الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université Akli Mohand Oulhadj - Bouira -
Tasdawit Akli Muḥend Ulḥağ - Tubirett -

وزارة التعليم العالي والبحث العلمي
جامعة أكلي محند أولحاج
-البويرة-

جامعة البويرة

Faculté des Sciences et des Sciences Appliquées

كلية العلوم والعلوم التطبيقية

Référence : ...../MM/2021

المرجع:...../م م/ 2021

Mémoire de Master

Présenté au

Département : Génie Électrique
Domaine : Sciences et Technologies
Filière : Electronique
Spécialité : Electronique des systèmes embarqués

Réalisé par :

ALOUACHE Adel

Thème

Reconnaissance des chiffres en utilisant Deep Learning

Soutenu le : **21/09/2023**

Devant le Jury composé de :

Mr :	MADI Saida	M.C.B	Univ. Bouira	Président
	BENSAFIA Yacine	M.C.A	Univ. Bouira	Rapporteur
	LADJOUZI Samir	M.C.B	Univ. Bouira	Examinateur

Année Universitaire : 2022-2023

Remerciements

Ce travail a été effectué au sein du Département des Sciences et sciences appliquées de l'Université de Bouira.

Je tiens à remercier, en premier lieu, Mr. BENSAFIA Yacine, Directeur de ce mémoire pour sa guidance précieuse, ses conseils éclairés et son soutien constant tout au long de ce projet. Votre expertise a été une source d'inspiration et a grandement enrichi ce travail.

Je souhaite aussi adresser mes remerciements à ma famille et à mes amis pour leur soutien indéfectible et leur compréhension tout au long de ce parcours académique exigeant.

Je remercie également tous les membres du jury pour l'intérêt qu'ils ont porté à mon travail :

Mme. MADI Saida -la présidente du Jury-

Mr. LADJOUZI Samir -l'examinateur-

Enfin, je tiens à exprimer ma profonde gratitude envers toutes les personnes qui ont contribué à la réalisation de cette étude.

Table des Matières

Liste des Figures

Liste des Tableaux

Listes des Acronymes et Symboles

- **Acronymes**

CNN	Convolutional neural network / Réseau de neurones convolutif
DL	Deep Learning
IA	Intelligence Artificielle
MNST	Modified National Institute of Standards and Technology
RNA	Réseau de neurones artificielle
RNP	Réseau de neurones profond
SVHN	Street View House Numbers
SVM	Support Vector Machine

- **Symboles**

$\boldsymbol{\alpha}$	Le taux d'apprentissage
$\boldsymbol{b}$	Le biais
$\boldsymbol{\omega}$	Matrice des poids associés à chaque entrée
$\boldsymbol{y_{pred}}$	Le vecteur de prédictions du modèle
$\boldsymbol{y_{true}}$	Le vecteur des vraies étiquettes
$\frac{\partial y_{pred_i}}{\partial b}$	Le gradient de la prédiction par rapport au biais $\boldsymbol{b}$
$\frac{\partial y_{pred_i}}{\partial \omega}$	Le gradient de la prédiction par rapport au poids $\boldsymbol{\omega}$

Introduction Générale [9]

L'avènement du Deep Learning a révolutionné de nombreux domaines de l'intelligence artificielle, notamment celui de la reconnaissance des chiffres. La reconnaissance des chiffres est une tâche fondamentale dans le traitement d'images et le domaine de la vision par ordinateur. Elle consiste à développer des algorithmes et des modèles capables d'identifier et d'interpréter automatiquement les chiffres présents dans des images numériques, que ce soit des numéros de téléphone, des codes de produits, des plaques d'immatriculation ou des chiffres manuscrits.

Le Deep Learning, une sous-branche du machine learning, a émergé comme une méthode particulièrement puissante pour aborder ces problèmes complexes de reconnaissance des chiffres. À la base du Deep Learning se trouvent les réseaux de neurones artificiels, qui sont inspirés du fonctionnement du cerveau humain.

Dans le contexte de la reconnaissance des chiffres, les réseaux de neurones profonds, ont démontré une remarquable capacité à apprendre à partir d'exemples. Lorsqu'ils sont alimentés avec de grandes quantités de données contenant des images de chiffres annotées (c'est-à-dire des images accompagnées des chiffres correspondants), ces réseaux peuvent apprendre à identifier des motifs et des caractéristiques discriminantes dans les images.

L'une des réalisations emblématiques du Deep Learning dans le domaine de la reconnaissance des chiffres est la création de modèles de réseaux de neurones convolutifs (CNN). Les CNN ont été spécialement conçus pour traiter des données structurées en grille, telles que les images. Grâce à des opérations de convolution et de pooling, ces modèles sont capables d'extraire automatiquement des caractéristiques visuelles importantes, telles que les bords, les textures et les formes, à partir des images.

L'apprentissage profond dans les CNN se fait par rétropropagation, où le modèle ajuste itérativement ses poids et ses biais pour minimiser l'erreur entre les prédictions et les étiquettes réelles des chiffres. Ce processus d'apprentissage permet au modèle de généraliser à de nouvelles images et de reconnaître des chiffres qu'il n'a jamais rencontrés auparavant.

Grâce à ces avancées du Deep Learning, les performances de la reconnaissance des chiffres ont considérablement progressé. Les modèles de réseaux neuronaux profonds, en particulier les CNN, ont atteint des niveaux de précision impressionnants, rivalisant souvent avec, voire dépassant, les capacités humaines dans de nombreux scénarios.

En conclusion, la reconnaissance des chiffres en utilisant le Deep Learning a ouvert de nouvelles perspectives passionnantes dans le domaine de la vision par ordinateur. Les modèles de réseaux neuronaux profonds, notamment les CNN, ont radicalement amélioré la capacité des machines à

interpréter et à comprendre les chiffres présents dans les images. Cette technologie a des applications potentielles dans divers domaines, tels que la numérisation de documents, la sécurité, la classification d'objets et bien plus encore.

Problématique :

Dans le contexte de l'apprentissage profond, comment peut-on optimiser la sélection des données d'entraînement en apprentissage profond, en tenant compte des contraintes liées au volume initial du jeu de données, afin d'améliorer la performance des modèles tout en évitant les écueils du sur-apprentissage et du sous-apprentissage.

Objectif :

Dans le cadre de ce projet de fin d'études, nous envisageons de résoudre cette problématique tout en concevant un système de reconnaissance de chiffres manuscrits qui s'appuie sur des techniques d'apprentissage profond.

Pour atteindre cet objectif, la structure de ce mémoire sera la suivante :

Chapitre 1 : Fondements du Deep Learning et de la Reconnaissance des Chiffres :

Dans ce premier chapitre, nous allons explorer l'historique de la reconnaissance de chiffres écrits à la main, comparer les approches traditionnelles aux modèles de deep learning, et examiner les diverses applications de cette technologie. En résumé, nous allons présenter un aperçu historique, une comparaison des méthodes et une exploration des utilisations de la reconnaissance de chiffres manuscrits

Chapitre 2 : Présentation de la méthodologie :

Le deuxième chapitre approfondit notre compréhension du sujet en explorant les généralités et l'état de l'art dans la reconnaissance de chiffres manuscrits avec le deep learning. Nous plongeons dans les concepts théoriques, en mettant en lumière les modèles de réseaux neuronaux convolutifs (CNN) et les méthodologies d'entraînement. Nous discutons également des techniques d'optimisation, des hyperparamètres, et des aspects mathématiques sous-jacents à cette discipline. Ce chapitre vise à établir une base solide pour la phase pratique qui suivra.

Chapitre 3 : Réalisation et Résultats :

Au cœur de ce chapitre se trouve notre modèle, fruit de notre compréhension des concepts théoriques exposés précédemment. Nous dévoilerons comment nous avons préparé les données, construit le modèle, et défini les paramètres pour optimiser les performances. Puis, nous analyserons en détail les résultats obtenus lors des tests, démontrant la capacité de notre modèle à la reconnaissance de chiffres manuscrits.

Chapitre 1 :

Fondements du Deep Learning et de la Reconnaissance des Chiffres

1. Introduction :

La reconnaissance des chiffres manuscrits a longtemps été une quête dans le domaine de l'informatique et de l'intelligence artificielle. Cette tâche, en apparence simple pour les êtres humains, est en réalité l'une des énigmes les plus complexes à résoudre pour les machines. L'avènement du Deep Learning a toutefois révolutionné notre capacité à aborder ce défi d'une manière inédite.

Ce premier chapitre marque le point de départ de notre exploration du domaine de la reconnaissance des chiffres manuscrits en utilisant le Deep Learning. Notre voyage nous conduira à travers les profondeurs des réseaux neuronaux, des données massives et des méthodes innovantes. Avant de plonger dans les détails de notre méthodologie et de nos résultats, il est essentiel de comprendre les généralités et l'état de l'art de ce domaine en évolution constante.

Nous commencerons par une introduction aux concepts fondamentaux du Deep Learning, en explorant comment les réseaux neuronaux artificiels s'inspirent du fonctionnement du cerveau humain pour apprendre à partir de données. Nous découvrirons comment ces réseaux sont devenus le socle sur lequel repose notre quête de la reconnaissance des chiffres manuscrits.

Nous parcourrons également l'histoire de cette discipline, depuis les premières tentatives de reconnaissance de caractères jusqu'à la révolution apportée par le Deep Learning. Nous comprendrons comment les approches traditionnelles ont cédé la place à des méthodes basées sur des données massives et des réseaux neuronaux profonds.

Une partie importante de ce chapitre sera consacrée aux réseaux de neurones convolutifs (CNN), des architectures spécifiquement conçues pour traiter des données spatiales telles que les images. Nous découvrirons comment les CNN extraient automatiquement des caractéristiques visuelles, jouant un rôle central dans la reconnaissance des chiffres manuscrits.

Enfin, nous explorerons les applications concrètes de la reconnaissance des chiffres manuscrits, des numérisations de documents à la lecture automatique de plaques d'immatriculation, en passant par la classification de textes manuscrits. Ces applications témoignent du potentiel transformateur du Deep Learning dans notre vie quotidienne.

2. Généralités : [4]

Les réseaux de neurones et l'apprentissage profond offrent actuellement les meilleures solutions à de nombreux problèmes de reconnaissance d'images, de reconnaissance vocale et de traitement du langage naturel.

Un réseau neuronal artificiel est un modèle informatique inspiré par le fonctionnement des neurones biologiques (voir la figure 1.1). Il est composé de couches d'unités interconnectées, qui transmettent et traitent l'information en ajustant les poids des connexions entre elles. Les signaux sont pondérés et activés, ce qui permet au réseau de "comprendre" des modèles dans les données et d'apprendre à partir d'exemples.

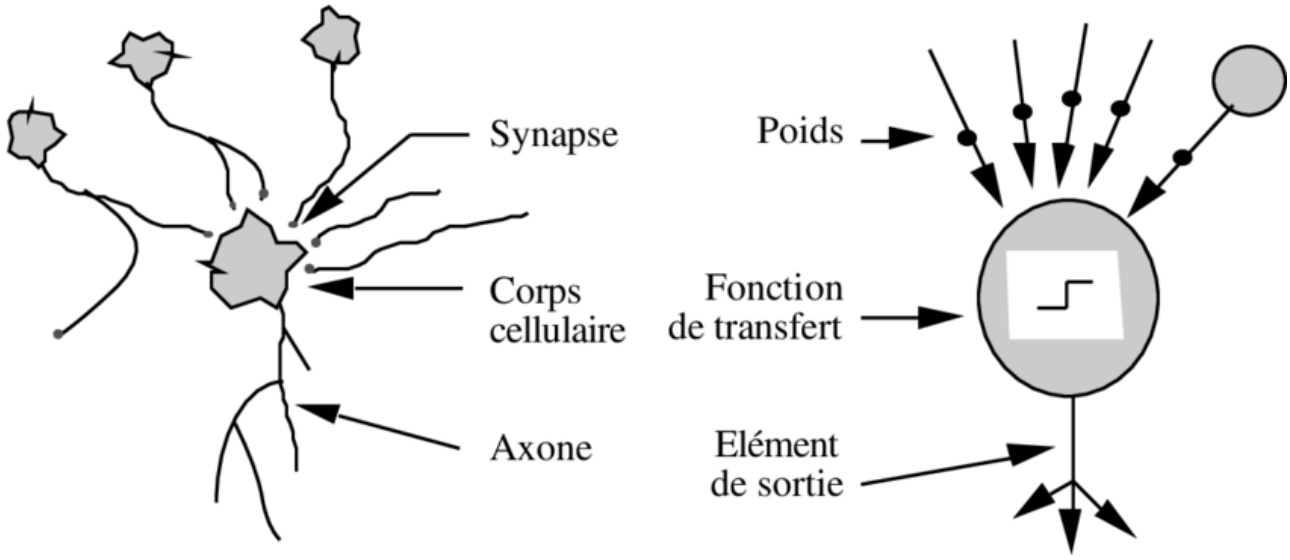


Fig. 1.1. Neurone biologique et Neurone artificiel

Le fonctionnement des neurones biologiques repose sur la transmission d'informations électrochimiques à travers des connexions synaptiques (voir la figure 1.2). De manière simplifiée, un neurone reçoit des signaux d'entrée de plusieurs autres neurones via ses dendrites. Si la somme pondérée de ces signaux dépasse un certain seuil, le neurone émet un signal électrique le long de son axone, qui peut à son tour influencer d'autres neurones.

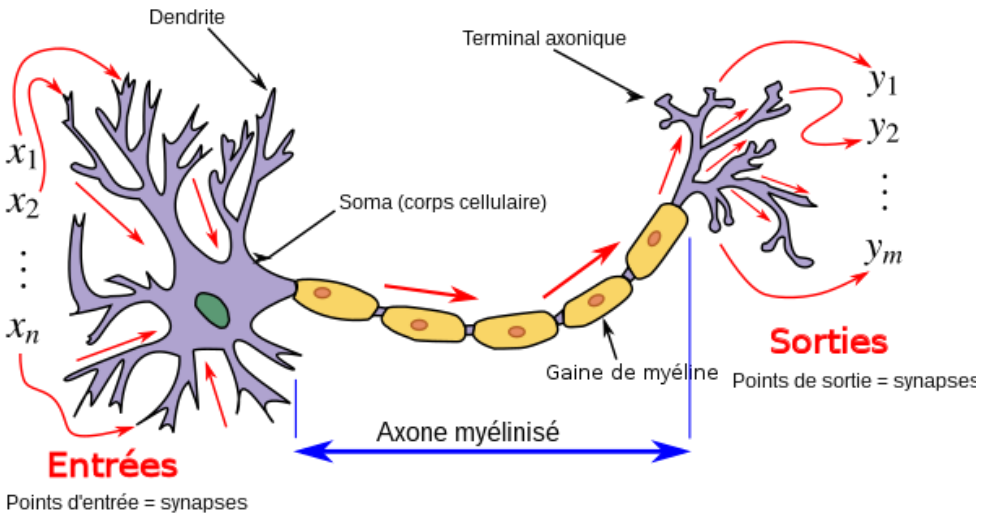


Fig. 1.2. Neurone Biologique

Les RNA imitent ce processus en utilisant des "neurones" artificiels, également appelés "unités". Chaque unité reçoit des entrées pondérées (voir la figure 1.3) et, si la somme de ces entrées dépasse un seuil (ou est soumise à une fonction d'activation), l'unité émet une sortie. Les unités sont organisées en couches, telles que la couche d'entrée, une ou plusieurs couches cachées et la couche de sortie.

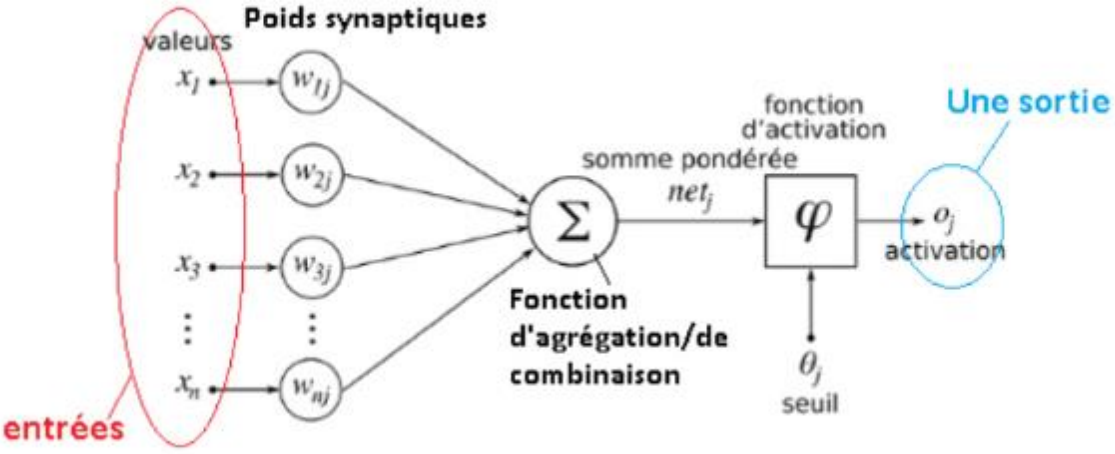


Fig. 1.3. Neurone artificiel

La transmission d'informations entre les unités est gérée par des poids synaptiques, qui déterminent l'importance relative des signaux provenant d'autres unités. L'apprentissage dans les RNA implique l'ajustement de ces poids en fonction des exemples d'entraînement, de manière à améliorer les performances du réseau sur une tâche spécifique. Cela se fait souvent à l'aide d'algorithmes d'optimisation tels que la rétropropagation du gradient.

En résumé, les RNA sont des modèles mathématiques qui s'inspirent du fonctionnement des neurones biologiques pour traiter l'information. Bien qu'ils simplifient grandement la complexité du cerveau, ils ont montré leur efficacité dans une variété de domaines grâce à leur capacité à apprendre des modèles à partir de données.

3. Apprentissage supervisé : [4]

Comme on a dit précédemment, les RNA apprennent à partir de données étiquetées .

Le deep learning est basé sur un type d'apprentissage appelé "apprentissage supervisé" dont le processus se déroule généralement de la manière suivante :

- ✓ **Collecte de données :**

Nous rassemblons un ensemble de données qui contient des exemples d'entrées et les étiquettes correspondantes. Par exemple, dans la classification d'images, les données pourraient être des images et les étiquettes correspondraient aux catégories auxquelles appartiennent ces images.

- ✓ **Prétraitement des données :**

Les données sont généralement prétraitées pour être normalisées, redimensionnées ou transformées de manière à faciliter le processus d'apprentissage.

- ✓ **Construction du modèle :**

Nous concevons et configurons un modèle de réseau de neurones profond avec des couches de neurones interconnectées. Chaque couche effectue des transformations sur les données d'entrée.

- ✓ **Initialisation des poids :**

Les poids du modèle sont initialisés de manière aléatoire ou à l'aide de méthodes spécifiques.

- ✓ **Propagation avant (Forward Propagation) :**

Les données d'entrée sont introduites dans le modèle et passent à travers les couches du réseau. Chaque couche effectue des calculs basés sur les poids et les fonctions d'activation pour générer des prédictions.

✓ **Calcul de la perte (Loss Calculation) :**

Les prédictions du modèle sont comparées aux étiquettes réelles dans les données. La perte (ou coût) mesure à quel point les prédictions diffèrent des étiquettes réelles.

✓ **Rétropropagation (Backpropagation) :**

Le processus de rétropropagation calcule les gradients de la perte par rapport aux poids du modèle. Ces gradients indiquent comment les poids doivent être ajustés pour minimiser la perte.

✓ **Optimisation des poids :**

Les algorithmes d'optimisation sont utilisés pour ajuster les poids du modèle en utilisant les gradients calculés pendant la rétropropagation. Cela permet au modèle d'apprendre à faire des prédictions plus précises au fil du temps.

✓ **Répétition du processus :**

Le processus - de Propagation avant jusqu'à Rétropropagation - est répété sur plusieurs itérations (époques) en ajustant progressivement les poids du modèle pour réduire la perte et améliorer les prédictions.

✓ **Évaluation du modèle :**

Une fois que le modèle a été formé, il est évalué sur des données qu'il n'a jamais vues auparavant pour estimer sa performance en généralisation.

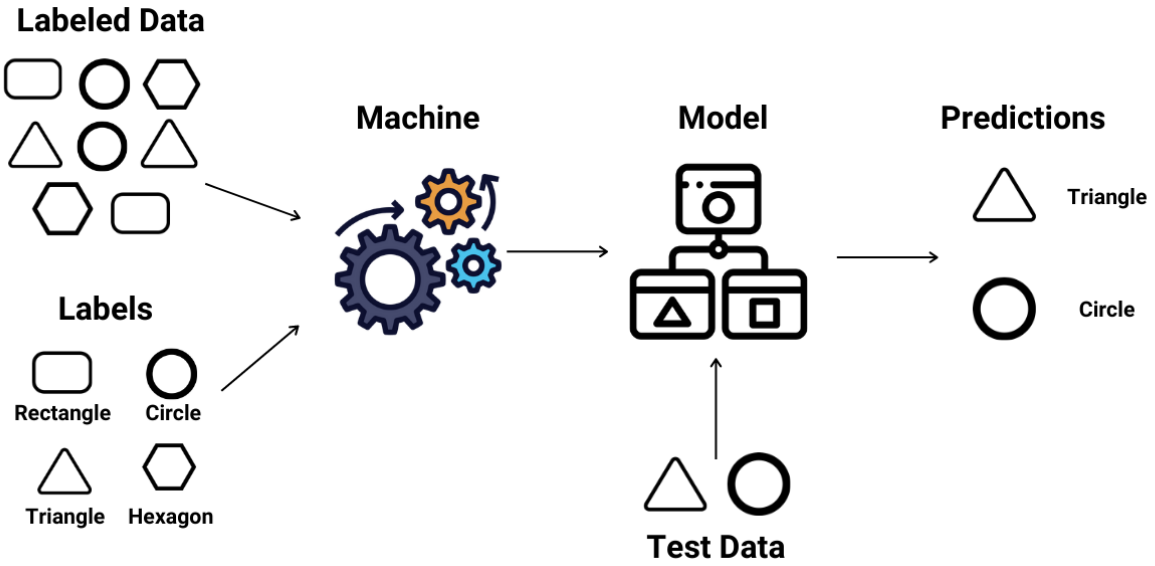


Fig. 1.4. Apprentissage Supervisé

En résumé, le DL est une approche d'apprentissage automatique qui utilise des RNP pour résoudre des tâches complexes en exploitant des données supervisées, où les exemples d'entrée sont associés à des sorties correspondantes.

4. Historique : [16]

L'histoire de la reconnaissance des chiffres remonte à plusieurs décennies et a été une partie importante du développement de la vision par ordinateur et de l'intelligence artificielle. Voici un aperçu de l'évolution de la reconnaissance des chiffres au fil du temps :

- **Années 1960 - 1970 : Les débuts**

À ses débuts, la reconnaissance des chiffres était souvent réalisée à l'aide de méthodes basées sur des règles simples. Les chercheurs développaient des algorithmes manuels pour identifier des motifs spécifiques dans les images de chiffres. Cependant, ces approches étaient limitées en termes de complexité et de capacité à traiter des variations dans les images, comme les différences d'écriture.

- **Années 1980 - 1990 : L'essor des méthodes basées sur les caractéristiques**

Pendant cette période, les chercheurs ont commencé à explorer des méthodes plus sophistiquées basées sur l'extraction de caractéristiques. Cela impliquait l'identification et la sélection de caractéristiques visuelles pertinentes dans les images de chiffres, telles que les angles, les intersections et les boucles.

Des modèles mathématiques, tels que les modèles de reconnaissance de formes, ont été appliqués pour définir des règles et des caractéristiques discriminantes pour la classification des chiffres. Les réseaux de neurones artificiels peu profonds ont également été utilisés à cette époque pour la reconnaissance des chiffres, bien que leur complexité soit restée limitée en raison des contraintes informatiques.

- **Années 2000 - 2010 : L'avènement de l'apprentissage automatique**

L'introduction de l'apprentissage automatique a marqué une transition significative dans la reconnaissance des chiffres. Les méthodes basées sur l'apprentissage automatique, telles que les machines à vecteurs de support (SVM) et les arbres de décision, ont permis aux ordinateurs d'apprendre à partir des données.

Cependant, l'avènement du deep learning a été le véritable tournant. Les réseaux de neurones artificiels profonds, et en particulier les réseaux de neurones convolutifs (CNN), ont considérablement augmenté la compétence des machines à gérer et identifier des images. Les CNN ont la capacité d'apprendre automatiquement des caractéristiques à différents niveaux d'abstraction, ce qui permet de capturer des motifs complexes et hiérarchiques dans les données.

❖ **Années 2010 - Aujourd'hui : La domination du deep learning**

Avec l'essor du DL, les performances de la reconnaissance des chiffres ont considérablement augmenté. Des architectures plus avancées, telles que les réseaux de neurones convolutifs profonds, ont été développées. Ces architectures ont réussi à surpasser les méthodes traditionnelles et à établir de nouveaux records de précision.

Des bases de données d'images de chiffres, comme la base de données MNIST, ont joué un rôle clé dans le développement et l'évaluation de ces modèles. En outre, la montée en puissance des cartes graphiques (GPU) a accéléré la formation de modèles de deep learning, ce qui a contribué à leur adoption généralisée.

En résumé, voici une figure qui explique la différence entre la méthode traditionnelle et la méthode du DL :

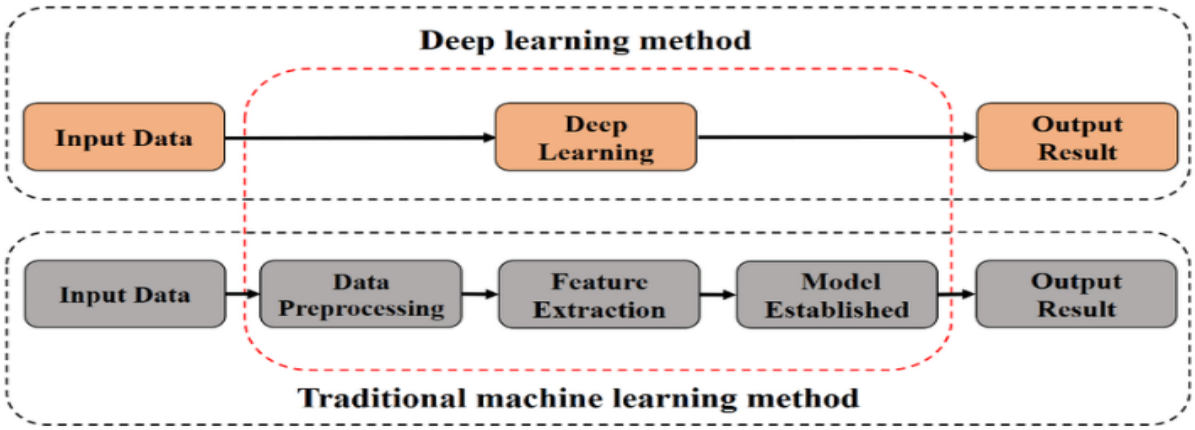


Fig. 1.5. Méthode traditionnelle vs méthode de DL

Et voici une comparaison des avantages et des inconvénients des deux méthodes sur quelques points :

T1.1 : Comparaison entre la méthode traditionnelle et la méthode de DL

	Méthode traditionnelle	Méthode du DL
Interprétabilité	Facile	Difficile
Simplicité	Simple	Complexe
Quantité de données	Petite	Enorme
Performance	Limitée	Elevée

Comme on note que les modèles de DL peuvent traiter des données brutes, telles que des images en pixels, ce qui permet d'exploiter des informations complexes et subtiles.

5. Réseaux de Neurones Convolutifs (CNN) :

Vous avez certainement remarqué que chaque fois que le terme "reconnaissance de chiffres" est mentionné, il est presque inévitable que le terme "CNN" (Réseaux de Neurones Convolutifs) suive immédiatement.

- Je pense qu'il est maintenant opportun d'explorer en profondeur les arcanes des CNN.

Les Réseaux de Neurones Convolutifs (CNN) ont révolutionné la reconnaissance d'images, y compris la reconnaissance des chiffres manuscrits, en exploitant leur capacité à apprendre des caractéristiques hiérarchiques et à capturer des motifs complexes dans les données visuelles. Voici comment les CNN sont adaptés à la tâche de reconnaissance des chiffres et comment ils sont devenus la méthode privilégiée pour cette tâche : [4]

5.1. Apprentissage hiérarchique des caractéristiques :

Les images, y compris les chiffres manuscrits, contiennent des motifs complexes qui varient en échelle et en orientation. Les réseaux de neurones classiques ont du mal à capturer ces variations avec une seule couche de neurones. Les CNN, en revanche, sont spécialement conçus pour gérer de telles données.

Les couches de convolution dans les CNN filtrent l'image d'entrée avec des noyaux (filtres) qui sont capables de détecter des caractéristiques simples telles que des bords et des coins (voir la figure 1.6). En empilant plusieurs couches de convolution, les réseaux de neurones peuvent apprendre des caractéristiques de plus en plus complexes et hiérarchiques, comme des formes, des textures et des motifs spécifiques.

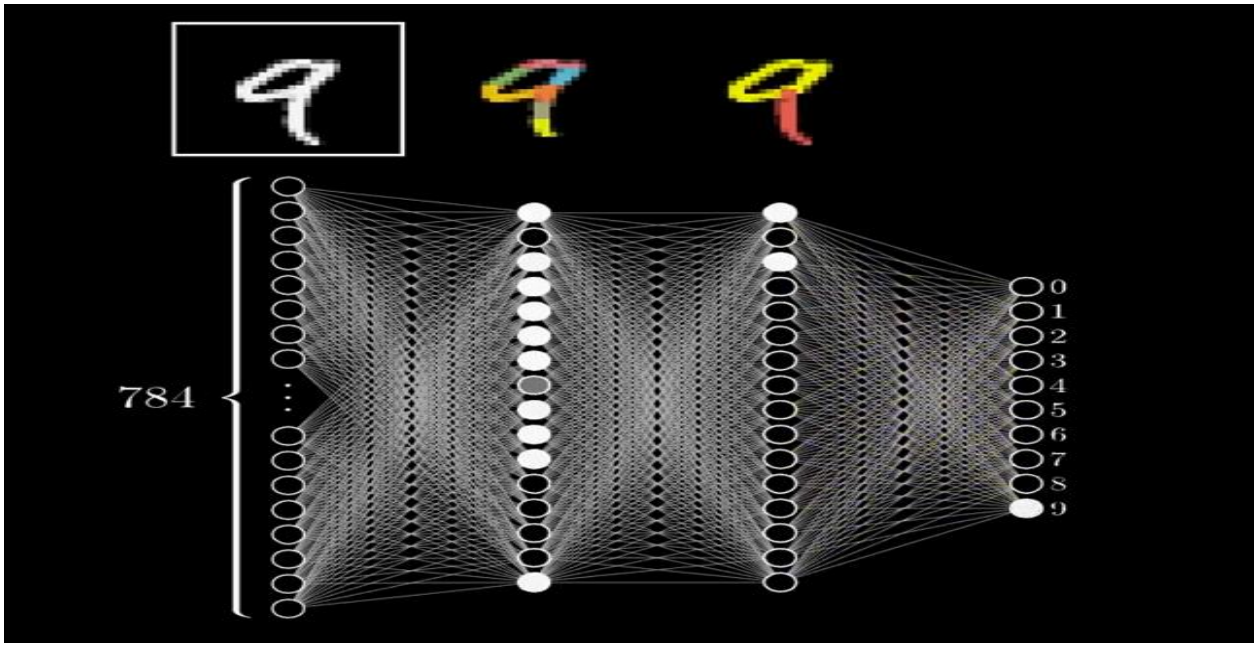


Fig. 1.6. Fonctionnement de CNN théoriquement

Par exemple, dans le cas de la reconnaissance des chiffres, les premières couches de convolution pourraient apprendre à détecter des bords et des lignes dans les chiffres. Ensuite, les couches suivantes pourraient apprendre à combiner ces bords et ces lignes pour former des formes plus complexes, comme des boucles, des angles, etc. Les couches supérieures du réseau pourraient ensuite apprendre à associer ces formes pour former des représentations de chiffres entiers.

5.2. Translation invariance :

Les CNN sont conçus pour être sensibles aux invariances de translation, ce qui signifie qu'ils peuvent reconnaître des motifs indépendamment de leur position dans l'image. Cela rend les CNN particulièrement adaptés à la reconnaissance des chiffres, car un chiffre écrit à différents endroits de l'image devrait toujours être reconnu comme le même chiffre.

5.3. Réduction du nombre de paramètres :

Les CNN incorporent des opérations de sous-échantillonnage (pooling) qui réduisent la dimension spatiale des données tout en préservant les caractéristiques essentielles. Cela réduit le nombre de paramètres du modèle, ce qui le rend plus gérable et moins susceptible de surapprentissage, tout en conservant la capacité de détecter des motifs importants.

5.4. Évolution vers la méthode de prédilection :

Le succès du CNN LeNet-5 de Yann LeCun dans la compétition MNIST en 1998 a jeté les bases de l'utilisation des CNN pour la reconnaissance des chiffres. La base de données MNIST, qui est constituée de chiffres manuscrits, est devenue un banc d'essai important pour évaluer les performances des modèles de reconnaissance d'images.

Par la suite, avec l'avènement de matériel informatique plus puissant, d'architectures de réseaux plus profondes et d'ensembles de données plus vastes (comme ImageNet), les CNN sont devenus de plus en plus sophistiqués et ont atteint des performances de pointe dans la reconnaissance d'images en général, y compris la reconnaissance des chiffres.

6. Applications et Domaines d'Utilisation :

Le deep learning, a apporté des avancées majeures dans de nombreuses applications, en particulier dans le domaine de la reconnaissance des chiffres manuscrits. Cette technologie a révolutionné la manière dont nous traitons et comprenons les chiffres écrits à la main. Avant le développement du deep learning, la reconnaissance des chiffres manuscrits était une tâche complexe nécessitant des règles manuelles et des caractéristiques artificielles pour extraire des informations à partir de ces données variées et parfois imprévisibles.

Dans les sections suivantes, nous explorerons l'impact significatif de la reconnaissance des chiffres manuscrits basée sur le deep learning dans de multiples domaines d'utilisation. Découvrons comment elle a transformé la manière dont nous interagissons avec des chiffres manuscrits dans notre vie quotidienne et dans le monde professionnel :

- **Reconnaissance de caractères pour la numérisation de documents :**

La technologie de reconnaissance des chiffres manuscrits est largement utilisée dans les scanners de documents pour convertir des textes écrits à la main en texte numérique. Cela facilite la recherche, la classification et l'archivage de documents papier.

- **Systèmes de traitement de formulaires :**

Les entreprises utilisent la reconnaissance de chiffres pour automatiser la saisie de données à partir de formulaires manuscrits, tels que les formulaires fiscaux, les questionnaires médicaux et les sondages.

- **Reconnaissance de chèques bancaires :**

Les banques utilisent des systèmes de reconnaissance de chiffres manuscrits pour traiter les chèques et extraire automatiquement les informations importantes, telles que le montant, le numéro de compte et la signature.

- **Reconnaissance d'écriture pour l'apprentissage automatique :**

Dans le domaine de l'apprentissage automatique, la reconnaissance des chiffres manuscrits est utilisée pour la création de bases de données étiquetées, ce qui est essentiel pour l'entraînement de modèles d'IA, par exemple, dans la classification de texte manuscrit.

- **Reconnaissance de codes postaux et d'adresses :**

Les services postaux utilisent la reconnaissance de chiffres pour trier et acheminer automatiquement le courrier en fonction des codes postaux et des adresses écrites à la main.

- **Saisie de données sur les tablettes et les appareils mobiles :**

Les applications de saisie de données sur les tablettes et les smartphones utilisent souvent la reconnaissance de chiffres pour interpréter l'écriture manuscrite des utilisateurs et convertir des entrées manuscrites en texte numérique.

- **Reconnaissance de numéros de téléphone et de fax :**

Dans les systèmes de télécopie et de numérotation automatique, la reconnaissance des chiffres permet d'interpréter les numéros de téléphone et de fax écrits à la main, ce qui simplifie la composition et la communication.

- **Reconnaissance de l'écriture pour l'accès aux services bancaires en ligne :**

Les applications de reconnaissance de chiffres manuscrits sont également utilisées dans les services bancaires en ligne pour permettre aux clients de déposer des chèques en prenant simplement une photo de ces chèques.

- **Édition de texte manuscrit :**

Les outils de reconnaissance de chiffres manuscrits peuvent être utilisés pour convertir l'écriture manuscrite en texte éditable dans des applications de traitement de texte.

- **Automatisation de la réponse aux courriers et formulaires :**

Dans les centres de services clientèle, les entreprises utilisent la reconnaissance de chiffres pour automatiser la réponse aux courriers et formulaires des clients.

- **Reconnaissance de chiffres pour l'authentification :**

La reconnaissance de chiffres manuscrits peut également être utilisée dans des systèmes de sécurité pour l'authentification des signatures manuscrites.

7. Conclusion :

La reconnaissance des chiffres manuscrits a des applications diverses et étendues dans de nombreux domaines, notamment la numérisation de documents, la saisie de données, les services postaux, les applications mobiles, les services bancaires en ligne, l'authentification, etc. Elle permet d'automatiser des tâches qui seraient autrement fastidieuses et sujettes aux erreurs humaines, ce qui entraîne des gains d'efficacité et de précision significatifs.

En somme, ce premier chapitre a posé les fondations essentielles de notre étude, nous offrant une compréhension approfondie des concepts, des obstacles à surmonter, ainsi que des perspectives prometteuses liées à la reconnaissance des chiffres manuscrits à travers le prisme du Deep Learning. Ces connaissances préliminaires nous préparent à une plongée plus approfondie dans notre méthodologie et nos découvertes à mesure que nous progresserons dans cette étude captivante.

Chapitre 2 :

Présentation de la méthodologie

1. Introduction :

Dans ce deuxième chapitre, nous allons entrer dans le cœur de notre méthodologie. Nous explorerons en détail les méthodes, les approches mathématiques et les techniques que nous avons employées pour résoudre le problème complexe de la reconnaissance des chiffres manuscrits. Ce chapitre servira de pont entre la compréhension théorique des concepts abordés précédemment et l'application pratique de ces concepts pour obtenir des résultats significatifs.

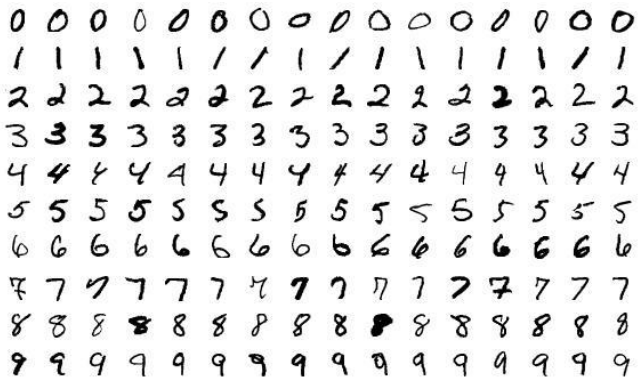
Nous commencerons par aborder la question cruciale des données, en examinant les ensembles de données que nous avons utilisés et en décrivant comment nous les avons préparés pour l'entraînement de nos modèles. Ensuite, nous plongerons dans l'architecture de nos modèles, en expliquant en quoi ils sont spécifiquement conçus pour résoudre le défi de la reconnaissance des chiffres manuscrits. Nous discuterons également des méthodes d'entraînement et d'optimisation que nous avons appliquées pour maximiser les performances de nos modèles.

Nous allons parler de la manière dont nous avons traduit les concepts théoriques en actions concrètes, et comment nous avons préparé nos modèles pour la tâche complexe de la reconnaissance des chiffres manuscrits. Les choix méthodologiques que nous avons faits joueront un rôle clé dans la réussite de notre démarche, et nous aborderons également les raisons derrière ces choix.

2. Ensembles de Données :

Pour mener à bien la reconnaissance des chiffres manuscrits avec le Deep Learning, il est essentiel de disposer de jeux de données appropriés. Plusieurs ensembles de données sont couramment utilisés dans la recherche et le développement de modèles de reconnaissance des chiffres manuscrits. Parmi les plus célèbres, nous pouvons citer :

2.1. MNIST (Modified National Institute of Standards and Technology): [6]



MNIST est sans doute l'ensemble de données le plus emblématique pour la reconnaissance des chiffres manuscrits. Il contient 60 000 images d'apprentissage et 10 000 images de test, chacune étant un chiffre manuscrit de 0 à 9, de résolution 28x28 pixels en niveaux de gris. MNIST est souvent utilisé comme point de départ pour tester de nouvelles approches en raison de sa taille gérable.

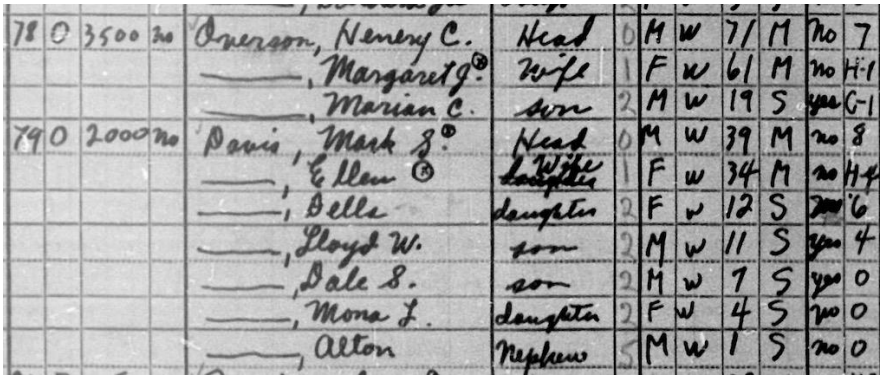
2.2. SVHN (Street View House Numbers): [18]

SVHN est une autre base de données populaire qui contient des chiffres manuscrits extraits de photos de rue. Il est plus complexe que MNIST car il présente des variations de taille, d'orientation et d'éclairage. SVHN compte environ 600 000 images de chiffres, ce qui en fait un ensemble de données plus volumineux.

2.3. EMNIST (Extended MNIST): [8]

EMNIST est une extension de MNIST qui comprend non seulement les chiffres, mais également des lettres de l'alphabet. Cela en fait un ensemble de données plus diversifié pour la reconnaissance de caractères manuscrits.

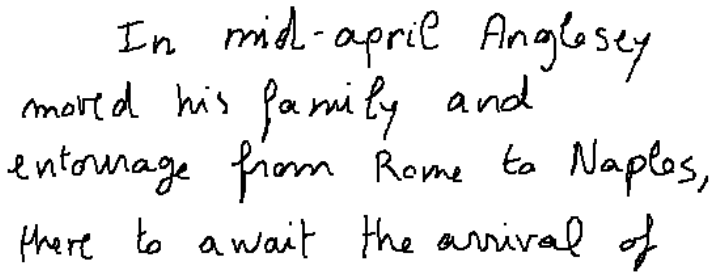
2.4. Census Bureau Handwriting : [1]



Cet ensemble de données contient des chiffres manuscrits extraits de formulaires fiscaux américains. Il est souvent utilisé dans des contextes de reconnaissance de caractères manuscrits pour des applications telles que la numérisation de documents fiscaux.

2.5. IAM Handwriting Database : [3]

Cet ensemble de données comprend des textes manuscrits provenant de documents divers. Il est utilisé pour des tâches de reconnaissance de texte manuscrit, mais les chiffres manuscrits peuvent également être extraits pour des tâches spécifiques.



3. Préparation des Données pour l'Entraînement des Modèles :

La qualité des données est cruciale pour la formation de modèles de reconnaissance des chiffres manuscrits. Voici les étapes typiques de préparation des données :

3.1. Collecte et Annotation des Données :

Les images de chiffres manuscrits sont collectées et annotées avec les chiffres correspondants. Cette étape peut être réalisée manuellement ou à l'aide d'outils d'annotation automatique.

3.2. Division en ensembles d'apprentissage, de validation et de test :

Les données sont divisées en trois ensembles distincts : un ensemble d'apprentissage pour entraîner le modèle, un ensemble de validation pour ajuster les hyperparamètres et un ensemble de test pour évaluer les performances finales. C'est une étape essentielle de la préparation des données pour l'entraînement et l'évaluation de modèles de Deep Learning. Voici plus de détails sur cette procédure :

3.2.1. Ensemble d'Apprentissage (Training Set) :

- L'ensemble d'apprentissage est la partie des données réservée à l'entraînement du modèle.
- Il s'agit du jeu de données sur lequel le modèle va apprendre à reconnaître les chiffres manuscrits en ajustant ses poids et ses biais.
- Cet ensemble doit être suffisamment grand pour permettre au modèle d'apprendre des modèles significatifs tout en évitant le surapprentissage (overfitting), où le modèle mémorise les données d'entraînement au lieu d'apprendre des modèles généraux.

3.2.2. Ensemble de Validation (Validation Set) :

- L'ensemble de validation est utilisé pour évaluer les performances du modèle pendant l'entraînement et ajuster les hyperparamètres.
- Les hyperparamètres sont des paramètres du modèle qui ne sont pas appris pendant l'entraînement, tels que le taux d'apprentissage ou la taille du lot (batch size). Ils doivent être ajustés de manière empirique pour optimiser les performances du modèle.
- L'ensemble de validation permet de mesurer comment le modèle se comporte sur des données qu'il n'a pas encore vues, ce qui aide à détecter le surapprentissage.

3.2.3. Ensemble de Test (Test Set) :

- L'ensemble de test est réservé à l'évaluation finale du modèle, une fois qu'il a été entraîné et ses hyperparamètres ajustés.
- Il s'agit d'un jeu de données indépendant de ceux d'apprentissage et de validation, ce qui signifie que le modèle ne l'a jamais vu auparavant.
- L'ensemble de test permet d'estimer la capacité du modèle à généraliser à de nouvelles données. Les performances mesurées sur cet ensemble reflètent à quel point le modèle est efficace dans des conditions réelles.

La division des données en ces trois ensembles vise à éviter le biais de surapprentissage. Le modèle est formé sur l'ensemble d'apprentissage, ajusté sur l'ensemble de validation, et ses performances sont enfin évaluées sur l'ensemble de test. Cette séparation garantit que les performances mesurées sont un indicateur fiable de la capacité du modèle à fonctionner avec de nouvelles données et à éviter la sur-optimisation des hyperparamètres pour un ensemble de données spécifique.

Typiquement, les données sont divisées en proportions telles que 60-80 % pour l'ensemble d'apprentissage, 10-20 % pour l'ensemble de validation, et 10-20 % pour l'ensemble de test. Les proportions exactes peuvent varier en fonction de la taille totale du jeu de données et des besoins spécifiques du projet.

4. Prétraitement des Données :

Le prétraitement des données est une étape cruciale dans la préparation des. Voici des détails sur les techniques couramment utilisées pour améliorer la qualité des données :

✓ **Normalisation :**

- La normalisation consiste à mettre à l'échelle les valeurs des pixels de l'image pour qu'elles se situent dans une plage spécifique, généralement entre 0 et 1 ou -1 et 1.
- Cette technique permet de rendre les données comparables et facilite la convergence de l'apprentissage, car elle assure que les valeurs des caractéristiques ont une plage plus homogène.

✓ **Redimensionnement :**

- Le redimensionnement implique de modifier la taille des images pour qu'elles aient une résolution uniforme.
- Dans le contexte de la reconnaissance des chiffres manuscrits, cela peut signifier redimensionner toutes les images à une taille spécifique, par exemple, 28x28 pixels, pour garantir que le modèle reçoit des données d'entrée de taille constante.

✓ **Suppression du Bruit :**

- Le bruit dans les images peut provenir de diverses sources, telles que des erreurs de numérisation, des taches, ou des irrégularités de fond.

- Les techniques de suppression du bruit, telles que le filtre médian ou le filtre gaussien, peuvent être appliquées pour améliorer la qualité des images en réduisant les artefacts indésirables.

✓ **Augmentation des Données (Data Augmentation) :**

- L'augmentation des données est une technique qui consiste à créer de nouvelles données d'apprentissage en appliquant des transformations aléatoires aux images existantes.
- Cela peut inclure des rotations légères, des translations, des zooms, des changements d'inclinaison, et même des modifications de la luminosité ou du contraste.
- L'augmentation des données augmente la diversité des exemples d'apprentissage, ce qui aide à prévenir le surapprentissage.

✓ **Binarisation (Thresholding) :**

- La binarisation consiste à convertir une image en niveaux de gris en une image binaire, où les pixels sont soit noirs (0) soit blancs (1) en fonction d'un seuil prédéfini.
- Cela peut être utile lorsque seuls les contours des chiffres sont nécessaires, par exemple, dans la segmentation de chiffres.

✓ **Filtrage :**

- L'application de filtres, tels que les filtres de détection de contours (par exemple, le filtre de Sobel), peut aider à mettre en évidence les caractéristiques importantes dans les images, facilitant ainsi la reconnaissance des chiffres.

✓ **Égalisation d'Histogramme :**

- L'égalisation d'histogramme est une technique de traitement d'image qui répartit les niveaux de gris dans une image de manière uniforme.
- Cela peut améliorer le contraste dans les images, ce qui est utile lorsque les chiffres sont mal visibles en raison d'un faible contraste.

✓ **Suppression du Fond :**

- Dans certains cas, il peut être nécessaire de supprimer le fond de l'image pour ne laisser que le chiffre manuscrit. Cela peut être réalisé à l'aide de techniques de segmentation d'images.

Le choix des techniques de prétraitement dépend des besoins. Une combinaison appropriée de ces techniques peut considérablement améliorer la qualité des données et aider le modèle à apprendre des représentations plus robustes et pertinentes pour la tâche.

5. Architecture des Modèles :

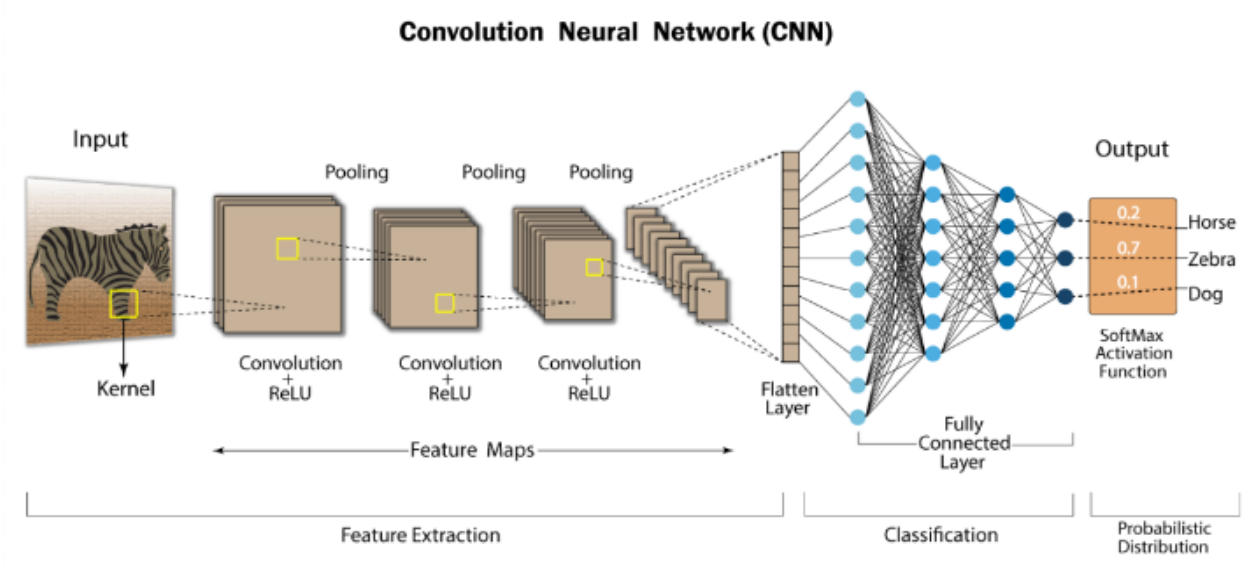


Fig. 1.7. Architecture de CNN

Les CNN sont la pierre angulaire de nombreuses applications de traitement d'images, y compris la reconnaissance des chiffres manuscrits. Ils sont conçus pour capturer des motifs spatiaux dans des données visuelles, ce qui les rend idéaux pour ce type de tâche.

Un CNN est généralement composé de plusieurs couches (voir la figure 1.7). Les couches initiales sont chargées de la détection des caractéristiques de bas niveau telles que les bords, tandis que les couches ultérieures détectent des caractéristiques de plus en plus complexes. Une architecture typique comprend des couches de convolution, de pooling et des couches entièrement connectées (fully connected).

5.1. Couches de Convolution :

Les couches de convolution utilisent des filtres (ou noyaux) pour balayer l'image d'entrée. Ces filtres apprennent automatiquement à détecter des motifs spécifiques, tels que des bords, des coins, ou des formes plus complexes. Les couches de convolution sont cruciales pour extraire des caractéristiques pertinentes dans les chiffres manuscrits.

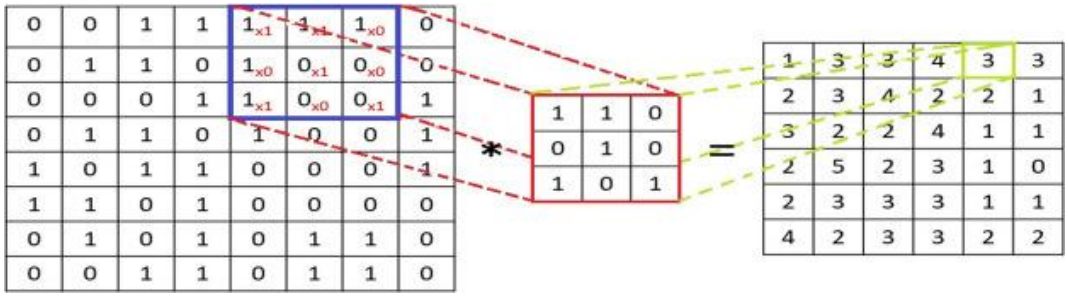


Fig. 1.8. Convolution avec un filtre 3*3

5.2. Couches de Pooling :

Les couches de pooling réduisent la dimension spatiale de l'image, ce qui réduit le nombre de paramètres du modèle et permet de capturer des caractéristiques invariantes à l'échelle et à la translation. Le pooling le plus couramment utilisé est le max-pooling, qui retient la valeur maximale dans chaque région d'un filtre.

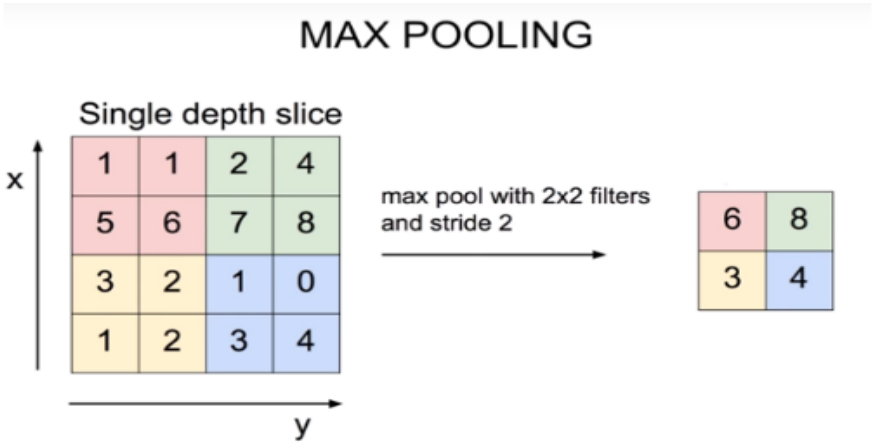


Fig. 1.9. Max_Pooling avec un filtre 2*2

5.3. Couches Entièrement Connectées :

Après avoir extrait des caractéristiques à partir de l'image, les couches entièrement connectées sont utilisées pour effectuer la classification. Ces couches sont semblables à celles d'un réseau neuronal

traditionnel, et elles combinent les informations spatiales extraites des couches de convolution pour produire une prédiction finale du chiffre.

5.4. Fonctions d'Activation :

5.4.1. La fonction ReLU (Rectified Linear Unit) :

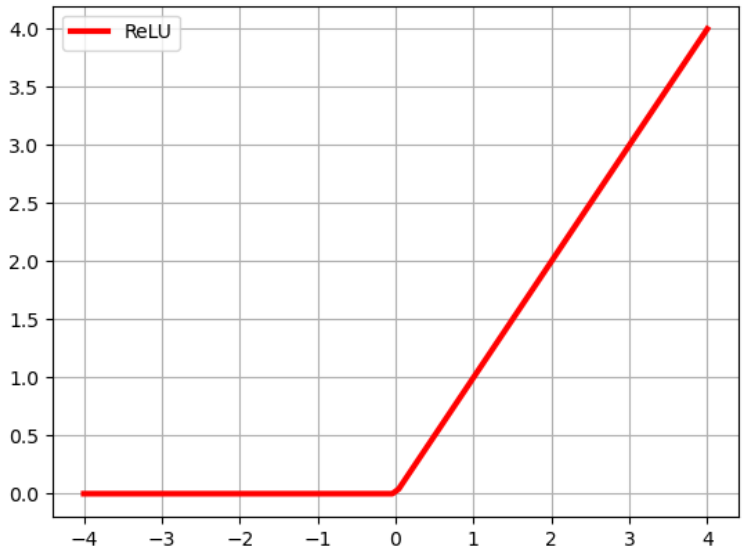


Fig. 1.10. La fonction ReLU

Entre chaque couche, des fonctions d'activation non linéaires, telles que ReLU, sont généralement appliquées. Cela introduit de la non-linéarité dans le modèle et permet au CNN d'apprendre des représentations complexes.

Fig. 1.11. Exemple d’application de ReLU

5.4.2. La fonction Softmax (Sigmoid-Function / Logistique-Function) :

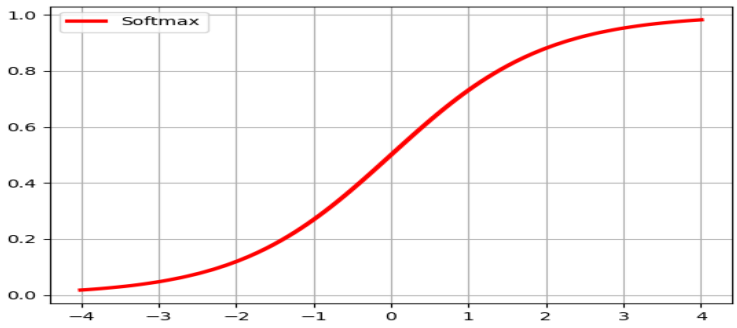


Fig. 1.12. La fonction Softmax

La fonction d'activation Softmax est couramment utilisée dans la couche de sortie d'un CNN pour la classification multi-classes, comme c'est le cas dans la reconnaissance des chiffres manuscrits. Elle fonctionne de la manière suivante :

- Dans notre cas, il y a généralement dix classes différentes (chiffres de 0 à 9). Le modèle doit donc être capable de classifier chaque entrée (une image de chiffre manuscrit) en l'une de ces dix classes.
- Avant d'appliquer Softmax, la couche de sortie génère des scores pour chaque classe. Ces scores ne sont pas encore des probabilités, mais ils indiquent à quel point l'image d'entrée ressemble à chacune des classes.
- C'est là que la fonction d'activation Softmax intervient. Elle prend les scores de classe et les transforme en une distribution de probabilité sur les différentes classes. Chaque classe obtient une probabilité, et l'ensemble de ces probabilités somme à 1.
- La classe avec la probabilité la plus élevée est alors choisie comme prédiction du modèle pour l'image donnée. Par exemple, si la probabilité que l'image représente le chiffre "6" est la plus élevée, le modèle prédit que l'image est un "6".

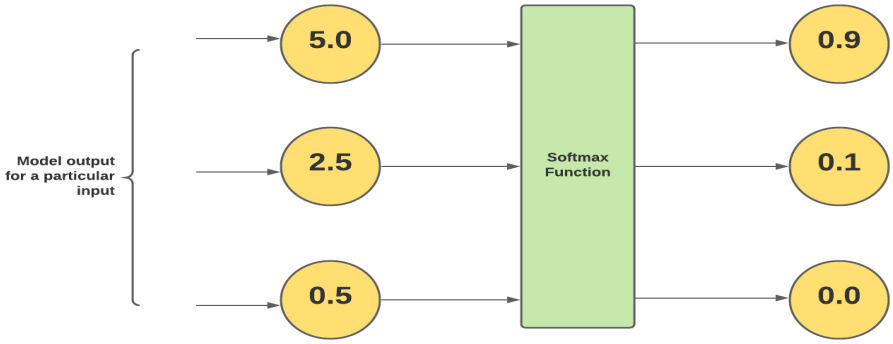


Fig. 1.13. Exemple sur Softmax

- Softmax est également utilisé conjointement avec une fonction de perte telle que la perte d'entropie croisée catégorielle (categorical cross-entropy loss) pour évaluer à quel point les prédictions du modèle correspondent aux étiquettes réelles (les chiffres corrects). Cette fonction de perte est utilisée pour guider l'apprentissage du modèle pendant l'entraînement.

En résumé, Softmax permet de convertir les scores bruts en probabilités et de sélectionner la classe prédite. Cela permet au modèle d'apprendre à attribuer des probabilités correctes aux différentes classes, ce qui est essentiel pour une classification précise des chiffres manuscrits.

5.5. Réduction du Nombre de Paramètres :

Les CNN exploitent l'idée que les motifs utiles dans les images sont souvent locaux et partagés. Plutôt que d'avoir des poids distincts pour chaque pixel, les CNN utilisent des filtres partagés pour réduire considérablement le nombre de paramètres.

5.6. Flattening (Aplatissage) :

Après l'étape de convolution et de pooling, la carte de caractéristiques 2D est aplatie en un vecteur unidimensionnel. C'est ce qu'on appelle l'étape de "Flattening". Elle consiste à prendre toutes les valeurs de la carte de caractéristiques et à les aligner dans une seule séquence linéaire.

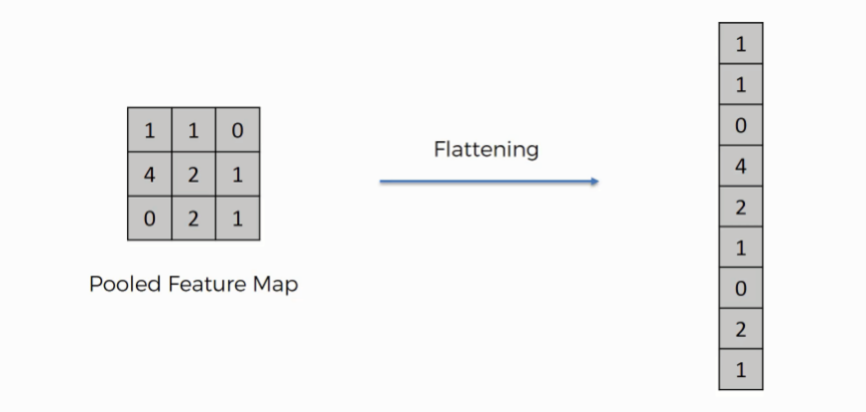


Fig. 1.14. Exemple d'aplatissage d'une image

Une fois que la carte de caractéristiques est aplatie, elle peut être utilisée comme entrée pour les couches entièrement connectées du réseau. Contrairement aux couches de convolution et de pooling, où les connexions sont locales et partagées, les couches entièrement connectées ont des connexions entre chaque neurone, ce qui signifie que chaque neurone de la couche entièrement connectée est connecté à chaque valeur du vecteur aplati.

Les couches entièrement connectées sont responsables de la classification finale. Elles apprennent à associer les caractéristiques extraites par les couches précédentes à des classes spécifiques.

6. Entraînement des Modèles :

L'entraînement des modèles de reconnaissance des chiffres manuscrits implique le calcul de la fonction coût, l'utilisation d'algorithmes d'optimisation tels que la descente de gradient et l'ajustement des hyperparamètres pour obtenir de bonnes performances.

Commençant par l'initialisation des poids d'une manière aléatoire, la propagation avant, arrivant à la 1ère étape la plus importante qui est le calcul de la fonction coût :

6.1. La Fonction coût (Loss Function) - Entropie Croisée Catégorielle :

Comme nous avons mentionné au 1er chapitre, La fonction coût est une mesure de l'erreur entre les prédictions du modèle et les étiquettes réelles (les chiffres corrects).

La perte d'entropie croisée catégorielle mesure la divergence entre les probabilités prédites par le modèle et les probabilités réelles des classes. Pour une seule observation (un exemple d'entraînement), elle est définie comme suit :

$$Loss(y_{true}, y_{pred}) = -\sum_{i=1}^{m} y_{true_i} * \log(y_{pred_i}) \quad \dots\dots\dots\dots\dots (2.1)$$

Où :

m : le nombre d'exemples.

y_{true} : est le vecteur des vraies étiquettes.

y_{pred} : est le vecteur de prédictions du modèle

6.2. Calcul des Gradients :

La rétropropagation (backpropagation) est utilisée pour calculer les gradients de la fonction de coût par rapport aux poids (weights) et biais (biases) du modèle. Pour une seule observation, les gradients sont donnés par :

$$\frac{\partial Loss}{\partial \omega} = -\sum_{i=1}^{m}\left(\frac{y_{true_i}}{y_{pred_i}} - 1\right)\frac{\partial y_{pred_i}}{\partial \omega} \quad \dots\dots\dots\dots\dots (2.2)$$

$$\frac{\partial Loss}{\partial b} = -\sum_{i=1}^{m}\left(\frac{y_{true_i}}{y_{pred_i}} - 1\right)\frac{\partial y_{pred_i}}{\partial b} \quad \dots\dots\dots\dots\dots (2.3)$$

Où :

y_{pred_i} : est la probabilité prédite pour la classe i par le modèle.

$\frac{\partial y_{pred_i}}{\partial \omega}$: est le gradient de la prédiction par rapport au poids ω.

$\frac{\partial y_{pred_i}}{\partial b}$: est le gradient de la prédiction par rapport au biais b.

Ce calcul est effectué pour chaque poids ω et biais b du modèle.

6.3. La Descente de Gradient :

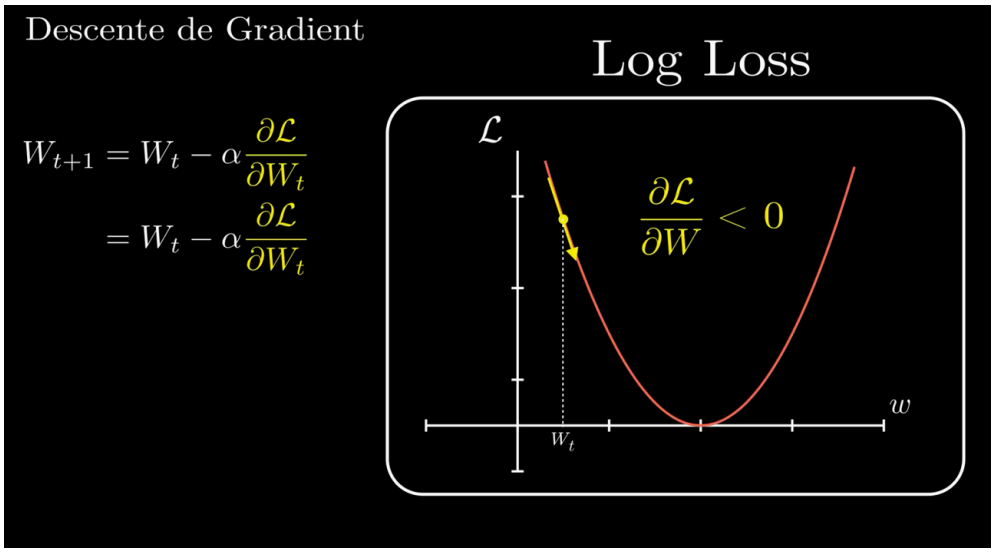


Fig. 1.15. Le Principe de la descente de gradient

La descente de gradient est l'algorithme d'optimisation utilisé pour ajuster les poids et les biais du modèle afin de minimiser la perte. Le poids ω et le biais b sont mis à jour à chaque itération de l'entraînement selon les formules suivantes (en utilisant le taux d'apprentissage α) :

$$\omega = \omega - \alpha \frac{\partial Loss}{\partial \omega} \quad \dots\dots\dots\dots\dots (2.4)$$

$$b = b - \alpha \frac{\partial Loss}{\partial b} \quad \dots\dots\dots\dots\dots (2.5)$$

Où :

$\frac{\partial Loss}{\partial \omega}$: est le gradient de la perte par rapport à ce poids spécifique.

$\frac{\partial Loss}{\partial b}$: est le gradient de la perte par rapport à ce biais spécifique.

6.4. Hyperparamètres :

Les hyperparamètres sont des paramètres du modèle et du processus d'entraînement qui doivent être ajustés pour obtenir de bonnes performances :

6.4.1. Taux d'apprentissage (Learning_rate) :

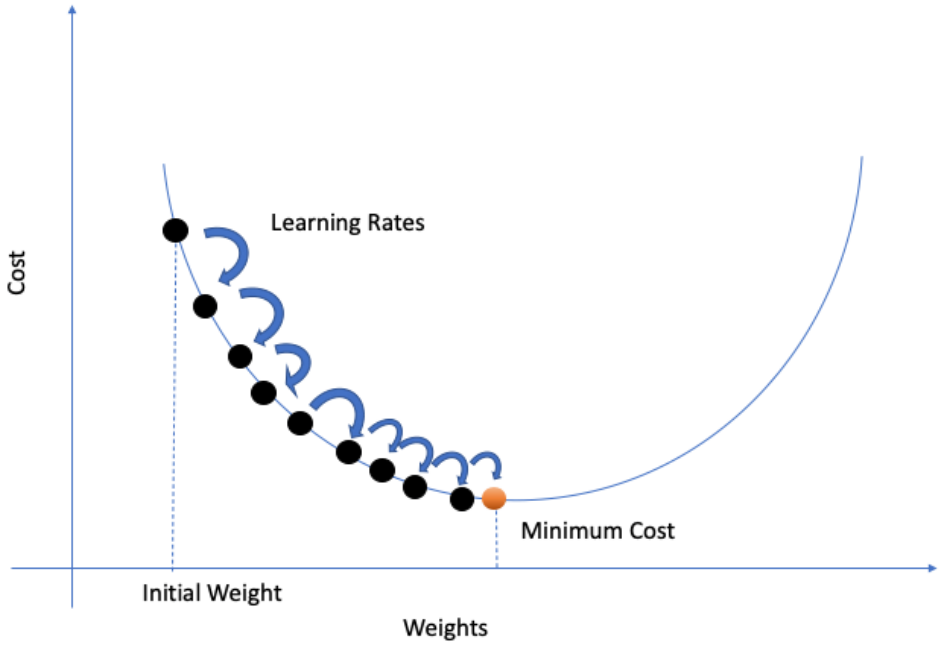


Fig. 1.16. Illustration du taux d'apprentissage

Le taux d'apprentissage contrôle la taille des pas que la descente de gradient prend lors de la mise à jour des poids du modèle. Un taux d'apprentissage inapproprié peut entraîner des problèmes tels que la convergence lente ou la divergence de l'entraînement. Voici comment il peut être ajusté :

- **Taux d'apprentissage élevé :**

Un taux d'apprentissage élevé peut accélérer la convergence, mais il peut également entraîner une divergence si les pas sont trop grands. Pour l'ajuster, il faut commencer par un taux élevé et diminuez-le progressivement si la perte ne diminue pas ou si le modèle diverge.

- **Taux d'apprentissage faible :**

Un taux d'apprentissage faible assure généralement une convergence stable, mais l'entraînement peut être lent. Si la perte diminue lentement, il faut essayer d'augmenter légèrement le taux d'apprentissage.

- **Taux d'apprentissage adaptatif :**

Les méthodes d'optimisation adaptatives, telles qu'Adam, ajustent automatiquement le taux d'apprentissage pendant l'entraînement en fonction de la réponse du modèle.

6.4.2. Nombre d'Itérations (Époques) :

Le nombre d'itérations, ou époques, détermine combien de fois l'ensemble d'apprentissage est parcouru lors de l'entraînement. Le choix du nombre d'époques dépend de la convergence du modèle :

- **Sous-apprentissage (Under-fitting) :**

Si la perte diminue rapidement mais n'atteint pas un plateau après un petit nombre d'époques, le modèle sous-apprend. Dans ce cas, nous pouvons augmenter le nombre d'époques pour permettre une convergence plus lente.

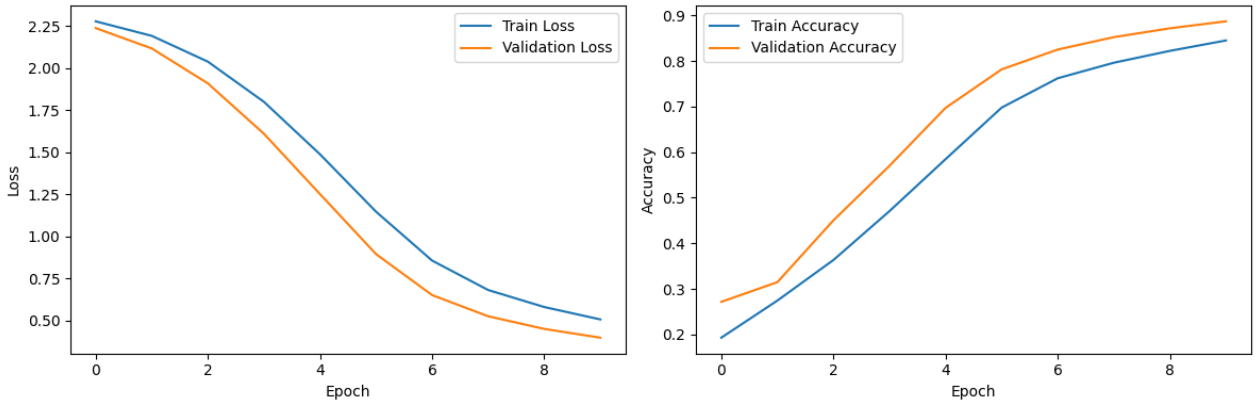


Fig. 1.17. Exemple sur le Sous-apprentissage

- **Sur-apprentissage (Over-fitting):**

Si la perte diminue sur l'ensemble d'apprentissage mais commence à augmenter sur l'ensemble de validation, le modèle surapprend. Nous devons surveiller l'ensemble de validation et arrêter l'entraînement dès que la performance commence à se détériorer.

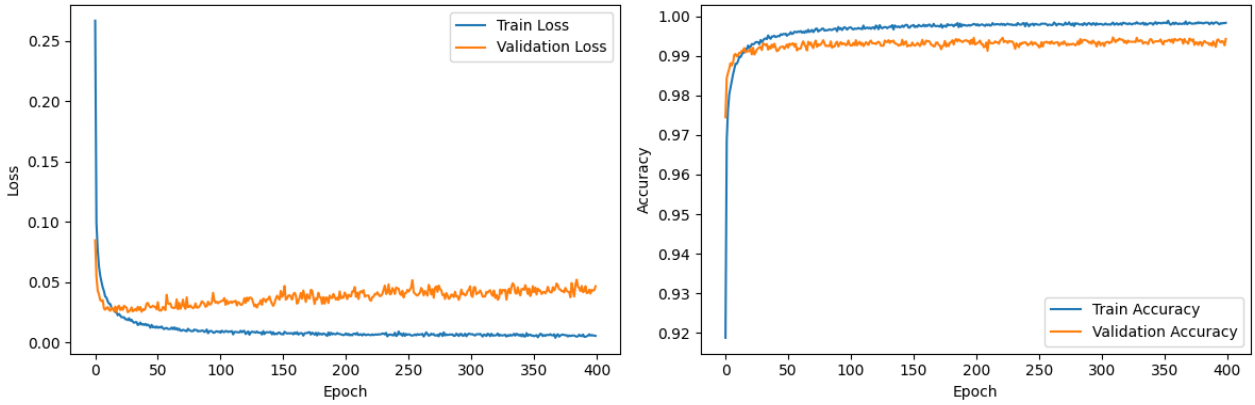


Fig. 1.18. Exemple sur le Sur-apprentissage

6.4.3. Taille du Lot (Batch Size) :

La taille du lot détermine combien d'exemples d'entraînement sont utilisés à chaque itération lors de la mise à jour des poids. Le choix de la taille du lot peut influencer la stabilité de l'entraînement :

- **Taille du lot plus grande :**

Une taille de lot plus grande accélère généralement l'entraînement, car les mises à jour des paramètres sont moins fréquentes. Cependant, cela nécessite plus de mémoire GPU et peut conduire à une convergence plus bruyante.

- **Taille du lot plus petite :**

Une taille de lot plus petite peut améliorer la stabilité de l'entraînement en introduisant plus de variabilité dans les mises à jour des paramètres. Cependant, cela peut ralentir l'entraînement.

Le choix de ces trois hyperparamètres est généralement basé sur des expérimentations itératives, en évaluant les performances du modèle sur l'ensemble de validation à chaque étape.

7. Conclusion :

Dans ce chapitre, nous avons exploré en détail la méthodologie et les processus mathématiques sous-jacents à la création de notre modèle de deep learning pour la reconnaissance des chiffres manuscrits. Nous avons discuté des principaux éléments de la conception du modèle, y compris l'architecture du réseau neuronal convolutif (CNN) et la manière dont il est adapté à la tâche de

reconnaissance des chiffres. Nous avons également abordé les hyperparamètres critiques tels que le taux d'apprentissage, le nombre d'itérations et la taille du lot, en mettant en avant l'importance de leur ajustement pour obtenir des performances optimales. De plus, nous avons examiné la fonction de coût (entropie croisée catégorielle) ainsi que le calcul des gradients et la descente de gradient, qui sont essentiels à l'entraînement du modèle.

Maintenant que nous avons acquis une compréhension approfondie de ces aspects théoriques, nous sommes prêts à passer à la phase pratique de programmation, où nous allons mettre en œuvre et entraîner notre modèle de deep learning sur des données réelles de chiffres manuscrits. Ce faisant, nous espérons obtenir un modèle performant capable de reconnaître avec précision les chiffres manuscrits, ce qui constituera un jalon important dans notre quête de comprendre et d'utiliser efficacement les techniques de deep learning dans le domaine de la reconnaissance des caractères manuscrits.

Chapitre 3 :

Implémentation et Résultats

1. Introduction :

Après avoir posé les fondations théoriques dans les deux premiers chapitres, nous abordons maintenant la phase pratique de notre étude. Le chapitre 3 constitue le cœur de notre exploration, où nous mettons en œuvre concrètement notre modèle de deep learning pour la reconnaissance des chiffres manuscrits.

Ce chapitre débute en détaillant la manière dont nous avons préparé nos données, en utilisant des ensembles de données bien établis et en effectuant les transformations nécessaires pour les rendre appropriés à l'entraînement de notre modèle. Ensuite, nous exposons l'architecture précise de notre réseau neuronal convolutif (CNN) et les hyperparamètres que nous avons ajustés pour obtenir les meilleures performances. Après l'entraînement, nous présentons les résultats de notre modèle sur des données de test réelles.

Enfin, ce chapitre s'achève par une discussion approfondie des résultats, de leurs implications, des limites potentielles de notre modèle, et des pistes d'amélioration futures. Nous explorons également l'utilité de notre approche dans divers contextes d'application et son impact global sur la reconnaissance de caractères manuscrits.

Ce chapitre représente le point culminant de notre étude, où la théorie se traduit en action, et où nous tirons des conclusions substantielles à partir de nos résultats concrets.

2. Description du Data-set : [6]

La base de données MNIST de chiffres manuscrits comprend un ensemble de formation 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.

Il s'agit d'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 consacrant un minimum d'efforts au prétraitement et au formatage.

Quatre fichiers sont disponibles sur ce site :

- Train_images : images de l'ensemble d'entraînement.
- Train_labels : étiquettes de l'ensemble d'entraînement.
- Test_images : images de l'ensemble de test.
- Test_labels : étiquettes d'ensemble de test.

3. Présentation d'outils :

3.1. Le langage de programmation :

Python est un langage de programmation de haut niveau interprété, orienté objet et doté d'une sémantique dynamique. Ses structures de données intégrées de haut niveau, combinées au typage et à la liaison dynamiques, le rendent très attrayant pour le développement rapide d'applications, ainsi que pour une utilisation comme langage de script ou de collage pour connecter des composants existants entre eux. La syntaxe simple et facile à apprendre de Python met l'accent sur la lisibilité et réduit donc le coût de maintenance du programme. Python prend en charge les modules et les packages, ce qui encourage la modularité des programmes et la réutilisation du code. L'interpréteur Python et la vaste bibliothèque standard sont disponibles gratuitement sous forme source ou binaire pour toutes les principales plates-formes et peuvent être distribués librement. [14]

C'est un langage qui prend en charge la création d'une large gamme d'applications. Les développeurs le considèrent comme un excellent choix pour les projets d'intelligence artificielle (IA), d'apprentissage automatique et d'apprentissage profond.

Il dispose d'un grand nombre de bibliothèques et de frameworks : Le langage Python est livré avec de nombreuses bibliothèques et frameworks qui facilitent le codage. Cela permet également de gagner un temps considérable.[15]

3.2. L'interface d'exécution : [13]

Jupyter Notebook est une application web open source populaire utilisée pour la création, le partage et l'exécution de documents interactifs appelés "notebooks". Ces notebooks peuvent contenir du texte, des images, des formules mathématiques et surtout, du code informatique exécutable en temps réel. Jupyter Notebook prend en charge plusieurs langages de programmation, mais il est principalement associé à Python.

Il est largement utilisé dans des domaines tels que la science des données, l'apprentissage automatique, la recherche scientifique et l'éducation.

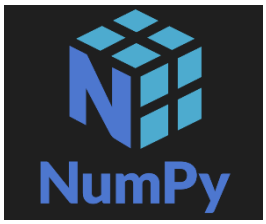
Voici quelques caractéristiques clés de Jupyter Notebook :

- **Support multilingue :** Bien qu'il soit couramment utilisé avec Python, Jupyter Notebook prend en charge plusieurs langages de programmation, tels que R, Julia, Scala et d'autres, grâce à des noyaux (kernels) spécifiques.

- **Environnement interactif :** Les notebooks permettent aux utilisateurs d'exécuter du code cellule par cellule, ce qui facilite le prototypage, la visualisation et le débogage du code.
- **Documentation intégrée :** Nous pouvons ajouter du texte, des commentaires et des explications aux notebooks, ce qui en fait un excellent outil pour la documentation et la communication de vos analyses et de vos projets.
- **Visualisation des données :** Jupyter Notebook offre une intégration transparente avec des bibliothèques de visualisation de données telles que Matplotlib, Seaborn et Plotly, ce qui en fait un choix populaire pour l'analyse de données et la création de graphiques.

3.3. Les Bibliothèques utilisées :

❖ **Numpy :** [10]



NumPy (abréviation de "Numerical Python") est une bibliothèque open source populaire et essentielle pour le calcul scientifique en Python. Elle fournit des structures de données et des fonctions permettant de manipuler efficacement des tableaux multidimensionnels et de réaliser des opérations mathématiques et statistiques sur ces tableaux.

❖ **Matplotlib :** [7]

Matplotlib est une bibliothèque de visualisation de données en Python, très populaire et puissante. Elle permet de créer une grande variété de graphiques, de diagrammes et de représentations visuelles à partir de données, ce qui en fait un outil essentiel pour l'analyse de données, la création de graphiques scientifiques, la production de graphiques pour la publication et bien plus encore.

Tensorflow : [17]

TensorFlow est une bibliothèque open source populaire développée par Google pour le machine learning (apprentissage automatique) et le deep learning (apprentissage profond). Elle est conçue pour permettre la création, l'entraînement et le déploiement de modèles d'apprentissage automatique, en particulier des réseaux neuronaux profonds. TensorFlow est largement utilisé dans l'industrie et la recherche pour une variété d'applications, y compris la vision par ordinateur, le traitement du langage naturel, la reconnaissance vocale, la traduction automatique, la robotique et plus encore.

❖ **Keras :** [5]

Keras est une interface de haut niveau pour la création, la formation et le déploiement de modèles d'apprentissage automatique, en particulier de réseaux neuronaux, qui s'intègre bien avec divers moteurs d'apprentissage automatique sous-jacents, notamment TensorFlow, Microsoft Cognitive Toolkit (CNTK) et Theano. Keras a été initialement développé par François Chollet et est devenu une partie intégrante de TensorFlow à partir de sa version 2.0.

❖ **Pillow :** [12]

Pillow est une bibliothèque Python largement utilisée pour le traitement d'images. Elle est principalement utilisée pour effectuer diverses opérations sur des images, notamment le chargement et l'enregistrement de différents formats d'images, la manipulation d'images, le redimensionnement, la conversion de couleurs, le rognage, la fusion, le dessin sur des images, etc. Pillow est une bibliothèque conviviale et populaire pour le traitement d'images en Python en raison de sa simplicité et de sa facilité d'utilisation.

❖ **OpenCV :** [11]

OpenCV (Open Source Computer Vision Library) est une bibliothèque open source très populaire et largement utilisée dans le domaine de la vision par ordinateur. Elle offre un ensemble complet de fonctions et d'outils pour le traitement et l'analyse d'images et de vidéos. OpenCV est écrit en C++ mais offre également des interfaces pour de nombreux autres langages de programmation, notamment Python, Java et MATLAB.

3.4. Outil d'entrainement :

❖ **Google Colaboratory :** [2]

Google Colaboratory, souvent appelé Colab, est un service cloud gratuit proposé par Google qui permet aux utilisateurs d'exécuter du code Python dans des environnements de notebook Jupyter hébergés en ligne. Colab offre un accès gratuit à des ressources de calcul, y compris des GPU (unités de traitement graphique), ce qui en fait un choix populaire pour l'apprentissage automatique, l'exploration de données et la recherche en science des données. Voici quelques caractéristiques et avantages de Google Colab :

- **Environnement basé sur le cloud :** Colab fonctionne entièrement dans le cloud, ce qui signifie que nous n'avons pas besoin d'installer ou de configurer un environnement Python sur votre propre machine. Nous pouvons y accéder depuis n'importe quel navigateur web.
- **Accélération matérielle :** Colab offre l'accès à l'unités de traitement graphique (GPU) gratuitement, ce qui est très utile pour accélérer les calculs, en particulier dans les tâches d'apprentissage automatique profond.
- **Intégration avec Google Drive :** Nous pouvons facilement sauvegarder et charger des notebooks Colab directement depuis Google Drive, ce qui facilite la gestion de projets et le partage avec d'autres utilisateurs.
- **Bibliothèques pré-installées :** Colab est pré-configuré avec de nombreuses bibliothèques populaires pour la science des données, l'apprentissage automatique, la visualisation et plus encore. Il est également possible d'installer des bibliothèques supplémentaires selon les besoins.

  - **Remarque :** L'entraînement d'un modèle de deep learning requiert généralement l'utilisation d'un processeur graphique (GPU), et c'est précisément ce que Google Colab met à notre disposition.

4. Explication du projet :

Notre projet est un programme écrit en Python, où nous allons entrainer un model CNN en utilisant « train_images » les images de l'ensemble d'entraînement fournies par MNIST. Valider ce modèle à l'aide de « test_images », et enfin le tester sur ma propre liste d'images de chiffres manuscrits.

5. Réalisation :

5.1. Phase d'entrainement :

Commençons par charger notre ensemble de données :

```
[ ]  import tensorflow as tf

     mnist = tf.keras.datasets.mnist

     (X_train, y_train), (X_test, y_test) = mnist.load_data()

     Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
     11490434/11490434 [==============================] - 1s 0us/step
     (60000, 28, 28)
```

+ Code + Texte

Fig. 1.19. Chargement de MNIST Dataset

Après le chargement de notre Dataset, nous allons le normaliser et le redimensionner :

```
X_train = tf.keras.utils.normalize(X_train, axis=1)
X_test = tf.keras.utils.normalize(X_test, axis=1)
```

Fig. 1.20. Dataset Normalisé

```
import numpy as np
IMG_SIZE = 28
X_train_r = np.array(X_train).reshape(-1, IMG_SIZE, IMG_SIZE, 1)
X_test_r = np.array(X_test).reshape(-1, IMG_SIZE, IMG_SIZE, 1)
print(X_train_r.shape)
print(X_test_r.shape)
```

```
(60000, 28, 28, 1)
(10000, 28, 28, 1)
```

Fig. 1.21. Dataset Redimensionné

Passons maintenant à l'étape d'initialisation de notre réseaux neuronaux convolutifs.

- **Remarque :** Pour parvenir à obtenir un modèle de qualité, il est essentiel d'effectuer plusieurs expérimentations en ajustant à la fois le nombre d'époques (epochs) et la taille des lots (batch size).

Dans notre modèle, nous avons élaboré une architecture de CNN qui intègre les couches suivantes :

- Une couche de convolution à 32 filtres avec activation ReLU, suivie d'une opération de max pooling et une couche de dropout (pour prévenir le sur-apprentissage).
- Une couche de convolution à 64 filtres avec activation ReLU, suivie d'une opération de max pooling et une couche de dropout.
- Une couche Flatten.
- Une couche Dense à 32 neurones avec activation ReLU.
- Une dernière couche Dense de classification à 10 neurones (chiffres de 0 à 9) avec activation Softmax.

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 conv2d_14 (Conv2D)           (None, 26, 26, 32)        320

 activation_32 (Activation)   (None, 26, 26, 32)        0

 max_pooling2d_14 (MaxPoolin  (None, 13, 13, 32)        0
 g2D)

 dropout_9 (Dropout)          (None, 13, 13, 32)        0

 conv2d_15 (Conv2D)           (None, 11, 11, 64)        18496

 activation_33 (Activation)   (None, 11, 11, 64)        0

 max_pooling2d_15 (MaxPoolin  (None, 5, 5, 64)          0
 g2D)

 dropout_10 (Dropout)         (None, 5, 5, 64)          0

 flatten_8 (Flatten)          (None, 1600)              0

 dense_18 (Dense)             (None, 32)                51232

 activation_34 (Activation)   (None, 32)                0

 dense_19 (Dense)             (None, 10)                330

 activation_35 (Activation)   (None, 10)                0

=================================================================
Total params: 70,378
Trainable params: 70,378
Non-trainable params: 0
_________________________________________________________________
```

Fig. 1.22. Résumé du modèle

- **Explication de « Output Shape (None, 26, 26, 32) » :**

"none" : Ce nombre indique généralement la taille du lot (batch size) que le modèle traite à la fois. Dans ce contexte, "none" signifie qu'il peut s'agir de n'importe quelle taille de lot, ce qui est typique lors de l'entraînement ou de l'inférence d'un modèle, car les modèles de CNN sont souvent conçus pour traiter des lots de données de différentes tailles.

"26" : Il s'agit de la hauteur (ou du nombre de lignes) de la sortie de la couche de convolution. Cela signifie que la convolution a été appliquée à l'image d'entrée ou à la couche précédente, et la sortie a maintenant une hauteur de 26 pixels.

"26" : Il s'agit de la largeur (ou du nombre de colonnes) de la sortie de la couche de convolution. Cela indique que la convolution a également été appliquée horizontalement, et la sortie a maintenant une largeur de 26 pixels.

"32" : Ce nombre représente le nombre de canaux (ou de "filtres") dans la sortie de la couche de convolution. Chaque canal est associé à un filtre appris qui extrait des caractéristiques spécifiques de l'image d'entrée. Dans ce cas, il y a 32 canaux de sortie.

Voici un schéma qui explique ce résumé :

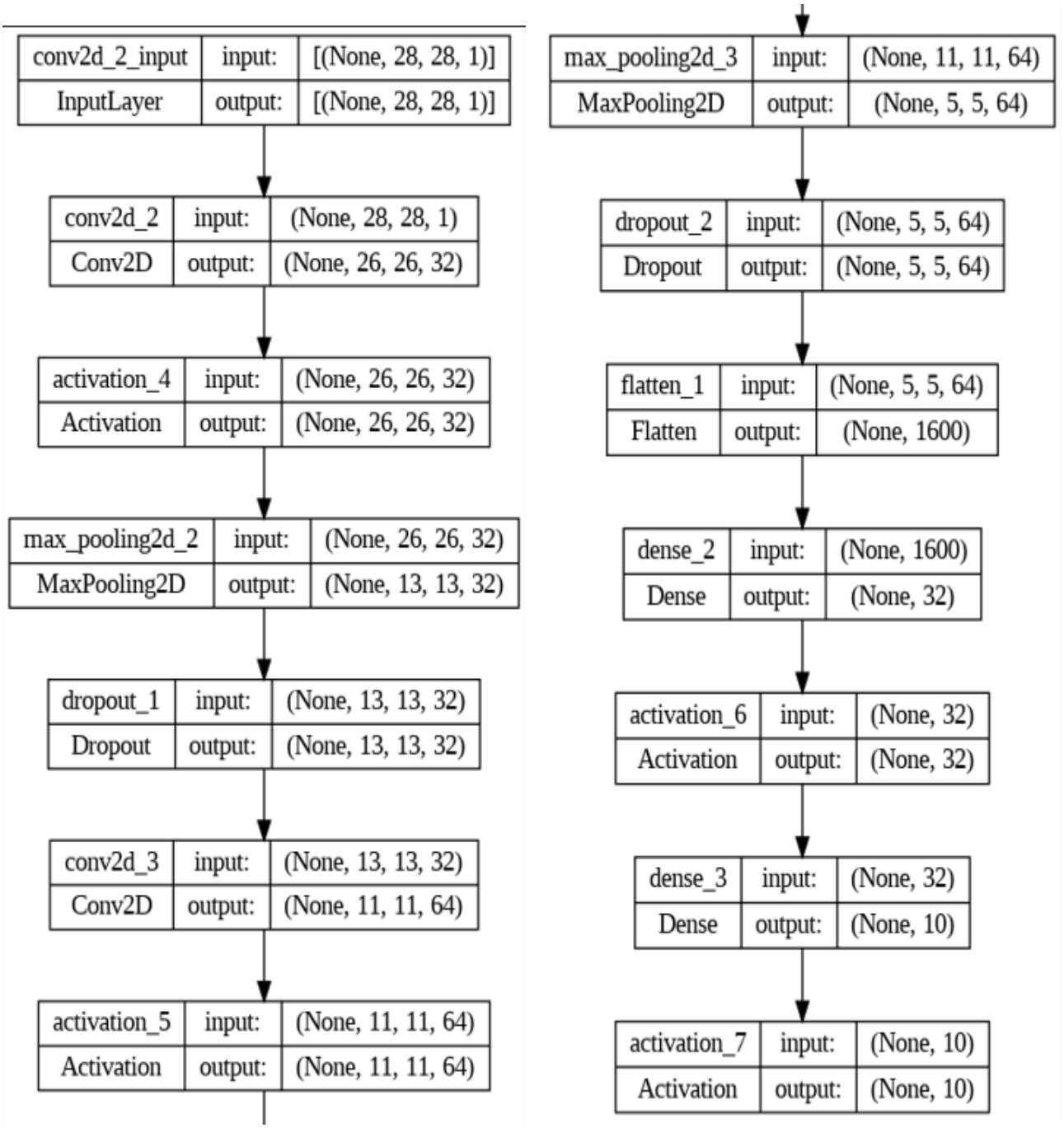


Fig. 1.23. Schéma de CNN en détails

Dans l'étape qui suit, nous allons choisir un optimiseur, une fonction de coût, et une métrique qui nous permettra de visualiser l'évaluation de notre modèle :

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

Fig. 1.24. Compilation du modèle

C'est bon nous avons fait tout le nécessaire, il ne reste plus qu'à entraîner notre modèle :

```
train_history = model.fit(X_train_r, y_train, batch_size=100, epochs=400, validation_data=(X_test_r, y_test))
```

Fig. 1.25. Entrainement du modèle

Pour l'entrainement, nous avons choisi le « train_set ».

Pour la validation, nous avons choisi le « test_set ».

5.2. Phase d'évaluation :

Il est maintenant temps d'évaluer notre modèle. Pour cela nous visualiserons l'évaluation de la fonction de coût (loss), et de la précision (accuracy) par rapport au nombre d'époques choisi.

Et voici les deux graphiques obtenus :

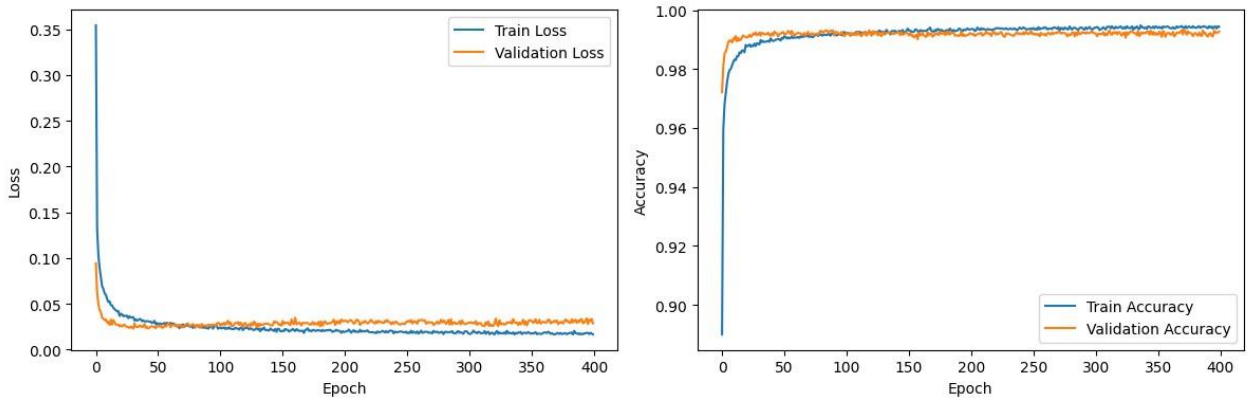


Fig. 1.26. Entrainement et Validation

5.3. Phase de tests :

Dans cette partie, nous importerons 12 images de nombres écrits dans le logiciel paint depuis le bureau :

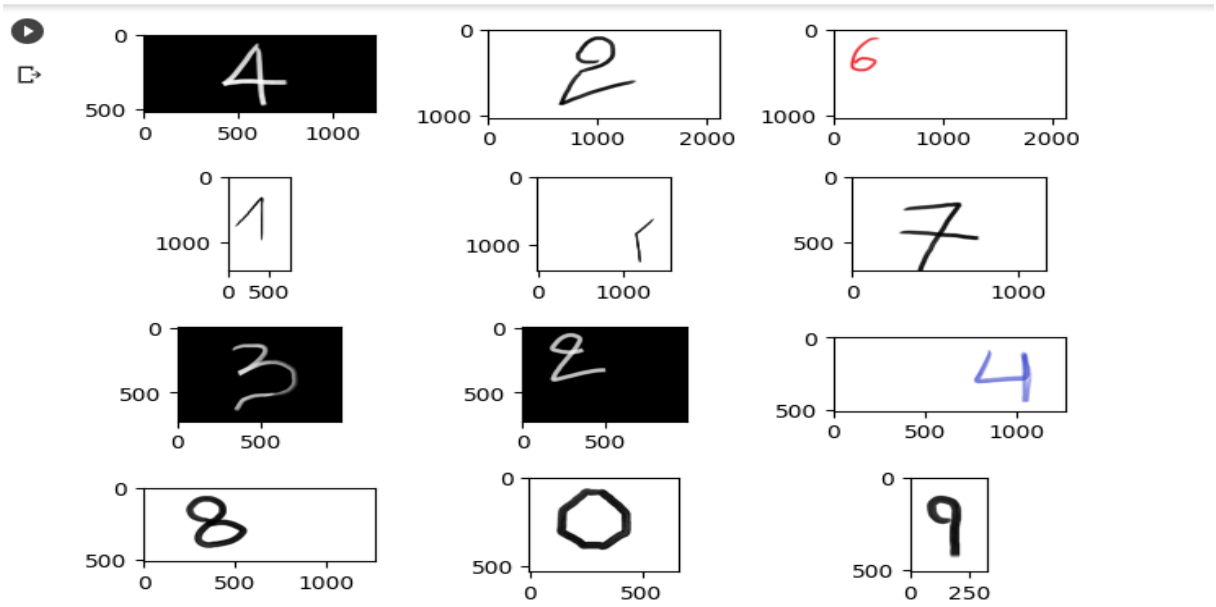


Fig. 1.27. Images de test

Ensuite, nous allons segmenter, normaliser et redimensionner ces images :

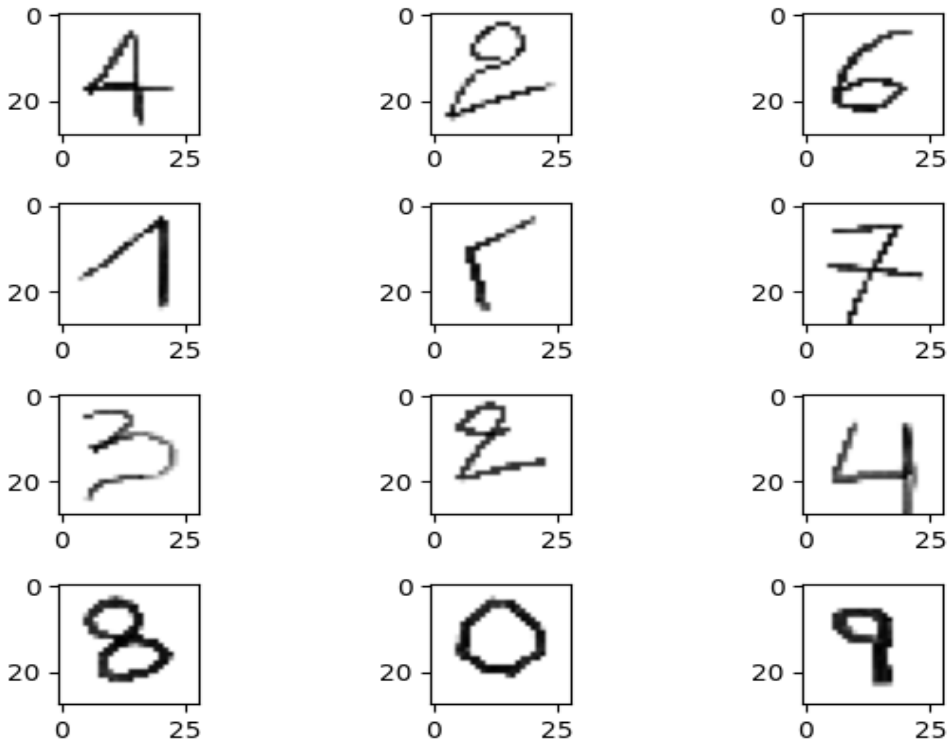


Fig. 1.28. Images pré-traitées

Enfin, il ne reste plus qu'à prédire les chiffres à l'aide de notre modèle entraîné :

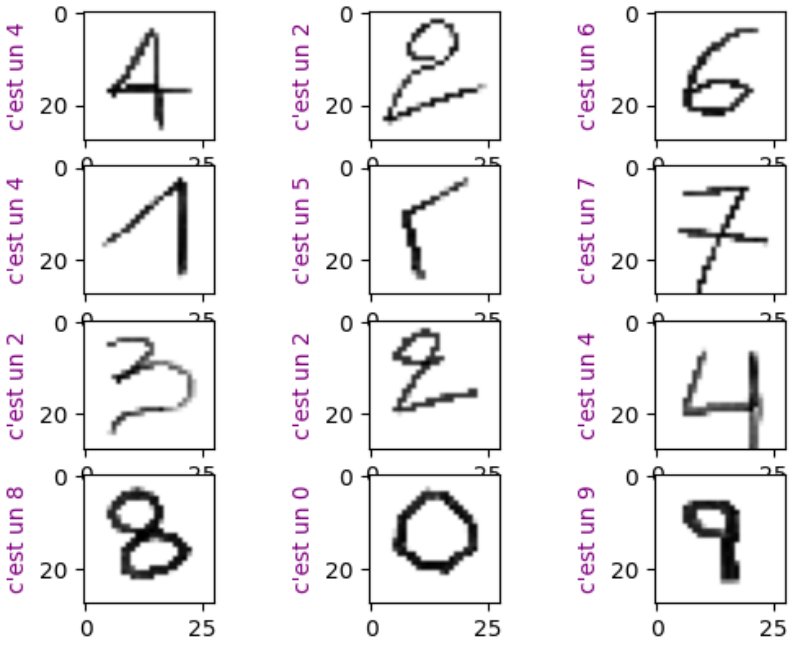


Fig. 1.29. Prédiction du modèle

- Le modèle a obtenu des réponses correctes dans 10 cas sur 12, ce qui équivaut à une précision d'environ 83 %.

6. Résultat :

Ce chapitre marque l'aboutissement de notre travail pratique dans la reconnaissance de chiffres manuscrits à l'aide du deep learning. Nous avons minutieusement planifié, implémenté et évalué notre modèle, et les résultats que nous avons obtenus sont prometteurs.

Les résultats de notre modèle sur l'ensemble de test ont démontré une précision encourageante, confirmant son efficacité dans la reconnaissance de chiffres manuscrits.

7. Conclusion :

En conclusion, ce chapitre représente l'essence de notre travail pratique, où la théorie s'est transformée en action. Nous sommes encouragés par les résultats obtenus, mais restons conscients des défis à relever. Ce chapitre prépare le terrain pour la conclusion générale de notre étude, où nous synthétiserons nos découvertes et discuterons de leur impact global.

Conclusion Générale

Au terme de cette étude sur la reconnaissance de chiffres manuscrits à l'aide du deep learning, nous pouvons tirer des conclusions significatives qui éclairent notre compréhension de ce domaine en pleine expansion. Notre recherche a été guidée par l'objectif de développer un modèle de deep learning efficace et polyvalent pour la reconnaissance de chiffres manuscrits, et les résultats que nous avons obtenus nous encouragent à envisager son potentiel dans divers contextes d'application.

Au cœur de notre recherche, les réseaux neuronaux convolutifs (CNN) se sont avérés être des outils puissants pour la reconnaissance de chiffres manuscrits. Ces architectures profondes ont la capacité unique d'apprendre automatiquement des caractéristiques pertinentes à partir des images, ce qui les rend particulièrement adaptées à la tâche de reconnaissance de chiffres.

Au fil de notre exploration, nous avons minutieusement conçu l'architecture de notre CNN, en accordant une attention méticuleuse à chaque couche et en ajustant les hyperparamètres pour maximiser les performances. Les résultats que nous avons obtenus sont un témoignage de la capacité des CNN à extraire des informations discriminantes à partir d'images complexes, et ce, avec une précision remarquable.

Cependant, notre recherche ne s'arrête pas ici. Les CNN continuent d'évoluer et de s'améliorer, et leur potentiel dans la reconnaissance de chiffres manuscrits est encore loin d'être pleinement exploité. Des architectures plus avancées, des techniques de régularisation innovantes et des approches d'augmentation de données créatives promettent d'améliorer encore davantage les performances des CNN dans ce domaine.

Dans l'ensemble, notre étude démontre que les CNN sont au cœur de l'avenir de la reconnaissance de chiffres manuscrits. Leur adaptabilité, leur efficacité et leur polyvalence les positionnent comme une technologie essentielle pour la résolution de nombreux problèmes du monde réel, ouvrant la voie à des applications innovantes et à une amélioration continue de notre compréhension de ce domaine en constante évolution.

En conclusion, cette étude a contribué à élargir notre compréhension de la reconnaissance de chiffres manuscrits en utilisant le deep learning. Elle démontre que cette technologie peut apporter des solutions efficaces et polyvalentes à des problèmes du monde réel. Ma recherche ne constitue qu'un point de départ, et nous espérons que d'autres travaux continueront à explorer et à étendre les possibilités offertes par cette discipline passionnante.

Références bibliographiques

[1] Bureau of the Census. (2021). Digitizing Handwritten Data: A Review of Techniques and Recent Experiments. [En ligne] Disponible sur : https://www.census.gov/library/working-papers/2021/econ/digitizing-hw-data.html

[2] Google Colab. (s.d.). [En ligne] Disponible sur : https://colab.google/

[3] IAM Handwriting Database. (s.d.). [En ligne] Disponible sur : https://fki.tic.heia-fr.ch/databases/iam-handwriting-database

[4] IBM. (s.d.). L'apprentissage supervisé. [En ligne] Disponible sur : https://www.ibm.com/fr-fr/topics/

[5] Keras. (s.d.). Keras. [En ligne] Disponible sur : https://keras.io/

[6] LeCun, Y. (1998). MNIST Database. [En ligne] Disponible sur : http://yann.lecun.com/exdb/mnist/

[7] Matplotlib. (s.d.). Matplotlib. [En ligne] Disponible sur : https://matplotlib.org/

[8] National Institute of Standards and Technology (NIST). (s.d.). EMNIST Dataset. [En ligne] Disponible sur : https://www.nist.gov/itl/products-and-services/emnist-dataset

[9] Nielsen, M. A. (s.d.). Neural Networks and Deep Learning. [En ligne] Disponible sur : http://neuralnetworksanddeeplearning.com/

[10] NumPy. (s.d.). NumPy. [En ligne] Disponible sur : https://numpy.org/

[11] OpenCV. (s.d.). OpenCV. [En ligne] Disponible sur : https://opencv.org/

[12] Pillow. (s.d.). Pillow Documentation. [En ligne] Disponible sur : https://pillow.readthedocs.io/en/stable/

[13] Project Jupyter. (s.d.). Jupyter. [En ligne] Disponible sur : https://jupyter.org/

[14] Python Software Foundation. (s.d.). Python. [En ligne] Disponible sur : https://www.python.org/doc/essays/blurb/

[15] Section.io. (s.d.). Engineering Education. [En ligne] Disponible sur : https://www.section.io/engineering-education/

[16] Sicara. (s.d.). Histoire du Deep Learning. [En ligne] Disponible sur : https://www.sicara.fr/fr/parlons-data/deep-learning#:~:text=%C3%A0%20la%20r%C3%A9alit%C3%A9.-,L'histoire%20du%20Deep%20Learning,repr%C3%A9sentation%20informatique%20de%20cerveau%20humain.

[17] TensorFlow. (s.d.). TensorFlow. [En ligne] Disponible sur : https://www.tensorflow.org/?hl=fr

[18] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, Andrew Y. Ng. (2011). Reading Digits in Natural Images with Unsupervised Feature Learning. [En ligne] Disponible sur: http://ufldl.stanford.edu/housenumbers/

ملخص

تركز هذه الدراسة على التعرف على الأرقام المكتوبة بخط اليد باستخدام الشبكات العصبية التلافيفية (CNN) في مجال التعلم العميق. لقد بدأنا بوضع أهمية هذه التكنولوجيا في سياق عصرنا الرقمي المتنامي. وبالنظر إلى تاريخ هذا التخصص، لاحظنا الانتقال من الأساليب التقليدية إلى الأساليب الحديثة القائمة على التعلم العميق. أدى بحثنا إلى إنشاء نموذج CNN وتحسينه، مما يوفر أداءً واعدًا في التعرف على الأرقام المكتوبة بخط اليد. واستكشفنا أيضًا تطبيقات مختلفة لهذه التقنية، بدءًا من عمليات التحقق من القراءة وحتى تحليل البيانات المكتوبة بخط اليد.

الكلمات المفتاحية: الشبكات العصبية التلافيفية , التعلم العميق, الأرقام المكتوبة بخط اليد .

Résumé

Cette étude se concentre sur la reconnaissance de chiffres manuscrits en utilisant des réseaux neuronaux convolutifs (CNN) dans le domaine de l'apprentissage profond.

Nous avons commencé par contextualiser l'importance de cette technologie dans notre ère numérique croissante.

En parcourant l'historique de cette discipline, nous avons noté le passage des méthodes traditionnelles aux approches modernes basées sur le deep learning. Notre recherche a abouti à la création et à l'optimisation d'un modèle de CNN, offrant des performances prometteuses dans la reconnaissance de chiffres manuscrits.

Nous avons également exploré diverses applications de cette technologie, allant de la lecture de chèques à l'analyse de données manuscrites.

Mots clés : la reconnaissance de chiffres manuscrits, des réseaux neuronaux convolutifs, apprentissage profond.

Abstract

This study focuses on handwritten digit recognition using networks convolutional neural networks (CNN) in the field of deep learning. We started with contextualize the importance of this technology in our growing digital age. While traveling through the history of this discipline, we have noted the transition from traditional methods to approaches modern technologies based on deep learning. Our research resulted in the creation and optimization of a CNN model, offering promising performance in handwritten digit recognition. We also explored various applications of this technology, ranging from reading checks to the analysis of handwritten data.

Keywords: handwritten digit recognition, convolutional neural networks, deep learning.