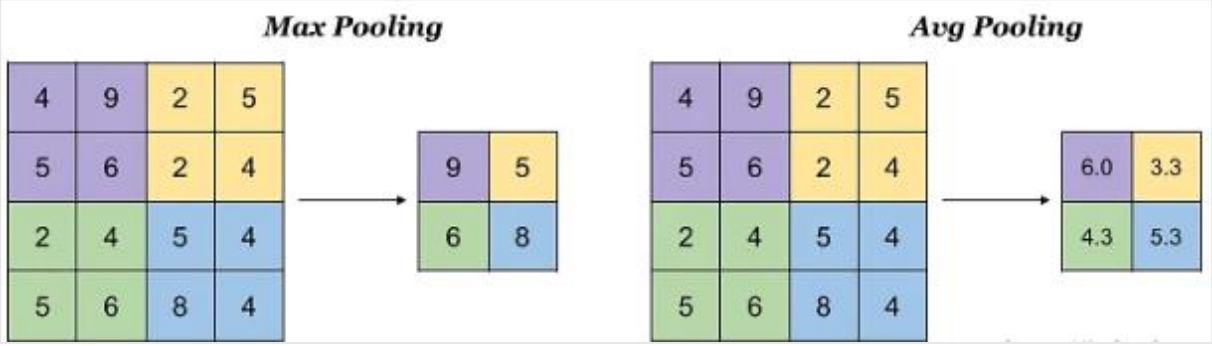


Figure 33: Les types de couches de regroupement. [75]

Chaque couche convolutive est suivie d'une couche de regroupement. Ainsi, les couches de convolution et de mise en commun sont utilisées ensemble par paires. [9]

L'idée de base de l'utilisation d'une couche de regroupement est Extrayez les fonctionnalités les plus importantes (pertinentes) en obtenant le nombre maximum ou en faisant la moyenne des nombres et Réduisez le nombre de paramètres dans le réseau pour Augmentez la précision des CNN. [9]

❖ **Fonction d'activation (non-linéarité)**

Des couches d'activation non linéaires sont utilisées après toutes les couches avec des poids (appelées couches apprenables, telles que les couches FC et les couches convolutives) dans l'architecture CNN. Cette performance non linéaire des couches d'activation signifie que le mappage de l'entrée à la sortie sera non linéaire ; de plus, ces couches donnent au CNN la capacité d'apprendre des choses très compliquées. [15]

❖ **Les différentes fonctions d'activation**

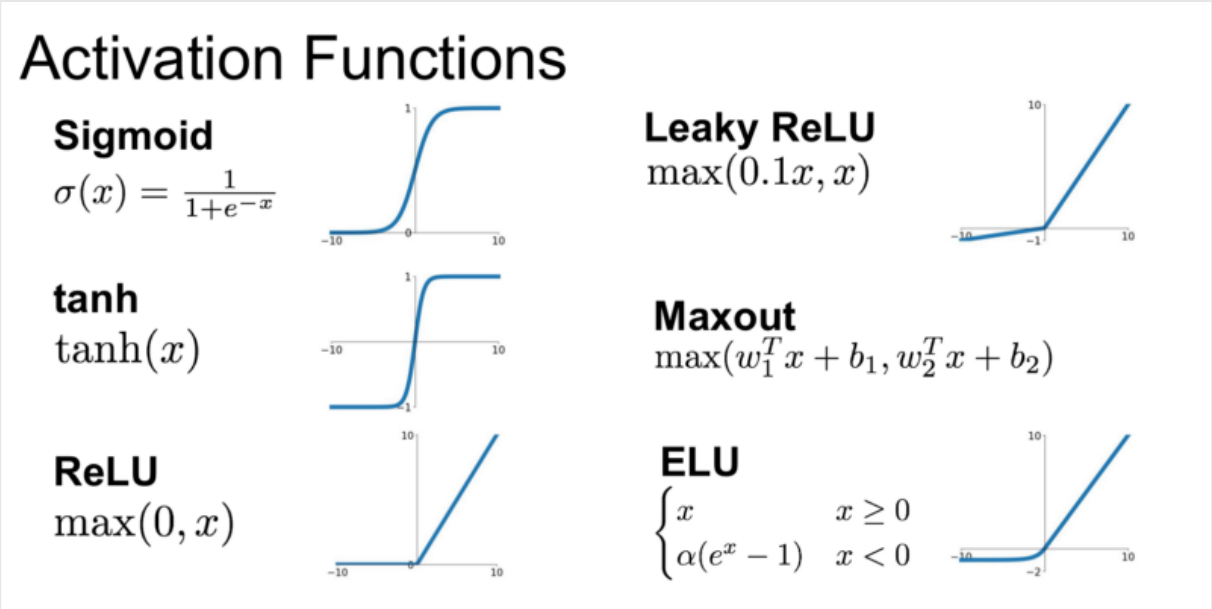


Figure 34: Les fonctions d'activation. [7]

❖ **ReLU :** La *fonction d'activation linéaire rectifiée,* ou ReLU en abrégé, Cette fonction permet d'effectuer un filtre sur les données. [53]

Si l'entrée est positive, il génère directement l'entrée ; sinon, il sort zéro.Parce qu'un modèle qui l'utilise est plus rapide à former et produit généralement de meilleures performances, il est devenu la fonction d'activation par défaut pour de nombreux types de réseaux de neurones. [2]

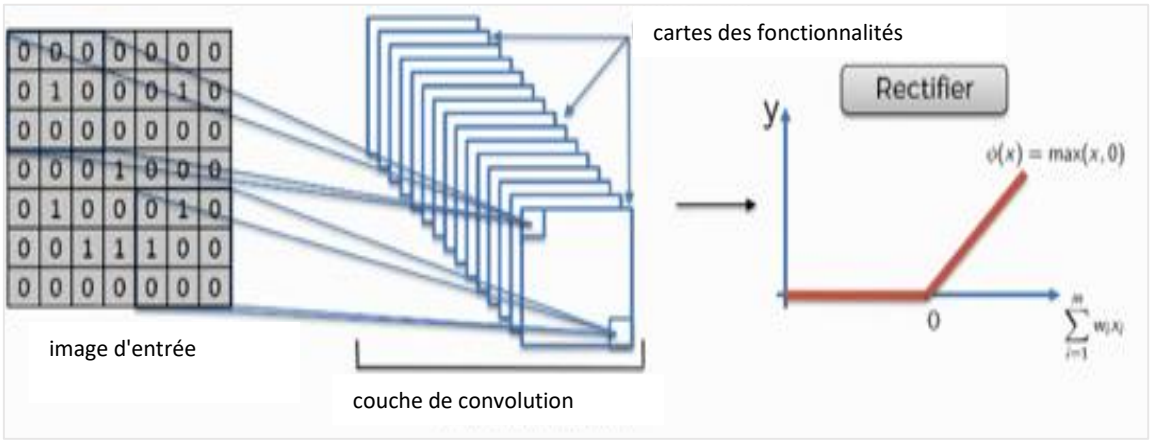


Figure 35: la fonction d'activation linéaire rectifiée. [2]

❖ **Softmax :** La fonction Softmax permet de transformer un vecteur réel en vecteur de probabilité. On l'utilise souvent dans la couche finale d'un modèle de classification, notamment pour les problèmes multiclasses. [53]

fonction_Softmax(x) = exp(x) / sum(exp(x_i))

❖ Opération d'aplatissement (Flattening)

L'aplatissement est le processus de conversion de tous les tableaux bidimensionnels résultants en un seul long vecteur linéaire continu [47]. Et il est relié au modèle de classification final, appelé couche entièrement connectée. [62]

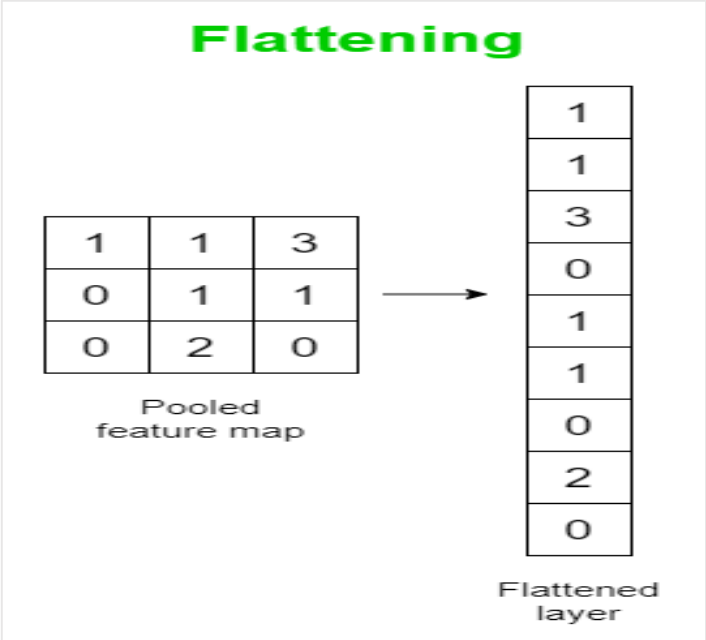


Figure 36: Aplatissement d'une carte de fonctionnalités regroupées sur un seul canal. [9]

Le diagramme suivant montre comment aplatir une carte de caractéristiques regroupées qui contient plusieurs canaux. [9]

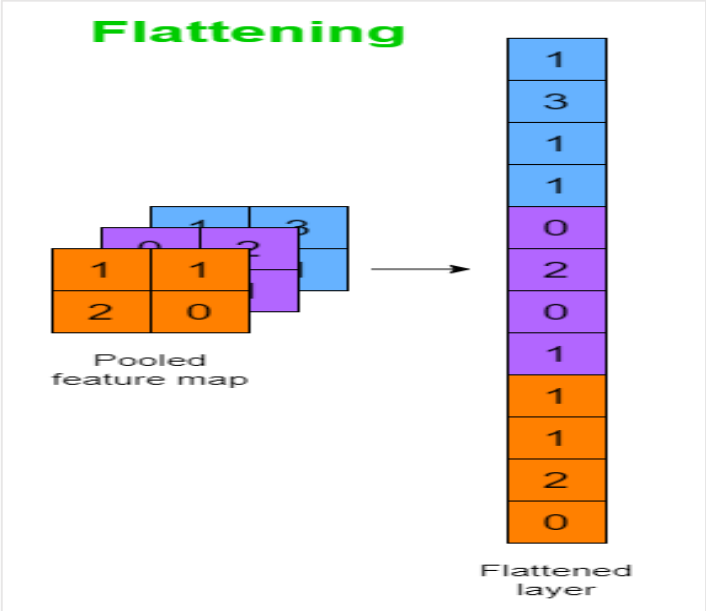


Figure 37: Aplatir une carte de fonctionnalités regroupées multicanaux. [9]

2.4.3.1.3. Couches entièrement connectées (denses)

Généralement, cette couche est située à la fin de chaque architecture CNN. A l'intérieur de cette couche, chaque neurone est connecté à tous les neurones de la couche précédente, l'approche dite Fully Connected (FC). [15]

Une opération d'aplatissement est appliquée : l'entrée est aplatie dans un vecteur de caractéristiques, puis transmise à un réseau de neurones pour prédire les probabilités de sortie. [7]

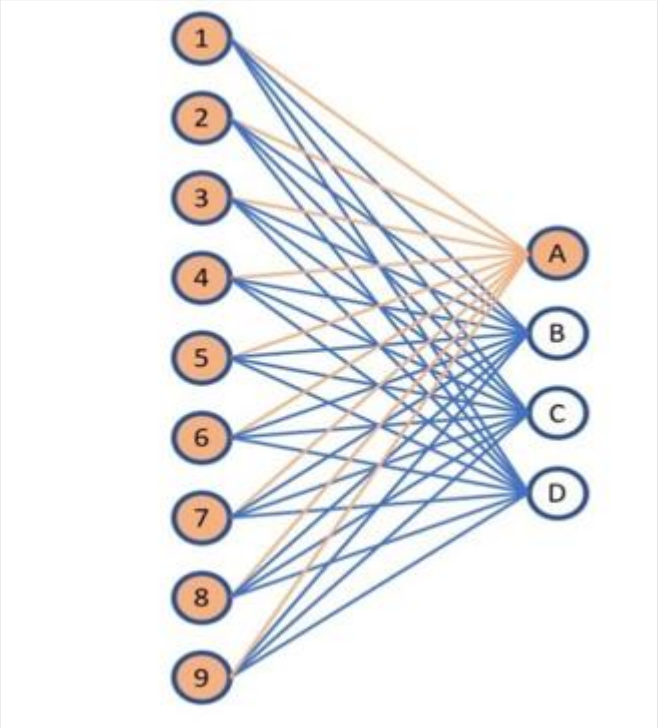


Figure 38: Illustration d'une couche entièrement connectée. [34]

L'image ci-dessus montre pourquoi nous appelons ces types de couches "entièrement connectées" ou parfois "densément connectées". Toutes les connexions possibles couche à couche sont présentes, ce qui signifie que chaque entrée du vecteur d'entrée influence chaque sortie du vecteur de sortie. Cependant, tous les poids n'affectent pas toutes les sorties. Regardez les lignes entre chaque nœud ci-dessus. Les lignes oranges représentent le premier neurone (ou perceptron) de la couche. Les poids de ce neurone n'affectent que la sortie A, et n'ont pas d'effet sur les sorties B, C ou D. [34]

2.4.3.2. Architectures CNN

Au fil des ans, un certain nombre d'architectures CNN largement utilisées ont été créées, chacune avec une structure et un ensemble de paramètres distincts. Voici quelques-unes des architectures CNN les plus connues :

2.4.3.2.1. Alex Net

Dans l'architecture CNN profonde, AlexNet est très utilisée, car il a obtenu des résultats innovants dans les domaines de la reconnaissance et de la classification d'images. Krizhevesky et al. a d'abord proposé AlexNet et a par conséquent amélioré la capacité d'apprentissage de

CNN en augmentant sa profondeur et en mettant en œuvre plusieurs stratégies d'optimisation des paramètres. La figure (40) illustre la conception de base de l'architecture AlexNet. [15]

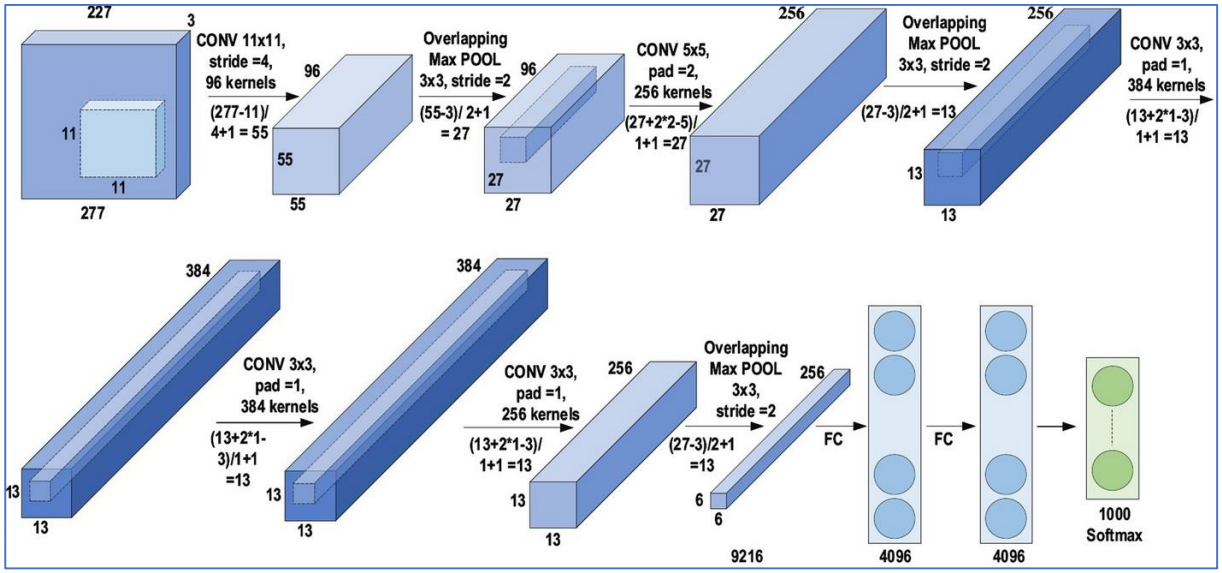


Figure 39: L'architecture d'AlexNet. [15]

Avec une précision de test de 84,6 %3, Alex Krizhevsky, Ilya Sutskever et Geoff Hinton ont remporté le défi de reconnaissance visuelle à grande échelle ImageNet en 2012. Les modèles CNN pourrait être formés plus rapidement, suscitant un regain d'intérêt et conduisant à de nouveaux travaux basés sur les CNN. [100]
Trois couches entièrement liées et cinq couches convolutionnelles constituent le réseau. [100]

2.4.3.2.2. GoogLeNet

La profondeur de GoogLeNet est de 22 couches. Ce modèle a remporté le concours ImageNet 2014 dans les tâches de classification et de détection avec une précision de 93,3 %. [100]

En outre, il utilise le regroupement moyen au lieu des couches entièrement connectées en haut du ConvNet, éliminant une grande quantité de paramètres qui semblent peu importants. Il existe également plusieurs versions de suivi du GoogLeNet, le plus récemment Inception-v4. [24]

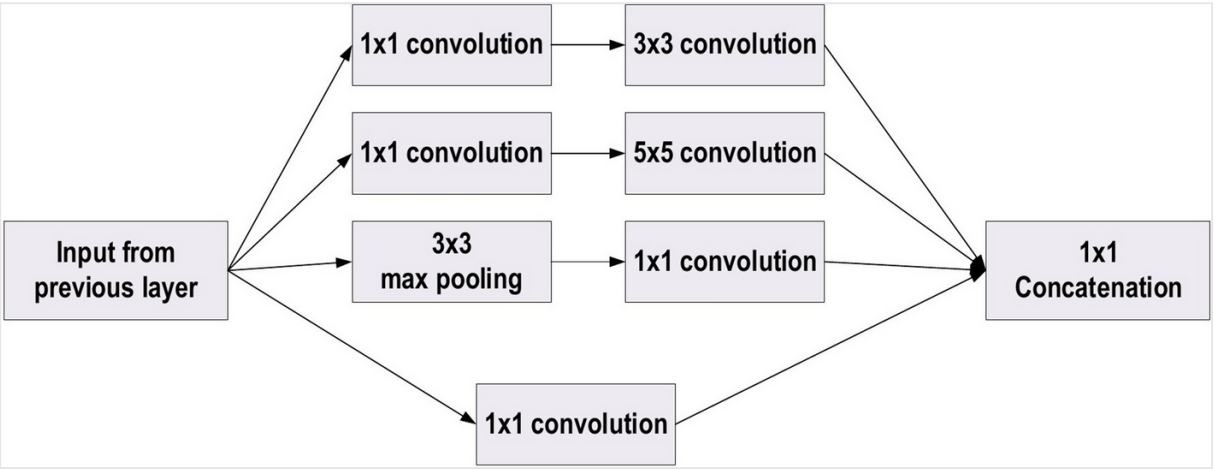


Figure 40: La structure de base de google block. [15]

2.4.3.2.3. VGGNet

VGG signifie Visual Geometry Group ; il s'agit d'une architecture standard de réseau de neurones à convolution profonde (CNN) à plusieurs couches. Le "profond" fait référence au nombre de couches avec VGG-16 ou VGG-19 composé de 16 et 19 couches convolutionnelles. [39]

Un avantage supplémentaire de réduction des complications de calcul a été obtenu en utilisant des filtres de petite taille. Ces résultats ont établi une nouvelle tendance de recherche pour travailler avec des filtres de petite taille dans CNN. De plus, en insérant1 × 11×1convolutions au milieu des couches convolutives, VGG régule la complexité du réseau. Il apprend un groupement linéaire des cartes d'entités suivantes. En ce qui concerne le réglage du réseau, une couche de mise en commun maximale est insérée après la couche convolutive, tandis que le rembourrage est mis en œuvre pour maintenir la résolution Spatiale. [15]

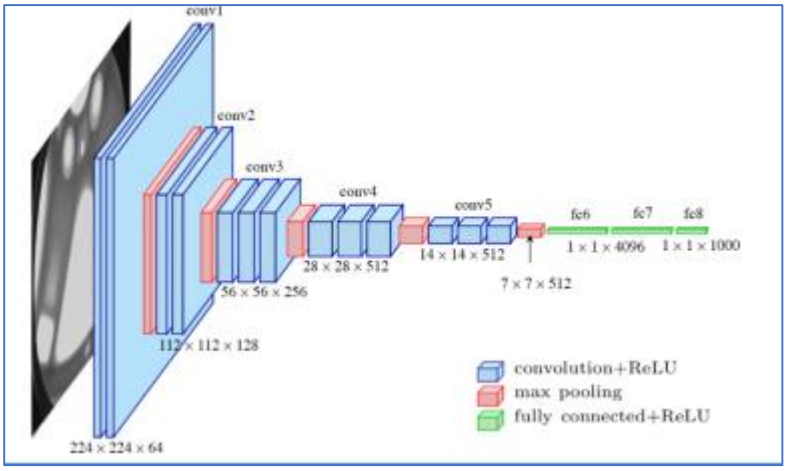


Figure 41: Architecture de réseau neuronal VGG. [39]

2.4.3.2.4. ResNet

Réseau résiduel développé par Kaiming He et al. a été le gagnant d'ILSVRC 2015. Il propose des *connexions* spéciales et un usage intensif de la normalisation des lots . L'architecture ne contient pas de couches entièrement connectées à la fin du réseau. (la présentation de Kaiming

(vidéo, diapositives) et quelques expériences récentes reproduisant ces réseaux dans Torch). [24]

Microsoft a conçu et mis en place le réseau. Ce modèle a remporté le concours ImageNet 2016 avec un taux de précision de 96,4 %. En raison de sa profondeur (jusqu'à 152 couches) et de l'ajout de blocs résiduels, il est bien connu. [100]

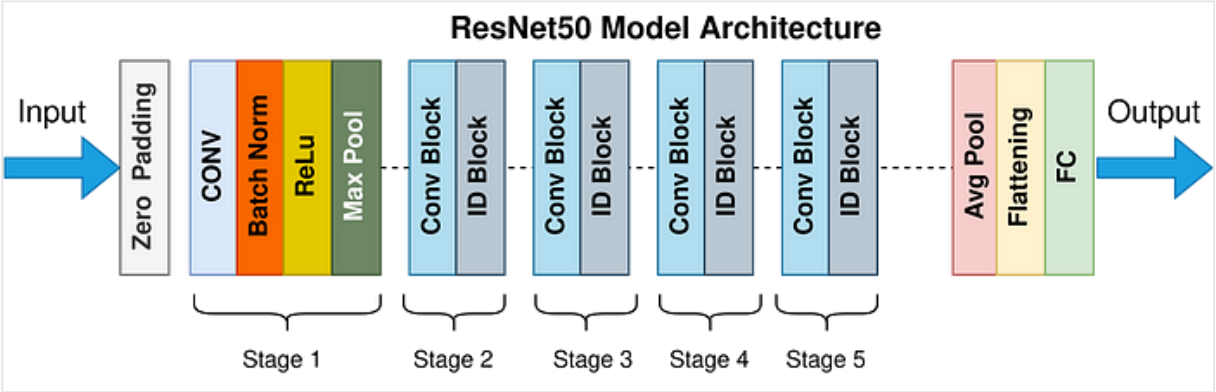


Figure 42: Architecture de ResNet. [85]

Ce ne sont là que quelques-unes des nombreuses architectures CNN qui ont été créées au fil du temps. La meilleure architecture à utiliser dépendra du travail exact à accomplir ainsi que des ressources de calcul disponibles. Chaque conception a ses propres avantages et inconvénients.

2.5. Conclusion

De nombreuses applications, notamment la reconnaissance de la parole et des images, la traduction de langues et la modélisation prédictive, ont montré l'efficacité des réseaux de neurones. Pour atteindre des niveaux élevés de précision, ils nécessitent de grandes quantités de données et leur formation peut être coûteuse en termes de calculs. Malgré cela, ils ont transformé le domaine de l'apprentissage automatique et ils sont un instrument important dans l'avancement de l'intelligence artificielle. Dans ce qui suit, nous présenterons la conception de notre projet.

Chapitre 03

Modélisation et Mise en œuvre d'un système de reconnaissance automatique des chiffres manuscrits isolés

3 1 Introduction

Dans ce chapitre, nous proposons une stratégie de modélisation et d'implémentation d'un système de reconnaissance automatique de chiffres manuscrits isolés.

La première section du chapitre traite l'importance crucial de l'extraction des caractères et l'étape du prétraitement dans l'amélioration de la précision de la reconnaissance, des techniques telles que la normalisation des images et l'élimination des bruit. La section suivante présente la conception du notre modèle et nous choisirons les architectures populaires les plus précis et nous les entrainerons avec la base de données MNIST via le langage de programmation Python. Ensuite nous avons comparé les architectures et abordons l'évaluation et les mesures de performance du chaque système. Enfin, nous avons réalisé une application web pour la reconnaissance des chiffres et puis nous avons appliqué quelques exemples.

3 2 L'environnement du développement

Pour le développement de notre application web ; nous avons utilisé les outils matériels et logiciels suivants :

3.2 1 Outils matériels

Pour réaliser notre projet, nous avons utilisé un PC portable marque HP avec les caractéristiques suivantes :

- Processeur : Intel(R) Core (TM) i5-8250U CPU @ 1.60GHz 1.80 GHz.
- Capacité Mémoire (RAM) : 8.00Go.
- Capacité disque dur : 256Go.
- Type de système : Système d'exploitation 64bits, processeur x64.
- Spécifications du Windows : Windows 10 Professionnel.
- Pour la réalisation de notre système nous avons choisi le langage de programmation Python, ainsi que les différents bibliothèques (Tensorflow, Keras, Numpy…).

3.2 2 Outils Logiciels

- **Visual Studio Code**

Visual Studio Code (**VSC**) est un éditeur de code open-source, gratuit et multi-plateforme (Windows, Mac et Linux), développé par Microsoft. Il ne faut pas le confondre avec Visual Studio, l'IDE propriétaire de Microsoft. VSC est développé avec Electron et exploite des fonctionnalités d'édition avancées du projet Monaco Editor. Principalement conçu pour le développement d'application avec JavaScript, TypeScript et Node.js, cet éditeur peut également être adapté à d'autres langages grâce à un système d'extension bien fourni. [94]

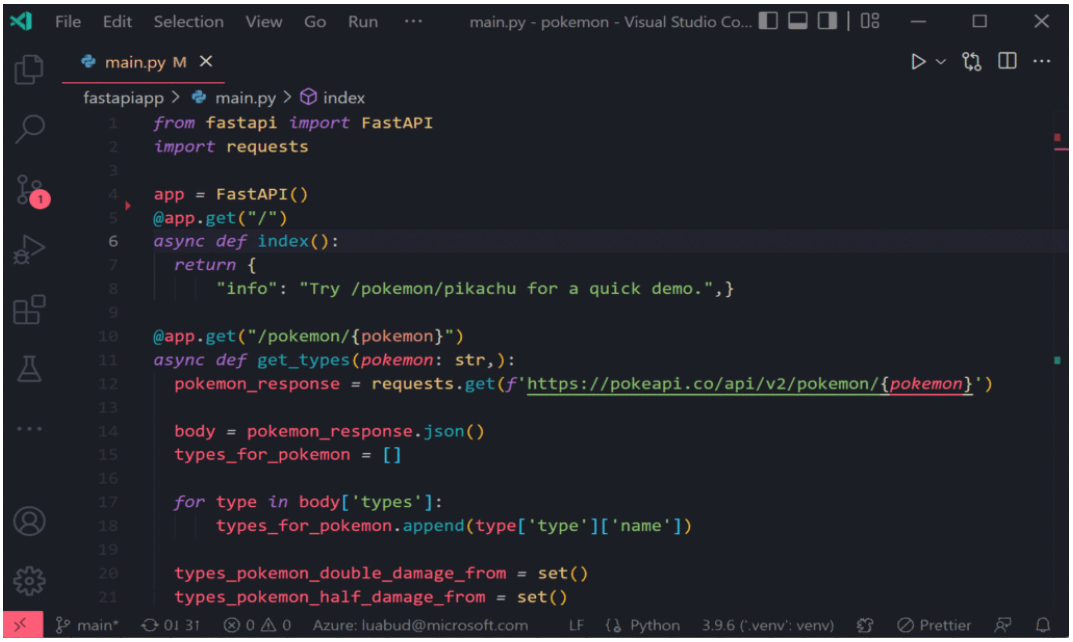


Figure 43: L'environnement de travail de Visual studio code. [94]

- **Google Colab (Colaboratory)**

Colab (ou "Colaboratory") vous permet d'écrire et d'exécuter du code Python dans un navigateur, Il est très largement utilisé par la communauté du machine Learning, par exemple dans les applications suivantes [43] :

- Développement et entraînement de réseaux de neurones.
- Expérimentation avec les TPU.
- Dissémination de la recherche en IA.

❖ **Les avantages de Google Colab :**

- Gratuit : les utilisateurs de Google Colab ont un accès gratuit une machine virtuelle équipée de GPU et de TPU pour les activités de machine Learning.
- Basé sur le cloud : étant donné que Google Colab est basé sur le Web, toute personne ayant accès à Internet peut l'utiliser, ce qui élimine le besoin d'équipement coûteux.
- Bibliothèques préinstallées : TensorFlow, Keras et PyTorch ne sont que quelques-unes des bibliothèques d'analyse de données et d'apprentissage automatique bien connues qui sont préinstallées avec Google Colab. La collaboration est simplifiée pour les équipes avec Google Colab, qui permet à plusieurs personnes de travailler sur le même ordinateur portable.
- Simple à utiliser : Google Colab offre aux utilisateurs une variété d'outils et de fonctionnalités pour créer et exécuter des modèles d'apprentissage automatique. Avec une simple configuration et utilisation.

❖ **Les inconvénients de Google Colab :**

- Ressources limitées : la RAM, l'espace disque et le temps GPU que Google Colab offre aux clients font partie des ressources limitées qui peuvent affecter la taille et la complexité des modèles pouvant être formés.
- Les sessions Google Colab se terminent automatiquement après une période définie, ce qui peut interférer avec de longues sessions de formation.
- Personnalisation limitée : Google Colab ne permet pas aux utilisateurs d'installer leurs propres applications ou bibliothèques pour personnaliser l'ordinateur virtuel. Google Colab a besoin d'une connexion Internet stable pour fonctionner, et des connexions lentes ou incohérentes peuvent dégrader l'expérience utilisateur.
- Les utilisateurs doivent faire confiance à Google pour protéger leurs données et maintenir la sécurité de la machine virtuelle, car Google Colab est un service basé sur le cloud.

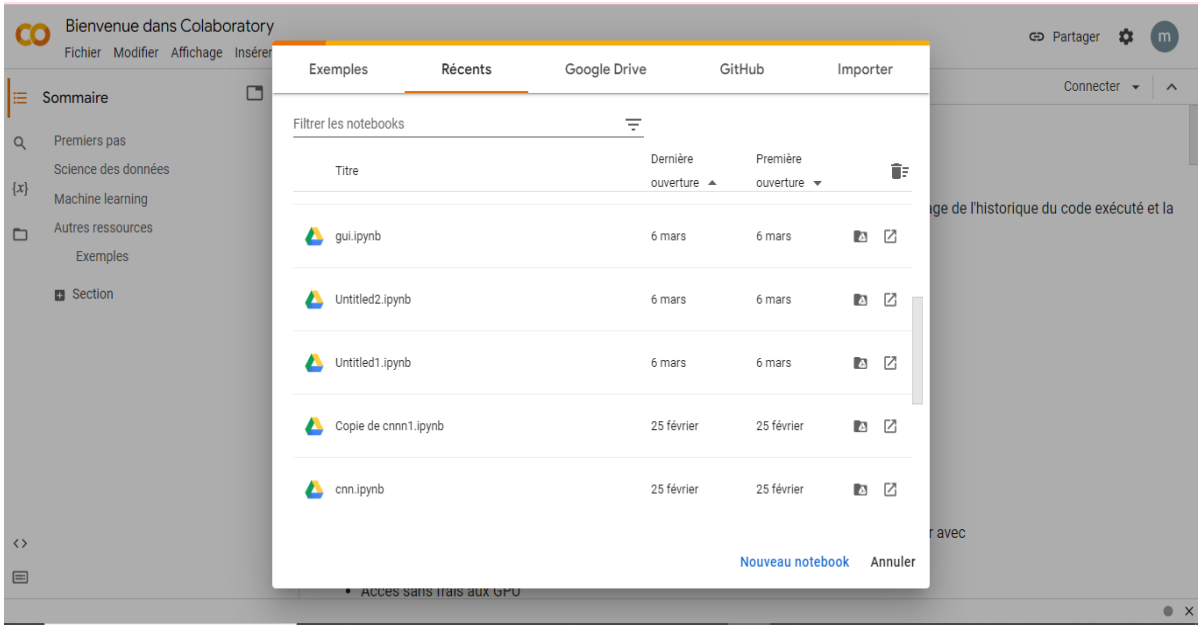


Figure 44: L'environnement de travail de google colab.

▪ **Python 3.9.13**

Une version spécifique du langage de programmation Python a été mise à disposition. Il fait partie de la série Python 3.x, qui, par rapport aux versions 2.x précédentes, a entraîné un certain nombre de mises à niveau et d'améliorations. La liste suivante montre les principales fonctionnalités et les mises à jour de Python 3.9.13 :

Python 3.9.13 fournit un certain nombre d'améliorations et d'optimisations des performances. Ces améliorations tentent d'accélérer l'exécution du code Python et de réduire l'utilisation de la mémoire, améliorant ainsi les performances globales.

Pour la sécurité : Python 3.9.13 contient des corrections de bogues et résout les problèmes signalés pour les versions antérieures. Ces correctifs de bogues améliorent la fiabilité et la stabilité du langage, ce qui se traduit par un environnement de développement plus satisfaisant.

Python dispose d'un vaste ensemble de bibliothèques pour l'intelligence artificielle et l'apprentissage automatique. En voici quelques-unes : [20]

- Keras, TensorFlow et Scikit-learn pour l'apprentissage automatique
- NumPy pour le calcul scientifique haute performance et l'analyse de données
- SciPy pour l'informatique avancée
- Pandas pour l'analyse de données à usage général
- Seaborn pour la visualisation des données
- OpenCV pour la vision par ordinateur.

3.2 3 Les bibliothèques

- **Tensorflow**

Tensorflow est un écosystème qui facilite l'utilisation du machine Learning pour résoudre des problèmes concrets complexes. Il propose de nombreux outils pour consolider, nettoyage et prétraiter des données à grande échelle, telles que : [88]

- Des ensembles de données standard pour la formation initiale et la validation.
- Des couches de prétraitement pour effectuer des transformations courantes sur les données d'entrée
- Des outils de validation et de transformation pour gérer de grands ensembles de données

De plus, les outils d'IA responsables vous aident à découvrir et à éliminer les biais dans vos données afin de produire des résultats équitables et éthiques à partir de vos modèles.

TensorFlow vous permet de créer et entraîner des modèles de pointe sans compromettre la vitesse ni les performances. Il offre une grande flexibilité et un contrôle accru grâce à des fonctionnalités telles que l'API fonctionnelle de Keras et l'API de sous-classification de modèle, qui permettent de créer des topologies complexes. Il facile également le prototypage et le débogage rapide. [88]

- **Keras**

Keras est une API d'apprentissage en profondeur écrite en Python, exécutée sur la plate-forme d'apprentissage automatique TensorFlow. Il a été développé dans le but de permettre une expérimentation rapide et d'offrir une expérience de développement agréable. Le but de Keras est de donner un avantage injuste à tout développeur cherchant à livrer des applications alimentées par ML. [86]

Keras facilite l'extension par la composition des éléments de base personnalisés pour exprimer de nouvelles idées de recherche. Créez des calques, des métriques et des fonctions de perte, et développez des modèles de pointe. Les modèles Keras sont créés en connectant des composants configurables, avec quelques restrictions. [89]

Keras est : [86]

- Simple : Keras réduit la charge cognitive du développeur et se concentre sur la facilité d'utilisation, la vitesse de débogage, l'élégance et la concision du code, la maintenabilité et la déployabilité (via TFServing, TFLite, TF.js).
- Flexible : Keras adopte le principe de la divulgation progressive de la complexité : les flux de travail simples doivent être rapides et faciles, tandis que les flux de travail arbitrairement avancés doivent être possibles via un chemin clair qui s'appuie sur ce que vous avez déjà appris.
- Puissant : Keras offre des performances et une évolutivité de pointe : il est utilisé par des organisations et des entreprises telles que la NASA, YouTube et Waymo.

❖ **Taille du lot**

La taille du lot (Batch size) est un terme utilisé dans l'apprentissage automatique et fait référence au nombre d'exemples de formation utilisés dans une itération. La taille du lot peut être l'une des trois options suivantes : [67]

- Mode de traitement par lots (mode batch) : où la taille du lot est égale à l'ensemble de données total, ce qui rend les valeurs d'itération et d'époque équivalentes
- Mode mini-lot (mini-batch mode) : lorsque la taille du lot est supérieure à un mais inférieure à la taille totale de l'ensemble de données. Généralement, un nombre qui peut être divisé en taille totale de l'ensemble de données.
- Mode stochastique : où la taille du lot est égale à un. Par conséquent, le gradient et les paramètres du réseau de neurones sont mis à jour après chaque échantillon.

La taille du lot est paramètre qui contrôle le nombre d'exemples d'entraînement utilisés dans chaque itération d'entraînement. L'augmentation de la taille du lot peut accélérer la formation, mais peut également entraîner un surajustement. [58]

❖ **Lots À Une Époque**

Une seule époque est un seul passage de toutes les données à travers le réseau, il faudra des lots pour rattraper l'époque complète. Nous avons des images divisées par une taille de lot de, ce qui équivaut au nombre total de lots. [74]

Lots à l'époque (nombre d'itérations) = taille de l'ensemble d'entraînement / batch_size

❖ **Comment le choix de la taille et les époques des lots**

Le nombre d'époques est le nombre de passages complets dans l'ensemble de données d'apprentissage. La taille d'un lot doit être supérieure ou égale à 1 et inférieure ou égale au nombre d'échantillons dans l'ensemble de données d'apprentissage. Le nombre d'époques peut être défini sur une valeur entière comprise entre un et l'infini. [58]

▪ **Optimiseurs**

Les optimiseurs sont le concept général utilisé dans les réseaux de neurones car il s'agit d'initialiser et de manipuler de manière aléatoire la valeur des poids pour chaque époque afin d'augmenter le potentiel de précision du réseau modèle. Une comparaison est effectuée à chaque époque entre la sortie des données d'entraînement et les données réelles, ce qui nous aide à

calculer les erreurs et à découvrir les fonctions de perte et la mise à jour ultérieure des poids correspondants. [54]

Les optimiseurs, lorsqu'ils sont associés à l'initialisation du poids, jouent un rôle clé dans la dynamique d'entraînement des réseaux neuronaux et sont actuellement un sujet de recherche brûlant. La question pour de nombreux praticiens est de savoir quel optimiseur particulier utiliser pour quel projet particulier. [60]

L'utilisation d'optimiseurs Keras peut aider à déterminer comment modifier le poids afin d'obtenir le plus haut niveau de précision. Nous pouvons obtenir une fonction de perte entièrement optimisée et les poids optimaux à l'aide de l'optimiseur Keras. La descente de gradient est l'un des optimiseurs les plus souvent utilisés. Il existe de nombreux autres optimiseurs keras qui sont facilement disponibles et utilisés dans de nombreuses applications du monde réel. Pour la mise en œuvre de différents optimiseurs Keras, de nombreuses API sont disponibles. [54]

❖ **Types d'optimiseurs Keras**

a. Optimiseur de descente de gradient (SGD)

La descente de gradient est considérée comme un algorithme d'optimisation pour minimiser la fonction de perte J (θ), L'étape de mise à jour en Descente de gradient est donnée par : [96]

$$\theta k+1=\theta k-\alpha \nabla J(\theta)$$

Il existe de nombreux autres algorithmes qui ont été créés au-dessus de la descente de gradient comme Adagrad et Adam. Le roi de tous les optimiseurs et il est très rapide, robuste et flexible. [61]

b. AdaGrad

Cet optimiseur de Keras utilise des paramètres spécifiques dans les taux d'apprentissage. Il a sa base des fréquences faites dans les mises à jour par la valeur des paramètres, et en conséquence, le travail se produit. Les caractéristiques individuelles affectent le taux d'apprentissage et sont ajustées en conséquence. Il y a aussi le scénario où différentes valeurs du taux d'apprentissage pour certains poids correspondent. [54]

c. **AdaDelta**

C'est une extension d'AdaGrad qui tend à éliminer le problème du taux d'apprentissage décroissant de celui-ci. Une autre chose avec AdaDelta est que nous n'avons même pas besoin de définir un taux d'apprentissage par défaut. [28]

d. Adam

Adam représente l'estimation adaptative du moment, qui est une autre façon d'utiliser les gradients passés pour calculer les gradients actuels, pour l'explication mathématique profonde, Adam utilise le concept de moment en ajoutant des fractions de gradients précédents à l'actuel, il est pratiquement accepté dans de nombreux projets lors de la formation de réseaux de neurones. [61]

Cet algorithme est avéré efficace pour Un large éventail d'architectures de réseaux neuronaux, et est recommandé comme un outil efficace pour des exemples aléatoires Parce qu'il n'a besoin que de quelques pentes à partir du premier gradient et réduit la capacité de stockage Obligatoire. [7]

- **Numpy**

Le terme NumPy est en fait l'abréviation de **«** Numerical Python **»**. Il s'agit d'une bibliothèque Open Source en langage Python. On utilise cet outil pour la programmation scientifique en Python, et notamment pour la programmation en Data Science, pour l'ingénierie, les mathématiques ou la science. [32]

❖ **Les avantage : [72]**

- Outils de calcul numérique : NumPy propose des fonctions mathématiques complètes, des générateurs de nombres aléatoires, des routines d'algèbre linéaire, des transformées de Fourier, etc. [72]
- Des tableaux n-dimensionnels puissants : Rapides et polyvalents, les concepts de vectorisation, d'indexation et de diffusion de NumPy. [72]
- Source ouverte : NumPy est développé et maintenu publiquement sur GitHub par une communauté dynamique, réactive et diversifiée. [72]
- Performant : Le cœur de NumPy est un code C bien optimisé. Profite de la flexibilité de Python avec la vitesse du code compilé. [72]

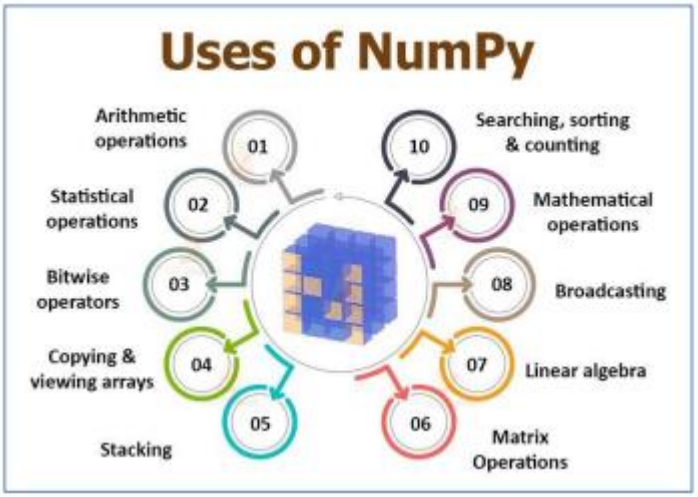


Figure 45: Utilisations de NumPy. [62]

- **Pillow**

Pillow est une bibliothèque de traitement d'image, qui est un fork et successeur du projet PIL (*Python Imaging Library*). Elle est conçue de manière à offrir un accès rapide aux données contenues dans une image, et offre un support pour différents formats de fichiers tels que PPM, PNG, JPEG, GIF, TIFF et BMP. [93]

Pillow dispose de capacités de traitement d'images relativement puissantes, et a pour but d'offrir une solide base à toute application générale de traitement d'images. [93]

La bibliothèque de fonctions peut être utilisée pour différents types d'activités, telles que : [93]

- **Archivage d'images** : Création de miniatures, conversion d'images d'un format de fichier à un autre.
- **Traitement d'images** : Offre un support pour quelques fonctions de bases telles que le filtrage, la convolution ou encore la conversion d'espaces couleurs. Il est également possible de redimensionner et d'appliquer des transformations géométriques à l'image (rotation, …). [93]
- **Filtrage et améliorations d'image** : Pillow propose un certain nombre de filtres d'image, notamment la réduction du bruit, l'amélioration des contours, le flou des contours et la netteté. Les images peuvent utiliser ces filtres pour améliorer leur qualité, éliminer les défauts.
- **Analyse des métadonnées et des images** : Pillow vous permet d'extraire une variété de métadonnées des photos, y compris la résolution, le mode couleur et les données EXIF (telles que les données de l'appareil photo, les horodatages et les positions GPS). Les images peuvent être traitées ou analysées à l'aide de ces métadonnées.

De nombreuses applications différentes, telles que le développement Web, la vision par ordinateur et la conception graphique, utilisent largement Pillow. C'est une bibliothèque incontournable pour les travaux de traitement d'images Python en raison de sa vaste documentation, de son interface conviviale et de son riche ensemble de fonctionnalités.

Pillow peut être utilisé en exécutant la commande pip install pillow dans votre environnement Python pour l'installer à l'aide du gestionnaire de paquets pip.

- **Matplotlib**

Matplotlib est une bibliothèque complète pour créer des visualisations statiques, animées et interactives en Python. Matplotlib rend les choses difficiles possibles. [63]

Avec Matplotlip on peut

- Créer des tracés de qualité de publication.
- Créer des figures interactives qui permettent le zoom, le panoramique et la mise à jour.
- Personnaliser le style visuel et la mise en page.
- Exporter vers de nombreux formats de fichiers.
- Intégrer dans Jupiter Lab et les interfaces utilisateur graphiques.
- Profiter d'un riche éventail de packages tiers construits sur Matplotlib.

Matplotlib est une boîte à outils flexible et puissant pour créer d'excellents tracés et visualisations en Python. C'est un outil essentiel pour la visualisation des données, l'analyse exploratoire des données et la recherche scientifique en raison de sa large gamme de fonctionnalités, d'options de personnalisation et d'intégration avec d'autres bibliothèques scientifiques.

3.2 4 Développement web

- **Flask**

Flask est un petit Framework web Python léger, qui fournit des outils et des fonctionnalités utiles qui facilitent la création d'applications web en Python. Il offre aux développeurs une certaine flexibilité et constitue un cadre plus accessible pour les nouveaux développeurs, Flask est également extensible et ne force pas une structure de répertoire particulière ou ne nécessite pas de code standard compliqué avant de commencer. [36]

Flask utilise le moteur de modèle Jinja pour construire dynamiquement des pages HTML en utilisant des concepts Python familiers tels que les variables, les boucles, les listes, etc. [36]

3 3 Base de données

La reconnaissance manuscrite des caractères est une vaste recherche qui contient des modalités de l'implémentation détaillées qui comprennent de grands ensembles de données d'apprentissage, des algorithmes populaires, Caractéristiques et méthodes d'extraction des caractéristiques. [8]

La base de données MNIST (**Modified National Institute of Standards and Technology**) est une grande collection de chiffres manuscrits, contient un ensemble d'apprentissage de 60 000 exemples et un ensemble de test de 10 000 exemples. Il s'agit d'un sous-ensemble d'un ensemble plus vaste disponible auprès 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 souhaitent essayer des techniques d'apprentissage et des méthodes de reconnaissance de formes sur des données du monde réel tout en dépensant un minimum d'efforts sur le prétraitement et le formatage. [6]

Les images originales en noir et blanc (à deux niveaux) du NIST ont été normalisées pour tenir dans une boîte de 20 x 20 pixels tout en préservant leur format d'image. Les images résultantes contiennent des niveaux de gris en raison de la technique d'anticrénelage 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 translatant l'image de manière à positionner ce point au centre du champ 28x28. [6]

Avec certaines méthodes de classification le taux d'erreur s'améliore lorsque les chiffres sont centrés par la boîte englobante plutôt que par le centre de masse. [6]

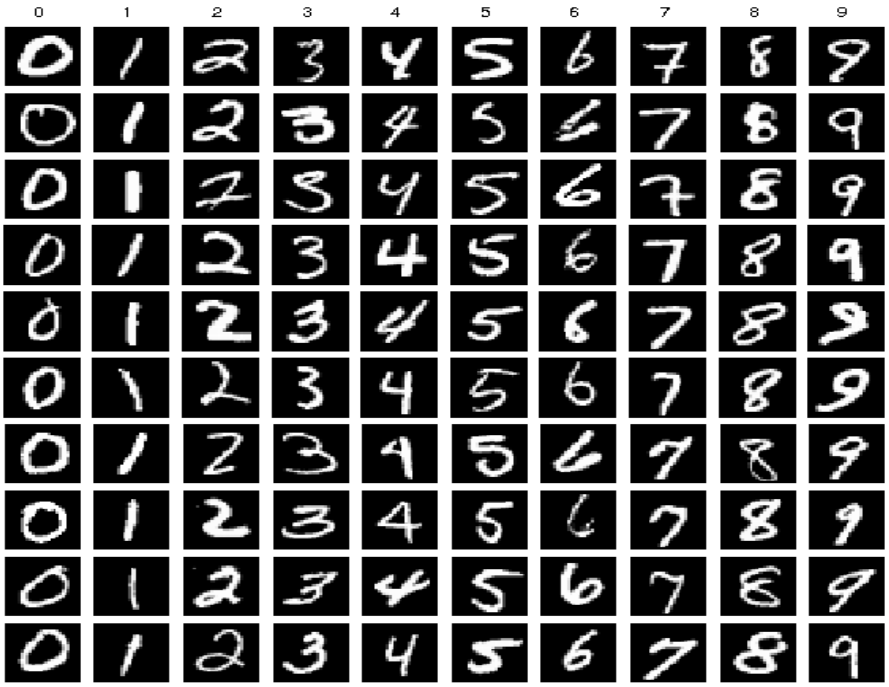


Figure 46: Quelques échantillons extraits de la base MNIST. [6]

La base de données MNIST a été construite à partir de les bases de données spéciales 3 et 1 du NIST qui contiennent des images binaires de chiffres manuscrits. Le NIST a initialement désigné SD-3 comme ensemble d'entraînement 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 le SD-3 a été collecté parmi les employés du Census Bureau, tandis que le SD-1 a été collecté parmi les élèves du secondaire. Tirer des conclusions sensées des expériences d'apprentissage nécessite que le résultat soit indépendant du choix de l'ensemble d'apprentissage et du test parmi l'ensemble complet d'échantillons. Il était donc nécessaire de construire une nouvelle base de données en mélangeant les jeux de données du NIST. [6]

Le tableau suivant représente la répartition de la base MNIST pour les chiffres :

Classes	Apprentissage	Test	Totale
0	5923	980	6903
1	6742	1135	7877
2	5958	1032	6990
3	6131	1010	7141
4	5842	982	6824
5	5421	892	6313
6	5918	958	6876
7	6265	1028	7293
8	5851	974	6825
9	5949	1009	6958
Totale	60000	10000	70000

Tableau 6: la répartition de la base MNIST pour les chiffres. [22]

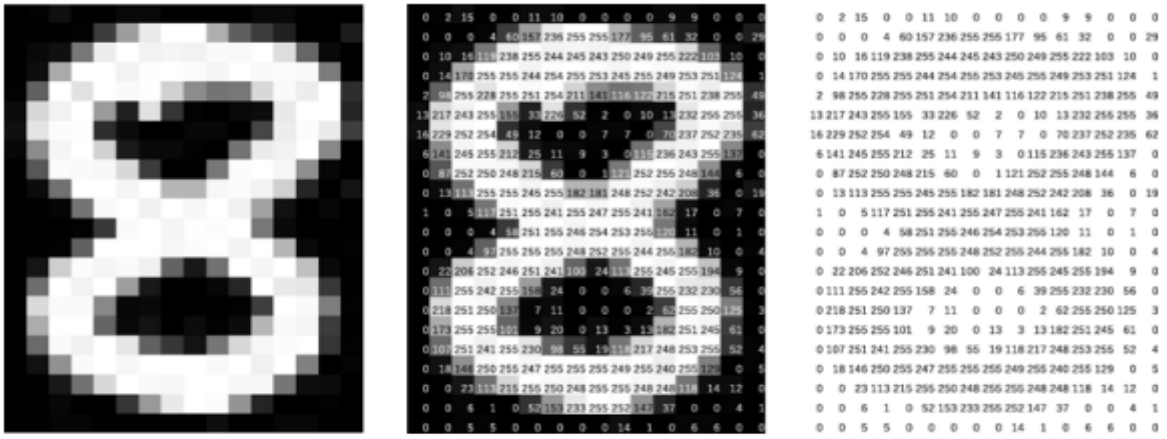


Figure 47: exemple de numéro 8 dans la base MNIST. [22]

3 4 Expérimentation

3.4 1 Le modèle proposé

Un modèle d'apprentissage en profondeur basé sur les réseaux de neurones convolutifs (CNN) est suggéré pour reconnaître les chiffres manuscrits. Les CNN sont une bonne option pour la reconnaissance manuscrite des chiffres car ils fonctionnent exceptionnellement bien sur les tâches de classification des images.

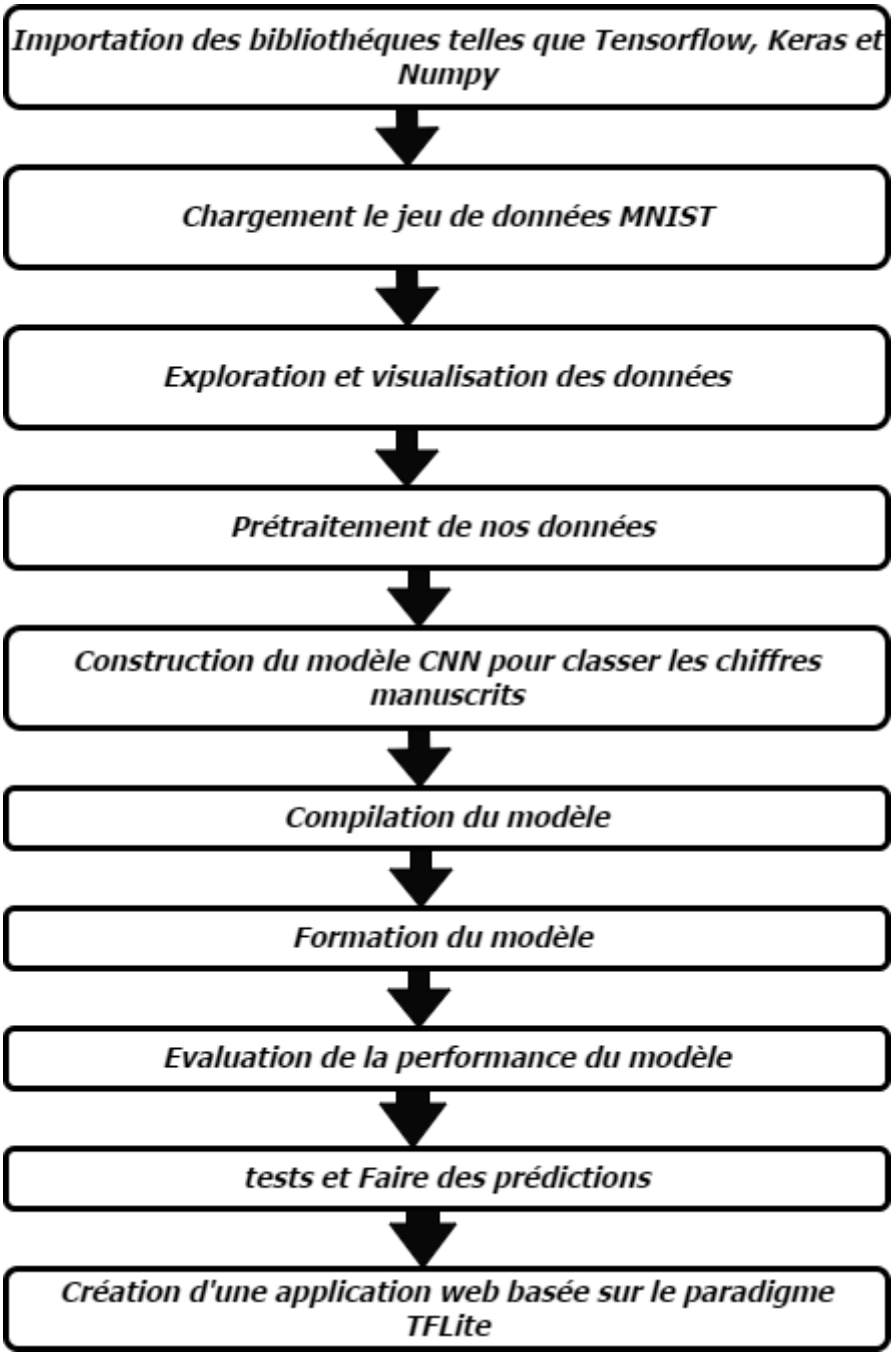


Figure 48: Les étapes de l'implémentations du projet.

1) Importation des bibliothèques nécessaires :

En utilisant TensorFlow et le base de données MNIST, nous allons commencer par créer un modèle pour reconnaître les chiffres écrits à la main. Cette base de données inclus 60 000 images des chiffres 0 à 9. Après cela, nous allons transformer ce modèle en un modèle TFLite, une variante plus condensée et efficace du modèle original TensorFlow. Enfin, nous allons créer une application web basée sur le paradigme TFLite qui permet aux utilisateurs de créer et de soumettre leurs propres images numériques manuscrites pour la reconnaissance. Le logiciel prévoit le chiffre des photographies fournies en utilisant le modèle TFLite et le montre à l'utilisateur.

Nous utiliserons principalement Google Colab comme notre plate-forme tout au long de ce manuel. Sans nécessiter d'installations ou de configurations logicielles compliquées, Google Colab est une solution accessible et conviviale pour commencer à travailler sur le projet.

2) Chargement des données

Nous utiliserons l'API Keras Datasets pour charger les échantillons du MNIST car elle offre une méthode simple et sans problème pour le faire. En réduisant la quantité de code supplémentaire nécessaire pour gérer le chargement de données, cette méthode facilite un lancement du projet plus rapide et plus facile.

```
from keras.datasets import mnist
(X_train, y_train), (X_test, y_test) = mnist.load_data()
```

3) Exploration des données

Examinons maintenant l'organisation de nos données. En analysant les données, nous pouvons en apprendre davantage sur sa structure et identifier les tendances ou les problèmes qui peuvent nuire à l'efficacité de notre algorithme de reconnaissance des chiffres.

```
print("Training Images:", len(X_train))
print("Testing Images:", len(X_test))
print("Shape:", X_train[0].shape)
```

```
Training Images: 60000
Testing Images: 10000
Shape: (28, 28)
```

4) Visualisation des images

60 000 images de formation et 10 000 images de test constituent notre ensemble de données. Chaque image est une matrice Numpy avec la forme de (28, 28). Nous pouvons afficher quelques photographies aléatoires pour avoir une meilleure compréhension de la collection. Avec l'aide de cette visualisation, nous serons en mesure de voir des schémas ou des anomalies dans les données ainsi que de nous assurer que le pré-traitement et le chargement des photographies sont effectués correctement.

```
import random
import matplotlib.pyplot as plt
random_image = random.choice(X_train)
plt.imshow(random_image, cmap="gray")
```

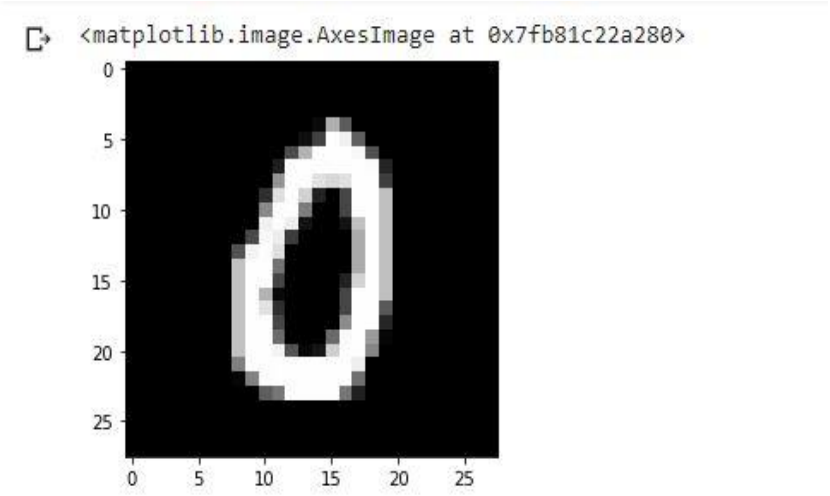


Figure 49: Example de la base de données MNIST.

5) Prétraitement de nos données

Nous devons apporter quelques modifications à nos données car nous avons l'intention d'utiliser un réseau neuronal convolutif (CNN) pour notre modèle de reconnaissance numérique. Étant donné que les CNN sont faits expressément pour traiter les données d'image, nous devons modifier nos photos pour se conformer au format d'entrée de la CNN. Les photographies peuvent avoir besoin d'être remodelées ou d'autres procédures de pré-traitement peuvent être appliquées dans le cadre de ces modifications. Ce processus de préparation des données aidera notre CNN à être plus précis et plus efficace.

Les images d'entrée pour les réseaux neuronaux convolutifs (CNN) doivent avoir les dimensions suivantes : largeur, hauteur et canaux de couleur. Cependant, nos photos MNIST ne comportent que des mesures de largeur et de hauteur et pas de données de canaux de couleur. Parce que les images MNIST sont à l'échelle grise, nous devons changer leur forme pour incorporer le canal de couleur, qui dans ce cas est tout simplement un. Pour ce faire, nous restructurerons les photos dans une nouvelle forme avec les coordonnées (28, 28, 1), donnant ainsi aux données une nouvelle dimension de canal de couleur. Pour s'assurer que notre CNN peut traiter les données d'entrée dans le format approprié, cet ajustement est nécessaire.

Par conséquent, nous allons :

```
X_train = X_train.reshape(X_train.shape[0], 28, 28, 1)
X_test = X_test.reshape(X_test.shape[0], 28, 28, 1)
X_train.shape
```

```
(60000, 28, 28, 1)
```

Normalisation

La norme d'image à l'échelle grise de 8 bits exige que les valeurs de pixel pour nos images MNIST soient comprises entre 0 et 255. La formation des modèles d'apprentissage automatique, en particulier des réseaux neuronaux, peut être entravée par de telles vastes gammes de valeurs. Nous normaliserons les valeurs de pixels pour tomber dans la plage [0, 1] afin de résoudre ce problème, qui est une méthode d'apprentissage profond standard. Bien que les informations d'image ne soient pas altérées par le redescalage, cela aidera notre modèle à apprendre plus rapidement et à reconnaître les nombres plus précisément.

```
X_train = X_train / 255.
X_test = X_test / 255.
```

Nous modifierons les valeurs de pixel de nos images MNIST du type de données par défaut float64 à float32 comme dernière étape de notre procédure de préparation. Cette modification améliorera les performances globales de notre modèle tout en réduisant le coût informatique de sa formation. Pour notre cas d'utilisation, la petite différence de précision entre les deux types de données rend souhaitable d'utiliser float32 car il utilise moins de mémoire et effectue des calculs plus rapidement.

```
import numpy as np
X_train = X_train.astype(np.float32)
X_test = X_test.astype(np.float32)
```

6) Construction du modèle CNN pour Classer les Chiffres manuscrits

Nous pouvons maintenant procéder à la création du modèle.

Ce modèle a un total de 13 couches : 3 couches Conv2D, 3 couches BatchNormalization, 3 couches MaxPool2D, 1 couche Flatten, 2 couches Dense et 1 couche Dropout.

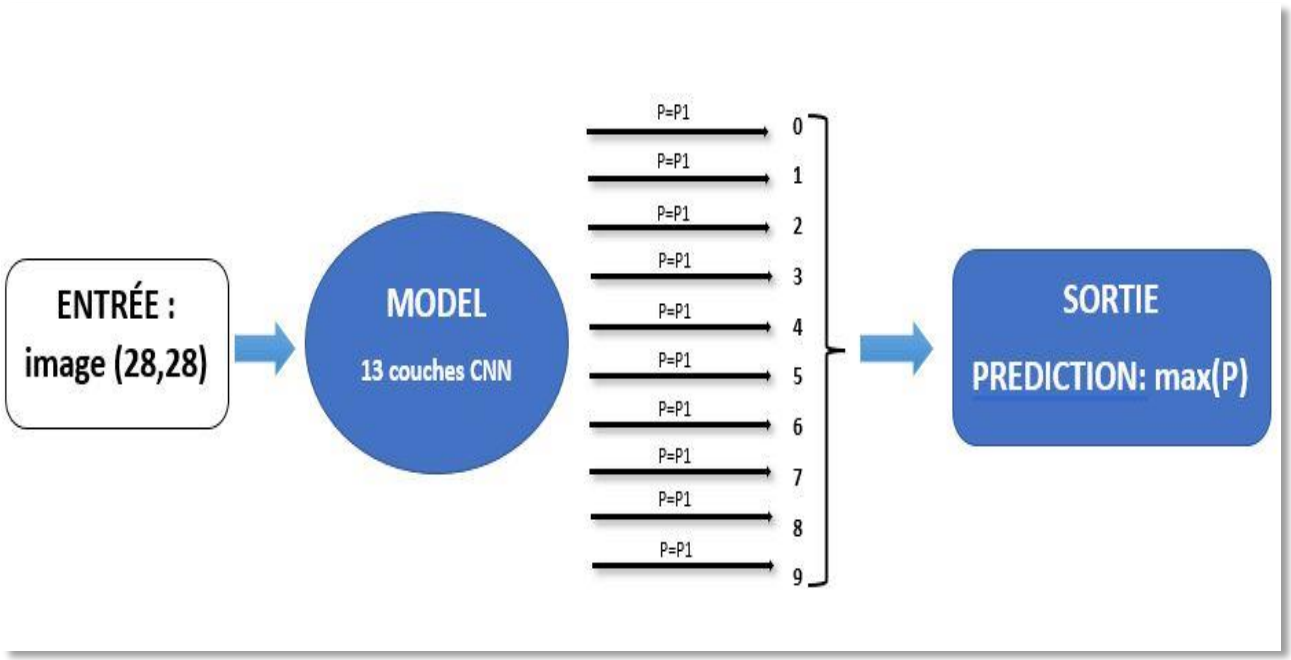


Figure 50: Schéma explicatif pour la construction du modèle CNN.

1- Couche "Conv2D" avec 32 filtres, une taille de noyau de 3, et une fonction d'activation ReLU : Cette couche applique 32 filtres de convolution à l'image d'entrée avec une taille de noyau de 3x3. La fonction d'activation ReLU est utilisée pour introduire une non-linéarité à la sortie de cette couche.

2- Couche "BatchNormalization" : Cette couche normalise la sortie de la couche précédente, ce qui permet d'accélérer le processus d'apprentissage et d'améliorer la précision du modèle.

3- Couche "MaxPool2D" : Cette couche effectue une opération de regroupement 2D max sur la sortie de la couche précédente, ce qui réduit les dimensions spatiales de la carte d'entités et rend le modèle plus robuste aux petites translations et distorsions dans l'image d'entrée.

4- Une autre couche "Conv2D" avec 64 filtres, une taille de noyau de 3, et une fonction d'activation ReLU : Cette couche applique 64 filtres de convolution à la sortie de la couche précédente avec une taille de noyau de 3x3. La fonction d'activation ReLU est à nouveau utilisée pour introduire la non-linéarité.

5- Une autre couche "BatchNormalization".

6- Une autre couche "MaxPool2D".

7- Une autre couche "Conv2D" avec 128 filtres, une taille de noyau de 3 et une fonction d'activation ReLU.

8- Une autre couche "BatchNormalization".

9- Une autre couche "MaxPool2D".

10- Couche "Aplatir" : Cette couche aplatit la sortie de la couche précédente, qui est un tenseur 3D, en un vecteur 1D.

11- Couche "Dense" avec 256 unités et fonction d'activation ReLU : Cette couche est une couche entièrement connectée avec 256 unités, qui prend l'entrée aplatie de la couche précédente comme entrée. La fonction d'activation ReLU est à nouveau utilisée pour introduire la non-linéarité.

12- Une autre couche "BatchNormalization".

13- Couche "Dropout" avec un taux de 0,5 : Cette couche abandonne aléatoirement 50% des activations de la couche précédente pendant l'entraînement, ce qui permet d'éviter le surajustement.

14- Une autre couche "Dense" avec 10 unités et fonction d'activation softmax : Cette couche est une autre couche entièrement connectée avec 10 unités, ce qui correspond aux 10 classes du jeu de données MNIST. La fonction d'activation softmax est utilisée pour convertir la sortie de cette couche en une distribution de probabilité sur les 10 classes de chiffres potentiels (0 à 9).

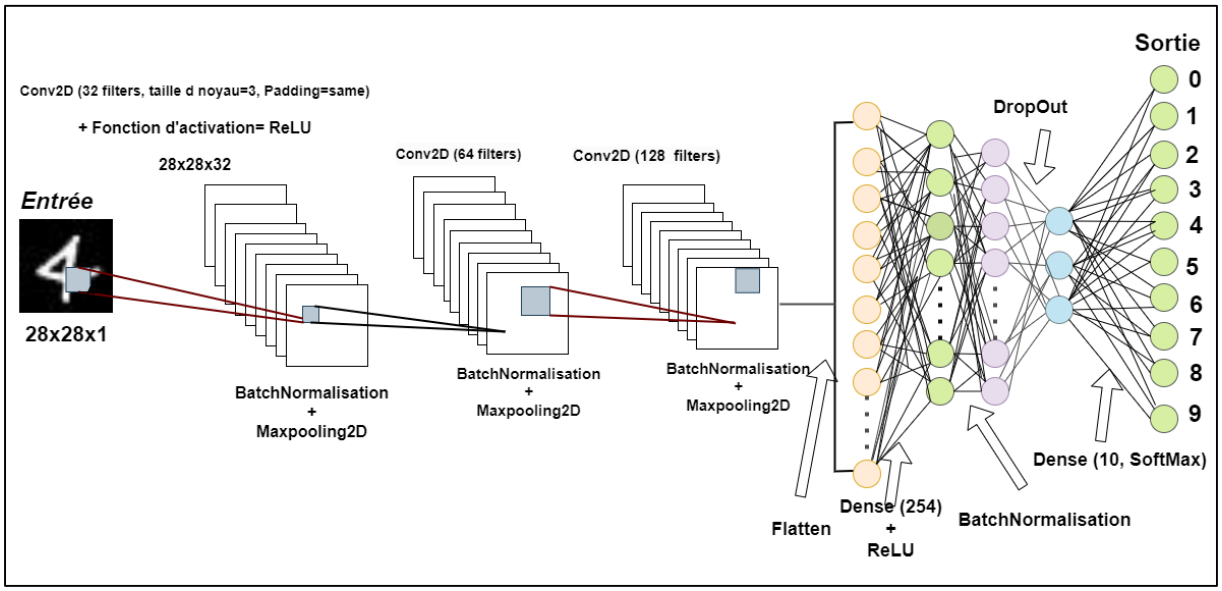


Figure 51: Les couches du modèle CNN.

Augmentation des données

Le code utilise également la fonction ImageDataGenerator de Keras pour enrichir les données en plus du modèle. Afin de diversifier l'ensemble de formation et d'améliorer la généralisation et la performance du modèle, des transformations aléatoires telles que la rotation, le déplacement et le retournement sont appliqués aux images de formation. En outre, les valeurs des pixels sont réévaluées pour tomber entre 0 et 1.

```
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras.callbacks import TensorBoard
import os
model = tf.keras.Sequential([
  layers.Conv2D(filters=32, kernel_size=3, activation="relu", padding="same", input_shape=(28, 28, 1)),
  layers.BatchNormalization(),
  layers.MaxPool2D(),

  layers.Conv2D(64, 3, activation="relu", padding="same"),
  layers.BatchNormalization(),
  layers.MaxPool2D(),

  layers.Conv2D(128, 3, activation="relu", padding="same"),
  layers.BatchNormalization(),
  layers.MaxPool2D(),

  layers.Flatten(),
  layers.Dense(256, activation="relu"),
  layers.BatchNormalization(),
  layers.Dropout(0.5),
  layers.Dense(10, activation="softmax")
])

# Data augmentation
train_datagen = tf.keras.preprocessing.image.ImageDataGenerator(
    rotation_range=10,
    width_shift_range=0.1,
    height_shift_range=0.1,
    zoom_range=0.1,
    horizontal_flip=True,
    vertical_flip=True,
    rescale=1./255
)
```

7) Compilation de modèle

Nous construisons un modèle CNN puis le compilons à l'aide de l'optimisateur Adam avec un taux d'apprentissage de 1e-3. Notre fonction de perte de choix est 'Sparse-Categorical-Crossentropy', qui fonctionne bien pour les problèmes de classification multi-classe où les étiquettes sont entières. La métrique de précision, qui mesure le pourcentage de prévisions précises sur l'ensemble de validation, est utilisée pour évaluer le modèle. Le modèle peut modifier ses poids pendant l'entraînement de manière à minimiser les pertes et à améliorer les performances dans le but d'identifier des images de chiffres manuscrits en ajustant la fonction de perte et l'optimisateur.

```
# Compile the model
model.compile(optimizer=tf.keras.optimizers.Adam(lr=1e-3),
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

8) Formation de modèle

Nous utiliserons les 20 époques de données de l'ensemble de formation pour former notre modèle, chaque époque désignant 1875 itérations à travers l'ensemble de formation. Le modèle analysera les données d'entraînement au cours de chaque époque en lots de taille 32, en mettant à jour ses poids en fonction des gradients moyens de 32 échantillons à la fois. En permettant au

modèle d'apprendre de ses erreurs et de modifier ses poids de manière appropriée, nous espérons améliorer les performances du modèle en l'entraînant pour de nombreuses époques.

```
model.fit(X_train, y_train, epochs=20, batch_size=32, validation_data=(X_test, y_test))
# Fit the model
```

Il est recommandé d'utiliser la GPU comme temps de fonctionnement pour une formation plus rapide car la procédure peut prendre quelques minutes pour être terminée. Une fois la formation terminée, nous pouvons évaluer l'exactitude de notre modèle en utilisant les données d'essai.

```
Epoch 18/20
1875/1875 [==============================] - 11s 6ms/step - loss: 0.0073 - accuracy: 0.9974 - val_loss: 0.0272 - val_accuracy: 0.9941
Epoch 19/20
1875/1875 [==============================] - 11s 6ms/step - loss: 0.0063 - accuracy: 0.9981 - val_loss: 0.0284 - val_accuracy: 0.9930
Epoch 20/20
1875/1875 [==============================] - 10s 6ms/step - loss: 0.0067 - accuracy: 0.9979 - val_loss: 0.0292 - val_accuracy: 0.9934
<keras.callbacks.History at 0x7f292853d2a0>
```

On a utilisé matplotlib pour tracer la courbe qui montre la variation de la précision (Accuracy) et la fonction de perte (Loss) en fonction du nombre d'époques.

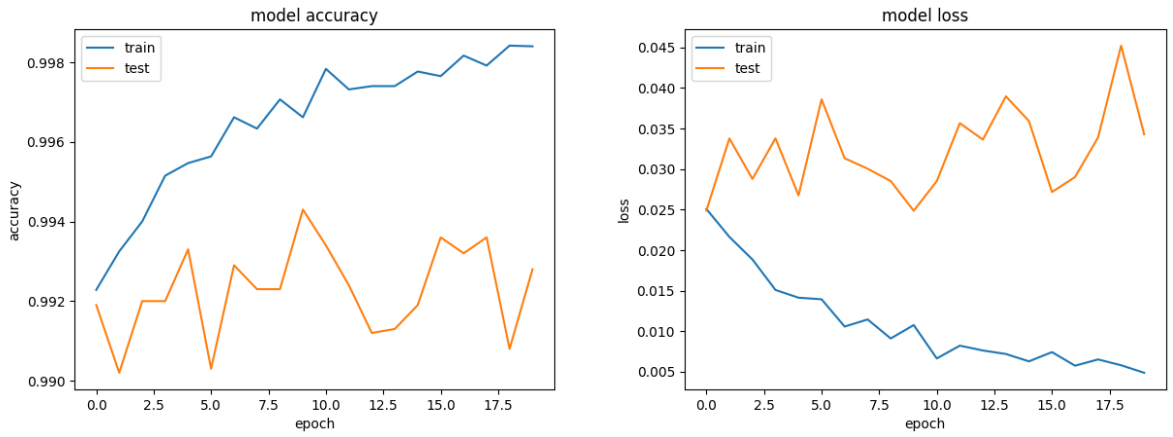


Figure 52: Précision et fonction de perte sur l'ensemble de formation et de test en fonction du nombre d'époques.

9) Évaluation de notre modèle

Les performances du modèle sont évaluées pendant l'entraînement sur un autre ensemble de validation pour vérifier sa précision et effectuer les ajustements d'hyperparamètres nécessaires.

Après la formation, la performance du modèle est ensuite évaluée sur un ensemble de tests séparé pour obtenir une évaluation équitable de sa précision et de son potentiel de généralisation.

Le modèle a réalisé des performances admirables sur le dataset de test, avec une précision de 99,39%. Le modèle a été formé avec succès pour distinguer les différentes classes de chiffres dans le groupe de données en fonction de son degré élevé de précision.

```
# Evaluate the model
test_loss, test_acc = model.evaluate(X_test, y_test)
print('Test accuracy:', test_acc)

# Make predictions
predictions = model.predict(X_test[:10])
print('Predictions:', predictions)
model.summary()
```

```
313/313 [==============================] - 1s 3ms/step - loss: 0.0283 - accuracy: 0.9939
Test accuracy: 0.9939000010490417
```

Résumé du modèle de formation

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 28, 28, 32)        320

batch_normalization (BatchN  (None, 28, 28, 32)        128
ormalization)

max_pooling2d (MaxPooling2D  (None, 14, 14, 32)        0
)

conv2d_1 (Conv2D)            (None, 14, 14, 64)        18496

batch_normalization_1 (Batc  (None, 14, 14, 64)        256
hNormalization)

max_pooling2d_1 (MaxPooling  (None, 7, 7, 64)          0
2D)

conv2d_2 (Conv2D)            (None, 7, 7, 128)         73856

batch_normalization_2 (Batc  (None, 7, 7, 128)         512
hNormalization)

max_pooling2d_2 (MaxPooling  (None, 3, 3, 128)         0
2D)

flatten (Flatten)            (None, 1152)              0

dense (Dense)                (None, 256)               295168

batch_normalization_3 (Batc  (None, 256)               1024
hNormalization)

dropout (Dropout)            (None, 256)               0

dense_1 (Dense)              (None, 10)                2570

=================================================================
Total params: 392,330
Trainable params: 391,370
Non-trainable params: 960
_________________________________________________________________
```

Tableau 7:L'architecture des couches du modèle CNN.

10) Proposition des prédictions :

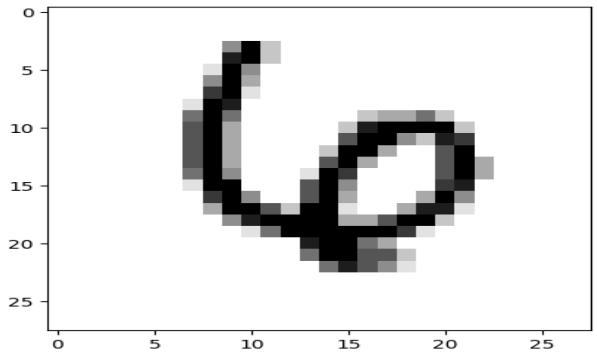
Des prédictions peuvent être effectuées à l'aide du modèle après qu'il a été entraîné et évalué à l'aide de nouvelles images numériques manuscrites. En réduisant la taille de l'image d'entrée pour qu'elle corresponde à la taille d'entrée prévue par le modèle et en normalisant les

valeurs des pixels, l'image d'entrée est prétraitée. Le modèle prédira la classe de chiffres lorsque vous lui fournirez l'image prétraitée.

On visualise quelques exemples de test après on fait une prédiction pour observer la performance du model et où il fait des erreurs.

Exemple 1 :

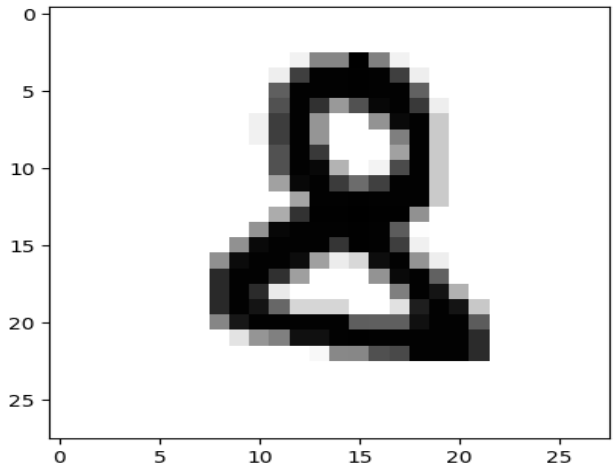
```
[35] plt.imshow(x_test[9999],cmap=plt.cm.binary)
     plt.show()
```



```
np.argmax (predcta[9999])
```

```
6
```

Exemple 2 :



```
np.argmax (predcta[1955])
```

```
8
```

Nous pouvons voir que notre modèle a correctement prédit le résultat et identifié le chiffre.

3.4 1 CNN avec architecture TinyVGG

Version condensée du célèbre modèle VGG (Visual Geometry Group), l'architecture TinyVGG de Keras a été créée pour obtenir de bonnes performances avec moins de paramètres. Plusieurs couches convolutives et de regroupement sont incluses, suivies de couches entièrement connectées pour la classification, et il adhère aux concepts de réseaux de neurones convolutifs (CNN). Voici un résumé de l'architecture TinyVGG de Keras :

```
model.summary()

Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_12 (Conv2D)          (None, 28, 28, 10)        100

 conv2d_13 (Conv2D)          (None, 24, 24, 10)        910

 max_pooling2d_6 (MaxPooling  (None, 12, 12, 10)       0
 2D)

 conv2d_14 (Conv2D)          (None, 10, 10, 10)        910

 conv2d_15 (Conv2D)          (None, 8, 8, 10)          910

 max_pooling2d_7 (MaxPooling  (None, 4, 4, 10)         0
 2D)

 flatten_3 (Flatten)         (None, 160)               0

 dense_3 (Dense)             (None, 10)                1610

=================================================================
Total params: 4,440
Trainable params: 4,440
Non-trainable params: 0
```

Tableau 8:L'architecture des couches du modèle TinyVGG.

Evaluation du modéle :

```
313/313 [==============================] - 1s 4ms/step - loss: 0.0942 - accuracy: 0.9715
test accuracy: 0.9714999794960022
```

Les performances du modèle sont évaluées à l'aide de mesures telles que la précision et la fonction de perte.

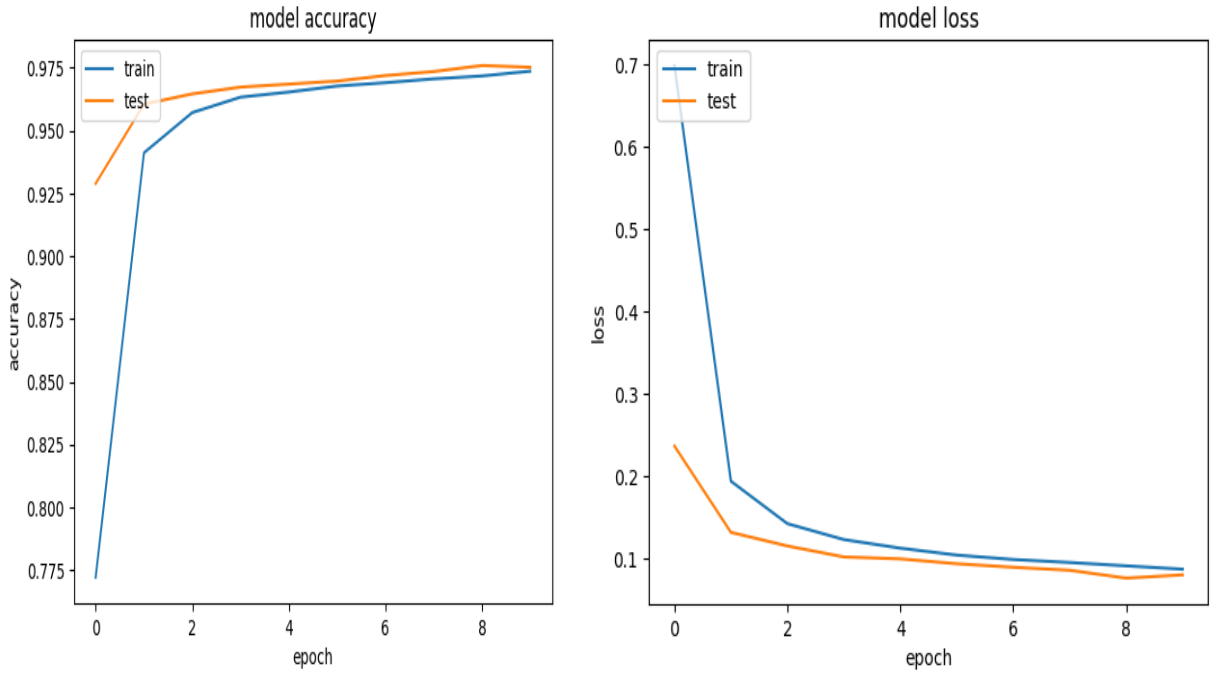


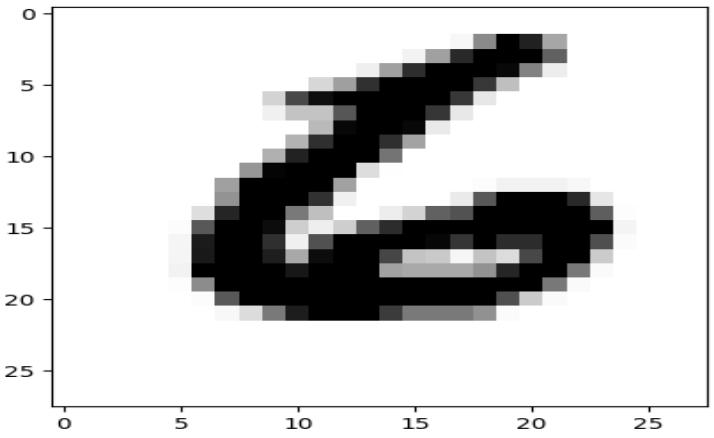
Figure 53: La précision et la fonction de perte sur l'ensemble de formation et de test en fonction du nombre d'époques dans le modèle TinyVGG.

Proposition des prédictions :

Nous essayons de prédire avec les mêmes images qui correspond au numéro d'image 9999 et 1955 dans l'ensemble de données de test :

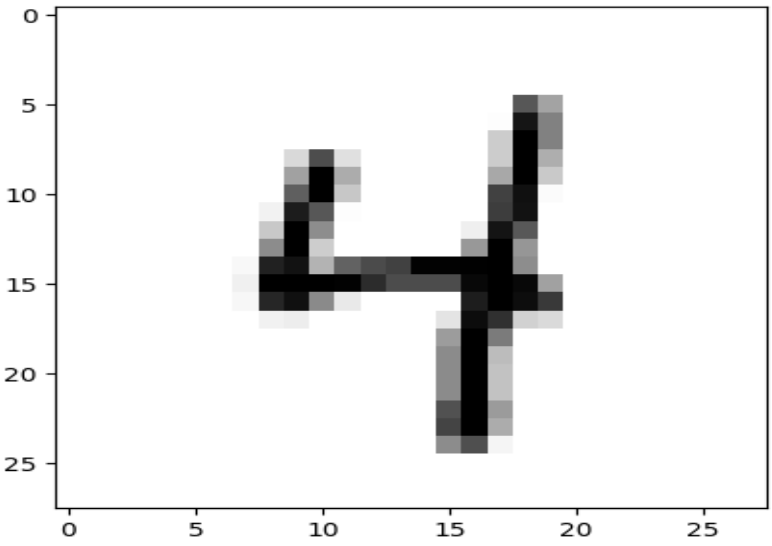
Exemple 1 :

```
[35] plt.imshow(x_test[9999],cmap=plt.cm.binary)
     plt.show()
```



```
np.argmax(predcta[9999])
6
```

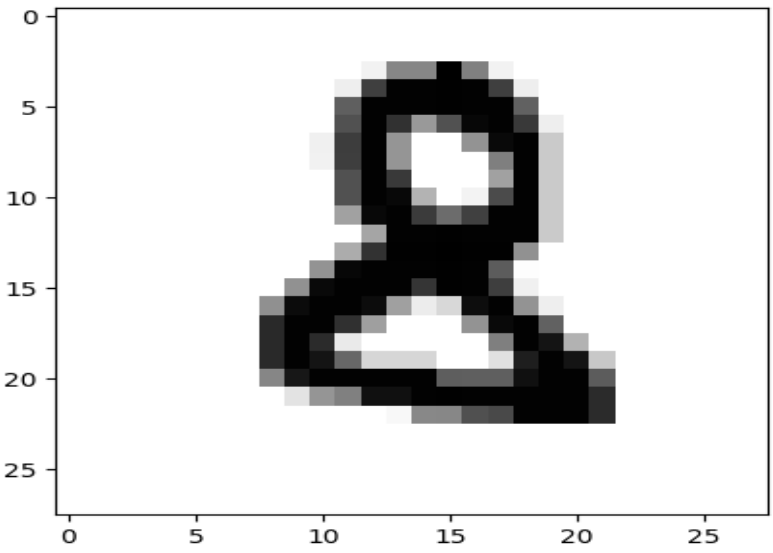
Exemple 2 :




```
[36] np.argmax (predcta[19])

     4
```

Exemple 3 :




```
np.argmax (predcta[1955])

2
```

Le modèle n'a pas reconnu l'image du chiffre 8 et il a prédit qu'elle est 2 donc ce modèle n'a pas bien réussi à classer l'image manuscrites car la prédiction était incorrecte.

3.4 2 CNN avec architecture LeNet

Dans le but de reconnaître les chiffres manuscrits, l'architecture LeNet de Keras sert de point de départ pour les travaux de classification d'images et de guide pour comprendre les éléments essentiels d'un CNN grâce à son architecture modulaire et sa simplicité.

Voici un résumé de l'architecture LeNet de Keras :

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_16 (Conv2D)          (None, 28, 28,32)         320

 max_pooling2d_16 (MaxPoolin  (None, 13, 13, 32)       0
 g2D)

 conv2d_17 (Conv2D)          (None, 11, 11, 64)        18496

 max_pooling2d_17 (MaxPoolin  (None, 5, 5, 64)         0
 g2D)

 flatten_8 (Flatten)         (None, 1600)              0

 dense_16 (Dense)            (None, 128)               204928

 dense_17 (Dense)            (None, 10)                1290

=================================================================
Total params: 225,034
Trainable params: 225,034
Non-trainable params: 0
_________________________________________________________________
```

Tableau 9:L'architecture des couches du modèle LeNet.

Evaluation du modèle :
L'architecture LeNet dans Keras est évaluée par ses performances sur la tâche de classification des images. L'évaluation de l'architecture sont les suivantes:

```
313/313 [==============================] – 0s 1ms/step – loss: 0.2827 – accuracy:
0.9210

Test accuracy: 0.9210000038146973
```

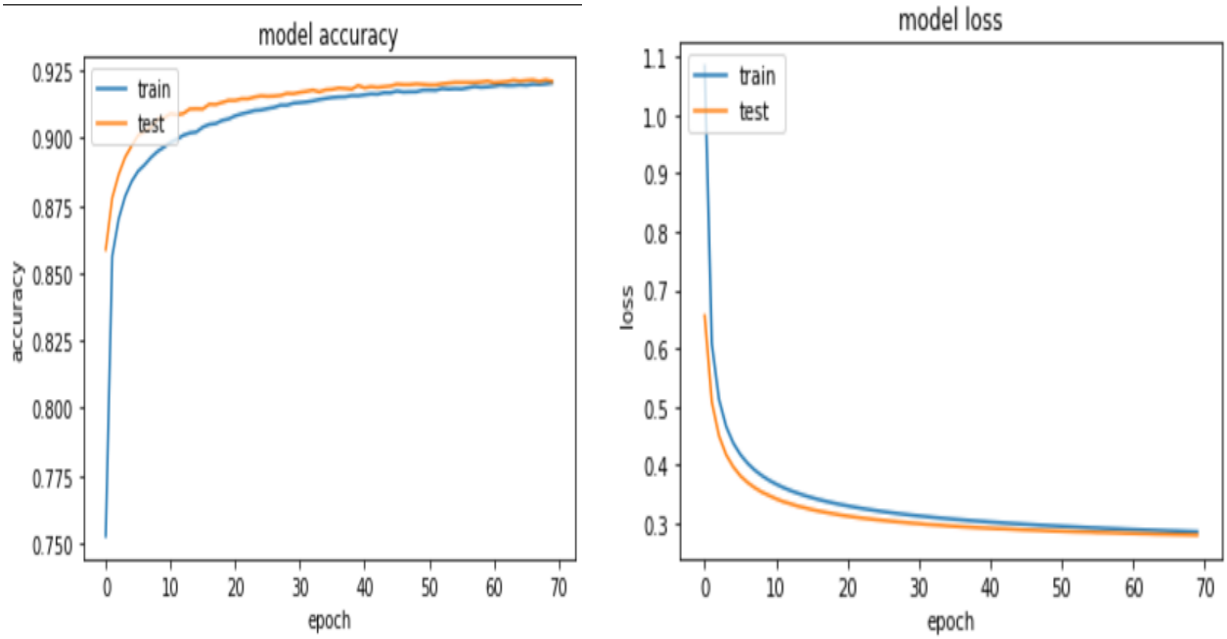
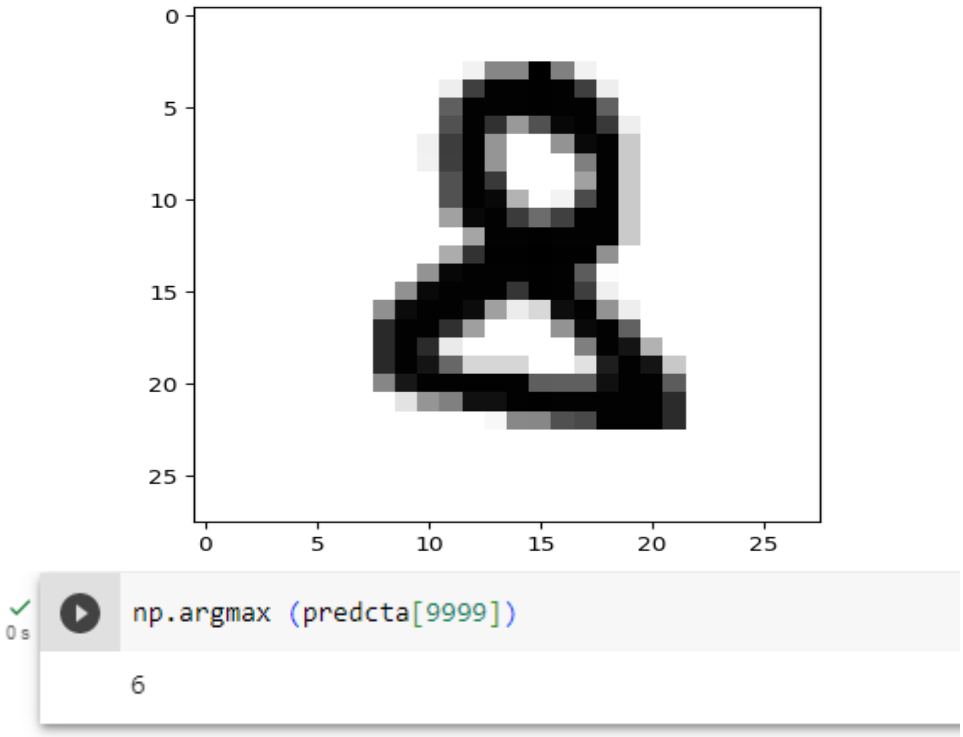


Figure 54: La précision et la fonction de perte sur l'ensemble de formation et de test en fonction du nombre d'époques dans le modèle LeNet.

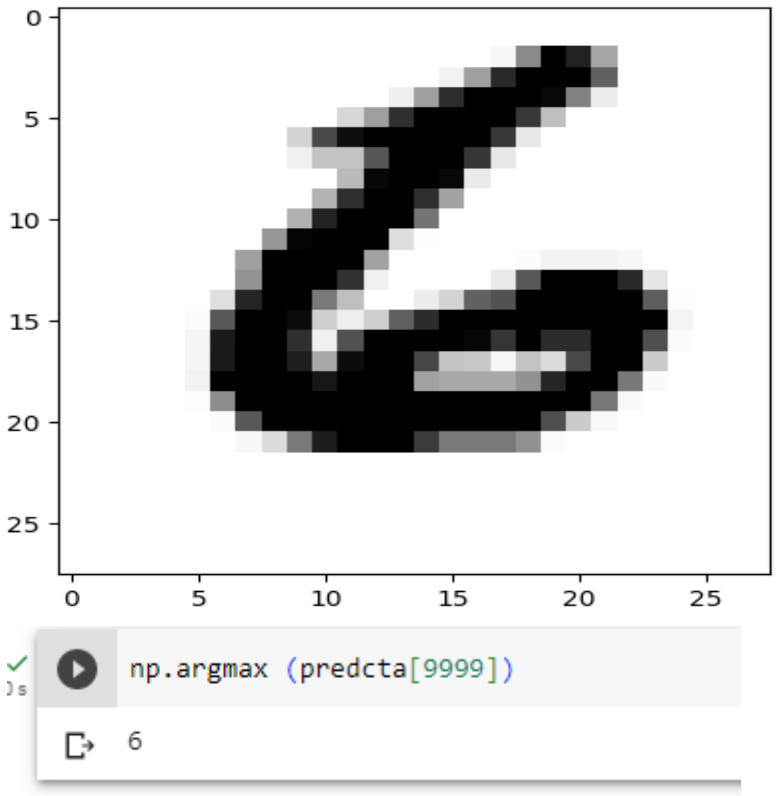
Proposition des prédictions :

Exemple 1 :

Nous avons testé deux images du chiffres 8 et 6, nous avons appliqué le modèle sur ces deux images, et nous pouvons voir les résultats au-dessus des photos.



Exemple 2 :



3.4 3 CNN avec ResNet-50

ResNet-50, un réseau à 50 couches qui comprend des blocs résiduels, a affiché des performances exceptionnelles dans une gamme d'applications de vision par ordinateur. U Un échantillon de l'architecture ResNet-50 dans Keras est décrite comme suite :

conv5_block3_1_bn (BatchNormali	(None, 1, 1, 512)	2048	conv5_block3_1_conv[0][0]
conv5_block3_1_relu (Activation	(None, 1, 1, 512)	0	conv5_block3_1_bn[0][0]
conv5_block3_2_conv (Conv2D)	(None, 1, 1, 512)	2359808	conv5_block3_1_relu[0][0]
conv5_block3_2_bn (BatchNormali	(None, 1, 1, 512)	2048	conv5_block3_2_conv[0][0]
conv5_block3_2_relu (Activation	(None, 1, 1, 512)	0	conv5_block3_2_bn[0][0]
conv5_block3_3_conv (Conv2D)	(None, 1, 1, 2048)	1050624	conv5_block3_2_relu[0][0]
conv5_block3_3_bn (BatchNormali	(None, 1, 1, 2048)	8192	conv5_block3_3_conv[0][0]
conv5_block3_add (Add)	(None, 1, 1, 2048)	0	conv5_block2_out[0][0] conv5_block3_3_bn[0][0]
conv5_block3_out (Activation)	(None, 1, 1, 2048)	0	conv5_block3_add[0][0]
flatten_2 (Flatten)	(None, 2048)	0	conv5_block3_out[0][0]
Abhishek_layer (Dense)	(None, 10)	20490	flatten_2[0][0]

Total params: 23,608,202
Trainable params: 20,490

Tableau 11:L'architecture des couches du modèle ResNet-50.

Tableau 10:

L'architecture ResNet-50 dans Keras est évaluée par ses performances sur la tâche de classification des images. L'évaluation de l'architecture sont les suivantes:

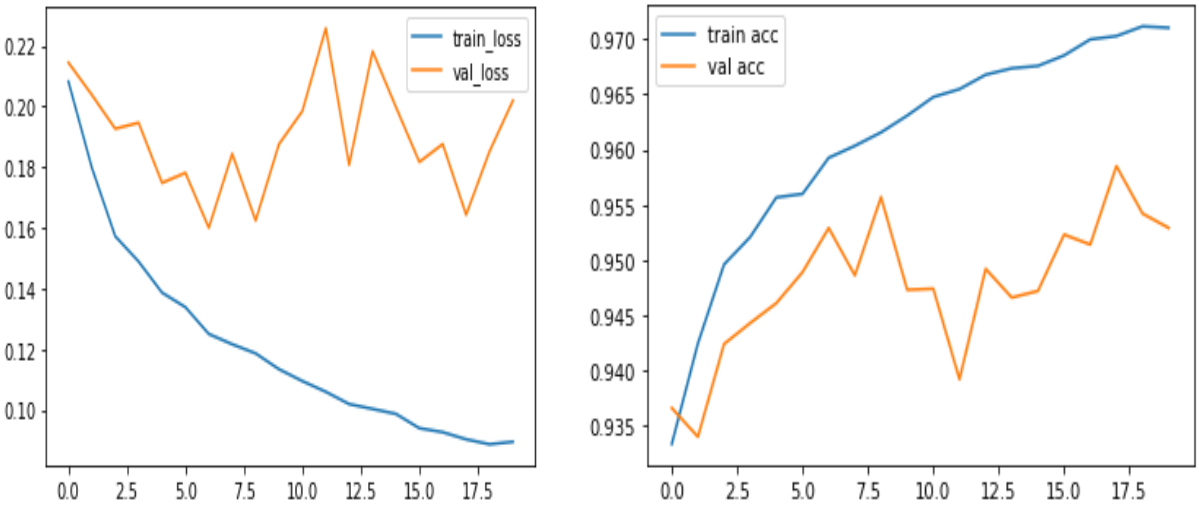
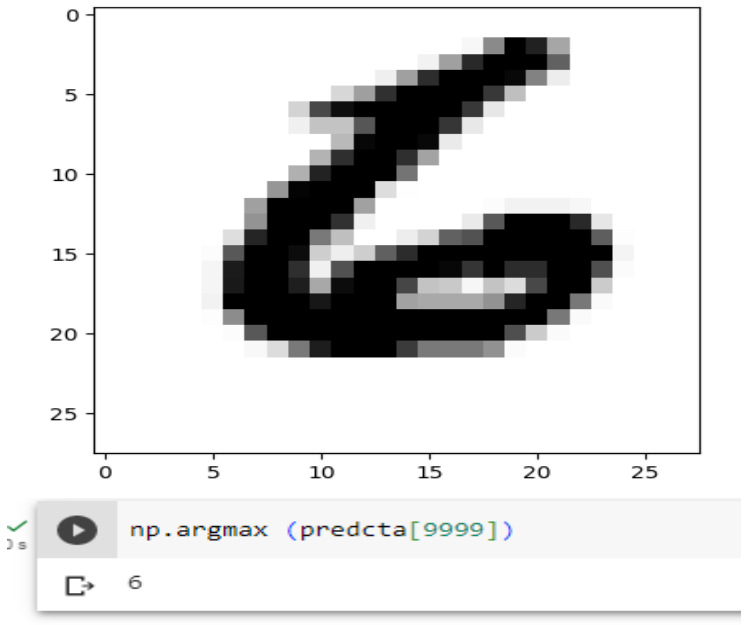


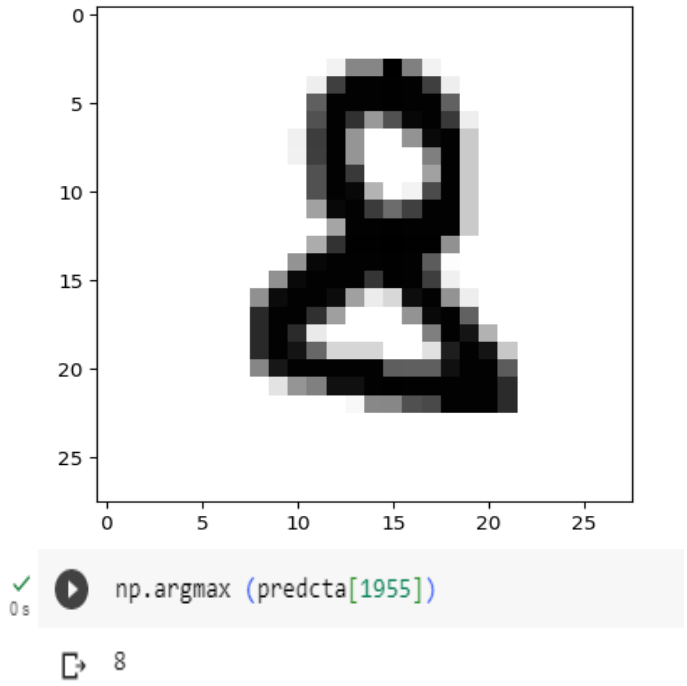
Figure 55 La précision et la fonction de perte sur l'ensemble de formation et de test en fonction du nombre d'époques dans le modèle ResNet-50.

Proposition des prédictions :

Example 1 :



Exemple 2 :



3 5 Comparaison

	Modèle amélioré	TinyVGG	LeNet	ResNet-50
Précision de l'entraînement	99,41%	96%	92%	95,29%
Certitude d'essai	99,39%	97%	92,1%	97,1%
Perte	0,0283	0,0942	0,2827	0,0894
Nombre de paramètres	391370	4440	225034	23608202

Tableau 12: Tableau de comparaison entre différentes architectures du CNN.

3 6 Présentation de l'application

L'application est un logiciel accessible via un navigateur Web qui permet aux utilisateurs d'identifier et de prédire des chiffres écrit à la main. Pour comprendre et évaluer les images saisis ou les traits, l'application utilise le modèle modifié qui été entraînés sur un ensemble de données MNIST de nombres manuscrits.

3.6 1 Interface « Accueil »

Au démarrage de l'application, les utilisateurs peuvent interagir avec l'interface < Accueil > en désigner directement sur un Canvas virtuel à l'aide d'une souris ou d'un écran tactile le champ (1), il y a la possibilité de télécharger une image d'un chiffre manuscrit depuis l'appareil le champ (2). Les utilisateurs comprennent facilement le processus de saisie à l'aide d'indices clair et virtuels dans l'interface qui est conçue pour être simple à utiliser et intuitive.

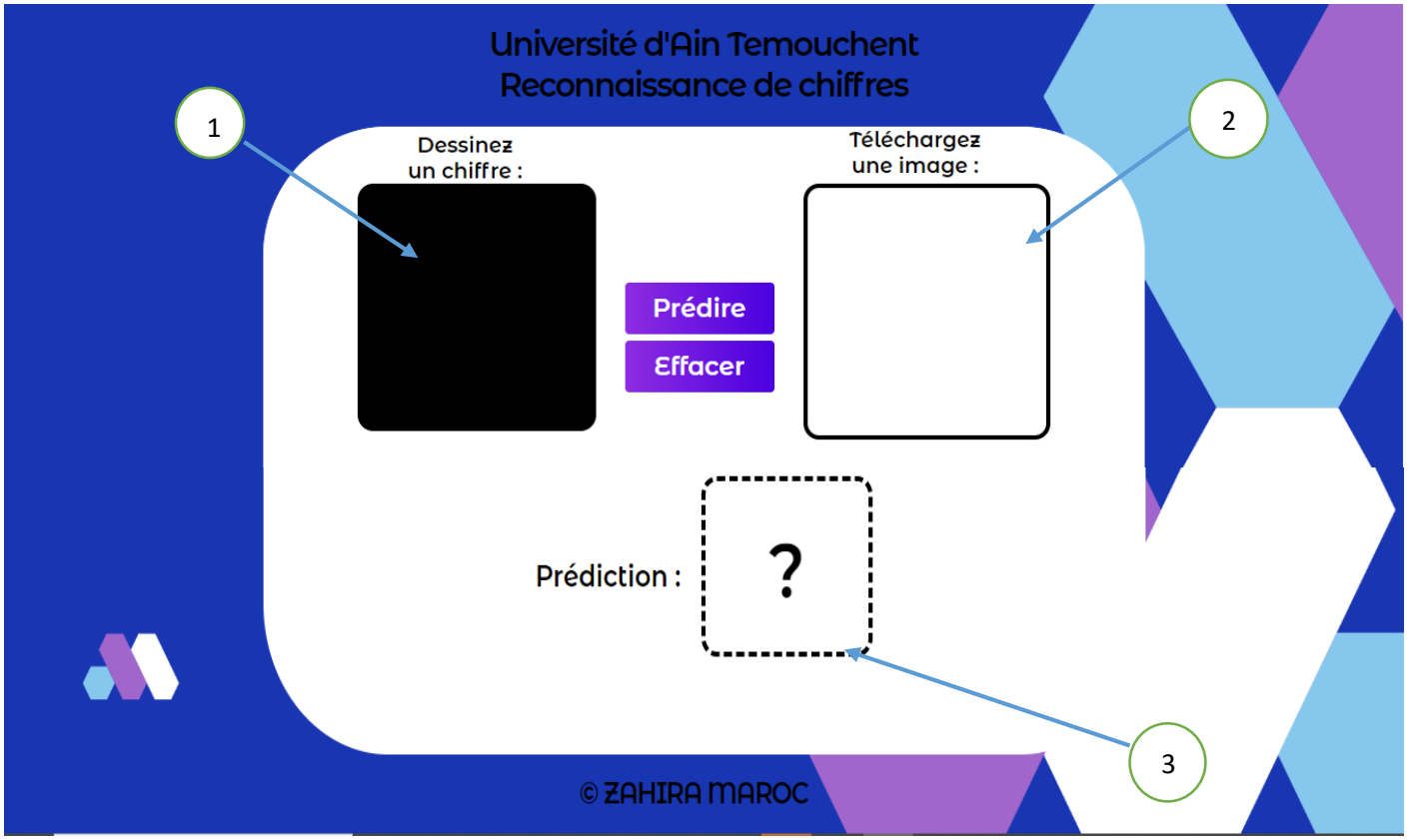


Figure 56: Interface d'accueil.

Lorsque l'utilisateur clique sur le champ (2), il donne l'accès à choisir une image parmi l'ensemble de test, l'interface illustrée à la figure 3.13.

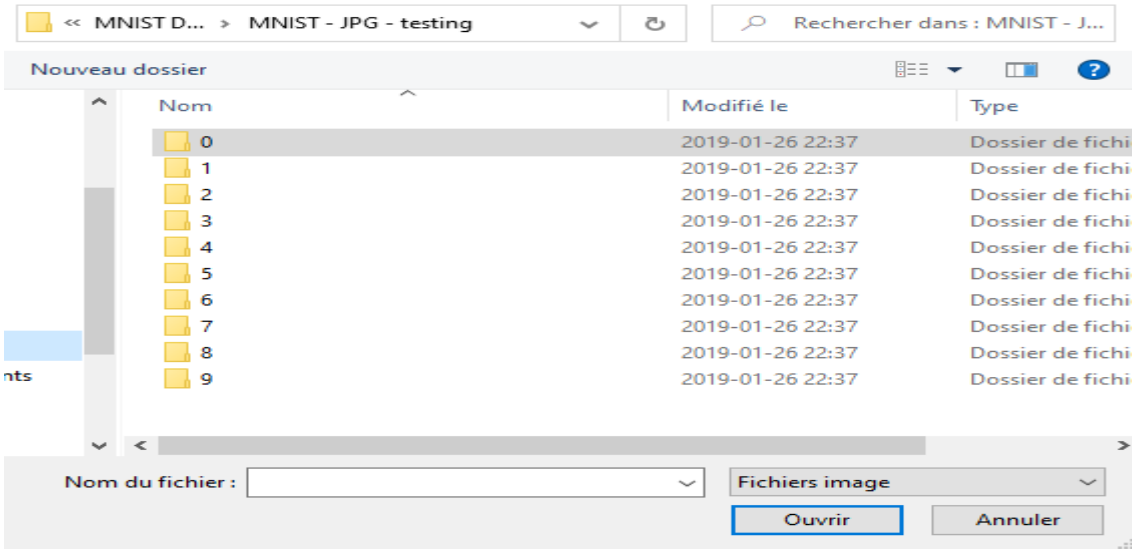


Figure 57: Interface de choix d'images.

Lors du le téléchargement ou dessinent un chiffre, le modèle d'apprentissage profond (CNN) examinent l'entrer, on clique sur le bouton <Prédire> pour identifier et reconnaitre les chiffres, <Effacer> permettre à l'utilisateur d'effacer et changer l'image ou dessin.

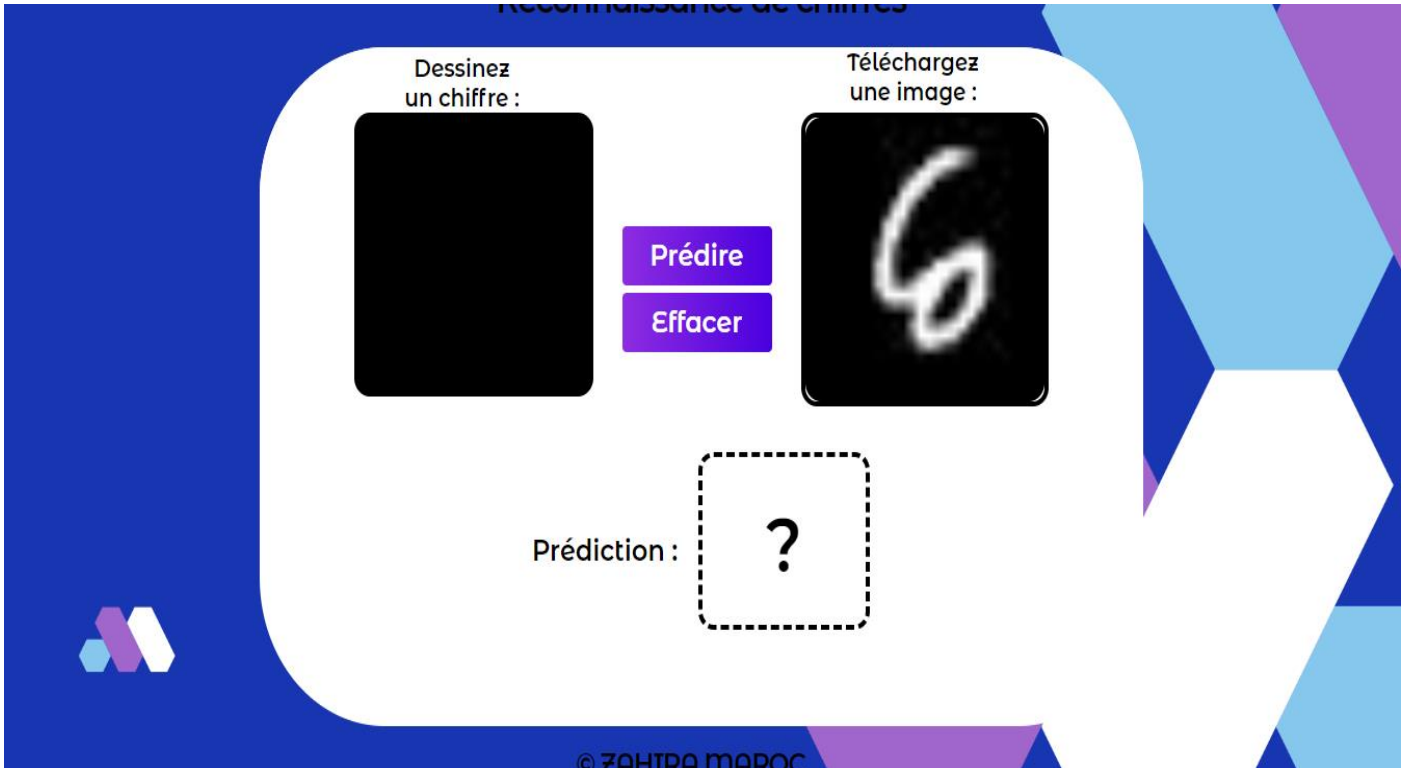


Figure 58: Interface après la sélection d'image.

Le champ (3) affiche à l'utilisateur le résultat de la reconnaissance de chiffre accompagnés avec la probabilité pour montrer la fiabilité et indique le niveau de certitude de reconnaissance.

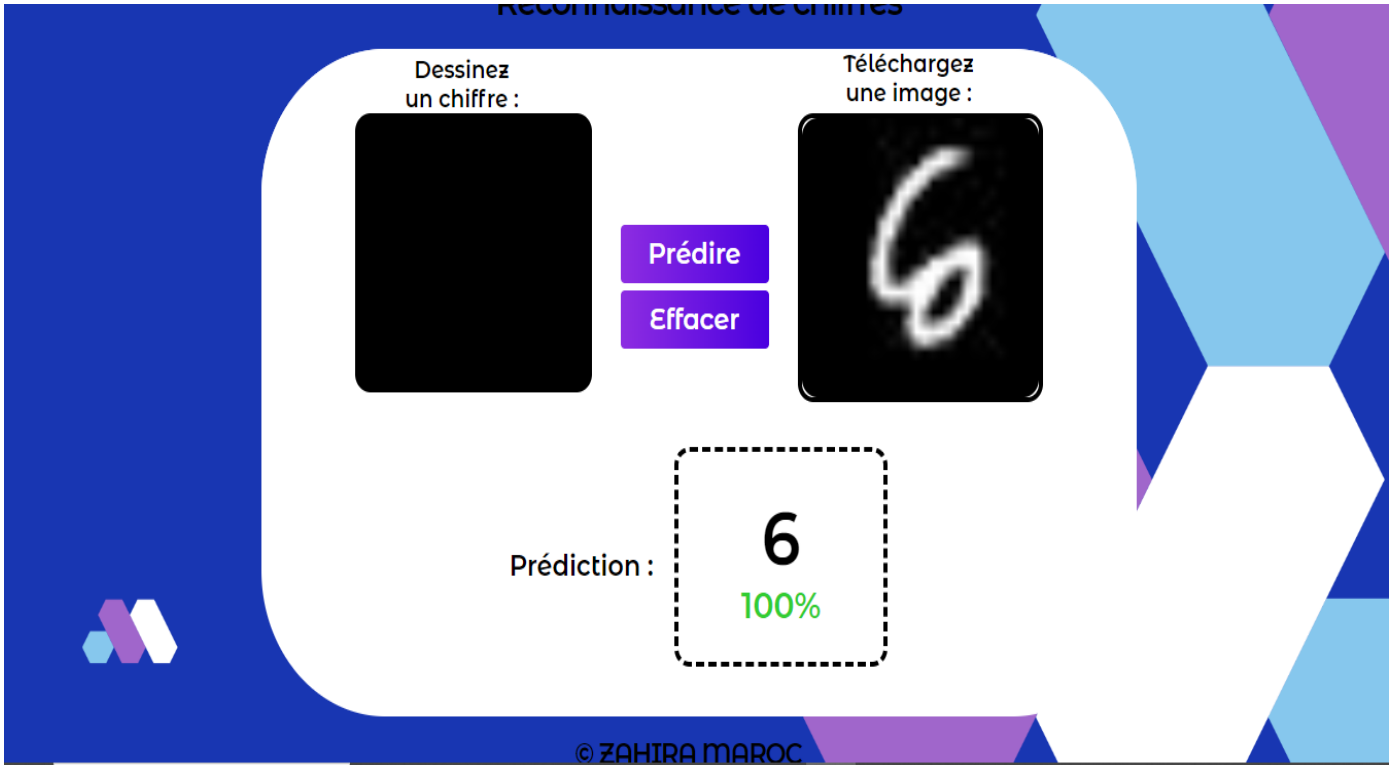


Figure 59: Interface après l'affichage du résultat.

3.6 2 Exemples

Voici des exemples de notre système de reconnaissance des chiffres manuscrits :

- La reconnaissance automatique avec le dessin sur le Canvas

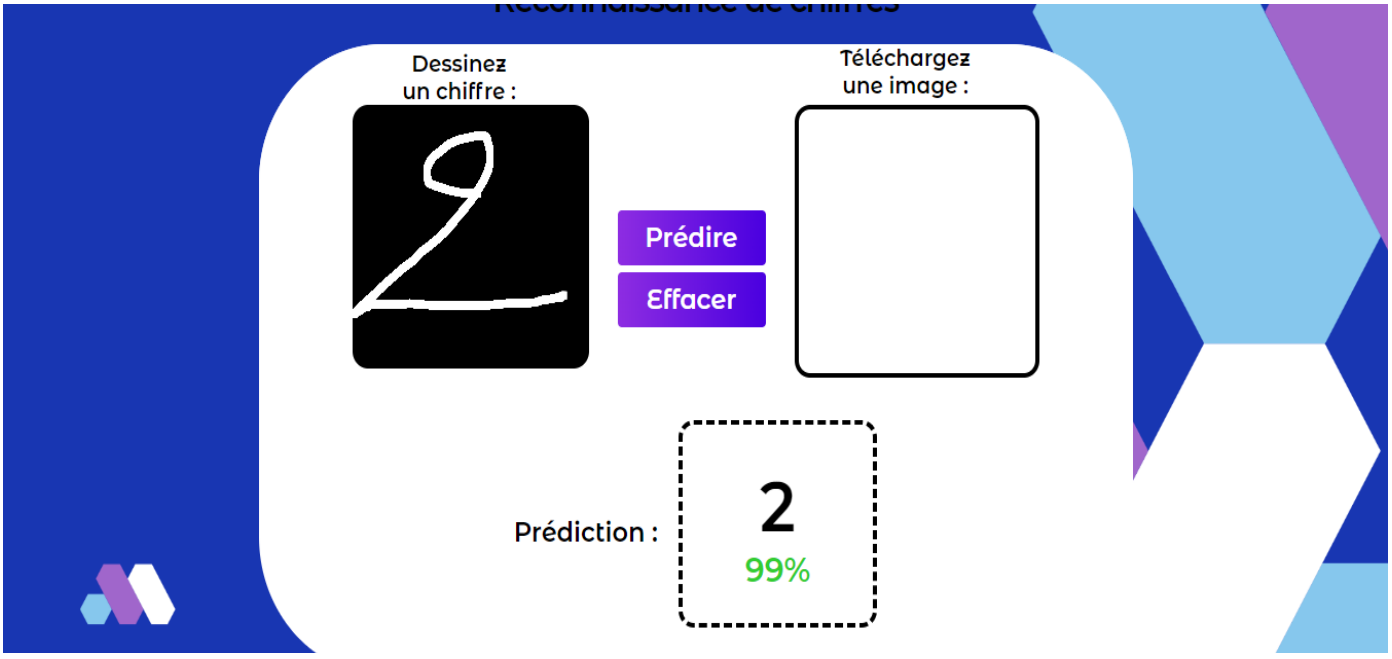


Figure 60: Exemple d'image contenant le chiffre "2".

Exemple 2 :

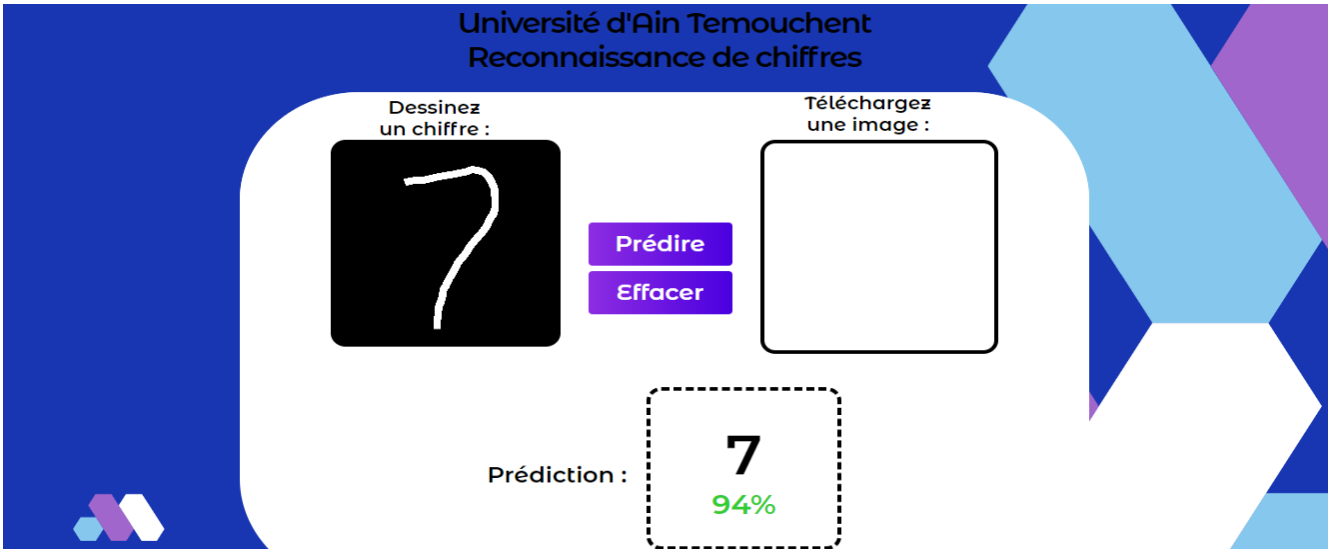


Figure 61: Exemple d'image contenant le chiffre "7".

Exemple 3 :

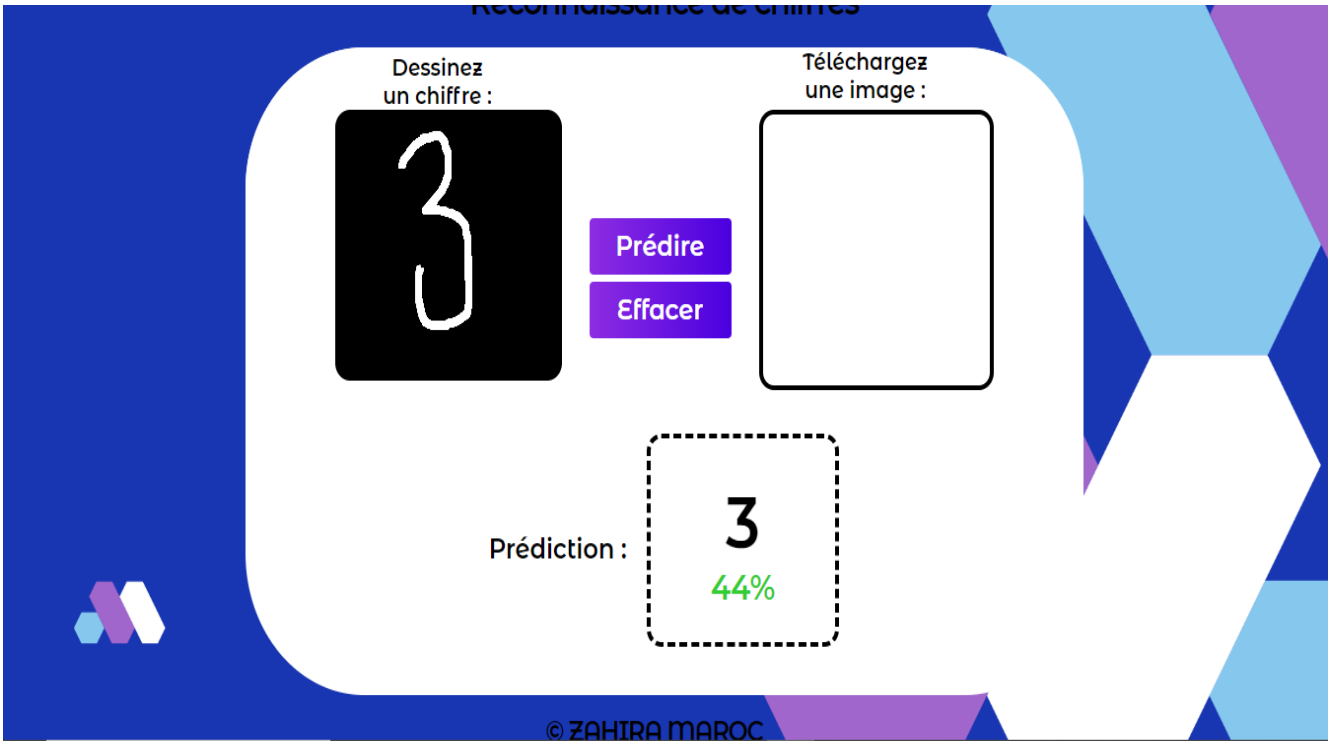


Figure 62: Exemple d'image contenant le chiffre "3".

Exemple 4 :

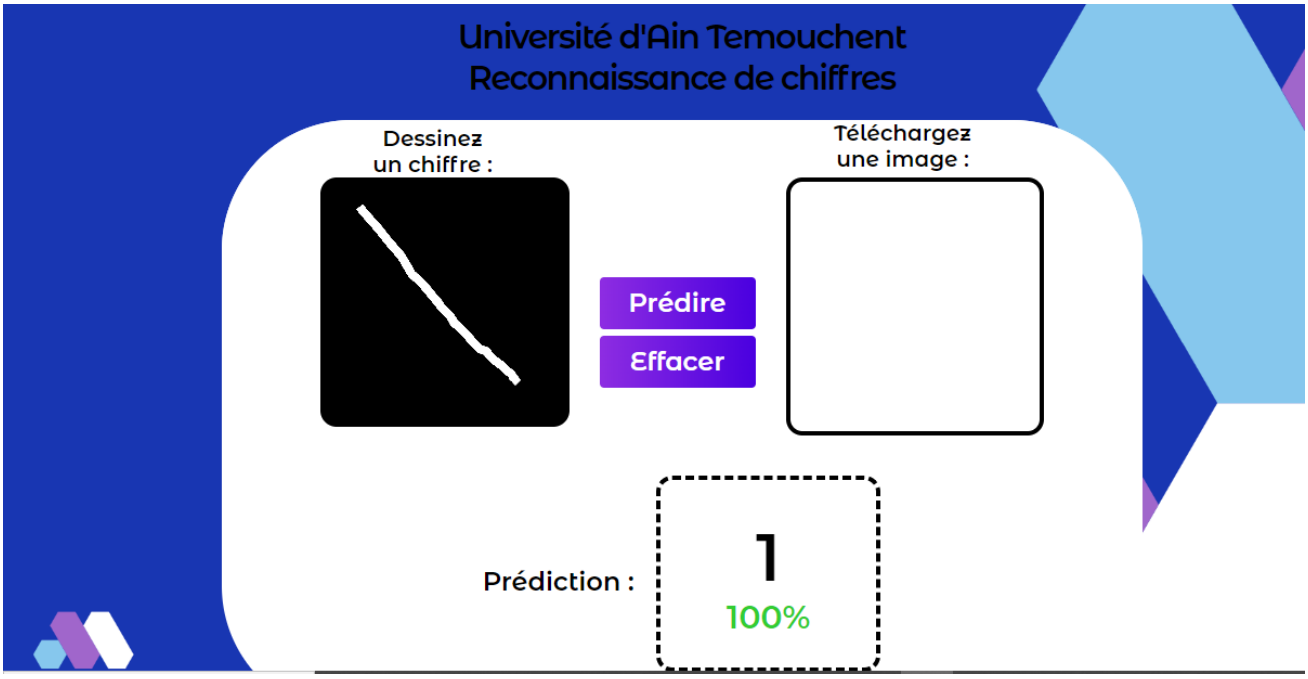


Figure 63: Exemple d'image contenant le chiffre "1".

- La reconnaissance automatique avec le téléchargement des images

Exemple 1 :

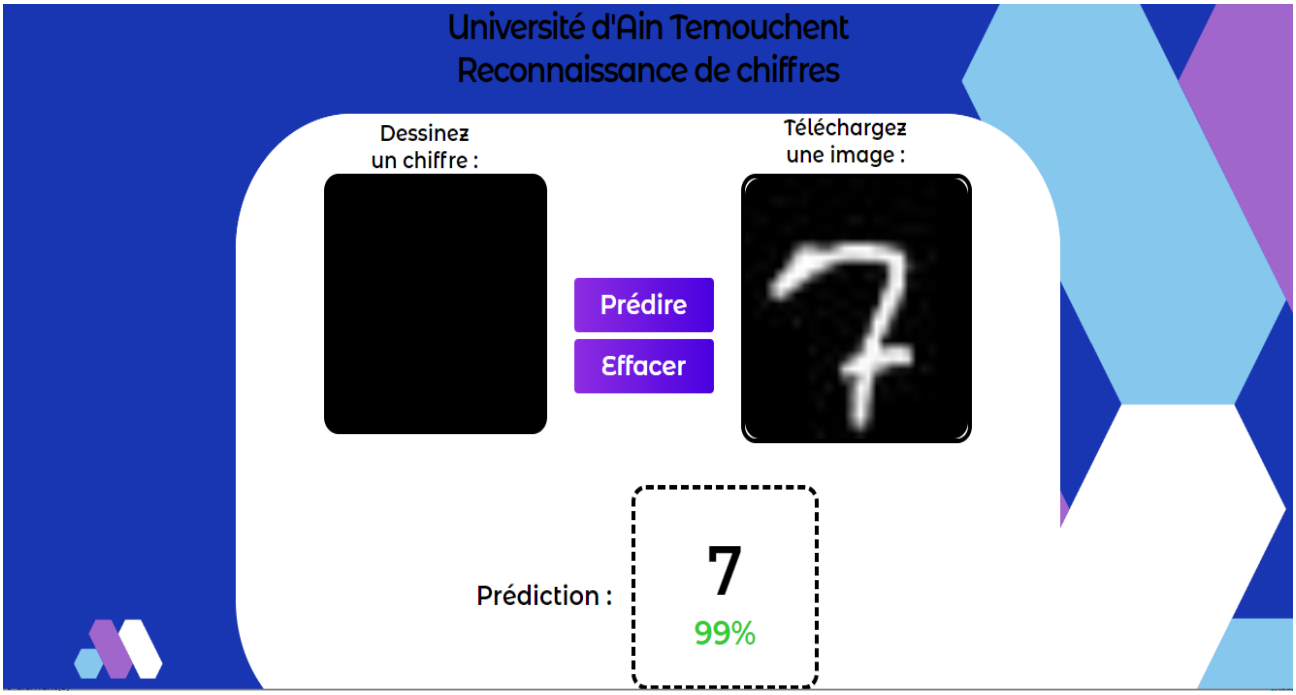


Figure 64: Exemple d'image contenant le chiffre "7".

Exemple 2 :

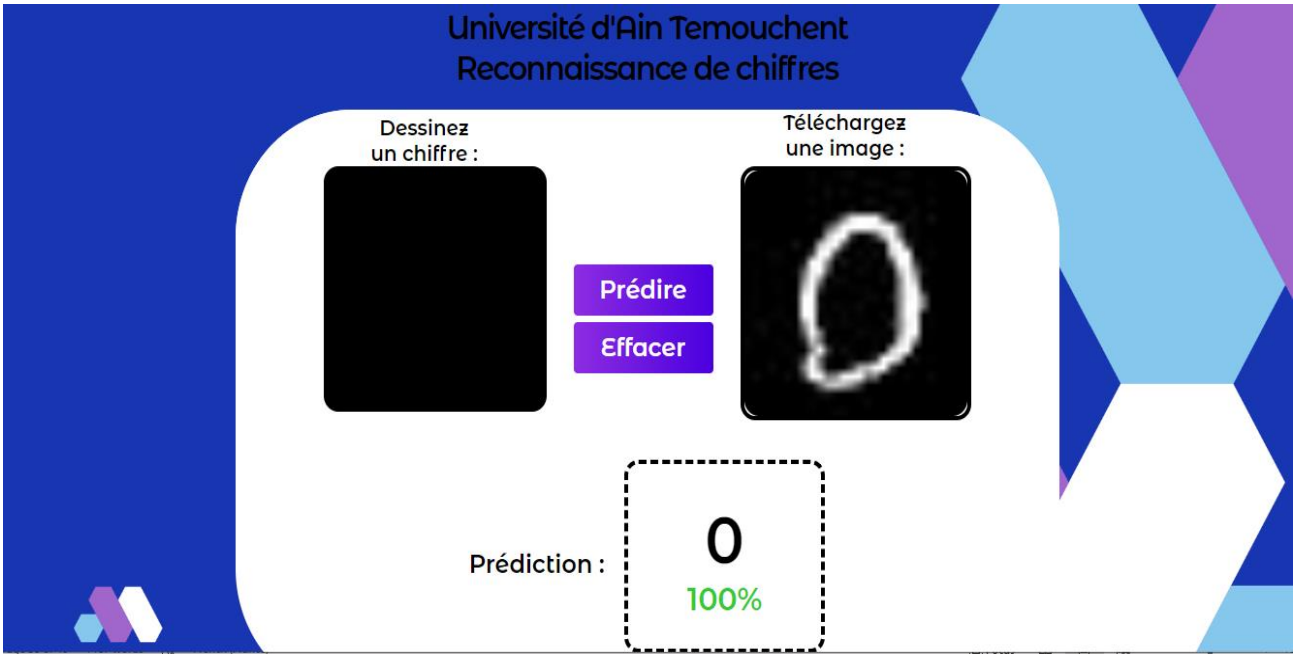


Figure 65: Exemple d'image contenant le chiffre "0".

3 7 Discussion des résultats

a) Comparaison de différents modèles

Nous avons comparé notre modèle avec d'autres architectures, notamment ResNet-50, TinyVGG et LeNet. Avec une amélioration de près de 7% par rapport aux autres architectures, malgré le nombre de paramètres de notre modèle amélioré est élevé par rapport TinyVGG et LeNet notre modèle s'est avéré bien supérieur en termes de performances lorsqu'il s'agit de lire des nombres manuscrits.

Nous avons utilisé la précision comme critère clé pour évaluer la performance de notre système. L'exactitude mesure la proportion d'images correctement classées par rapport à l'ensemble de test. De plus, nous avons calculé la fonction de perte pour permettre un entraînement efficace et des prédictions précises de classification par classe de chiffres.

b) Performances globales

Sur l'ensemble de tests, notre modèle de reconnaissance manuscrite des chiffres avait un taux de précision 99,39% et pour la précision de l'entrainement notre modèle atteint 99,41% et nous avons une valeur négligeable pour la fonction de perte par rapport les autres architectures. Cela démontre la puissance et l'efficacité de notre modèle basée sur CNN pour lire les chiffres manuscrits.

c) Impact des techniques de prétraitement

Avant la formation des modèles, nous avons observé que l'application du prétraitement de l'image avait un effet considérable sur les performances de reconnaissance. La normalisation a réduit les variations de luminosité et augmenté l'adaptabilité du modèle à divers scénarios d'éclairage.

d) Analyse des différentes classes de chiffres

Dans plusieurs classes de chiffres, nous avons constaté des différences de performances. Une précision de plus de 99% a été obtenue dans la reconnaissance de chiffres comme 1, 2 et 8. Les taux de reconnaissance des nombres 3 et 7 étaient légèrement inférieurs, avec des précisions de 44% et 74%, respectivement. Cet écart est causé par des similitudes structurelles entre certains chiffres, ce qui rend plus difficile pour le modèle de les distinguer.

e) Analyse des erreurs :

Certaines tendances ont émergé d'une analyse des erreurs de classification. En raison de leurs structures similaires, les nombres 8 et 2 sont parfois erronés. Notre modèle a également été mis à l'épreuve par des chiffres déformés ou mal écrits, ce qui a entraîné des erreurs de classification.

f) Comparaison de différents nombres d'époque de notre modèle

Les performances et le comportement d'un modèle d'apprentissage en profondeur peuvent être considérablement affectés par le nombre d'époques choisies. Les conséquences des différents numéros d'époque sont contrastées ci-dessous :

Paramètre d'époques	Précision (Accuracy)	Perte (Loss)
1	0 ,8589	0,4975
10	0,9915	0 ,0327
20	0 ,9939	0,0283
100	0,9932	0,0579

Tableau 13: Tableau de comparaison nombre d'époque.

Nous avons remarqué qu'en prenant 20 époques, la précision sur l'ensemble du test est meilleure et pour la fonction de perte on obtient un minimal valeur qui tente vers 0.

3 8 Conclusion

Dans ce chapitre, nous avons présenté les outils de développement et la base de données utilisé, nous avons expliqué le principe et la mise en œuvre de notre système et les trois architectures du CNN notamment ResNet, LeNet et TinyVGG.

Ce chapitre a été concentré sur l'évaluation des modèles sur l'ensembles de données et le développement d'application web qui permet aux utilisateurs d'entrer des chiffres isolés via une page web pour présenter quelques résultats expérimentaux, notre système a montré que l'architecture et les techniques d'amélioration de précision fonctionnent assez bien.

CONCLUSION GENERAL

Le développement d'un système hautement efficace et fiable a été rendu possible grâce à la révolution de l'apprentissage profond dans la reconnaissance des chiffres manuscrits. Ces systèmes offrent plusieurs avantages pour une variété d'application, ils ont considérablement réduit la saisie des données, économisant du temps et améliorant la productivité dans l'industrie.

Dans ce mémoire nous avons réalisé un système de reconnaissance des chiffres manuscrits isolés basé sur les techniques de prétraitement et les réseaux de neurones convolutifs CNNs qui permettent d'analyser et de traduire des chiffres manuscrits en caractères numériques lisibles et exploitables par la machine. Nous avons discuté des types de classifieurs utilisés dans les systèmes de reconnaissance des chiffres manuscrits.

Ensuite, nous avons construit les modèles de réseaux neuronaux et implémenté plusieurs architectures populaires. Grâce à l'algorithme de réseau neuronal convolutif notre système est capable de gérer et de reconnaître les chiffres écrits par différentes personnes, réalisant de meilleures performances en termes de taux de reconnaissance. Nous avons montré que le bon ajustement des hyperparamètres implique l'amélioration des performances du réseau neuronal convolutif.

L'évaluation de notre modèle après l'entraînement et la validation sur le jeu de données MNIST a atteint un taux de reconnaissance de précision de 99,39 %. Puis, nous avons développé une application web qui permet à l'utilisateur de prédire les chiffres manuscrits. Enfin, nous avons réalisé quelques tests et les résultats obtenus démontrent l'efficacité et la fiabilité de notre système.

Selon la mise en œuvre du système et la technique employée, un système de reconnaissance automatique des chiffres manuscrits isolés peut présenter des limitations différentes. Cependant, les difficultés et contraintes typiques suivantes que le système pourrait rencontrer : Ambiguïté entre les chiffres, Variabilité de l'écriture manuscrite et Données d'entraînement limitées.

Il y a plusieurs perspectives à considérer lorsqu'on envisage l'avenir des systèmes de reconnaissance des chiffres manuscrits :

- Prise en charge multilingue, il existe un besoin de systèmes capables de détecter les chiffres dans plusieurs langues à mesure que les industries et les services se mondialisent. Pour que ces systèmes soient largement utilisés, il sera essentiel d'étendre leur capacité à détecter les chiffres de différents caractères, tels que le chinois, l'arabe ou bengali.
- La reconnaissance en mode réel.
- Tester le modèle avec d'autres bases de données.
- Améliorer le taux de précision par combiner les CNNs avec d'autres classifieurs ou par l'ajustement des hyperparamètres.
- Elargir la bases de données avec plus de données.
- Enrichir notre système afin de reconnaître les images en couleurs.

Références Bibliographiques

3 Références

[1] Baheti, P. (2021). *Un guide complet sur les réseaux de neurones convolutifs*. Consulté le avril 2023, sur V7Labs: https://www.v7labs.com/blog/convolutional-neural-networks-guide

[2] Debasish, K. (2022, mars 3). *Bases de CNN dans l'apprentissage en profondeur*. Consulté le avril 2023, sur analytics vidhya: https://www.analyticsvidhya.com/blog/2022/03/basics-of-cnn-in-deep-learning/

[3] Guedri, m. (2019). *La reconnaissance des chiffres manuscrits isolés en utilisant l'apprentissage profond.* master Systèmes d'information, Univérsité Larbi Tébessa. Consulté le janvrier 2023, sur https://www.theses-algerie.com/1691451375396032/memoire-de-master/universite-larbi-tebessi---tebessa/la-reconnaissance-des-chiffres-manuscrits-isol%C3%A9s-en-utilisant-l-apprentissage-profond

[4] Guizhu, S., Qingping, T., & Haoyu, Z. (2018). Deep Learning with Gated Recurrent Unit Networks for Financial Sequence Predictions. Procedia Computer Science. Consulté le mars 2023, sur https://doi.org/10.1016/j.procs.2018.04.298

[5] Gurucharan, M. (2022, JUILLET 28). *Architecture CNN de base : explication des 5 couches du réseau neuronal convolutif*. Consulté le avril 2023, sur upgrad: https://www.upgrad.com/blog/basic-cnn-architecture/

[6] LeCun, Y., Institute, C., & NYU. (s.d.). *THE MNIST DATABASE of handwritten digits*. Consulté le mai 2023, sur http://yann.lecun.com/exdb/MNIST/

[7] NOUR, A. (2019). *Un guide complet pour construire un réseau de neurones convolutifs (CNN)*. Consulté le avril 2023, sur https://www.aliens-sci.com/convolutional-neural-network-cnn/

[8] Ritik , D., Rishika , K., & Samay , P. (2021, juin 23). *Handwritten Digit Recognition using Machine and.* Consulté le février 2023, sur https://doi.org/10.48550/arXiv.2106.12614

[9] Rukshan, P. (2022, juin 20). *L'architecture du réseau neuronal convolutif (CNN) expliquée en langage simple à l'aide de schémas simples*. Consulté le mars 2023, sur towardsdatascience: https://towardsdatascience.com/convolutional-neural-network-cnn-architecture-explained-in-plain-english-using-simple-diagrams-e5de17eacc8f

[10] A.G, Hochuli, O & A. S., B. (2018). Handwritten Digit Segmentation: Is it still necessary? Récupéré sur https://doi.org/10.1016/j.patcog.2018.01.004

[11] Abdeljalil , G., chawki , D., & Youcef , C. (2017). *Oriented Basic Image Features Column for isolated handwritten digit.* Récupéré sur https://doi.org/10.1145/3129186.3129189

[12] Abdeljalil, G., Djeddi , C., & Chibani, Y. (2017, janvier). Isolated Handwritten Digit Recognition Using oBIFs and Background Features . Récupéré sur https://ieeexplore.ieee.org/document/7490135.

[13] Abdeljalil, G., Djeddi , C., Chibani , Y., & Siddiqi , I. (2014). Improving Isolated Digit Recognition using a Combination of Multiple Features. Récupéré sur https://ieeexplore.ieee.org/document/6981060

[14] Abderrahim, H. S., & HAMMOUTI, M. A. (2022). *Développement d'un système de reconnaissance de caractères arabes manuscrits* . université Ain Témouchent-Belhadj Bouchaib.

[15] Alzubaidi , L., Zhang , J., & J.Humaidi , A. (2021, Mars 31). Revue du deep learning : concepts, architectures CNN, enjeux, applications, orientations futures. *Big Data* . Consulté le mars 2023, sur https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00444-8

[16] AMROUCH, M. (2012). *Reconnaissance de caractères imprimés et manuscrits,textes et documents basée sur les modèles de Markov cachés* . THÈSE DE DOCTORAT ÈS-SCIENCES, Université Ibn Zohr Faculté des Sciences d'Agadir. Récupéré sur https://www.ircam.ma/sites/default/files/doc/asinag-9/resumes-de-theses.pdf

[17] Andre G, H., Luiz S, O., Alceu de Souza, B., & Robert , S. (2018 , octobre). Segmentation-Free Approaches for Handwritten Numeral String Recognition. Récupéré sur https://ieeexplore.ieee.org/document/8489233

[18] Arnaud, W.-B. (2020). *Application des algorithmes d'apprentissage automatique pour la détection de défauts de roulements sur les machines tournantes dans le cadre de l'Industrie 4.0.* mémoire dde fin d'étude, Université du Québec à Chicoutimi, canada. Récupéré sur https://constellation.uqac.ca/id/eprint/5857/

[19] Bastien, L. (2021, juin). *Reinforcement Learning : qu'est-ce que l'apprentissage par renforcement ?* Consulté le mars 2023, sur lebigdata.fr: https://www.lebigdata.fr/reinforcement-learning-definition.

[20] Beklemysheva, A. (s.d.). *Why Use Python for AI and Machine Learning?* Consulté le mai 2023, sur Steel kiwi: https://steelkiwi.com/blog/python-for-ai-and-machine-learning/

[21] Belaïd, A. (s.d.). Reconnaissance automatique de l'écriture et du document. *239*(54506). Récupéré sur https://members.loria.fr/ABelaid/publis/pour_la_science.pdf

[22] BEN AISSA , c., & CHERGUI , A. (2022). *Reconnaissance des chiffres en utilisant un CNN.* MOHAMED BOUDIAF - M'SILA, Electronique. Consulté le mai 2023

[23] benahmed, n. (2002). *Optimisation de réseaux de neurones pour la reconnaissance de chiffres manuscrits isolés : sélection et pondération des primitives par algorithmes génétiques.* mémoire de fin d'étude, École de technologie supérieure, Montréal. Récupéré sur https://espace.etsmtl.ca/id/eprint/806/.

[24] Benlahmar. (2018, October 12). *Les réseaux de neurones convolutifs*. Consulté le mars 2023, sur datasciencetoday: https://www.datasciencetoday.net/index.php/en-us/deep-learning/173-les-reseaux-de-neurones-convolutifs

[25] Biswal , A. (2023, février 16). *Top 10 Deep Learning Algorithms You Should Know in 2023*. Consulté le mars 2023, sur simplilearn: https://www.simplilearn.com/tutorials/deep-learning-tutorial/deep-learning-algorithm

[26] Charles, T. (2021, septembre). *Qu'est-ce que le Machine Learning ?* Consulté le février 2023, sur kobia.fr: https://kobia.fr/quest-ce-que-le-machine-learning/

[27] Chawki, D. (2014). *Contribution à l'analyse et la caractérisation de l'écriture manuscrite.* UNIVERSITE BADJI MOKHTAR-ANNABA. Récupéré sur https://www.researchgate.net/profile/Chawki-Djeddi/publication/280979425_Contribution_a_l'analyse_et_la_caracterisat

[28] Chengwei. (s.d.). *Quick Notes on How to choose Optimizer In Keras*. Consulté le mai 2023, sur DLology: https://www.dlology.com/blog/quick-notes-on-how-to-choose-optimizer-in-keras/

[29] Christophe, O. (2015, août 27). *Comprendre les réseaux LSTM*. Consulté le mars 2023, sur le blog de cola: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[30] d, s. (s.d.). *Classification par K-Plus Proches Voisins*. Consulté le mars 2023, sur irrlicht-fr: irrlicht-fr.org

[31] DAP. (s.d.). *RÉSEAUX DE NEURONES RÉCURRENTS*. Consulté le mars 2023, sur data analytics post: https://dataanalyticspost.com/Lexique/reseaux-de-neurones-recurrents/

[32] datascientest. (s.d.). *NumPy : la bibliothèque Python la plus utilisée en Data Science*. Consulté le mai 2023, sur datascientest: https://datascientest.com/numpy

[33] Dharmaraj. (2022, juin 1). *Réseaux de neurones convolutifs (CNN) - Explication de l'architecture*. Consulté le février 2023, sur medium: https://medium.com/@draj0718/convolutional-neural-networks-cnn-architectures-explained-716fb197b243

[34] Diego, U. (2022, octobre 18). *Couche entièrement connectée vs couche convolutive : expliqué.* Consulté le mars 2023, sur builtin: https://builtin.com/machine-learning/fully-connected-layer

[35] Djerouni, H. (2021). *Développement d'un système de reconnaissance de chiffres manuscrits.* mémoire de fin d'étude, UNIVERSITÉ ECHAHID HAMMA LAKHDAR - D'EL OUED. Récupéré sur http://dspace.univ-eloued.dz/.

[36] Dyouri, A. (s.d.). *Comment créer une application web en utilisant Flask en Python 3*. Consulté le mai 2023, sur developpez.com: https://python.developpez.com/tutoriel/intro-flask-python3/

[37] E, S. N. (2012). *An online writer recognition system based on in-air and on-surface trajectories* . Université polytechnique de Catalogne. Récupéré sur https://upcommons.upc.edu/bitstream/handle/2117/94691/TESN1de1.pdf

[38] fre.myservername.com. (2022, juin). *Introduction aux algorithmes génétiques dans l'apprentissage automatique*. Consulté le mars 2023, sur fre.myservername.com: https://fre.myservername.com/introduction-genetic-algorithms-machine-learning.

[39] Gaudenz , B. (s.d.). *VGG Very Deep Convolutional Networks (VGGNet) – Ce que vous devez savoir*. Consulté le avril 2023, sur viso.ai: https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/

[40] Gayhardt , L., Franks , L., Gilley , S., C.J. , G., Quintanilla , L., & Withee , K. (2022, novembre 29). *Apprentissage profond et apprentissage automatique dans Azure Machine Learning*. Consulté le mars 2023, sur https://learn.microsoft.com/fr-fr/azure/machine-learning/concept-deep-learning-vs-machine-learning

[41] Gennaro, C. (2023, février). *Apprentissage Supervisé Ou Non Supervisé En Bref »,*. Consulté le mars 2023, sur fourweekmba.com: https://fourweekmba.com/fr/apprentissage-supervis%C3%A9-vs-non-supervis%C3%A9/.

[42] Gilliam, T. (2011). *Writer identification in medieval and modern.* University of York, Computer Science. Récupéré sur https://etheses.whiterose.ac.uk/2398/1/t.pdf

[43] *Google Colab*. (s.d.). Consulté le mai 2023, sur Page Officiel de google colab: https://colab.research.google.com/

[44] GUEDIRI, T., & FERHAT , S. (2017). *Reconnaissance des chiffres manuscrits à base des machines à vecteurs de supports* . mémoire de fin d'études, UNIVERSITE ECHAHID HAMMA LAKHDAR - EL OUED.

[45] Gundamotoko. (2019, décembre 05). *Le perceptron multi-couches - Deep learning*. Consulté le mars 2023, sur kongakura.fr: https://kongakura.fr/article/Le-perceptron-multicouches

[46] Gurucharan, M. (2022, JUILLET 28). *Architecture CNN de base : explication des 5 couches du réseau neuronal convolutif*. Consulté le mars 2023, sur upgrad: https://www.upgrad.com/blog/basic-cnn-architecture/

[47] Hammani , H., & Hamza , K. (2022). *Développement d'une application Android pour le diagnostic du Covid-19 basée deep learning.* master, Larbi Ben M'hidi , Oum el Bouaghi. Récupéré sur http://bib.univ-oeb.dz:8080/jspui/bitstream/123456789/14323/1/M%C3%A9moire.pdf

[48] ICHI.PRO. (2020). *Apprentissage automatique - Modèle de Markov caché (HMM)*. Consulté le mars 2023, sur ichi.pro/fr: https://ichi.pro/fr/apprentissage-automatique-modele-de-markov-cache-hmm-54314376723041

[49] JDN. (2022, janvier). *Apprentissage non-supervis : définition et algorithmes populaires*. Consulté le février 2023, sur journaldunet.fr: https://www.journaldunet.fr/web-tech/guide-de-l-intelligence-artificielle/1501309-apprentissage-non-supervise/

[50] Johnson, D. (2023, mars 25). *Tutoriel RNN (Réseau de neurones récurrent) : exemple TensorFlow*. Consulté le mars 2023, sur Guru99: https://www.guru99.com/rnn-tutorial.html#2

[51] Karandish, F. (2016, février). *Comprendre la reconnaissance optique de caractères : le guide moov ai.* Consulté le mars 2023, sur moov.ai: https://moov.ai/fr/blog/reconnaissance-optique-de-caracteres-ocr

[52] Katia, D., & OUALLI , W. (2018). *Détection d'objet basé sur l'estimation de la saillance* . mémoire de fin d'étude, UNIVERSITE MOULOUD MAMMERI DE TIZI-OUZOU . Récupéré sur https://www.ummto.dz/dspace/bitstream/handle/ummto/12375/DjailebKatia.pdf?sequence=1&isAllowed=y

[53] KELDENICH, T. (2021, FÉVRIER 8). *Fonction d'activation, comment ça marche ? – Une explication simple*. Consulté le avril 2023, sur insidemachine learning: https://inside-machinelearning.com/fonction-dactivation-comment-ca-marche-une-explication-simple/#Quest-ce_quune_fonction_dactivation

[54] *Keras Optimizers*. (s.d.). Consulté le mai 2023, sur Educba: https://www.educba.com/keras-optimizers/

[55] Kübra, G., & Çavuşoğlu , Ü. (2022). Handwritten Digit Recognition With Machine Learning Algorithms . *Academic Platform Journal of Engineering and Smart Systems (APJESS) , 10*(1). Récupéré sur https://doi.org/10.21541/apjess.1060753.

[56] L , B. (2019, avril 5). *Réseau de neurones artificiels : qu'est-ce que c'est et à quoi ça sert ?* Consulté le mars 2023, sur lebigdata.fr: https://www.lebigdata.fr/reseau-de-neurones-artificiels-definition

[57] lazzeri, F. (2020). *Machine Learning for Time Series Forecasting with Python.* wiley. Récupéré sur https://books.google.dz/books?id=5YUIEAAAQBAJ

[58] Limbo. (s.d.). *What is batch size in deep learning?* Consulté le mai 2023, sur openaichat: https://openaichat.info/what-is-batch-size-in-deep-learning/#:~:text=When%20the%20learning%20rates%20are%20high%2C%20the%20large,with%20a%20larger%20batch%20size%20requiring%20more%20memory.

[59] MAACHOU, K., & Aicha , S. (2013). *La reconnaissance automatique des chiffres manuscrits isolés par les modèles de Markov cachés.* MASTER en Informatique Option Réseaux et Systèmes Intelligents, Université d'Adrar.

[60] Maheshkar, S. (s.d.). *How to Compare Keras Optimizers in Tensorflow for Deep Learning*. Consulté le mai 2023, sur Weights & Biases: https://wandb.ai/sauravm/Optimizers/reports/How-to-Compare-Keras-Optimizers-in-Tensorflow-for-Deep-Learning--VmlldzoxNjU1OTA4

[61] Maithani, M. (2021, janvier 18). *Guide des optimiseurs Tensorflow Keras*. Consulté le mai 2023, sur Analyticsindiamag: https://analyticsindiamag.com/guide-to-tensorflow-keras-optimizers/

[62] MANKOURI , A., & ZENASNI , E. (2022). *Detecting Covid in chest X-ray using neural networks.* Université de Belhadj Bouchaib , Ain-Temouchent.

[63] *Matplotlib: Visualization with Python*. (s.d.). Consulté le mai 2023, sur Matplotlib: https://matplotlib.org/

[64] MEGAR, s., & Adel , G. (2016). *Contribution à la reconnaissance d'écriture arabe manuscrite en utilisant la classification neuronale.* UNIVERSITE KASDI MERBAH OUARGLA. Récupéré sur https://dspace.univ-ouargla.dz/jspui/bitstream/123456789/11824/1/MEGAR-GEUDDA.pdf.

[65] Mobiskill, L. b. (2021., mars). *Apprentissage supervisé vs apprentissage non supervisé*. Consulté le février 2023, sur mobiskill.fr: https://mobiskill.fr/blog/conseils-emploi-tech/apprentissage-supervise-vs-apprentissage-non-supervise/.

[66] Moustafa, B. (2017). *Reconnaissance Automatique des Chiffres Manuscrits.* mémoire de fin d'étude, Université Abou Bakr Belkaid– Tlemcen. Récupéré sur http://dspace.univ-tlemcen.dz/handle/112/9666.

[67] Murphy, A. (s.d.). *Batch size (machine learning)*. Consulté le mai 2023, sur Radiopaedia: https://radiopaedia.org/articles/batch-size-machine-learning

[68] NADIA, BENAHMED. « Optimisation de réseaux de neurones pour la reconnaissance de chiffres manuscrits isolés : sélection et pondération des primitives par algorithmes génétiques ». ÉCOLE DE TECHNOLOGIE SUPÉRIEURE UNIVERSITÉ DU QUÉBEC, 1 mars 2002.

[69] Nagesh Singh, C. (2022, avril). *Naïve Bayes Algorithm: Everything You Need to Know*. Consulté le mars 2023, sur kdnuggets.com: https://www.kdnuggets.com/2020/06/naive-bayes-algorithm-everything.html

[70] Nasr , A. (2018). *Perceptron Multicouche-Introduction à l'apprentissage automatique.* master Sciences Cognitives, Université Aix Marseille. Consulté le mars 2023, sur https://pageperso.lis-lab.fr/~alexis.nasr/Ens/MASCO_AA/mlp.pdf

[71] Nicolas, F., & . (1996, février). Système de reconnaissance de chiffres manuscrits hors lignes ,Traitement du Signal . pp. 85-90. Récupéré sur https://www.iieta.org/journals/ts/paper/10.3166/TS.13.85-98.

[72] Numpy. (s.d.). *NumPy*. Consulté le mai 2023, sur la page officielle NumPy: https://numpy.org/

[73] Patrick, H. (2021, septembre 6). *Les réseaux de neurones récurrents pour les séries temporelles*. Consulté le mars 2023, sur metalblog: https://metalblog.ctif.com/2021/09/06/les-reseaux-de-neurones-recurrents-pour-les-series-temporelles/

[74] *Principes de base du Deep Learning*. (s.d.). Consulté le mai 2023, sur DEEPLIZARD: https://deeplizard.com/learn/video/U4WB9p6ODjM

[75] ProjectPro. (2023, avril 26). *Introduction à l'architecture des réseaux de neurones convolutifs*. Consulté le mars 2023, sur projectpro.io:

https://www.projectpro.io/article/introduction-to-convolutional-neural-networks-algorithm-architecture/560#mcetoc_1fs0joo1hc

[76] projectpro. (2023, avril 17). *Top 10 des algorithmes d'apprentissage en profondeur dans l'apprentissage automatique*. Consulté le mars 2023, sur https://www.projectpro.io/article/deep-learning-algorithms/443#mcetoc_1glh52nvlejd

[77] Reconnaissance de formes avec Modèle de Markov caché (HMM). (2021). p. 47. Récupéré sur https://elearn.univ-tlemcen.dz/pluginfile.php/108511/mod_resource/content/1/MID-RdF-05.pdf

[78] *Reconnaissance de l'écriture manuscrite*. (2020). Consulté le janvier 2023, sur Wikipedia: https://fr.wikipedia.org/wiki/Reconnaissance_de_l%27%C3%A9criture_manuscrite.

[79] Sabine, C., & Anquetil , E. (2020). *Apprentissage automatique d'une distance d'édition dédiée à la reconnaissance de l'écriture manuscrite.* université de beaulieu avenue de général leclerc. Récupéré sur https://archivesic.ccsd.cnrs.fr/sic_00001198

[80] Sadiq H , A., Basheera M , M., & Marwah, A. (2021, mars). A robust handwritten numeral recognition using hybrid orthogonal polynomials and moments. Récupéré sur https://doi.org/10.3390/s21061999

[81] Sanpreet, S. (2021). *Quelle est la différence entre LSTM, RNN et séquence à séquence ?* Consulté le mars 2023, sur quora: https://www.quora.com/What-is-the-difference-between-LSTM-RNN-and-sequence-to-sequence

[82] Savita, A., & Choudhary , A. (2020). hybrid CNN-SVM Classifier for Handwritten Digit Recognition. *167*, 2554-2560. Récupéré sur https://doi.org/10.1016/j.procs.2020.03.309

[83] Shamim , S., Badrul , A., Sarker , A., Rana , M., & Al , J. (2018). *Handwritten Digit Recognition using Machine Learning Algorithms.* Consulté le mars 2023, sur arxiv.org: https://arxiv.org/abs/2106.12614

[84] StackLima. (2022, juillet). *Avantages et inconvénients des différents modèles de classification*. Consulté le mars 2023, sur stacklima.com: https://stacklima.com/avantages-et-inconvenients-des-differents-modeles-de-classification/

[85] Suvaditya, M. (2022, août 18). *Le ResNet-50 annoté*. Consulté le avril 2023, sur towards data science: https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758

[86] Team, K. (s.d.). *Keras*. Consulté le mai 2023, sur https://github.com/keras-team/keras

[87] TechTarget. (2018). *Réseau de neurones artificiels (RNA)*. Consulté le mars 2023, sur lemagit: https://www.lemagit.fr/definition/Reseau-de-neurones-artificiels-RNA

[88] TensorFlow. (s.d.). *Introduction à TensorFlow*. Consulté le mai 2023, sur TensorFlow: https://www.tensorflow.org/learn?hl=fr

[89] tensorflow. (s.d.). *TensorFlow Core*. Consulté le mai 2023, sur TensorFlow: https://www.tensorflow.org/guide/keras?hl=fr

[90] toufik, B. (2008). *le choix de paramétres pour la reconnaissance des chiffres manuscrites* . UNIVERSITE BADJI MOKHTAR ANNABA. Récupéré sur https://biblio.univ-annaba.dz/wp-content/uploads/2014/06/Bendjedou101.pdf

[91] Touzani, Y. (2021, Août 25). *Deep Learning : les réseaux de neurones récurrents*. Consulté le mars 2023, sur datavalue: https://datavalue-consulting.com/deep-learning-reseaux-neurones-recurrents-rnn/

[92] Tyler, W. (s.d.). *DeepLearning-MNIST-Écriture manuscrite*. Consulté le mai 2023, sur Github: https://github.com/WolfeTyler/DeepLearning-MNIST-Handwritting/blob/master/README.md

[93] Vaucher, Q. (s.d.). *Python Imaging Library (PIL)*. Consulté le mai 2023, sur Bibliothèques Python Navigation: https://he-arc.github.io/livre-python/pillow/index.html

[94] *Visual studio code*. (2022, octobre 16). Consulté le mai 2023, sur Edu tech wiki: https://edutechwiki.unige.ch/fr/Visual_studio_code

[95] Vonintsoa. (2022, mars). *Apprentissage supervisé et non supervisé : comment les différencier ?* Consulté le février 2023, sur intelligence-artificielle.com: https://intelligence-artificielle.com/apprentissage-supervise-et-non-supervise/.

[96] *What is batch size in neural network?* (s.d.). Consulté le mai 2023, sur stack exchange: https://stats.stackexchange.com/questions/153531/what-is-batch-size-in-neural-network

[97] Will, K. (2018, novembre 5). *Réseaux de neurones récurrents par exemple en Python*. Consulté le mars 2023, sur towardsdatascience: https://towardsdatascience.com/recurrent-neural-networks-by-example-in-python-ffd204f99470

[98] Wolfewicz, A. (2023, Février 15). *Deep Learning vs Machine Learning – Quelle est la différence ?* Consulté le mars 2023, sur levity.ai: https://levity.ai/blog/difference-machine-learning-deep-learning

[99] Zerougui, A., & nabil , S. (2017). *Traitement d'images monochromes Détection de contours, Filtrage(spacial et fréquentiel) et segmentation par réseaux de neurones* . mémoire de fin d'étude, Université Larbi Ben M'hidi, Oum El Bouaghi. Récupéré sur http://bib.univ-oeb.dz:8080/jspui/bits

[100] Zubair , H. (2022, ouat 22). *Best CNN Architecture For Image Processing*. Consulté le avril 2023, sur folio3: https://www.folio3.ai/blog/best-cnn-architecture-for-image-processing/#3_Fully_Connected_Layer