REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université 8 Mai 1945 – Guelma
Faculté des Sciences et de la Technologie
Département de Génie Electrotechnique et Automatique

Réf……/2025

MEMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER Académique**

Domaine : Sciences et Technologie

Filière : Automatique

Spécialité : Automatique et Informatique Industrielle

Par : BOUCHEBOUT Nor El Houda

Thème

Comparaison des performances de divers modèles CNN pour la classification d'images

Soutenu publiquement, le 22/06 /2025, devant le jury composé de :

Pr. BABOURI Abdesselam	Professeur	Univ. Guelma	Examinateur principale
Mr. Debeche Mehdi	MAA	Univ. Guelma	Encadreur
Dr.ELAGGOUNE Hocine	MCB	Univ. Guelma	Examinateur principale

Année Universitaire : 2024/2025

Remerciement

Je remercie Dieu Tout-Puissant de nous avoir donné la santé et le bien-être.

Je tiens tout d'abord à exprimer ma profonde gratitude à mon directeur de mémoire, Mr.Debeche mehdi, pour son encadrement, sa patience et sa confiance tout au long de ce travail de recherche. Ses précieux conseils, son expertise et son soutien inébranlable ont été d'une aide inestimable et ont grandement contribué à l'aboutissement de mon mémoire.

Mes remerciements vont également à l'ensemble des professeurs du département de génie électrotechnique et automatique de l'Université 8 mai 1945 Guelma, pour leur enseignement de qualité et les connaissances qu'ils m'ont transmises durant mes années d'études. Leur passion et leur dévouement pour la recherche m'ont inspiré et motivé à poursuivre mes propres questionnements scientifiques.

Chers membres du jury, je vous remercie de votre présence et de l'attention que vous avez portée à mon travail.

Je vous suis très reconnaissant d'avoir accepté d'évaluer mon mémoire. Votre expertise est précieuse pour moi.

Je souhaite exprimer ma gratitude à ma famille pour leur soutien indéfectible.

À tous ceux qui ont contribué de près ou de loin à ce mémoire, je vous adresse mes sincères remerciements.

Dédicace

À ma très chère mère Radia,
symbole de courage, de tendresse et de foi, qui n'a jamais cessé de croire en moi.
À mon père Aziz,
pour sa sagesse, sa patience et ses précieux conseils tout au long de mon parcours.
À ma sœur Bouchra,
pour sa douceur, son soutien constant et ses paroles réconfortantes dans les moments difficiles. Merci d'avoir toujours été là, discrètement mais profondément présente.
À mon frère badereddine,
pour sa force tranquille, ses encouragements sincères et sa confiance en moi. Ton exemple et ton appui m'ont donné le courage d'aller jusqu'au bout.
À mes amis les plus proches,
Pour leurs mots motivants, leur écoute bienveillante et leur amitié précieuse.
À tous ceux qui ont cru en moi,
Cette réussite est aussi la vôtre.

ملخص :

تقدم هذه المذكرة مقارنة أداء نماذج مختلفة للشبكات العصبية الالتفافية (CNN) المستخدمة في تصنيف الصور. يتعلق هذا المشروع بالذكاء الاصطناعي في مجال الرؤية الحاسوبية. تتمثل الفكرة في توضيح كيفية مقارنة الشبكات العصبية الالتفافية المختلفة من حيث الدقة، وزمن الحساب، وعدد المعاملات. نراجع العديد من البنى الشائعة ونطبقها باستخدام مجموعات بيانات MNIST, Fashion- MNIST وSVHN، مثل Resnet50 , Resnet18, Alexnet, Xception, Resnet101, وMobilnetV2. ومن خلال ذلك، نحدد بعض نقاط القوة والضعف لكل منها ونستكشف آليات عملها. وفي النهاية، يوفر هذا دليلاً لاختيار نموذج CNN المناسب لتطبيق معين.

الكلمات المفتاحية: CNN، تصنيف الصور، الرؤية الحاسوبية، الأداء، هندسة شبكات CNN، Alexnet، ResNet، Xception، MobilnetV2، مجموعات البيانات.

Résumé :

Ce mémoire présente la comparaison de la performance de différents modèles de réseaux de neurones convolutifs (CNN) utilisés pour la classification d'images. Il s'agit d'un projet lié à l'intelligence artificielle dans la vision informatique. L'idée est de montrer comment les différents réseaux de convolution peuvent être comparés en termes de précision, de temps de calcul et de nombre de paramètres. Nous passons en revue plusieurs architectures populaires et les implémentons avec les jeux de données MNIST, Fashion-MNIST et SVHN, telles que Alexnet, ResNet18, Resnet50, Resnet101, Xception et MobilnetV2. Ce faisant, nous identifions certaines de leurs forces et faiblesses et apprenons le fonctionnement interne de ceux-ci. En fin de compte, cela donne des indications pour choisir le modèle CNN adapté à une application en particulier.

Mots clés : CNN, Classification d'images, Vision par ordinateur, Performances, Architectures de CNN, Alexnet, ResNet, Xception, MobilnetV2**,** Jeux de données.

Abstract:

This thesis presents the performance comparison of different convolutional neural network (CNN) models used for image classification. This is a project related to artificial intelligence in computer vision. The idea is to show how different convolutional networks can be compared in terms of accuracy, computational time, and number of parameters. We review several popular architectures and implement them with the MNIST, Fashion-MNIST and SVHN, such as Alexnet, ResNet18, Resnet50, Resnet101, Xception and mobilnetv2.In doing so, we identify some of their strengths and weaknesses and learn their inner workings. Ultimately, this provides guidance for choosing the right CNN model for a particular application.

Key words: CNN, Image Classification, Computer Vision, Performance, CNN architectures, Alexnet, ResNet, Xception, MobilnetV2, Datasets.

Acronyme

RNA : Réseaux de Neurones Artificiels

CNN : Convolutional Neural Network

IA : Intelligence Artificielle

ML: Machine Learning

RNN: Recurrent Neural Network

SGD: Stochastic Gradient Descent

MSE: Mean Squared Error

GD: Gradient Descent

RELU: Rectified Linear Unit

TANH: Hyperbolic Tangent

ADAM: Adaptive Moment Estimation

RMSprop: Root Mean Square Propagation

OCR: Optical Character Recognition

NLP : Natural Language Processing

CAH : Classification Ascendante Hiérarchique

CHD : classification descendante hiérarchique

SVM: Support Vector Machine

K-NN: K-Nearest Neighbors

ACP : Analyse en Composantes Principales

NAS : Neural Architecture Search

FC: Fully Connected

LRN: Local Response Normalization

MNIST: Modified National Institute of Standards and Technology Database

SVHN: Street View House Numbers

Liste des figures

Liste des tableaux

Sommaire

Chapitre 2 : Classification

Chapitre 3 :Réseaux de neurone convolutionnel(CNN)

Chapitre 4 : Implémentation

Introduction générale

Introduction générale

Dans le contexte actuel de la révolution numérique, le traitement d'images occupe une place centrale dans les systèmes intelligents, grâce aux avancées de l'intelligence artificielle et de l'apprentissage profond. De nombreuses applications, telles que la reconnaissance faciale, la détection d'objets ou l'analyse médicale automatisée, reposent sur des techniques performantes de classification d'images. Ces techniques s'appuient, dans la grande majorité des cas, sur les réseaux de neurones convolutifs (CNN), devenus une solution nécessaire en vision par ordinateur. Les CNN permettent une interprétation efficace des images par les machines, en mettant en œuvre un mécanisme d'extraction automatique, hiérarchisée et optimisée des caractéristiques visuelles.

Depuis l'émergence d'AlexNet lors de la compétition ImageNet en 2012, de nombreuses architectures CNN ont été développées, telles que VGG, ResNet, Xception, DenseNet, MobileNet ou EfficientNet. Chacune de ces architectures propose des améliorations spécifiques en termes de précision, de vitesse d'exécution ou d'efficacité énergétique, notamment pour les dispositifs mobiles.

Cependant, le choix d'un modèle de CNN adapté à une tâche particulière n'est pas toujours évident, chaque architecture présentant des avantages et des limites selon le contexte d'application. C'est pourquoi nous proposons, dans le cadre de ce mémoire, une étude comparative des performances de plusieurs modèles CNN pour la classification d'images. Cette étude sera menée à l'aide de bases de données de référence telles que MNIST (reconnaissance de chiffres manuscrits), Fashion-MNIST (classification d'articles vestimentaires), et SVHN (reconnaissance de chiffres dans des images de la vie réelle). L'objectif est d'orienter les choix techniques en fonction des exigences pratiques et de la complexité des bases de données.

Description des chapitres

Ce travail est structuré en quatre chapitres :

1. Le premier chapitre présente une généralité sur les réseaux de neurones artificiels
2. Le deuxième chapitre traite de la classification, ses méthodes, l'élaboration de certains algorithmes, les critères d'évaluation d'un classificateur et les domaines d'application.
3. Le troisième aborde les modèles CNN et leurs architectures spécifiques.
4. Le quatrième chapitre est dédié à l'implémentation, où une comparaison expérimentale des modèles CNN est effectuée sur trois bases de données. Les performances sont évaluées à travers : la précision globale, la précision moyenne et le coefficient Kappa.

Chapitre 1

Généralités sur les réseaux de neurones

1.1. Introduction :

Les réseaux de neurones artificiels (RNA) sont des modèles informatiques puissants utilisés pour résoudre des problèmes complexes dans divers domaines tels que la reconnaissance d'image, la classification, la prédiction et le traitement du langage naturel. Ils reposent sur une structure composée de neurones artificiels interconnectés**,** organisés en couches successives, capables d'apprendre des relations non linéaires à partir de données brutes. Ces réseaux imitent, dans leur principe, le comportement des neurones biologiques pour apprendre automatiquement à partir d'exemples. Grâce à leur architecture flexible et leur capacité à modéliser des fonctions très complexes, les RNA sont devenus des outils incontournables dans le champ de l'intelligence artificielle et du machine learning.

L'un des principaux atouts des réseaux de neurones réside dans leur capacité à extraire automatiquement des caractéristiques pertinentes, sans qu'il soit nécessaire de définir manuellement des règles ou des descripteurs spécifiques. Cette propriété les rend particulièrement adaptés au traitement de données multidimensionnelles telles que les images, les sons ou les signaux.

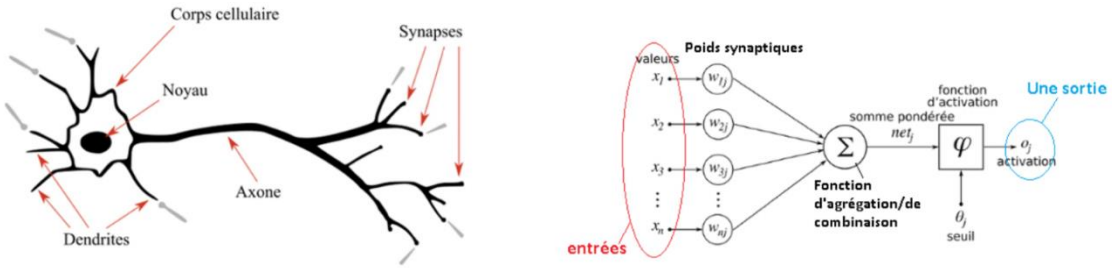
Ce chapitre vise à présenter les fondements des réseaux de neurones artificiels, en décrivant leur architecture générale, le rôle de chaque couche, le mécanisme d'apprentissage basé sur la rétropropagation, ainsi que les principales fonctions d'activation utilisées. Cette base théorique est essentielle pour comprendre les architectures plus avancées, telles que les réseaux de neurones convolutifs (CNN), abordés dans les chapitres suivants.

1.2. Définition :

Le réseau de neurones, formellement appelé « réseau de neurones artificiels » est un modèle de calcul conçue pour reproduire la manière dont le cerveau humain traite et analyse les informations. Il est constitué de nœuds reliés entre eux, structurés en plusieurs couches : la couche d'entrée, une ou plusieurs couches cachées et enfin la couche de sortie. Chaque neurone reçoit des informations, réalise des calculs puis transmet les conclusions à d'autres neurones.

Les réseaux de neurones constituent une percée technologique majeure qui a transformé le cadre de l'intelligence artificielle. S'inspirant du fonctionnement du cerveau humain, ils ont la capacité d'apprendre à partir de données, d'identifier des schémas et de réaliser des prédictions précises. Les réseaux de neurones, adoptés par une variété d'entités allant des grands noms de la technologie aux institutions financières, ont démontré leur utilité dans divers domaines tels que la santé, la finance et la technologie.

Le réseau de neurones est un outil d'apprentissage performant, apte à identifier des schémas et des tendances dans les données, et à exploiter ces renseignements pour réaliser des prédictions ou prendre des décisions. De nombreuses percées en intelligence artificielle reposent sur le réseau neuronal, qui demeure un champ de recherche actif.[1]



a). RN biologique b). RN artificial

Figure 1.1 : Définition d'un réseau de neurones

Tableau 1.1 : Correspondance neurone biologique/neurone artificiel

Neurone biologique	Neurone artificiel (formel)
Axones	Signal d'entrée
Synapses	Poids de la connexion
dendrites	Signal de sortie

1.3. Historique :

L'origine des réseaux de neurones artificiels remonte aux années 1940, grâce aux contributions pionnières de Warren McCulloch et Walter Pitts en 1943. Ils ont présenté le premier modèle mathématique simplifié du neurone biologique, qu'ils ont nommé neurone formel. Ce modèle puise son inspiration dans le fonctionnement des neurones biologiques et jette les fondations de la simulation informatique des réseaux de neurones

En 1949, Donald Hebb, neuropsychologue canadien, présente une théorie majeure connue sous le nom de « règle de Hebb ». Cette dernière explique comment les liaisons synaptiques entre neurones peuvent évoluer en fonction de leur activité, introduisant par conséquent un processus d'apprentissage.

Frank Rosenblatt a créé le perceptron en 1958, qui est le premier réseau de neurones artificiels opérationnel capable d'apprendre de l'expérience et de classer les données, représentant ainsi

une avancée significative dans l'évolution des réseaux de neurones. Ce modèle comprend une couche d'entrée et une couche de sortie, et il a bénéficié d'une importante couverture médiatique à l'époque.

Dans les années 80, le physicien John Hopfield ravive l'intérêt pour les réseaux de neurones grâce à son modèle de réseau récurrent, offrant une nouvelle vision sur la dynamique des réseaux neuronaux. Par la suite, d'importants progrès ont été accomplis avec les réseaux à plusieurs couches, les réseaux de neurones convolutifs, et l'apparition du deep learning à partir des années 2000, en particulier grâce aux contributions de Yann LeCun.[2]

1.4. Différence entre intelligence artificielle, apprentissage automatique et réseaux de neurones :

Les réseaux de neurones sont une composante de l'intelligence artificielle (IA), un champ d'étude qui englobe toutes les méthodes permettant aux machines de reproduire l'intelligence humaine. Le Machine Learning (ML), qui est un sous-secteur de l'IA, offre aux machines la capacité d'apprendre à partir des données sans nécessiter une programmation explicite.

En revanche, les Réseaux de Neurones Artificiels (RNA) représentent une méthode particulière d'apprentissage automatique inspirée du mode de fonctionnement du cerveau humain. Dans ce cadre, des neurones artificiels reliés entre eux traitent les informations par le biais de l'ajustement des poids synaptiques. À l'opposé des modèles de Machine Learning traditionnels qui s'appuient fréquemment sur des algorithmes statistiques tels que les arbres de décision ou les régressions, les réseaux de neurones se révèlent particulièrement performants pour la gestion de données complexes et non structurées telles que les images, le texte ou l'audio. On désigne souvent comme Deep Learning une forme avancée du Machine Learning, qui se caractérise par l'utilisation de multiples couches cachées et d'un grand nombre de paramètres. Ceci permet d'atteindre des résultats remarquables dans des domaines tels que la vision informatique et le traitement du langage naturel.

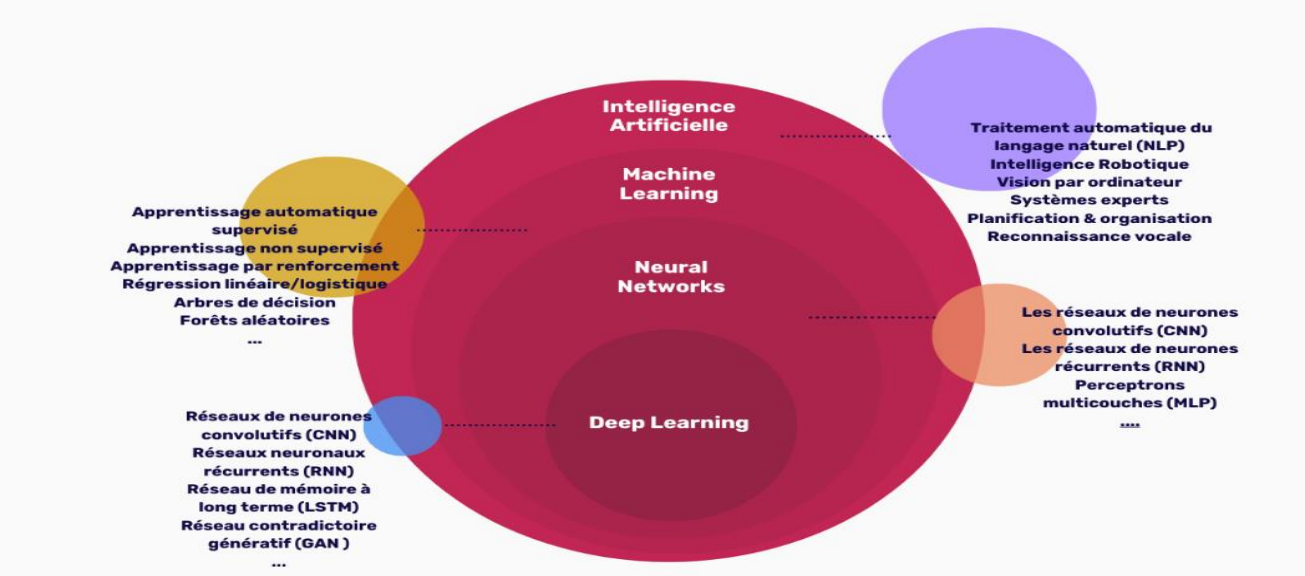


Figure 1.2 : Intelligence artificielle, apprentissage automatique et réseaux de neurones.

1.5. Architecture des réseaux de neurones :

1.5.1. Neurone artificiel

Un neurone artificiel est une composante essentielle des réseaux de neurones, modelée sur le principe de fonctionnement des neurones biologiques. Il reçoit des données, chaque donnée étant liée à un poids qui indique son importance dans le processus de traitement de l'information. Ces entrées sont multipliées par leurs coefficients respectifs, puis cumulées pour établir un total pondéré. Par la suite, cette somme est soumise à une fonction d'activation, ce qui détermine la sortie du neurone. Ce mécanisme d'activation est essentiel en apportant de la non-linéarité au sein du modèle, autorisant ainsi les réseaux neuronaux à saisir des liens complexes entre les données et à acquérir des représentations plus élaborées.

1.5.2. Couches d'un réseau de neurones

Un réseau de neurones est structuré en plusieurs couches, chacune jouant un rôle spécifique dans le traitement des données.

1. Couche d'Entrée (Input Layer) : L'entrée du réseau est chargée de recevoir les données initiales destinées à être traitées par le système. Habituellement, chaque neurone dans cette couche est associé à une variable d'entrée du problème à traiter. Par exemple, dans un cas de détection d'images, chaque neurone de la couche initiale pourrait symboliser un pixel de l'image.[3]
2. Couches Cachées (Hidden Layers) : Les couches cachées se trouvent entre la couche d'entrée et la couche de sortie. On les nomme « cachées » parce qu'elles n'ont aucun lien direct avec l'extérieur. Ces couches réalisent la majorité du traitement des données, en

convertissant les entrées en représentations plus sophistiquées et abstraites. La quantité de couches dissimulées peut fluctuer en fonction de la complexité du problème à traiter et des capacités de calcul disponibles.[3]

3. Couche de Sortie (Output Layer) : La couche de sortie génère les résultats finaux issus du traitement réalisé par le réseau. Le nombre de neurones dans cette couche est déterminé par le nombre de classes ou de résultats attendus dans le contexte du problème. Par exemple, dans le cas d'un problème de classification binaire, deux neurones seraient présents en sortie, alors que pour une classification multi-classe, le nombre de neurones correspondrait au nombre de classes possibles.[3]

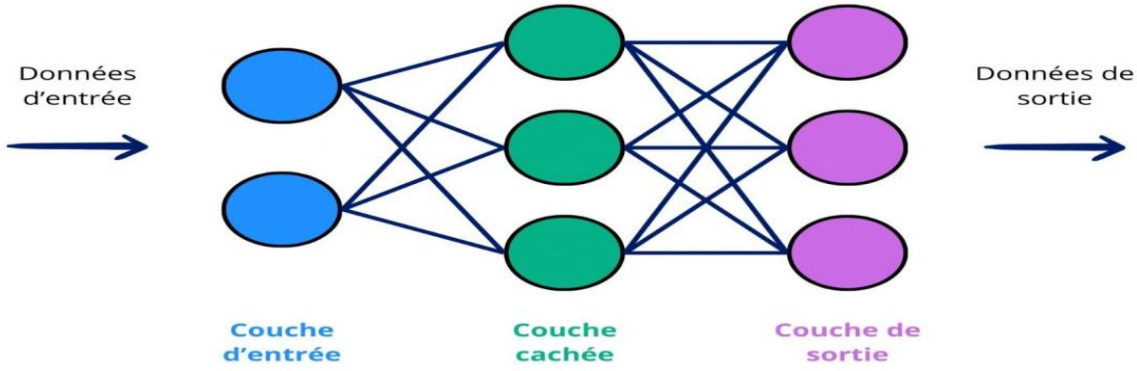


Figure 1.3 : Architecture des réseaux de neurones

1.5.3. Connexions et poids synaptiques

Les connexions et les poids synaptiques sont des éléments fondamentaux dans les réseaux de neurones artificiels, inspirés par le fonctionnement des neurones biologiques.

1. Connexions Synaptiques

Dans un réseau de neurones artificiels, les connexions synaptiques représentent les liens entre les neurones. Ces liens permettent aux neurones de communiquer entre eux en transmettant des signaux. Dans les réseaux artificiels, ces connexions sont souvent modélisées par des poids numériques qui influencent la force du signal transmis entre les neurones.

2. Poids Synaptiques

Les poids synaptiques sont des coefficients associés à chaque connexion entre les neurones. Ils déterminent l'importance du signal transmis d'un neurone à l'autre. Pendant l'apprentissage du réseau, ces poids sont ajustés pour minimiser l'erreur entre les prédictions du modèle et les sorties attendues. Cet ajustement permet au réseau de mieux apprendre à partir des données et d'améliorer ses performances.

3. Fonctionnement

Dans un neurone artificiel, les entrées sont multipliées par leurs poids respectifs, puis la somme de ces produits est calculée. Cette somme est ensuite passée à travers une fonction d'activation pour déterminer la sortie du neurone. Les poids synaptiques jouent donc un rôle crucial dans ce processus, car ils influencent directement la sortie du neurone.

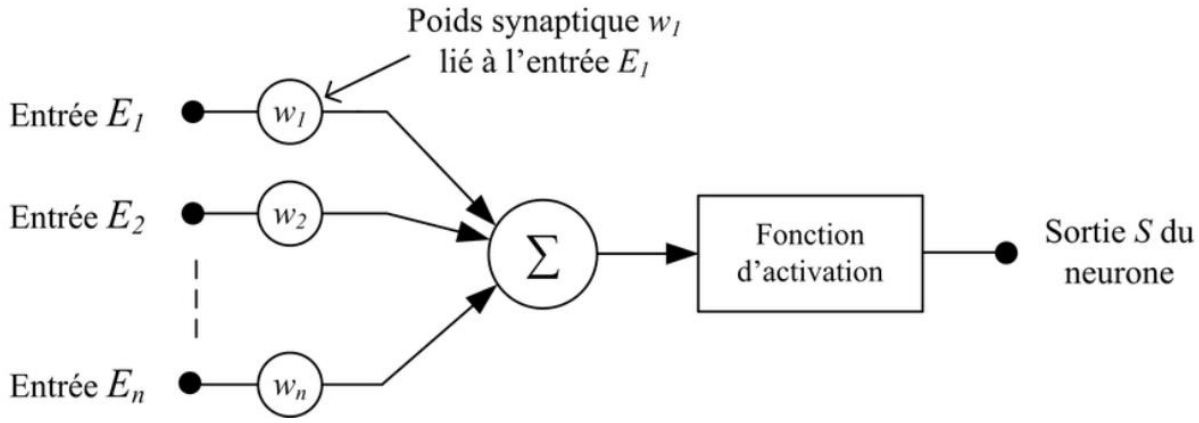


Figure 1.4 : Structure d'un neurone artificiel

4. Synapses Artificielles

Les chercheurs ont développé des synapses artificielles pour simuler le comportement des synapses biologiques. Ces synapses utilisent des composants électroniques comme les memristors, qui peuvent ajuster leur résistance en fonction des impulsions électriques reçues, simulant ainsi la plasticité synaptique des neurones biologiques.

1.6. Types de réseaux de neurones :

Les réseaux de neurones artificiels sont classés en plusieurs types en fonction de leur architecture et de leur mode de fonctionnement.

1.6.1. Le perceptron : est un algorithme d'apprentissage automatique développé pour la classification binaire. Il s'agit d'une simplification d'un neurone biologique, conçue pour imiter la manière dont le cerveau traite l'information

1.6.2. Réseaux de Neurones Feedforward (Feedforward Neural Networks)

Ces réseaux traitent les données dans une seule direction, de l'entrée vers la sortie, sans boucle. Chaque nœud d'une couche est connecté à chaque nœud de la couche suivante. Ils sont souvent utilisés pour des tâches de classification simples.[4]

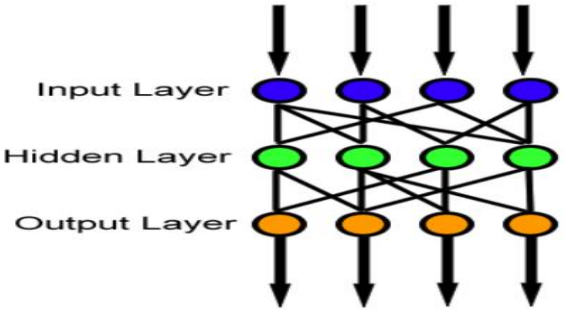


Figure 1.5 : Réseaux de Neurones Feedforward

1.6.3. Réseaux de Neurones Récurrents (Recurrent Neural Networks, RNN)

Les RNN utilisent le contexte des entrées lors du calcul de la sortie. La sortie dépend des entrées et des sorties calculées précédemment, ce qui les rend adaptés aux applications traitant des séquences temporelles, comme la prédiction de séries chronologiques ou le traitement du langage naturel.[4]

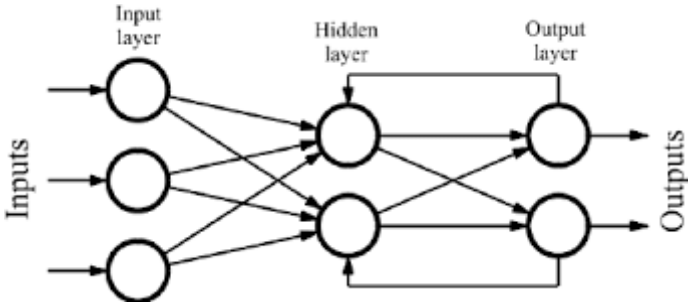


Figure 1.6 : Réseaux de Neurones Récurrents

1.6.3. Réseaux de Neurones Convolutionnels (Convolutional Neural Networks, CNN)

Les CNN sont spécialisés dans le traitement des images. Ils utilisent des couches de convolution et de pooling pour extraire des caractéristiques spatiales, ce qui les rend idéaux pour la reconnaissance visuelle et l'analyse d'images.[4]

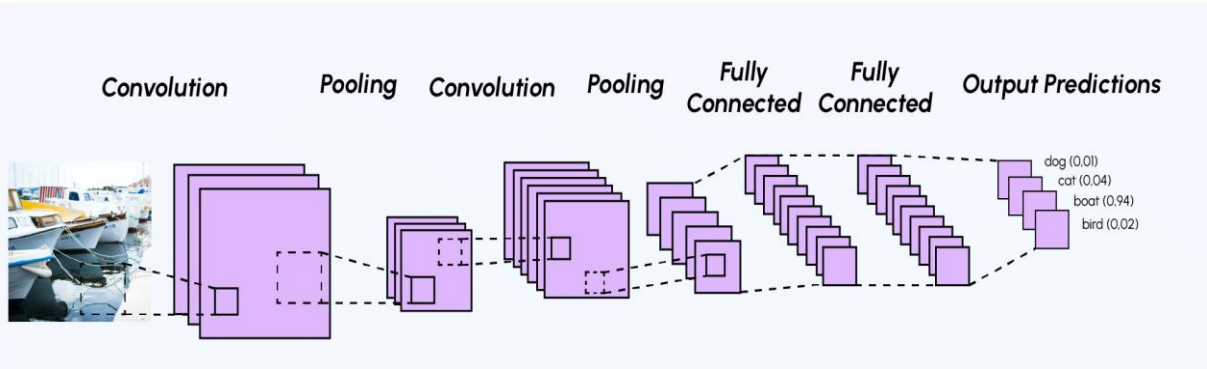


Figure 1.7 : Réseaux de Neurones Convolutionnels

1.6.5. Réseaux de Neurones Résiduels (residual neural network)

Ils permettent d'ignorer certaines données en utilisant des connexions de saut (skip connections), ce qui aide à éviter le surapprentissage et à améliorer la convergence.

1.7. Apprentissage des réseaux de neurones :

1.7.1. Propagation avant (Forward Propagation)

La propagation avant (ou forward propagation) est un processus fondamental dans les réseaux de neurones à propagation avant (feedforward neural networks).

1. Définition

La propagation avant est le mécanisme par lequel les données d'entrée sont traitées et transmises à travers les couches successives d'un réseau de neurones, depuis la couche d'entrée jusqu'à la couche de sortie, sans boucle ni rétroaction.[5]

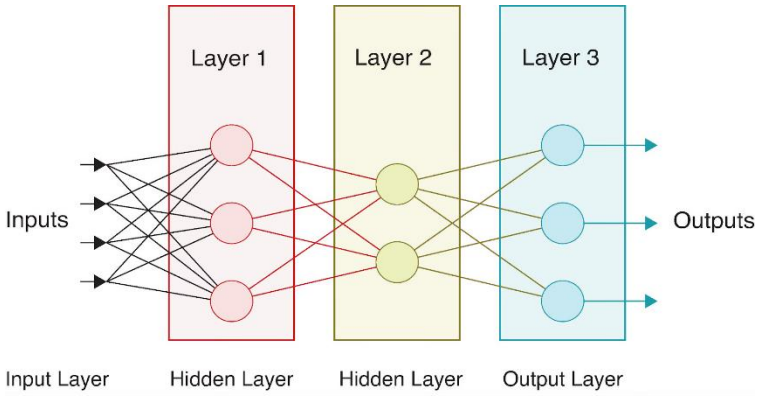


Figure 1.8 : Propagation vers l'avant

2. Fonctionnement
   - Entrée : Les données sont présentées à la couche d'entrée.
   - Traitement : Chaque neurone de la couche d'entrée transmet ses sorties aux neurones de la première couche cachée.
   - Propagation : Les données passent à travers chaque couche cachée, où chaque neurone applique une fonction d'activation à la somme pondérée des entrées qu'il reçoit.
   - Sortie : La sortie finale est générée par la couche de sortie.
3. Caractéristiques
   - Direction Unidirectionnelle : L'information ne se déplace que dans une seule direction, sans cycles ou boucles.
4. Utilisation

La propagation avant est utilisée pour des tâches de classification, de régression et de reconnaissance de schémas.

1.7.2. Propagation arrière et descente de gradient (Backpropagation & Gradient Descent)

Sont deux concepts fondamentaux dans l'entraînement des réseaux de neurones artificiels.

1. Propagation Arrière (Backpropagation)

La propagation arrière est une méthode utilisée pour entraîner les réseaux de neurones. Elle permet de calculer les gradients de la fonction de perte par rapport aux poids du réseau, ce qui est essentiel pour ajuster ces poids et minimiser l'erreur entre les prédictions du modèle et les sorties attendues.[6]

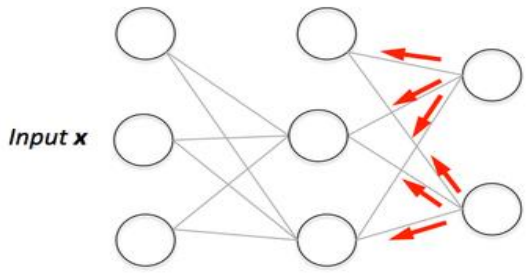


Figure 1.9 : Propagation vers l'arrière

Étapes de la Propagation Arrière :

- Propagation Avant : Les données d'entrée traversent le réseau couche par couche jusqu'à la couche de sortie, où une prédiction est faite.
- Calcul de l'Erreur : La différence entre la prédiction et la valeur réelle (l'erreur) est calculée.
- Propagation Arrière : Cette erreur est propagée en sens inverse à travers le réseau. Les gradients de la fonction de perte par rapport aux poids sont calculés à chaque couche.
- Mise à Jour des Poids : Les poids sont ajustés en fonction des gradients calculés pour minimiser l'erreur.

2. Descente de Gradient (Gradient Descent)

La descente de gradient est une méthode d'optimisation utilisée pour ajuster les poids du réseau de neurones. Elle consiste à déplacer les poids dans la direction opposée au gradient de la fonction de perte, ce qui permet de minimiser cette fonction et donc l'erreur globale du modèle.[6]

Types de Descente de Gradient :

- Descente de Gradient Stochastique (SGD) : Utilise une seule donnée à la fois pour calculer le gradient.

- Descente de Gradient par Lots (Mini-Batch Gradient Descent) : Utilise un sous-ensemble des données pour calculer le gradient.
- Descente de Gradient Complète (Batch Gradient Descent) : Utilise toutes les données pour calculer le gradient.

3. Relation Entre Propagation Arrière et Descente de Gradient

La propagation arrière est utilisée pour calculer les gradients nécessaires à la descente de gradient. En d'autres termes, la propagation arrière détermine la direction dans laquelle les poids doivent être ajustés pour minimiser l'erreur, tandis que la descente de gradient effectue réellement cette mise à jour en déplaçant les poids dans cette direction.

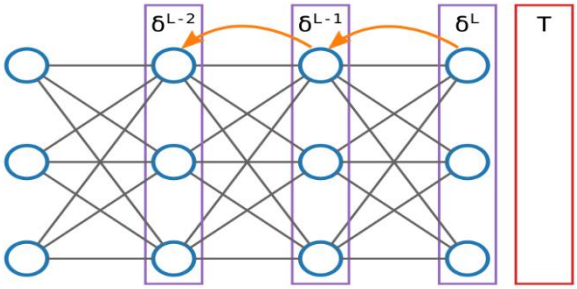


Figure 1.10 :La rétropropagation du gradient

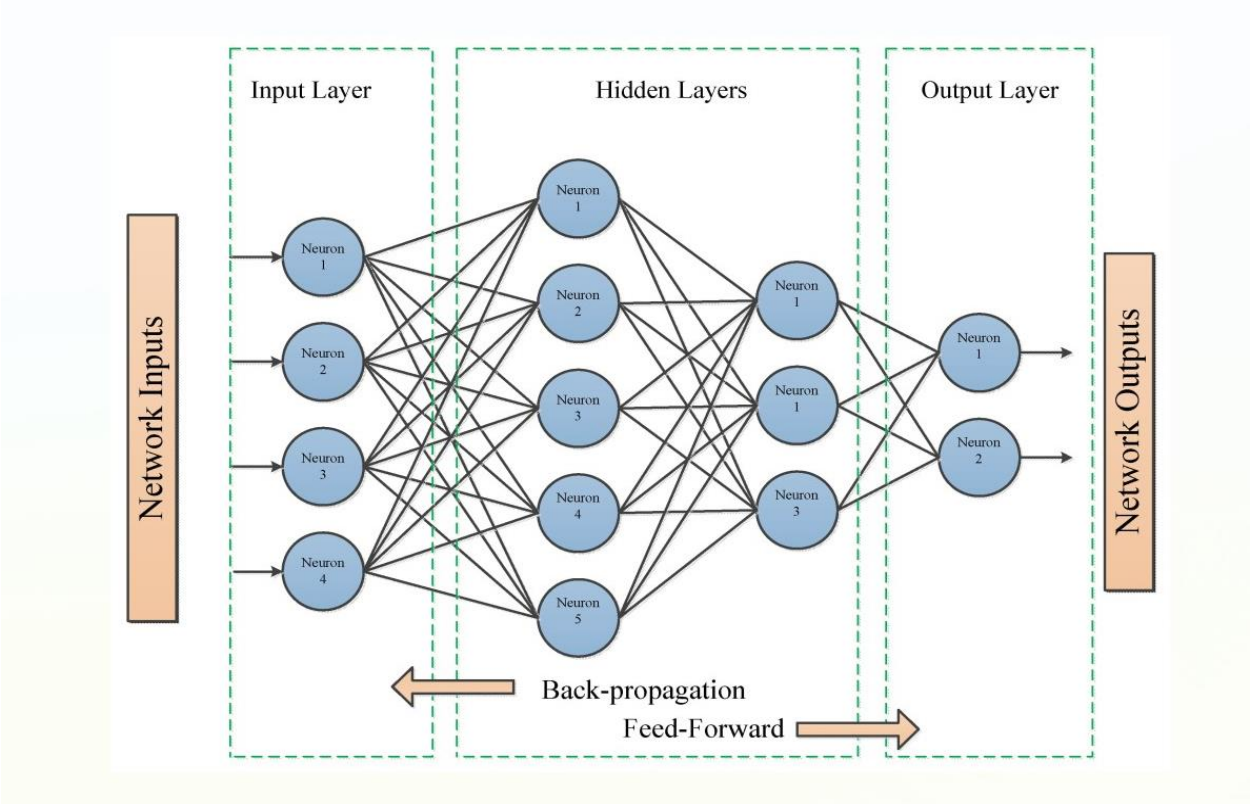


Figure 1.11: Apprentissage des réseaux de neurones

1.7.3. Fonction de coût et optimisation

La fonction de coût et l'optimisation sont des éléments fondamentaux dans l'entraînement des réseaux de neurones artificiels.

1. Fonction de Coût

La fonction de coût, également appelée fonction de perte, est une mesure de l'erreur entre les prédictions du modèle et les sorties attendues. Elle est utilisée pour évaluer les performances du réseau et guider l'optimisation des paramètres.

Exemples de Fonctions de Coût

- **Erreur Quadratique Moyenne (MSE)** : Utilisée pour les problèmes de régression, elle calcule la moyenne des carrés des erreurs entre les prédictions et les valeurs réelles.

  Formule :

$$\boldsymbol{J(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \qquad (\mathbf{1.1})$$

  Où : y_i est la valeur réelle, $\hat{y}_i$ la prédiction du modèle, et N le nombre d'exemples

- **Entropie Croisée** : Utilisée pour les problèmes de classification, elle mesure la différence entre les distributions de probabilité prédites et réelles.

2. Optimisation

L'optimisation dans les réseaux de neurones consiste à ajuster les paramètres (poids et biais) pour minimiser la fonction de coût. Les algorithmes d'optimisation les plus courants incluent :

- **Descente de Gradient (GD)** : Utilise le gradient de la fonction de coût pour ajuster les paramètres dans la direction qui minimise l'erreur.

  Formule :

$$\boldsymbol{\theta := \theta - \alpha \nabla_\theta J(\theta)} \qquad (\mathbf{1.2})$$

  Où : α est le taux d'apprentissage, et $\nabla_\theta J(\theta)$ le gradient de la fonction de coût par rapport aux paramètres.

- **Descente de Gradient Stochastique (SGD)** : Une variante de la descente de gradient qui utilise une seule donnée à la fois pour calculer le gradient, ce qui accélère l'apprentissage.

  Formule :

$$\boldsymbol{\theta := \theta - \alpha \nabla_\theta J\left(\theta, x^{(i)}, y^{(i)}\right)} \qquad (\mathbf{1.3})$$

- **Descente de Gradient par Lots (Mini-Batch GD)** : Utilise un sous-ensemble des données pour calculer le gradient, offrant un compromis entre la précision et la vitesse.

  Formule :

$$\boldsymbol{\theta} := \boldsymbol{\theta} - \boldsymbol{\alpha}\frac{\mathbf{1}}{\boldsymbol{m}}\sum_{\boldsymbol{j=1}}^{\boldsymbol{m}} \boldsymbol{\nabla_{\theta} J\left(\theta, x^{(i)}, y^{(i)}\right)} \qquad (\mathbf{1.4})$$

Où : m est la taille du mini-batch.

3. Régularisation

La régularisation est une technique utilisée pour éviter le surapprentissage en ajoutant un terme supplémentaire à la fonction de coût. Ce terme pénalise les poids élevés, favorisant ainsi des modèles plus simples et plus généralisables.

La fonction de coût est essentielle pour évaluer les performances du réseau, tandis que l'optimisation ajuste les paramètres pour minimiser cette fonction. Les algorithmes d'optimisation, tels que la descente de gradient, sont utilisés pour atteindre cet objectif.

1.8. Fonction d'activation :

➢ Définition :

Une fonction d'activation est une fonction mathématique appliquée à la somme pondérée des entrées d'un neurone, plus le biais. Elle détermine si le neurone doit être activé ou non en fonction du seuil de stimulation atteint.[7]

1.8.1. Sigmoïde :

La fonction d'activation sigmoïde, également connue sous le nom de fonction logistique, est une des fonctions d'activation les plus anciennes et les plus largement utilisées dans les réseaux de neurones artificiels.

1. Définition et Formule :

La fonction sigmoïde est définie par la formule suivante :

$$\boldsymbol{f(x) = \frac{1}{1 + e^{-x}}} \qquad (\mathbf{1.5})$$

Elle prend une valeur réelle en entrée et la réduit à une valeur comprise entre 0 et 1. La courbe de la fonction sigmoïde est en forme de "S", asymptotique à 0 pour les grandes valeurs négatives et à 1 pour les grandes valeurs positives.

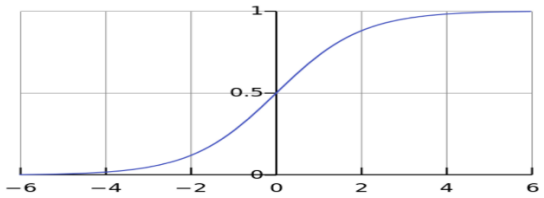


Figure 1.12 : Représentation graphique de la fonction sigmoïde

2. Utilisation :

La fonction sigmoïde est principalement utilisée dans les couches de sortie des modèles de classification binaire. Elle permet de transformer les sorties en probabilités, ce qui est très utile pour interpréter les résultats en termes de probabilité d'appartenance à une classe particulière.

3. Avantages

- Interprétation des Probabilités : Les sorties de la fonction sigmoïde peuvent être directement interprétées comme des probabilités, ce qui est avantageux pour les problèmes de classification binaire.
- Continuité et Différentiabilité : La fonction sigmoïde est continue et différentiable partout, ce qui facilite l'optimisation basée sur les gradients lors de l'apprentissage des réseaux de neurones.

4. Inconvénients

- Problème de la Disparition du Gradient : La fonction sigmoïde souffre du problème de la disparition du gradient, où les gradients deviennent très petits pour les valeurs d'entrée élevées ou très faibles. Cela peut ralentir l'apprentissage dans les réseaux profonds.
- Saturation : La fonction sigmoïde sature à 0 ou 1 pour les grandes valeurs négatives ou positives, ce qui peut limiter la capacité du réseau à apprendre des relations complexes.

1.8.2. ReLU (Rectified Linear Unit) :

La fonction d'activation ReLU (Rectified Linear Unit) est une des fonctions d'activation les plus couramment utilisées dans les réseaux de neurones artificiels, en particulier dans les réseaux profonds.

1. Définition

La fonction ReLU est définie par la formule suivante :

$$\boldsymbol{f(x) = \max(0, x)} \qquad \boldsymbol{(1.6)}$$

Elle renvoie la valeur d'entrée x si elle est positive ou nulle, et zéro si elle est négative.

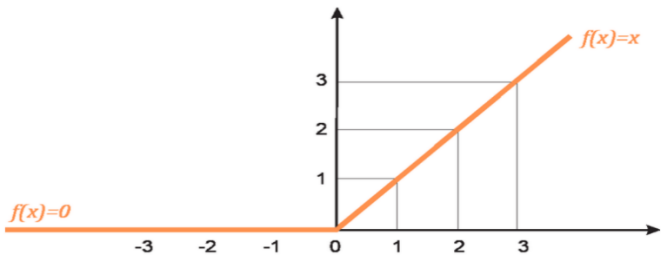


Figure 1.13 :La représentation graphique de la fonction d'activation ReLU

2. Fonctionnement

La ReLU joue un rôle crucial dans les réseaux de neurones en introduisant la non-linéarité nécessaire pour modéliser des relations complexes entre les données. Elle permet aux neurones de se concentrer sur les caractéristiques positives des données, en mettant à zéro les valeurs négatives.

3. Avantages
    - Simplicité de Calcul : La ReLU est facile à calculer et à dériver, ce qui accélère l'entraînement des réseaux.
    - Prévention de la Saturation des Gradients : Contrairement à la fonction sigmoïde, ReLU évite le problème de disparition des gradients, permettant une convergence plus rapide dans les réseaux profonds.
    - Efficacité Computationnelle : En mettant à zéro les valeurs négatives, ReLU réduit le nombre de calculs nécessaires, ce qui améliore l'efficacité computationnelle.
4. Inconvénients
    - Neurones "Morts" : Lorsque la sortie d'un neurone est constamment négative, la ReLU peut le "tuer" en mettant toujours sa sortie à zéro, ce qui empêche l'ajustement des poids pendant la rétropropagation.
    - Non-Différentiable en 0 : Bien que la ReLU soit facile à dériver, elle n'est pas différentiable en 0, ce qui peut causer des problèmes dans certains algorithmes d'optimisation.
5. Variations

Pour pallier les inconvénients de la ReLU, des variantes ont été développées, telles que la Leaky ReLU**.** Cette variante introduit un petit coefficient pour les entrées négatives, permettant ainsi aux neurones de ne jamais être complètement "morts".

- La fonction Leaky ReLU :

L'unité linéaire rectifiée à fuite, ou ReLU à fuite, est un type de fonction d'activation basée sur une ReLU, mais elle présente une faible pente pour les valeurs négatives au lieu d'une pente plate. Le coefficient de pente est déterminé avant l'apprentissage, c'est-à-dire qu'il n'est pas appris pendant l'apprentissage. Ce type de fonction d'activation est courant dans les tâches où les gradients peuvent être épars.

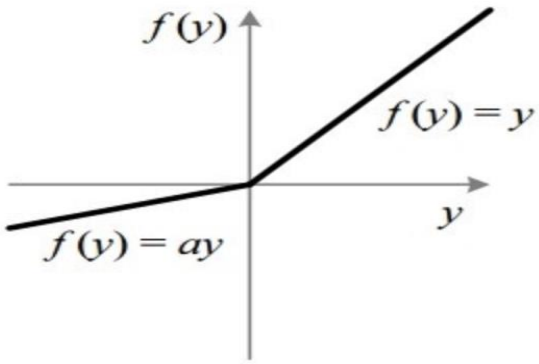


Figure 1.14 : La représentation graphique de la fonction d'activation Leaky ReLU.

1.8.3. La fonction Tanh :

La fonction d'activation Tanh (Tangente Hyperbolique) est une fonction mathématique couramment utilisée dans les réseaux de neurones artificiels.

1. Définition et Formule

La fonction Tanh est définie par la formule suivante :

$$\frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \qquad (1.7)$$

Elle produit une sortie comprise entre -1 et 1, ce qui la rend similaire à la fonction sigmoïde mais avec une plage différente.

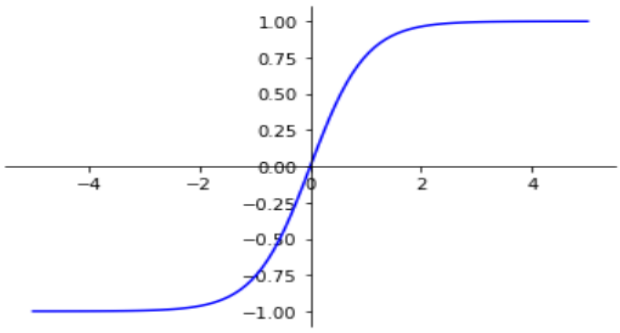


Figure 1.15 : La représentation graphique de la fonction d'activation Tanh

2. Caractéristiques

- Plage de Sortie : La fonction Tanh produit des valeurs comprises entre -1 et 1, ce qui permet de traiter efficacement les valeurs négatives et positives.
- Centrage sur Zéro : Contrairement à la sigmoïde, la Tanh est centrée sur zéro, ce qui signifie que sa sortie est symétrique par rapport à l'origine. Cela peut aider à accélérer la convergence lors de l'apprentissage des réseaux neuronaux.

- Gradients Plus Forts : Les gradients de la fonction Tanh sont généralement plus forts que ceux de la sigmoïde, ce qui peut améliorer la vitesse d'apprentissage et réduire le problème de disparition des gradients.

3. Utilisation

La fonction Tanh est souvent utilisée dans les réseaux de neurones récurrents (RNN) et peut être préférée à la sigmoïde dans certains cas en raison de sa symétrie et de sa capacité à traiter les valeurs négatives plus efficacement.

4. Avantages et Inconvénients

- Avantages : Centrage sur zéro, gradients plus forts, et traitement efficace des valeurs négatives.
- Inconvénients : Peut souffrir du problème de disparition des gradients, bien que moins que la sigmoïde

1.8.4. Softmax :

La fonction d'activation Softmax est une fonction mathématique utilisée principalement dans les couches de sortie des réseaux de neurones pour les problèmes de classification multi-classes.

1. Définition et Formule :

La fonction Softmax est définie par la formule suivante :

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}} \qquad (1.8)$$

Où x_i est le i-ième élément du vecteur d'entrée, et k est le nombre total de classes.

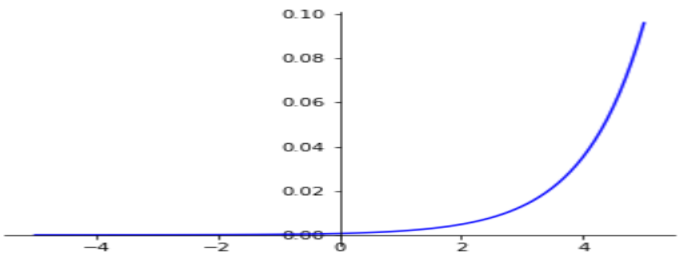


Figure 1.16 : La représentation graphique de la fonction d'activation Softmax

2. Fonctionnement

La Softmax prend un vecteur de valeurs brutes (appelées logits) et le transforme en un vecteur de probabilités. Chaque élément de la sortie représente la probabilité que l'entrée appartienne à une classe particulière. La somme des probabilités est toujours égale à 1, ce qui permet d'interpréter les sorties comme des distributions de probabilité.

3. Utilisation

La fonction Softmax est essentiellement utilisée dans les tâches de classification multi-classes, où plusieurs classes sont possibles. Elle est couramment employée dans des applications telles que la reconnaissance d'images, la traduction automatique et le traitement du langage naturel.

4. Avantages

- Interprétation des Probabilités : Les sorties de la Softmax peuvent être directement interprétées comme des probabilités, ce qui est utile pour évaluer la confiance du modèle dans ses prédictions.
- Flexibilité : La Softmax peut être appliquée à divers types de données, y compris les images, le texte et d'autres formes de données multimédia.
- Précision des Prédictions : En normalisant les logits en probabilités, la Softmax aide à faire des prédictions plus précises et fiables.

5. Inconvénients

Sensibilité aux Valeurs Extrêmes : La Softmax peut amplifier les différences dans les entrées, ce qui peut rendre le modèle trop confiant si une valeur est très élevée par rapport aux autres.[8]

1.9. Entraînement et optimisation :

L'entraînement et l'optimisation des réseaux de neurones impliquent plusieurs techniques clés pour améliorer les performances et éviter le surapprentissage. Voici une description détaillée des algorithmes d'optimisation, des techniques de régularisation et des problèmes de surapprentissage :

1.9.1-Algorithmes d'Optimisation

1. Descente de Gradient Stochastique (SGD) : Utilise une seule donnée à la fois pour calculer le gradient, ce qui accélère l'apprentissage mais peut être instable.
2. Adam : Une variante de SGD qui ajuste dynamiquement le taux d'apprentissage pour chaque paramètre, ce qui améliore la stabilité et la vitesse de convergence.

Étapes de l'algorithme Adam :

➢ Calcule de gradient :

$$\boldsymbol{g_t = \Delta_\theta L(\theta_t)} \qquad \mathbf{(1.9)}$$

Où g_t est le gradient de la fonction de perte L par rapport aux paramètres à l'étape t.

➢ Mise à jour de l'estimation du premier moment corrigée du biais :

$$\boldsymbol{m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t} \qquad \mathbf{(1.10)}$$

m_t Représente la moyenne mobile des gradients et β est le taux de décroissance de cette moyenne mobile.

- Mise à jour de l'estimation du deuxième moment brut corrigée du biais :

$$\boldsymbol{v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2} \quad \textbf{(1.11)}$$

Où : v_t est la moyenne mobile des gradients au carré et β est le taux de décroissance des gradients au carré.

- Calculer les estimations de moment corrigées du biais :

$$\boldsymbol{\hat{m}_t = \frac{m_t}{1-\beta_1^t}}, \boldsymbol{\hat{v}_t = \frac{v_t}{1-\beta_2^t}} \quad \textbf{(1.12)}$$

Ces corrections de biais aident à ajuster le biais d'initialisation vers zéro.

- Paramètres de mise à jour :

$$\boldsymbol{\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t} \quad \textbf{(1.13)}$$

Où η est la taille du pas (taux d'apprentissage) et ϵ est un petit nombre ajouté pour éviter la division par zéro.[8]

3. RMSprop : Similaire à Adam, mais il ajuste le taux d'apprentissage en fonction de la moyenne des carrés des gradients passés, ce qui aide à éviter les oscillations.

Étapes de l'algorithme RMSprop :[9]

Soit θ_t les paramètres à l'itération t, $\boldsymbol{g_t = \Delta_\theta L(\theta_t)}$ le gradient.

- Initialiser la moyenne mobile des carrés des gradients :$\boldsymbol{E[g^2]_0 = 0}$

Choisir le taux d'apprentissage α, le facteur de décroissance β (typiquement 0.9), et ϵ petite constante pour la stabilité.

- À chaque itération t :

Calculer le gradient g_t.

Mettre à jour la moyenne mobile des carrés des gradients :

$$\boldsymbol{E[g^2]_t = \beta E[g^2]_{t-1} + (1-\beta) g_t^2} \quad \textbf{(1.14)}$$

Mettre à jour les paramètres :

$$\boldsymbol{\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{E[g^2]_t} + \epsilon} g_t} \quad \textbf{(1.15)}$$

1.9.2 -Techniques de Régularisation

1. Dropout : Désactive aléatoirement des neurones pendant l'entraînement pour éviter que le réseau ne dépende trop fortement d'un petit nombre de neurones.

2. Batch Normalization : Normalise les entrées de chaque couche pour réduire l'effet de la disparité des échelles entre les couches, ce qui accélère l'apprentissage et améliore la stabilité.

1.9.3-Problèmes de Surapprentissage (Overfitting)

Le surapprentissage se produit lorsque le modèle est trop bien adapté aux données d'entraînement mais ne généralise pas bien aux nouvelles données. Les techniques de régularisation, comme le dropout et la normalisation par lots, aident à prévenir ce problème en favorisant des modèles plus simples et plus généralisables.

L'optimisation des réseaux de neurones implique le choix d'algorithmes appropriés pour ajuster les paramètres, l'utilisation de techniques de régularisation pour éviter le surapprentissage, et une gestion attentive des hyperparamètres pour améliorer les performances du modèle.

1.10. Applications des réseaux de neurones :

1.10.1. Vision par ordinateur

Les réseaux de neurones ont de nombreuses applications dans divers domaines, notamment dans la vision par ordinateur. Voilà quelques exemples d'applications des réseaux de neurones dans ce domaine et au-delà :

- Reconnaissance d'Images : Les réseaux de neurones convolutionnels (CNN) sont largement utilisés pour la reconnaissance d'images, la détection d'objets et la classification d'images.
- Reconnaissance Optique de Caractères (OCR) : Utilisée pour lire et interpréter les textes écrits à la main ou imprimés, comme dans les applications de tri du courrier.
- Déplacement Automatisé de Robots : Les réseaux de neurones aident à guider les robots en leur permettant de reconnaître et d'interpréter leur environnement visuel.

1.10.2. Traitement du Langage Naturel (NLP)

- Traduction Automatique : Les modèles de réseaux de neurones, comme les Transformers, sont utilisés pour améliorer la précision des traductions en comprenant les relations complexes entre les mots dans différentes langues.
- Résumé Automatique : Les réseaux de neurones peuvent condenser des textes en identifiant les informations principales, ce qui est utile pour les résumés d'articles ou les assistants numériques.
- Chatbots et Assistants Virtuels : Les chatbots utilisent le NLP pour interagir avec les utilisateurs via des conversations en langage naturel, comme Siri et Alexa.

- Analyse de Sentiments : Les réseaux de neurones, tels que RoBERTa, analysent les sentiments exprimés dans les textes, capturant même des nuances comme le sarcasme.

1.10.3. Reconnaissance vocale

- Assistants Virtuels : Les assistants comme Siri, Alexa et Google Assistant utilisent la reconnaissance vocale pour interagir avec les utilisateurs. Ils permettent de contrôler des appareils domestiques, de planifier des événements, et de rechercher des informations sur Internet.
- Logiciels de Reconnaissance Vocale : Des logiciels comme Dragon Professional et Google Docs permettent de transcrire des documents vocalement, ce qui est utile pour améliorer la productivité et l'accessibilité.
- Objets Connectés : La reconnaissance vocale est intégrée dans de nombreux objets connectés pour contrôler l'éclairage, la température, et la musique avec des commandes vocales.
- Traduction en Temps Réel : Les systèmes de reconnaissance vocale peuvent traduire des conversations en temps réel, facilitant la communication entre personnes parlant différentes langues.
- Secteur Bancaire : Les paiements vocaux sécurisés sont une application croissante, offrant une expérience client simplifiée et sécurisée.

➢ Techniques Utilisées

Les réseaux de neurones, notamment les réseaux profonds, sont essentiels pour la reconnaissance vocale. Ils apprennent à reconnaître les schémas vocaux à partir d'échantillons sonores et peuvent être combinés avec d'autres techniques comme les modèles de Markov cachés et les n-grammes pour améliorer la précision.

1.10.4. Prédiction et classification

1. Prédiction

- Prévisions Financières : Les réseaux de neurones sont utilisés pour prédire les fluctuations des marchés boursiers, évaluer les risques de crédit et détecter les entreprises en difficulté.
- Séries Temporelles : Ils sont employés pour prédire les tendances futures dans les données temporelles, comme les prévisions météorologiques ou les ventes futures.
- Énergie : Les réseaux de neurones aident à prédire la demande énergétique et à optimiser la production d'énergie renouvelable.

2. Classification
    - Classification d'Images : Les réseaux de neurones convolutionnels (CNN) sont très efficaces pour la classification d'images, comme dans la reconnaissance d'objets ou la détection de maladies à partir d'images médicales.
    - Traitement du Langage Naturel (NLP) : Les réseaux de neurones sont utilisés pour la classification de textes, l'analyse de sentiments et la traduction automatique.
    - Diagnostic Médical : Ils aident à classifier les images médicales pour diagnostiquer des maladies, comme le cancer ou les maladies cardiaques.

➢ Techniques Utilisées :

- Apprentissage Supervisé : Utilisé pour la classification et la prédiction lorsque les données sont labellisées.[10]
- Apprentissage Non Supervisé : Utilisé pour identifier des modèles dans les données non labellisées, comme dans le clustering ou la réduction de dimensionnalité.

1.11. Conclusion :

Actuellement, les réseaux de neurones artificiels constituent l'une des avancées majeures dans le domaine de l'intelligence artificielle. S'inspirant de l'activité du cerveau humain, ces dispositifs sont capables de gérer des données complexes via une structure hiérarchique faite de neurones interconnectés. Grâce à des processus tels que la propagation avant, la rétropropagation et l'optimisation par descente de gradient, ces modèles peuvent assimiler les données, optimiser leur performance et s'adapter à de nouveaux contextes.

Grâce à leur souplesse, une multitude de variantes ont vu le jour - feedforward, convolutif, récurrent ou résiduel- chacune étant perfectionnée pour des missions spécifiques, de la vision par ordinateur au traitement du langage naturel. Avec l'utilisation de fonctions d'activation non linéaires et des méthodes d'apprentissage avancées, les réseaux de neurones sont désormais une constante dans notre vie quotidienne. Ils se manifestent dans les assistants vocaux, la reconnaissance faciale, les systèmes de traduction et les diagnostics médicaux assistés par intelligence artificielle.

En définitive, les réseaux de neurones se sont imposés comme un élément fondamental de l'IA contemporaine, et leur progression constante laisse présager des possibilités encore plus prometteuses pour le futur.

Chapitre 2

Classification

2.1. Introduction :

« Le seul moyen de faire une méthode instructive et naturelle, est de mettre ensemble les choses qui se ressemblent et de séparer celles qui diffèrent les unes des autres. » M. Georges Buffon, Histoire naturelle, 1749.[11]

La classification est un processus clé en apprentissage automatique qui consiste à attribuer une ou plusieurs étiquettes à un objet en fonction de certaines de ses caractéristiques. Largement exploitée dans des domaines aussi variés que la reconnaissance d'images, la détection de spams, ou encore le diagnostic médical, elle vise à apprendre, à partir d'un jeu de données étiqueté, appelé jeu d'entraînement, un modèle prédictif capable de généraliser.

Les algorithmes de classification, en général, fonctionnent sur la base de méthodes statistiques et d'apprentissage supervisé dans lesquelles chaque exemple d'apprentissage est représenté par un vecteur de caractéristiques et une étiquette associée. Parmi les modèles d'apprentissage les plus fréquemment utilisés pour la classification, on peut citer les arbres de décision, les machines à vecteurs de support (SVM), les réseaux de neurones ou encore plus récemment les très populaires réseaux de neurones convolutifs (CNN) pour la classification d'images.

L'efficacité d'un modèle de classification est bien souvent mesurée en utilisant des métriques telle que la précision globale, la précision moyenne, le coefficient kappa, la précision, le rappel, la F-mesure ou bien encore la matrice de confusion qui permettent de nous fournir des indications sur sa capacité à reconnaître correctement les différentes classes.

Dans ce chapitre, nous commencerons par définir la classification, ses méthodes, ses grandes approches, ses domaines d'application, etc.

2.2. Définition :

La classification est une des approches les plus anciennes pour l'analyse et le traitement de données. Une classe représente un groupe d'éléments qui partagent des similarités entre eux, tout en se distinguant de ceux appartenant à d'autres classes. La classification est effectuée sur des éléments qui doivent être classés. Les objets sont situés dans un espace de variables, aussi appelées attributs, caractéristiques ou critères. L'objectif est de les situer dans un espace de catégories. Cette question n'a de pertinence que si l'on suppose qu'il existe une correspondance entre ces deux espaces. Le fait de résoudre un problème de classification consiste à déterminer comment chaque objet, caractérisé par les variables descriptives sélectionnées, peut être assigné à une classe spécifique parmi toutes les classes disponibles. L'application est effectuée par un algorithme ou une procédure qu'on appelle classifieur.

La classification est une tâche qui consiste à étiqueter une donnée que l'on veut classifier, ou aussi à trouver la classe d'une donnée.

2.3. Définition formelle :

Soit $X \subseteq R$ d un ensemble représentant un espace à d dimensions, appelé l'espace des instances. La donnée $x \in X$ est appelée une instance et représente un point dans l'espace X. L'instance x est présentée sous forme d'un vecteur de taille d, $x = (x(1),\ldots,x(d))$, où chaque composante $x(i) \in R$ est une valeur discrète ou continue. Soit Y un ensemble fini de classes où chaque classe $y \in Y$ est présentée sous forme d'une valeur discrète appelée étiquette de classe (un nom ou identifiant unique pour la classe). Le classifieur se présente alors sous forme d'une fonction de classification h (appelée aussi modèle de classification) permettant d'associer une donnée $x \in X$ à une étiquette de classe $y \in Y$.

$$\boldsymbol{h}: \begin{cases} \boldsymbol{X \to Y} \\ \boldsymbol{x \mapsto y = h(x)} \end{cases} \qquad (\mathbf{2.1})$$

Notez que pour le problème de classification, l'espace des réponses $Y \subset N$ est discret et fini, vu que chaque $y \in Y$ représente une classe. Lorsque Y est continu (c.à.d. $Y \subset R$), on parle alors du problème de régression qui sert à estimer la relation entre une ou plusieurs variables $x(i) \in R$ et une autre variable $y \in R$.[12]

2.4. Types de classification :

2.4.1. Classification binaire

La classification binaire est une opération de Machine Learning où l'opérateur doit assigner les données d'entrée à l'une des deux catégories, également connues sous le nom de « classes ».

En d'autres termes, la classification binaire consiste à segmenter un ensemble de données, également connu sous le nom de « dataset », en deux groupes distincts.

Dans une classification binaire, les attributs des éléments du jeu de données constituent les données d'entrée, tandis que les classes associées à ces éléments représentent les données de sortie.

Donc, pour la formation d'un algorithme, les composantes des données d'entrée sont attribuées à une catégorie. On qualifie généralement ces données de « labellisées » ou « étiquetées ». Généralement, un élément est classifié comme appartenant à la classe 0 ou, alternativement, comme appartenant à la classe 1.[13]

Figure 2.1 : Dataset d'élément « chats » et « chiens » étiquetés 0 et 1

2.4.2. Classification multi classe

La classification multi-classe est une tâche de Machine Learning où l'expert est amené à classer des données d'entrée parmi plus de deux catégories.

Donc, dans le cadre d'une classification à plusieurs classes, le nombre de catégories possibles pour une étiquette dépasse deux.

Dans une classification multi-classes, les classes sont habituellement représentées par des chiffres qui commencent à zéro.[13]

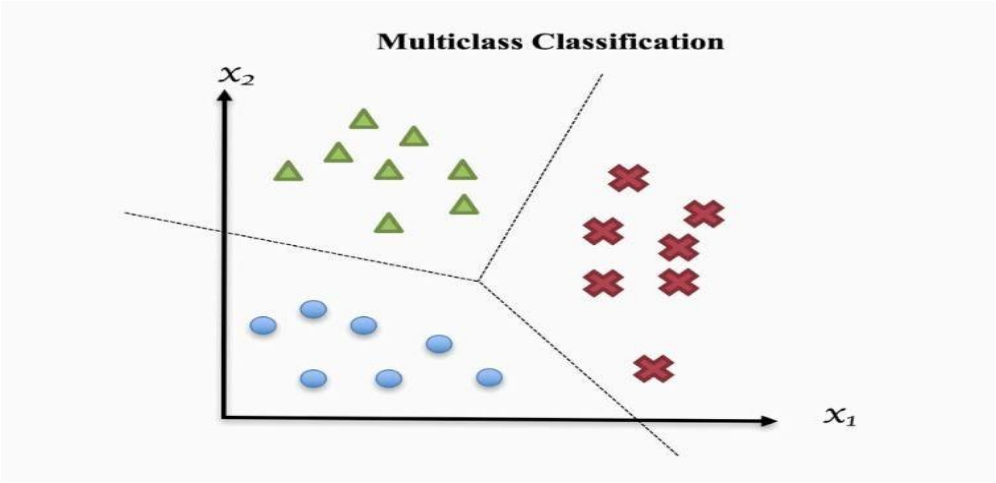


Figure 2.2 : Classification en multi classes

2.4.3. Multi-label classification

La classification multi-étiquettes est une méthode conçue pour gérer des scénarios complexes où des points de données peuvent appartenir à plusieurs catégories simultanément. Contrairement aux modèles binaires ou multi-classes traditionnels, cette technique permet une catégorisation détaillée. Elle est particulièrement utile pour des tâches telles que l'étiquetage de documents et l'annotation d'images.[14]

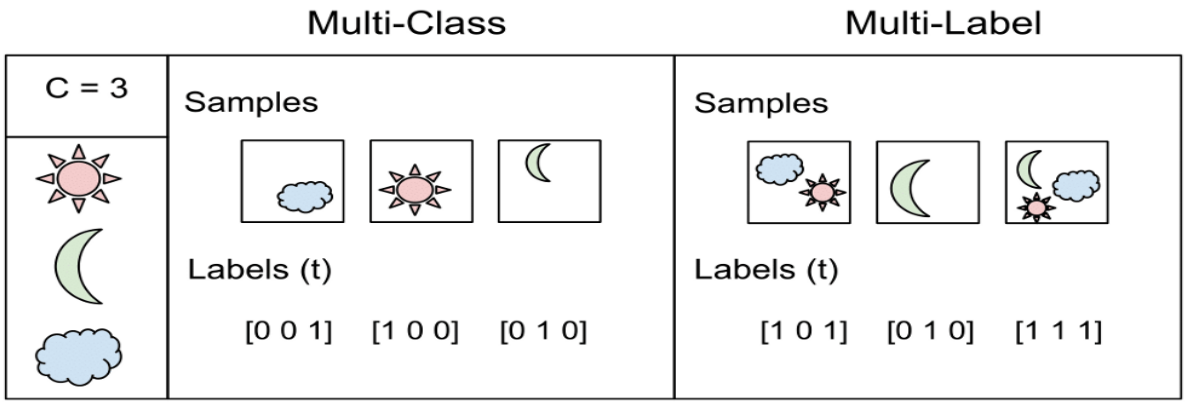


Figure 2.3 : Classification multi-étiquettes

2.4.4. Classification hiérarchique (Facultatif)

L'approche de classification hiérarchique, souvent non supervisée, structure les données en forme d'arbre (hiérarchie des classes), où les différents niveaux de granularité renferment des groupes de données imbriqués.

On distingue deux méthodes principales :

1. Classification ascendante hiérarchique (agglomérative) : Chaque composant débute dans son propre groupe, puis les groupes sont progressivement unis jusqu'à constituer une hiérarchie complète.

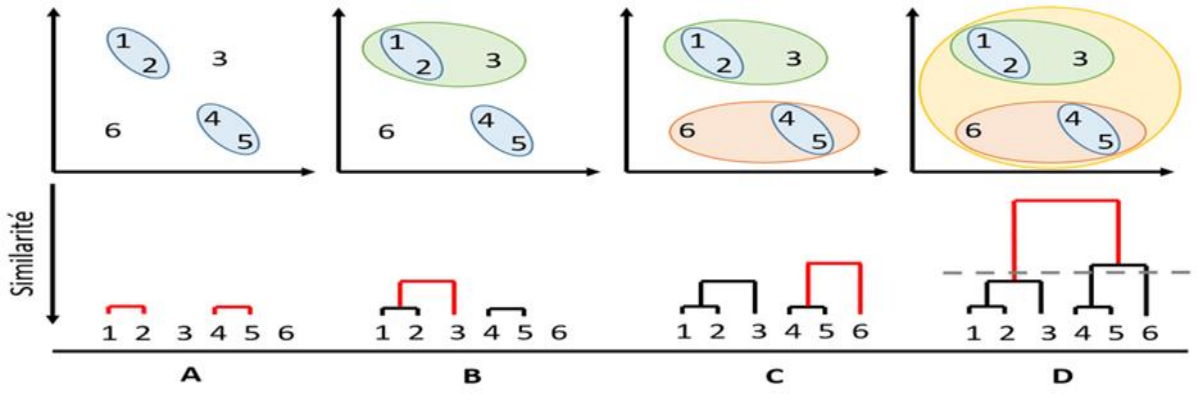


Figure 2.4 : Exemple de Classification Ascendante Hiérarchique (CAH)

2. Classification descendante hiérarchique (divisive) : Initialement, tous les éléments sont rassemblés, puis subdivisés de manière récursive en sous-groupes.[15]

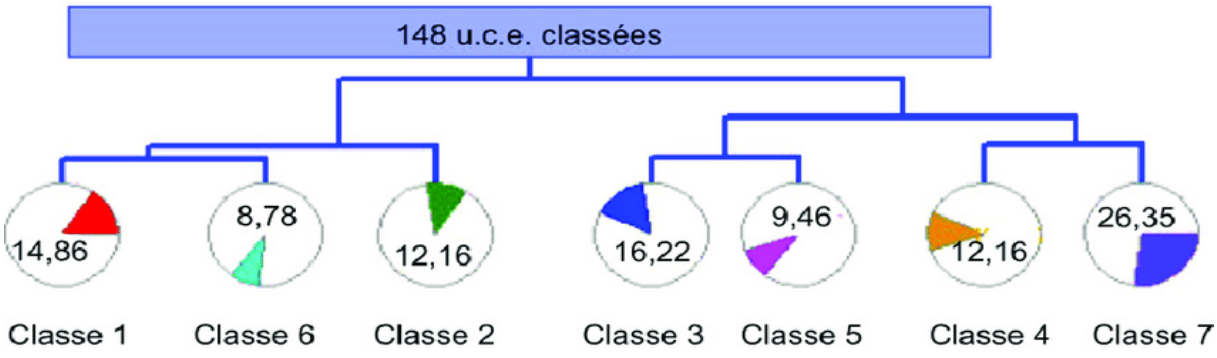


Figure 2.5 : Exemple de Classification descendante Hiérarchique (CHD)

2.5. Méthodes de classification :

Les techniques de classification portent sur l'analyse de données d'entrée (comme des images, des documents, des dossiers médicaux, etc.) dans le but d'obtenir des résultats sous forme de classes, catégories ou diagnostics. En d'autres termes, cela implique de donner une étiquette à chaque donnée en fonction de sa nature.

L'objectif de l'apprentissage automatique est d'automatiser cette procédure de classification. On identifie habituellement deux principales catégories de classification :

- **Classification supervisée** : se base sur l'apprentissage supervisé, où les résultats (ou classes) sont préétablis. Le modèle se forme sur des données étiquetées afin de prévoir les classes de nouvelles données.
- **Classification non supervisée** : s'appuie sur l'apprentissage non supervisé, où les résultats ne sont pas prédéterminés. Le modèle s'efforce alors d'identifier des regroupements ou des structures dissimulées dans les données, sans étiquettes préexistantes.
- Il y a aussi d'autres formes de classification, comme **l'apprentissage semi-supervisé**, où seule une partie des résultats est observée. Cette méthode d'apprentissage représente une synthèse entre l'apprentissage supervisé et non supervisé, en associant des données labellisées et non labellisées pour optimiser le rendement du modèle.

2.5.1. Méthodes supervisées :

2.5.1.1. Définition

L'apprentissage supervisé, également désigné comme classification supervisée, est une technique d'apprentissage automatique qui implique la formation d'un modèle à partir de données étiquetées, c'est-à-dire des données pour lesquelles les classes ou les labels sont déjà définis. L'intention est de maîtriser l'association correcte entre chaque entrée et sa classe, dans le but d'être en mesure de classer des nouvelles données ou anticiper des résultats avec précision.[16]

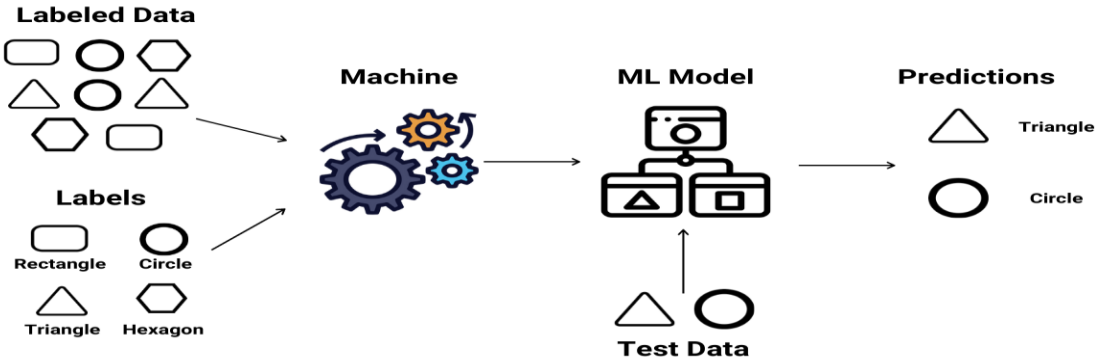


Figure 2.6 : Apprentissage supervisée

2.5.1.2. Types de problèmes traités par l'apprentissage supervisé

L'apprentissage supervisé est utilisé pour aborder deux catégories de problèmes lors de l'analyse de données : la classification et la régression.

- La classification fait appel à un algorithme afin d'attribuer précisément des données de test à des catégories spécifiques. L'algorithme identifie des entités particulières dans l'ensemble de données et cherche à déterminer comment ces entités devraient être labellisées ou caractérisées. Les classificateurs linéaires, les machines à vecteurs de support (SVM), les arbres décisionnels, les k plus proches voisins ainsi que les forêts d'arbres décisionnels figurent parmi les algorithmes de classification populaires.
- La régression aide à décrypter le lien entre les variables dépendantes et indépendantes. On l'utilise fréquemment pour effectuer des prévisions, comme par exemple le chiffre d'affaires d'une entreprise spécifique. Des algorithmes de régression couramment utilisés comprennent la régression linéaire, la régression logistique et la régression polynomiale.

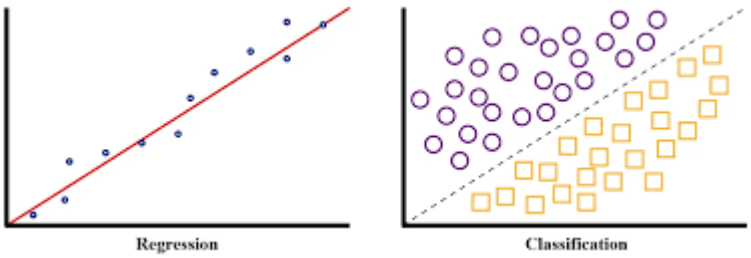


Figure 2.7 :La différence entre la classification et la régression

2.5.1.3. Algorithme de classification supervisée

1. Machines à Vecteurs de Support (SVM) :

1.1. Définition

Les Machines à Vecteurs de Support, souvent appelées SVM (Séparateur à Vaste Marge), représentent une catégorie d'algorithmes d'apprentissage initialement conçus pour la

discrimination, soit la prédiction d'une variable qualitative binaire. Par la suite, ils ont été étendus à la prévision d'une variable quantitative.[17]

Cet algorithme est considéré comme un classificateur. Il a pour fonction de séparer les ensembles de données à travers des lignes, connues sous le nom d'hyperplans. Pour atteindre cet objectif, l'algorithme se doit d'optimiser les écarts entre la frontière de séparation et les divers échantillons positionnés de chaque côté. On appelle vecteurs supports ceux qui sont les plus proches de la ligne.

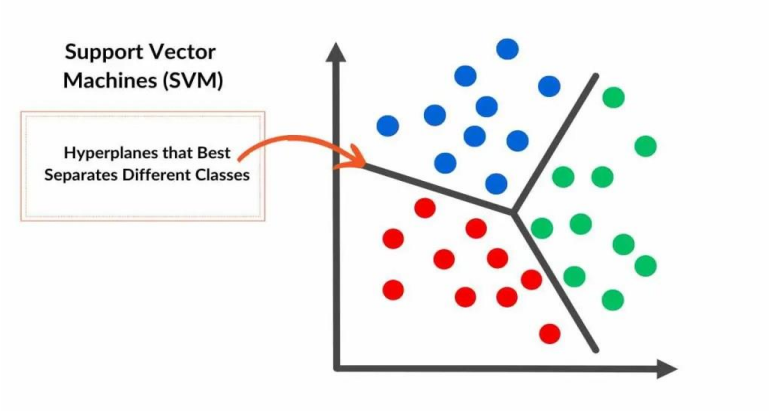


Figure 2.8 : Machines à Vecteurs de Support (SVM)

1.2. Composants des SVM :

- Hyperplan : Barrière de décision qui sépare les différentes classes dans l'espace des attributs.
- Vecteurs support : Les points de données qui se trouvent le plus près de l'hyperplan et qui ont une incidence sur sa localisation.
- Marges : L'écart perpendiculaire entre l'hyperplan et les vecteurs de support est essentiel pour évaluer la capacité de généralisation du modèle.[18]

2.L'arbre de décision :

2.1. Définition

L'algorithme de l'arbre de décision regroupe les diverses données en branches. Il se base sur une origine où chaque donnée suit un certain chemin en fonction de son comportement. Cela permet par la suite de prévoir les variables de réponse.

Tout comme pour les arbres, les points d'intersection sont désignés sous le terme de nœuds, tandis que leurs résultats sont nommés feuilles. Ainsi, les règles qui permettent de classifier les données en différentes catégories sont symbolisées par les nœuds, tandis que les feuilles correspondent aux informations proprement dites.[19]

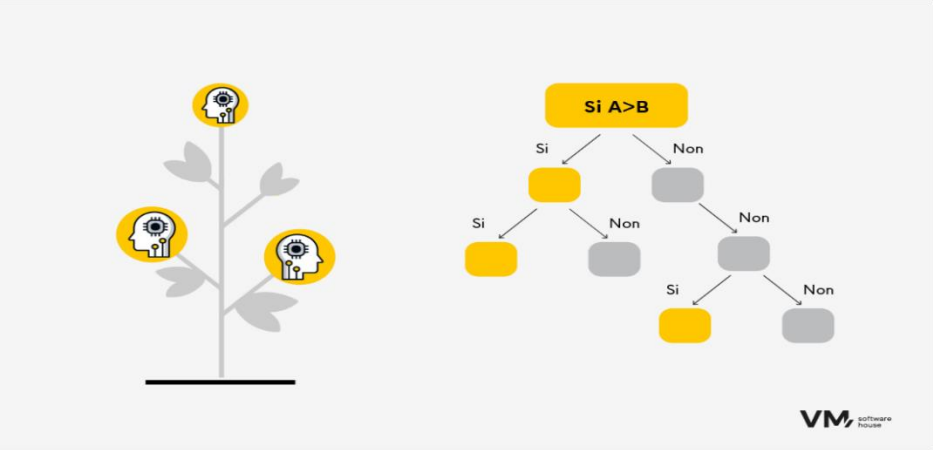


Figure 2.9 : Arbre de décision

2.2. Exemple :

Vous avez les résultats des tests. Vous voulez en savoir plus :

Séparation des sexes : femme ou homme ? Êtes-vous une femme ? Consultez la section ci-dessous.

Division par âge. Diviser, par exemple, en cinq tranches d'âge différentes :

- 0-18 ans
- 18-25 ans
- 25-40 ans
- 40-55 ans
- 55-65 ans

Division par caractéristique du résultat du test (détermination du niveau d'un résultat donné). L'algorithme vous guide pas à pas jusqu'au dernier niveau, après quoi le résultat final est déterminé et nommé en conséquence.

Les arbres de décision apprennent relativement vite et ne nécessitent pas une grande puissance de calcul. Cependant, pour qu'un algorithme sache comment construire cet arbre automatiquement et puisse « fouiller » les niveaux appropriés, il doit disposer d'une quantité de données suffisamment importante pour que les erreurs soient aussi minimes que possible.

3.k-NN (k-Nearest Neighbors):

3.1. Définition

L'algorithme des K plus proches voisins représente une technique d'apprentissage supervisé en machine learning qui utilise la notion de proximité afin d'effectuer des classifications ou des prédictions sur le rassemblement de points de données individuels. En gardant toutes les données d'apprentissage en mémoire et en ne procédant aux calculs qu'au

moment de la classification, KNN fonctionne comme un algorithme non paramétrique et paresseux. Par conséquent, plutôt que de développer un modèle durant la phase d'entraînement, KNN réalise des prédictions en mettant directement en comparaison les nouveaux points de données avec les données d'entraînement qu'il a préalablement stockées. Cet algorithme, versatile, est employé pour des missions de classification tout autant que de régression. Ses résultats dépendent du choix du K (le nombre de voisins les plus proches pris en compte) et de l'indicateur de distance utilisé pour évaluer la similarité.[20]

3.2. Fonctionnement

L'algorithme k-NN se caractérise par sa simplicité et son caractère intuitif. Il s'appuie sur le concept que des objets similaires dans l'espace des attributs possèdent des caractéristiques communes, en particulier leur catégorie.

Étapes du fonctionnement :

1. Choix du paramètre k :

On détermine un nombre entier positif k, qui correspond au nombre de voisins les plus proches à prendre en compte pour la prédiction.

La sélection de k a un impact sur la précision et la solidité du modèle.

2. Calcul des distances :

C'est la distance la plus fréquemment utilisée, restreinte aux vecteurs à valeurs réelles. En utilisant la formule suivante, elle évalue une ligne droite entre le point interrogé et l'autre point mesuré.

Pour une nouvelle donnée x à classer, on calcule la distance entre x et chaque point x_i du jeu d'entraînement.

$$\boldsymbol{d(x, x_i) = \sqrt{\sum_{j=1}^{p} \left(x_i - x_{i,j}\right)^2}} \qquad (\mathbf{2.2})$$

Selon le contexte, on peut recourir à d'autres types de distances (Manhattan, Minkowski, etc).

3. Recherche des k plus proches voisins :

On choisit les k exemples de l'ensemble d'entraînement qui sont les plus proches de x. L'ensemble $N_k(x)$ est constitué par ces voisins.

4.Vote majoritaire (classification) :

On compte les classes des k voisins.

La classe la plus fréquente parmi ces voisins est attribuée à x :

$$\hat{y} = \arg\max_{c \in y} \sum_{(x_i, y_i) \in N_k(x)} y_i = c \qquad (2.3)$$

5.Prédiction (régression) :

Dans le cas d'une tâche de régression, nous déterminons la moyenne des valeurs des voisins :

$$\hat{y} = \frac{1}{k} \sum_{(x_i, y_i) \in N_k(x)} y_i \qquad (2.4)$$

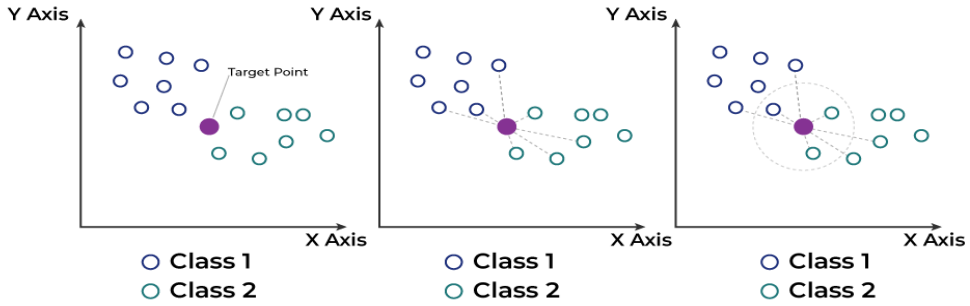
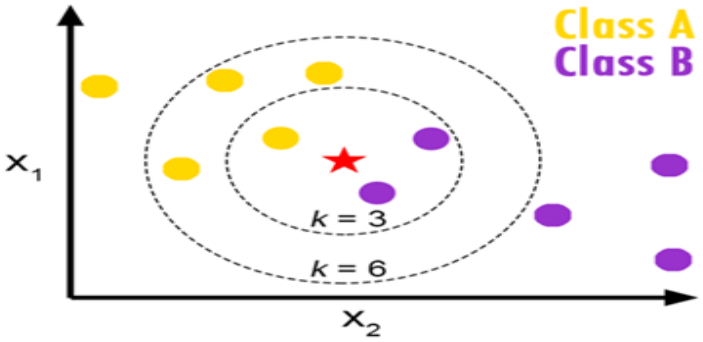


Figure 2.10 : Fonctionnement de l'algorithme KNN

3.3. Exemple : L'étoile (indiquée par un symbole rouge dans l'image ci-dessous) représente le x_q, et notre ensemble de données comporte deux classes : la Classe A (cercles en jaune - a_1, a_2, a_3, …) et la Classe B (cercles en violet - b_1, b_2, b_3, …), qui sont toutes deux regroupées sous d-train. La valeur de k est fixée à 3. Ces trois éléments constituent les données d'entrée pour l'algorithme kNN.



Étape 1 : De manière intuitive, le x_q sera représenté sur le graphique comme illustré dans le graphique en fonction de ses caractéristiques x_1 et x_2.

Étape 2 : À présent, lors de l'exécution du modèle, la distance entre tous les points de données de d-train et le point de requête (x_q) sera calculée.

Distance = [dist (x_q, a_1), dist (x_q, b_1), dist (x_q, a_2), …]

Étape 3 : Choisir les k points de données/étiquettes/classes les plus pertinentes, c'est-à-dire les k points qui sont les moins éloignés des autres par rapport à x_q.

Étape 4 : Effectuez un vote majoritaire. x_q est étiqueté sur la base de la classe majoritaire des k points de données.

4. Réseau de neurone :

Les réseaux de neurones artificiels sont des réseaux fortement connectés de processeurs élémentaires fonctionnant en parallèle. Chaque processeur élémentaire calcule une sortie unique sur la base des informations qu'il reçoit. Toute structure hiérarchique de réseaux est évidemment un réseau.

Il existe 3 couches principales d'un réseau de neurones :

- Couche d'entrée (Input Layer) : Elle prend en entrée des données non traitées au sein du réseau (par exemple : une image, un vecteur de caractéristiques ou un signal). La configuration dépend du genre et de la taille des données à manipuler.
- Couche cachée (Hidden Layers) : C'est principalement au niveau des couches internes que s'effectue la majorité du traitement. Ces dernières extraient et modifient les attributs des données à travers une séquence de calculs (tel que les fonctions d'activation, etc.).
- Couche de sortie (Output Layer) : Elle offre l'issue finale du réseau, à l'image de la classe estimée dans un problème de classification. Elle fait fréquemment appel à une fonction d'activation telle que Softmax ou Sigmoid, en fonction du type de résultat prévu.

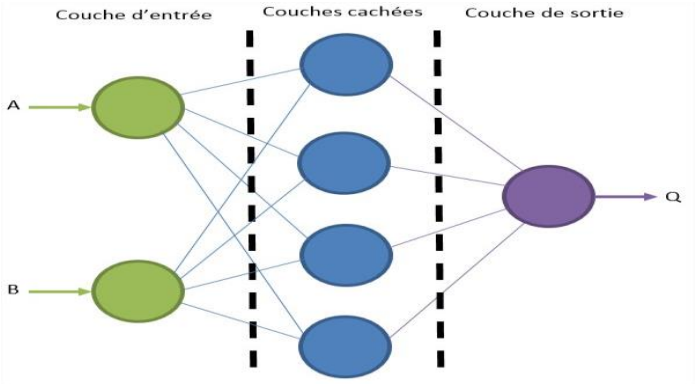


Figure 2.11 : Réseau de neurone artificiels.

5. Perceptron :

Le perceptron est un algorithme d'apprentissage supervisé ainsi qu'un modèle de neurone artificiel simplifié, créé par Frank Rosenblatt en 1957. Il est principalement utilisé pour le classement binaire, c'est-à-dire pour diviser les données en deux catégories distinctes.

1. Définition et fonctionnement

Le perceptron prend en entrée un vecteur de caractéristiques $x = (x_1, x_2, \ldots, x_n)$ et associe à chaque entrée un poids synaptique $w = (w_1, w_2, \ldots, w_n)$.

Il calcule une somme pondérée des entrées :

$$z = \sum_{i=1}^{n} w_i x_i + b \qquad (2.5)$$

Où b est un biais (ou seuil).

L'obtention de la sortie o se fait en utilisant une fonction d'activation, généralement la fonction de Heaviside (fonction seuil) :

$$o = \begin{cases} 1 \; si \; z \geq 0 \\ 0 \quad sinon \end{cases} \qquad (2.6)$$

Cette sortie correspond à la prédiction de la classe (par exemple 0 ou 1).

2. Apprentissage

Le perceptron modifie ses paramètres ω_i et le biais b en se basant sur un ensemble d'exemples annotés, dans le but de réduire l'erreur sur les données d'apprentissage.

La règle d'apprentissage ajuste les poids à chaque itération sur la base de l'écart entre le résultat prévu et le résultat effectif, jusqu'à ce qu'une convergence soit atteinte si les classes peuvent être séparées de manière linéaire.

3. Limite

Le perceptron simple ne peut traiter que des problèmes qui sont linéairement séparés. On fait appel à des perceptrons multicouches (MLP), qui intègrent des couches cachées et des fonctions d'activation non linéaires, pour aborder des problématiques plus sophistiquées.

4. Application

Classification binaire simple (spam ou non-spam, reconnaissance d'écriture manuscrite). Fondements historiques des réseaux de neurones et de l'apprentissage profond.

2.5.2. Méthode non supervisée :

2.5.2.1. Définition

L'analyse et la classification de données non étiquetées font partie du machine learning, une discipline également connue sous le nom d'apprentissage non supervisé. C'est pourquoi, ces algorithmes sont formés pour détecter des motifs ou des ensembles dans les données, avec une intervention humaine minimale. Sur le plan mathématique, l'apprentissage non supervisé consiste à observer plusieurs instances d'un vecteur X et à déterminer la distribution de probabilité $p(x)$ associée à ces instances.

- Représentation des données et objectif mathématique :

On suppose qu'on a plusieurs exemples, chacun étant un vecteur de nombres :

$$X = \{x^{(1)}, x^{(2)}, \dots, x^{(n)}\} \text{ Ou chaque } x^{(i)} \in \mathbb{R}^d$$

Par exemple, chaque $x^{(i)}$ peut représenter une image, un texte ou un signal, avec d caractéristiques (comme la taille, la couleur, la fréquence, etc.).

L'idée est d'étudier comment ces données sont réparties dans l'espace. On veut approcher leur distribution de probabilité :$p(x)$

Cela signifie que l'objectif est de comprendre la forme globale des données. Il s'agit d'identifier d'éventuels regroupements naturels, de repérer des zones où les données sont plus concentrées (zones de forte densité), ainsi que de mettre en évidence des structures ou des motifs cachés dans l'ensemble des observations.

Figure 2.12 : Classification non supervisée

2.5.2.2. Principe

Détection de structures cachées : L'algorithme vise à déceler des motifs ou des regroupements basés sur la similarité ou la distance entre les données.

Absence d'étiquettes : Le modèle n'a pas accès à des exemples préclassés, il doit donc s'auto-organiser et déterminer les groupes sans aucune référence externe.

Évaluation de la qualité : L'évaluation de la qualité d'une classification non supervisée se fait par sa propension à mettre en évidence des motifs cohérents et utilisables, généralement à travers des paramètres internes (liens, distinction) ou par une validation externe si cela est réalisable.

Méthodes habituelles : Les techniques employées incluent le regroupement (k-means, classification hiérarchique), la réduction de dimensionnalité (ACP) et les réseaux de neurones générateurs.

- **Réseaux de Neurones Générateurs**

Les réseaux de neurones générateurs sont un type de réseaux de neurones utilisés non pas pour prédire une étiquette ou classifier des données, mais pour créer de nouvelles données qui ressemblent à celles qu'on leur a montrées.

Leur objectif principal est de modéliser la distribution des données et de générer de nouvelles instances (images, textes, sons, etc.) qui suivent la même structure statistique que les données d'entraînement.

2.5.2.3. Types de problèmes d'apprentissage non supervisé

L'apprentissage non supervisé peut être divisé en deux catégories distinctes : le clustering et l'association.

- Le clustering : est une technique d'apprentissage non supervisé qui vise à rassembler des données sans étiquettes selon leur degré de similitude. Les objets semblables sont regroupés ensemble, tandis que ceux qui présentent des différences sont mis à part. Cette méthode s'appuie sur l'étude des propriétés des données afin de créer des groupes homogènes. On trouve souvent des techniques telles que k-means, la classification hiérarchique et la classification probabiliste.
- Les règles d'association : représentent une méthode non supervisée visant à déceler des relations récurrentes entre des éléments au sein d'un jeu de données. Elles trouvent leur utilisation fréquente en marketing, surtout pour l'analyse des comportements d'achat des consommateurs. Ces techniques détectent des corrélations du genre : « Si un client achète X, il achètera probablement Y ». On utilise principalement les algorithmes Apriori, Eclat et FP-growth.

2.5.2.4. Algorithmes principaux

On distingue de nombreuses familles d'algorithmes dont le but est de trouver des groupes ou des structures cachées dans des données non labélisée ; cet objectif est alors qualifié de classification non supervisée.

- **Les méthodes de clustering :** Le regroupement, également appelé clustering, sert à regrouper des données semblables en ensembles ou en clusters. On dispose de diverses méthodes et algorithmes pour atteindre cet objectif.

1. K-Means Clustering

L'algorithme de regroupement K-Means est l'un des plus couramment utilisés dans l'apprentissage non supervisé. Il vous faut lui fournir un ensemble de données non labellisé ainsi que le nombre de regroupements souhaités. L'algorithme initiera le processus en déterminant les K centroïdes, soit en les produisant de façon aléatoire, soit en les sélectionnant directement à partir du jeu de données. Cet algorithme procédera donc à des itérations dans le but d'arriver à un résultat.

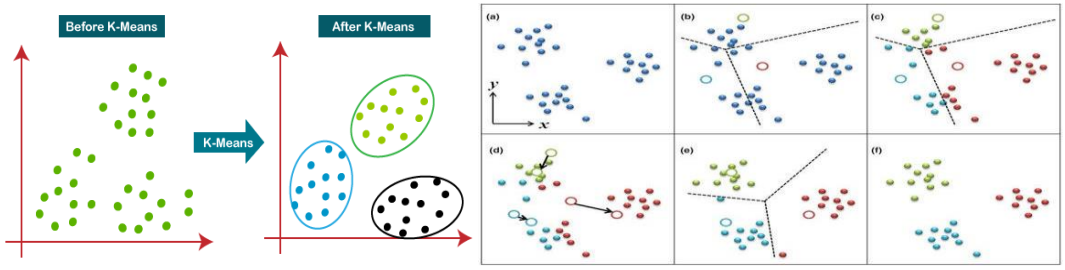


Figure 2.13 : Algorithme et fonctionnement de k-means clustring

2. L'analyse hiérarchique des données

Cette approche cherche à organiser des données comparables de façon hiérarchique, afin de créer un dendrogramme, une structure en forme d'arbre regroupant des clusters. Initialement, chaque point de données est traité comme un cluster distinct. Ensuite, les clusters les plus proches se fusionnent graduellement, formant ainsi une hiérarchie de regroupements. L'étape de classification est effectuée en continu jusqu'à ce que tous les points soient rassemblés au sein d'un unique regroupement.

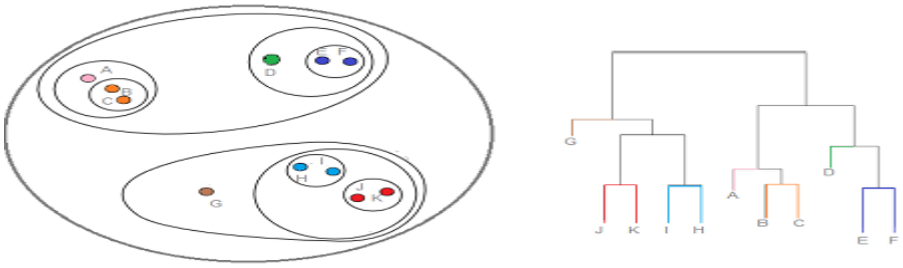


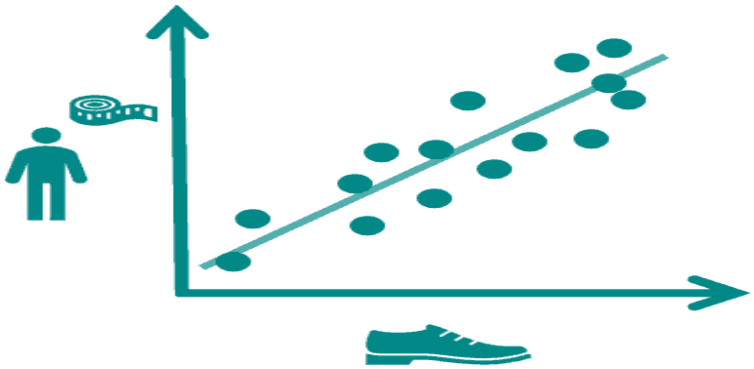
Figure 2.14 : Dendrogramme de clustering hiérarchique

3. Le regroupement par décalage moyen

Également connue sous le nom de clustering par déplacement moyen (Mean Shift Clustering), la conception fondamentale de cet algorithme consiste à faire glisser progressivement un groupe de fenêtres ou de noyaux sur les données jusqu'à ce qu'ils atteignent les points d'apogée locaux de la distribution des données.

- **Les méthodes d'association** : L'objectif de ces techniques est d'identifier des relations ou des corrélations importantes entre les différents éléments d'un jeu de données.

1. Les méthodes de Corrélation : Une analyse de corrélation est une méthode statistique qui permet de déterminer la relation entre deux variables, comme par exemple si la taille du corps a un rapport avec la pointure des chaussures.



On peut procéder à un calcul de corrélation pour examiner le lien entre les différentes variables. La force de la corrélation est jugée sur la base du coefficient de corrélation, qui peut aller de -1 à +1. On peut donc se servir des analyses de corrélation pour établir la force et la direction de la corrélation.[21]

- Le coeffcient de corrélation : Le coefficient de corrélation de Pearson, le plus utilisé, est calculé à partir de la covariance des deux variables normalisées par le produit de leurs écarts-types. La formule est :

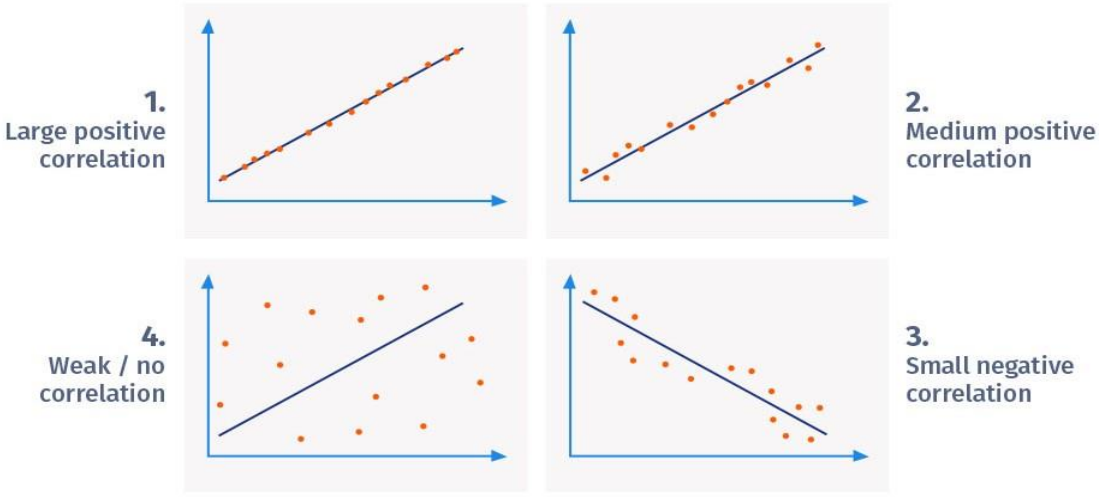
$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \Sigma (y_i - \bar{y})^2}} \qquad (2.7)$$

Où x_i et y_i sont les valeurs observées, $\bar{x}$ et $\bar{y}$ les moyennes des variables X et Y. [22]

- Interprétation :

Un coefficient r se rapprochant de +1 signifie une forte corrélation linéaire positive, c'est-à-dire que si X s'accroît, Y tend également à croître de façon prévisible.

Un coefficient r se rapprochant de -1 indique une relation linéaire négative significative. Un coefficient proche de 0 indique l'absence de lien linéaire notable entre les variables. Une corrélation ne sous-entend pas une relation de cause à effet, mais établit un lien statistique entre les variables.



2. L'algorithme A-priori : une création de Rakesh Agrawal et Ramakrishnan Sikrant en 1994, est un outil d'exploration de données conçu spécifiquement pour le domaine de l'apprentissage des règles d'association. Il est utilisé pour identifier des caractéristiques qui apparaissent régulièrement dans un jeu de données et en tirer une classification. A-Priori identifie les règles d'association présentes dans un ensemble de données, en se basant sur un seuil de support et un seuil de confiance fixés. L'utilisateur a la liberté de définir ces deux valeurs à sa guise.[23]

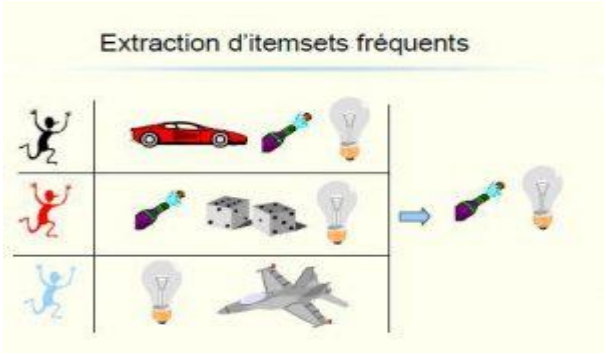


Figure 2.15 : L'algorithme A-priori

2.5.2.5. Domaines d'application

L'apprentissage non supervisé est largement utilisé dans divers domaines.

- En marketing, il permet la segmentation de clientèle pour mieux cibler les offres et campagnes.
- En traitement du langage naturel (NLP), il facilite le regroupement de documents, l'extraction de thèmes, le résumé automatique ou la détection de plagiat.
- En détection de fraude, il identifie des anomalies dans les transactions ou les activités suspectes en cybersécurité.
- Les moteurs de recommandation, comme ceux de Netflix, utilisent ces algorithmes pour proposer des contenus personnalisés.
- En vision par ordinateur, ils servent à reconnaître des objets ou des scènes dans des images. Enfin, en recherche génétique, le clustering hiérarchique aide à analyser les séquences d'ADN et à comprendre les relations évolutives.

2.5.2.6. Limites

La classification non supervisée, même si elle est performante pour l'exploration et l'organisation de données brutes, comporte plusieurs contraintes majeures :

- Interprétation complexe des résultats : Les groupes identifiés peuvent poser des problèmes d'interprétation, en raison de l'absence de référence externe pour confirmer leur validité.

L'algorithme est capable d'identifier des structures qui ne possèdent pas véritablement de signification ou d'utilité pour l'utilisateur.

- Absence de critère d'évaluation objectif : En l'absence d'étiquettes, il est difficile d'apprécier la qualité des partitions produites. Contrairement à l'apprentissage supervisé où un score de réussite est clairement défini, la validation dans ce contexte s'appuie fréquemment sur des critères internes ou des évaluations subjectives
- Paramètres et données : Les résultats sont fortement influencés par les décisions relatives aux paramètres (nombre de regroupements, méthode de distance, etc.) ainsi que par la qualité des données utilisées (présence de bruit, valeurs hors normes, échelles des variables).
- Danger de sur-segmentation ou de sous-segmentation : L'algorithme peut générer un nombre excessif de groupes (sur-segmentation) ou, à l'inverse, combiner différents groupes (sous-segmentation), en particulier si le choix du nombre de classes n'est pas approprié.
- Identification de structures non pertinentes : L'algorithme peut repérer des motifs ou des ensembles qui ne se conforment à aucune réalité ou utilité professionnelle, générant ainsi du bruit plutôt que de l'ordre dans les informations.

2.6. Critères d'évaluation d'un classificateur :

2.6.1. La précision globale (Overall Accuracy)

La précision est une mesure qui mesure la fréquence à laquelle un modèle d'apprentissage automatique prédit correctement le résultat. Pour calculer la précision en divisant le nombre de prédictions correctes par le nombre total de prédictions.

1. Formule :

$$\boldsymbol{Précision\ globale = \frac{Nombre\ de\ prédictions\ correctes}{Nombre\ total\ de\ prédiction}} \qquad (\mathbf{2.8})$$

2. Exemple de calcul :

Overall Accuracy

Reference data	Classified image			
	Water	Forest	Urban	Total
Water	21	5	7	33
Forest	6	31	2	39
Urban	0	1	22	23
Total	27	37	31	95

Correctly classified:
21 + 31 + 22 = 74

Total number reference sites = 95

Overall accuracy
= 74 / 95 = **77.9%**

En utilisant les termes de la matrice de confusion :

$$\boldsymbol{Précision\ globale} = \frac{\boldsymbol{VP+VN}}{\boldsymbol{VP+VN+FP+FN}} \qquad (\mathbf{2.9})$$

Où :

- VP (Vrai Positif) : nombre de cas positifs correctement prédits
- VN (Vrai Négatif) : nombre de cas négatifs correctement prédits
- FP (Faux Positif) : nombre de cas négatifs prédits à tort comme positifs
- FN (Faux Négatif) : nombre de cas positifs prédits à tort comme négatifs. [24]

3. Interprétation :

L'exactitude globale peut aller de 0 (aucune prédiction juste) à 1 (toutes les prédictions justes), ou exprimée en pourcentage, de 0% à 100%.

Une précision totale de 100% indique que le modèle a correctement classé toutes les classes. C'est une méthode facile à comprendre et intuitive, particulièrement adaptée lorsque les classes sont équilibrées (c'est-à-dire présentes en proportions comparables).

4. Limites :

Déséquilibre des classes : Si une classe est nettement plus courante que les autres, la précision générale pourrait être fallacieuse. Par exemple, si 95% des courriels ne sont pas du spam, un modèle qui prédit systématiquement « non spam » aura une précision de 95%, néanmoins il n'identifiera aucun courrier indésirable.

Indicateurs supplémentaires : Dans ces situations, il est recommandé de se référer à d'autres mesures telles que la précision (precision), le rappel (recall) ou le F-mesure (F1-score), qui fournissent une meilleure évaluation de la performance pour chaque classe.

L'indicateur le plus simple pour juger de l'efficacité d'un modèle de classification est la précision globale, qui indique la proportion des prédictions justes parmi toutes les données. Toutefois, son utilisation doit être faite avec précaution, notamment en présence d'un déséquilibre entre les classes, où des mesures plus spécifiques seront requises pour une évaluation totale.

2.6.2. La précision moyenne (Average Accuracy, AA) : est une mesure employée pour évaluer la performance des modèles dans les domaines de la classification, de la détection d'objets ou de la recherche d'informations, spécifiquement quand les résultats sont rangés selon leur pertinence.

1. Calcul de la précision moyenne :

Elle calcule l'exactitude moyenne par classe en prenant la moyenne de l'exactitude de chaque classe individuelle. Sa formule est :

$$AA = \frac{1}{N}\sum_{i=1}^{N}\frac{Prediction\ correct\ pour\ la\ clase\ i}{total\ de\ prediction\ de\ la\ classe\ i} \quad (2.10)$$

Où N est le nombre total de classe

2. Importance et applications :

La performance moyenne est particulièrement pertinente dans les situations où l'exactitude du tri des résultats est primordiale, à l'image de la détection d'objets en vision par ordinateur, de la recherche d'informations ou encore des systèmes de suggestion.

Elle offre une meilleure évaluation des performances sur des ensembles de données déséquilibrés, où une simple précision globale peut être fallacieuse.

Par exemple, dans le contexte de l'identification des tumeurs sur des images de santé, la précision moyenne est un indicateur permettant d'évaluer la pertinence et la qualité du classement des zones identifiées.

2.6.3. Le coefficient Kappa

Le Kappa (ou Kappa de Cohen) est une mesure statistique servant à déterminer l'accord entre deux évaluateurs (ou classifieurs), tout en prenant en compte le hasard. À l'inverse d'un simple pourcentage de concordance, Kappa prend en considération la probabilité que deux accords se produisent par hasard.

1. Formule :

$$kappa = \frac{P_o - P_e}{1 - P_e} \quad (2.11)$$

- P_o: proportion d'accord observé
- P_e: proportion d'accord attendu par hasard

2. Interprétation :

Valeur de Kappa	Niveau d'accord
≤ 0	Aucun accord (ou pire que le hasard)
0.01 – 0.20	Accord faible
0.21 – 0.40	Accord passable
0.41 – 0.60	Accord modéré
0.61 – 0.80	Accord substantiel
0.81 – 1.00	Accord presque parfait

3. Utilisation

- Évaluation de classifieurs (ex. : deux modèles de classification)

- Analyse d'accord inter-annotateurs (en traitement du langage naturel ou médecine)
- Études cliniques, psychométrie, systèmes de recommandation

4. Limites
   - Sensible à la prévalence des classes (si les classes sont déséquilibrées)
   - Moins fiable si les évaluateurs utilisent majoritairement une même catégorie

2.6.4. Matrice de confusion :

1. Définition

Une matrice de confusion est un outil couramment utilisé en apprentissage supervisé qui permet de fournir une représentation exhaustive du comportement d'un modèle de classification et d'évaluer précisément sa performance par rapport aux données réelles. Elle représente les performances d'un algorithme en signalant la qualité du modèle à travers quatre indicateurs clés, sans tenir compte de la distribution des classes (Figure 16).[25]

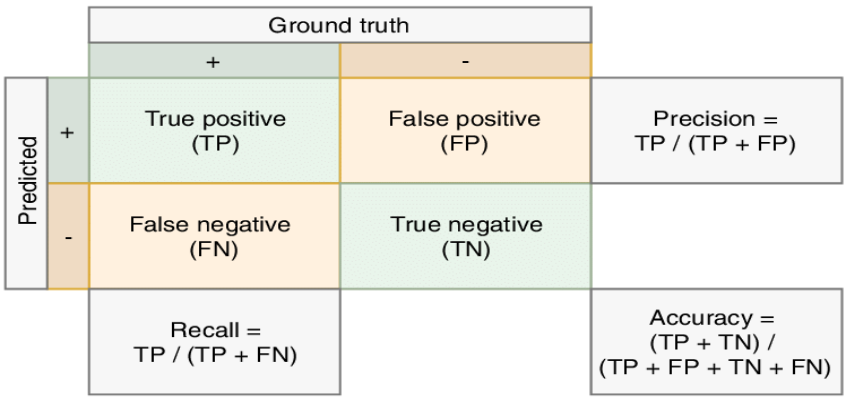


Figure 2.16 : Exemple de la matrice de confusion

Les quatre indicateurs sont :

- Vrais positifs (VP) : cas où le modèle a correctement prédit la classe.
- Vrais négatifs (VN) : cas où le modèle a correctement prédit l'absence d'une classe.
- Faux positifs (FP) : également appelés erreurs de type I, cas où le modèle a incorrectement prédit la présence d'une classe.
- Faux négatifs (FN) : également appelés erreurs de type II, cas où le modèle a incorrectement prédit l'absence d'une classe.

2. Indicateur de performance

La matrice de confusion sert de base au calcul de plusieurs indicateurs de performance, tels que :

➢ Accuracy : proportion de vrais résultats (vrais positifs et vrais négatifs) parmi le nombre total de cas examinés.[26]

- Précision : nombre de vrais positifs divisé par la somme des vrais positifs et des faux positifs. Également appelée valeur prédictive positive. Sous forme mathématique, on a :

$$\boldsymbol{Precision = \frac{VP}{VP + FP}} \quad (\mathbf{2.12})$$

- Rappel (ou sensibilité ou taux de vrais positifs) : nombre de vrais positifs divisé par la somme des vrais positifs et des faux négatifs. Sous forme mathématique, on a :

$$\boldsymbol{recall = \frac{VP}{VP + FN}} \quad (\mathbf{2.13})$$

- Score F1 (ou score F, ou mesure F) : moyenne pondérée de la précision et du rappel. Il prend en compte les faux positifs et les faux négatifs, permettant ainsi un équilibre entre les deux. Sous forme mathématique, on a :[26]

$$\boldsymbol{F1 - score = 2\frac{recall \times precision}{recall + precision}} \quad (\mathbf{2.14})$$

3. Exemple de calcul

		Actual class (ground truth)		
Total (n)=100		Dog (Positive)	Not a Dog (Negative)	
Predicted class	Dog (Positive)	15 (TP)	20 (FP, Type I Error)	**Precision** =TP/(TP+FP) =0.42
	Not a Dog (Negative)	5 (FN, Type II Error)	60 (TN)	
	Accuracy =(TP+TN)/Total =0.75	**Sensitivity,Recall, TPR** =TP/(TP+FN) = 0.75	**FPR** = FP/(FP+TN) =0.25	**F1 Score** =2*(Precision*Recall) / (Precision+Recall) =0.53
	Error Rate =(FP+FN)/Total =0.25	**Miss Rate, FNR** =FN/(TP+FN) =0.25	**Specificity, TNR** = TN/(FP+TN) =0.75	

2.7. Applications pratiques de la classification :

- Vision par ordinateur : L'identification et la catégorisation automatiques d'objets présents dans des images ou vidéos font appel à la technique de classification. Elle autorise, par exemple, l'identification faciale, la détection de panneaux de signalisation ou encore l'identification d'objets dans les systèmes de conduite autonome

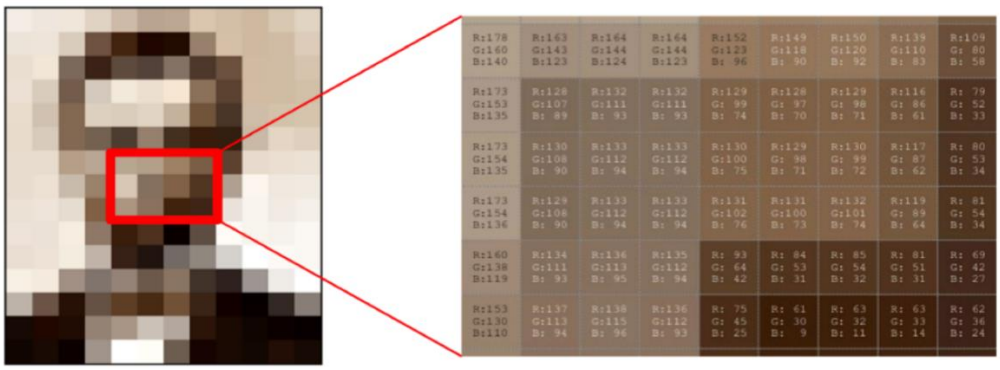


Figure 2.18 : Un modèle de vision par ordinateur

- Diagnostic médical : Les modèles de classification contribuent à l'identification de pathologies à partir d'images médicales (telles que les IRM ou les radiographies) ou d'informations cliniques. Ils facilitent la détection d'une pathologie, contribuant ainsi à un diagnostic précoce.
- Finance et marketing : Dans le domaine de la finance, on utilise la classification pour anticiper les risques liés au crédit ou pour identifier les fraudes. Dans le domaine du marketing, elle facilite la segmentation des clients, la prévision de leur comportement et l'optimisation des campagnes publicitaires.

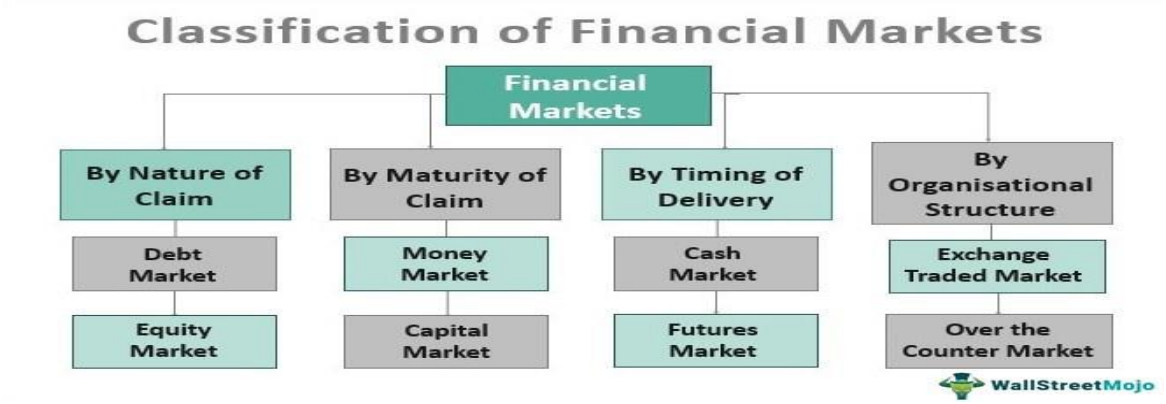


Figure 2.19 : classification des marchés financiers

- Traitement du langage naturel : En NLP, la classification sert à identifier le thème d'un texte, détecter les spams, analyser les sentiments, ou encore classifier les e-mails par catégories. Ces techniques sont essentielles pour comprendre et organiser de grandes quantités de texte automatiquement.

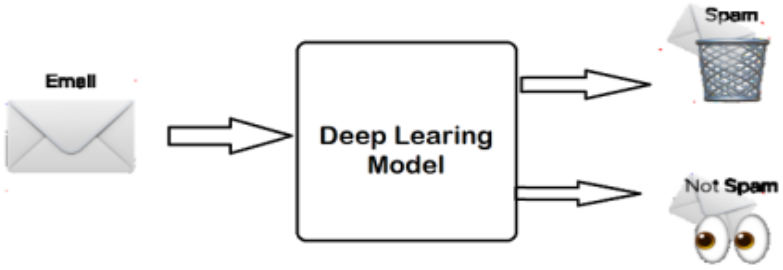


Figure 2.20 : Détection des spams

2.8. Conclusion :

La classification constitue un élément clé de l'apprentissage automatique, permettant de regrouper des données selon leurs caractéristiques communes. Elle joue un rôle essentiel dans de nombreux domaines d'application, qu'il s'agisse de la reconnaissance d'images, du diagnostic médical, de la détection de fraude ou de la segmentation de clients.

Qu'elle soit binaire, multi-classe ou multi-étiquette, la classification repose sur des algorithmes adaptés à la nature des données et aux objectifs poursuivis. Les approches supervisées exploitent des données annotées pour entraîner des modèles précis, tandis que les approches non supervisées cherchent à révéler des structures cachées sans connaissance préalable des classes.

La diversité des algorithmes telles que les SVM, les arbres de décision, les réseaux de neurones ou les techniques de clustering – permet de répondre à un large éventail de problématiques. Leur évaluation s'appuie sur des critères comme la précision, le rappel, le F-mesure ou encore la matrice de confusion, afin d'estimer l'efficacité et la robustesse des modèles.

Chapitre 3

Réseaux de neurones convolutifs (CNN)

3.1. Introduction :

L'apparition des réseaux de neurones convolutifs (CNN - Convolutional Neural Networks) a constitué une étape cruciale dans le secteur de la vision artificielle et de l'apprentissage machine. Les CNN, conçus pour gérer de manière efficace les images et les données organisées en grille, ont transformé la façon dont les machines perçoivent, analysent et interprètent les informations visuelles.

Ce chapitre vise à exposer de façon organisée la définition d'un CNN, à décrire les diverses couches qui le constituent comme la convolution, le pooling, l'activation, la normalisation et les couches entièrement connectées ainsi que les principes architecturaux qui guident son opération. Ces aspects essentiels donnent aux CNN la capacité de déduire de manière automatique les attributs hiérarchiques d'une image, diminuant ainsi notablement le besoin d'un travail manuel sur les descripteurs.

Par la suite, on porte une attention spéciale à l'analyse des modèles CNN les plus représentatifs de ces dernières années. On compte parmi eux LeNet-5 précurseur dans la reconnaissance de chiffres manuscrits , ResNet innovateur des connexions résiduelles , Xception qui repose sur les convolutions séparables en profondeur , DenseNet qui renforce les liens inter-couches , VGG célèbre pour sa simplicité et sa profondeur uniforme , MobileNet élaboré pour les appareils mobiles ,AlexNet précurseur de l'apprentissage profond sur GPU et fondement de la révolution actuelle des CNN, ainsi qu'EfficientNet, qui parvient à un équilibre optimal entre profondeur, largeur et résolution du réseau.

3.2. Définition :

Un réseau de neurones convolutif, aussi appelé CNN, est un type de réseau de neurones en apprentissage profond conçu pour manipuler des matrices de données structurées, comme les images. Les réseaux de neurones convolutifs, qui sont couramment utilisés en vision par ordinateur, se sont imposés comme la norme pour plusieurs applications visuelles telles que la classification d'images. Ils ont aussi démontré leur compétence dans le traitement du langage naturel pour la classification de textes. Les réseaux de neurones convolutifs excellent dans la détection des motifs dans l'image d'entrée, comme les lignes, les dégradés, les cercles, ou encore les yeux et les visages. C'est cette caractéristique qui les rend si efficaces en matière de vision par ordinateur. À la différence des précédents algorithmes de vision par ordinateur, ceux-ci sont capables d'agir directement sur une image brute sans nécessiter aucune étape de prétraitement.

Un réseau de neurones convolutif est un type de réseau de neurones feedforward, qui comprend couramment jusqu'à 20 ou 30 niveaux. L'efficacité d'un réseau de neurones convolutifs est due

à une couche particulière, nommée couche convolutive. Ils sont constitués de plusieurs couches convolutives empilées, chaque couche étant apte à identifier des motifs plus sophistiqués. En utilisant trois ou quatre couches convolutives, on peut identifier des chiffres écrits à la main, tandis qu'avec 25 couches, on est capable de distinguer des visages humains.[27]

3.3. Les couches des CNN :

Pour un réseau de neurones convolutif, on distingue quatre types de couches : la couche de convolution, la couche de pooling, la couche d'activation ReLU et la couche entièrement connectée.

3.3.1. Couche de convolution :

De manière simplifiée, la convolution consiste à appliquer un filtre mathématique sur une image. Sur le plan technique, cela consiste à faire glisser une matrice sur une image, et pour chaque pixel, on utilise la somme résultant de la multiplication de ce pixel par sa valeur sur la matrice. Cette méthode nous aide à identifier les sections de l'image qui pourraient susciter notre intérêt. Considérons la Figure (3.1) à gauche comme illustration d'image et à droite comme illustration de filtre.

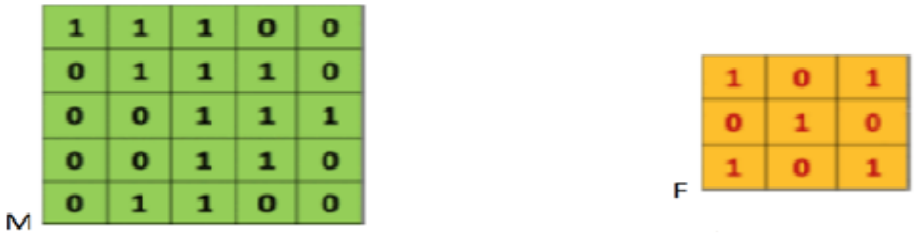
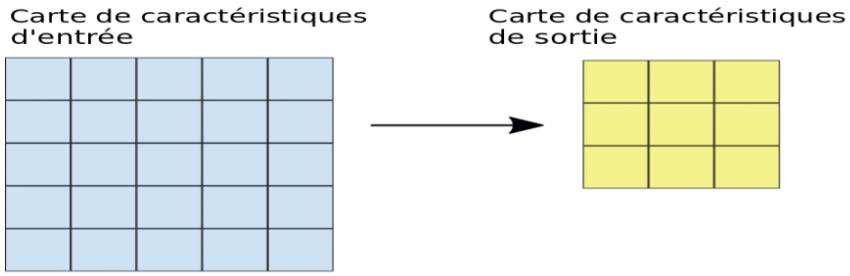
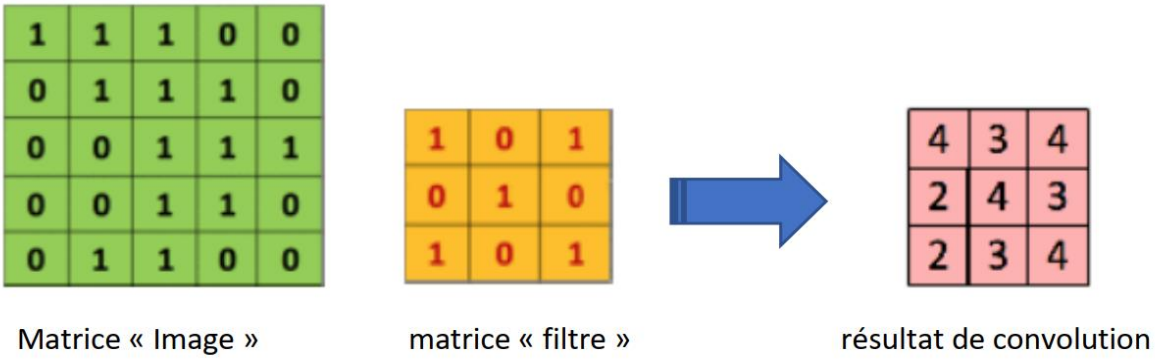


Figure 3.1 : A gauche exemple des valeurs des pixels d'une image 5x5 et à droite exemple de valeurs d'une matrice utilisée comme filtre

Le filtre doit être déplacé d'une case à chaque itération jusqu'à ce que la première ligne soit complétée. Quand nous avons terminé la première ligne, le filtre « descend » d'une case et le processus est reproduit pour chaque ligne et colonne. Regardez l'animation qui suit :





Il est à noter qu'une convolution 3x3 appliquée sur une carte de caractéristiques d'entrée 5x5, ayant également une profondeur 1. Étant donné qu'il existe neuf positions 3x3 possibles pour tirer les tuiles de la carte des caractéristiques 5x5, cette convolution produit une carte des caractéristiques de sortie de dimension 3x3.

Un réseau de neurones convolutif renferme plusieurs filtres qui sont appliqués à l'image source. Suite à la première phase, nous avons un nombre de nouvelles images équivalent au nombre de filtres. On peut également considérer la phase de convolution comme des couches de neurones dissimulés où chaque neurone est relié uniquement à certains neurones de la couche supérieure.[28]

- Exemple de calcule de convolution :

Dans l'image, la partie gauche montre la matrice des valeurs de pixels d'une image en niveaux de gris (c'est-à-dire la représentation numérique de l'image pour un ordinateur). Au centre se trouve la matrice du filtre de convolution, qui glisse sur l'image d'origine en partant du coin supérieur gauche. Le filtre calcule une valeur à chaque position, et ce processus est répété sur toute l'image. Les valeurs résultantes forment l'image de droite (carte des caractéristiques), qui contient les caractéristiques locales de l'image d'origine obtenues par convolution.

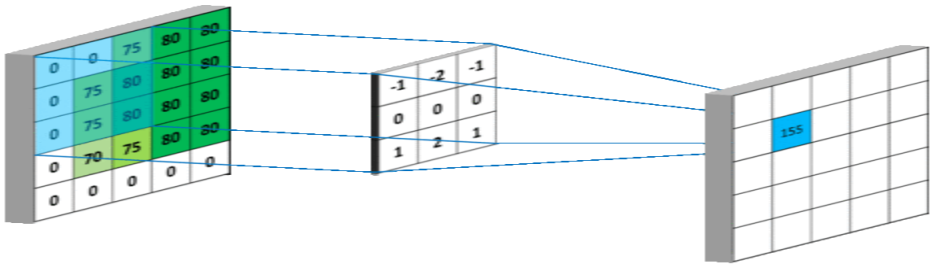


Figure 3.2 : Illustration d'une opération de convolution

Étapes du calcul de convolution :

1. Sélection d'une région de l'image

On choisit un bloc (fenêtre) de taille équivalente au filtre (le filtre de la **figure (3.2)** est de taille 3×3).

Exemple de bloc extrait de la figure (3.2) :

$$\begin{bmatrix} 0 & 0 & 75 \\ 0 & 75 & 80 \\ 0 & 75 & 80 \end{bmatrix}$$

2.application du filtre de convolution :

On applique un produit élément par élément entre les valeurs de l'image et celles du filtre.

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

3.Multiplication élément par élément :

On fait les multiplications :

$$Image * filtre = \begin{bmatrix} 0\times(-1) & 0\times(-2) & 75\times(-1) \\ 0\times 0 & 75\times 0 & 80\times 0 \\ 0\times 1 & 75\times 2 & 80\times 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -75 \\ 0 & 0 & 0 \\ 0 & 150 & 80 \end{bmatrix}$$

$*$: opération de convolution

4.Somme des résultats :

On additionne tous les produits :

$$Somme = 0+0+(-75)+0+0+0+0+150+80 = 155$$

5.Écriture dans la feature map :

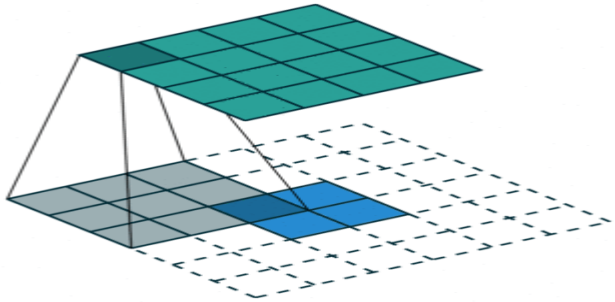
On place la valeur obtenue (ici 155) à la position correspondante dans l'image de sortie (feature map).

6. Glissement (striding)

On décale le filtre vers la droite ou le bas (selon le stride, généralement de 1).

On répète les étapes 1 à 5 jusqu'à avoir balayé toute l'image.

L'animation fonctionne comme ceci :



3.3.2. La couche d'activation ReLU

ReLu est une fonction qui s'applique à chaque pixel d'une image après la convolution, substituant chaque valeur négative par un 0. Si cette fonction n'est pas mise en œuvre, la fonction résultante sera linéaire et le problème XOR subsistera, car aucune fonction d'activation n'est appliquée dans la couche de convolution.

La fonction ReLu est largement adoptée dans les réseaux de neurones convolutifs en raison de sa rapidité de calcul : $f(u) = max\,(0, u)$. Ainsi, sa performance surpasse celle d'autres fonctions nécessitant la réalisation d'opérations onéreuses.

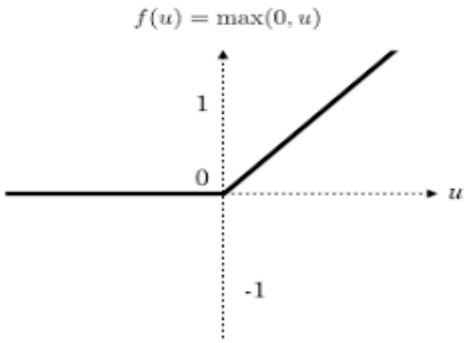
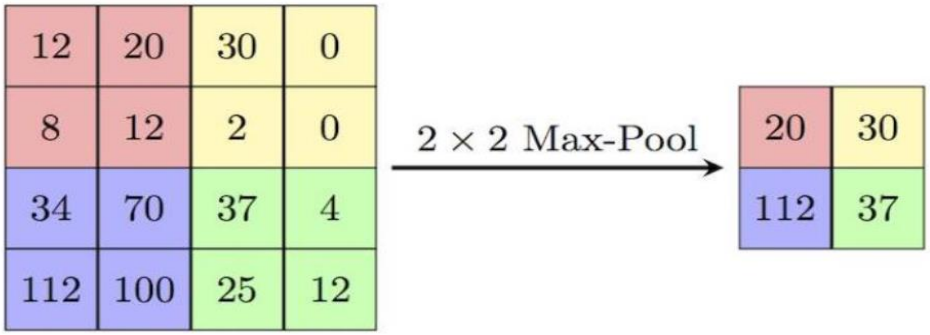


Figure 3.3 :La fonction ReLU

3.3.3. La couche de pooling

On positionne généralement ce genre de couche entre deux couches de convolution : elle reçoit en entrée divers feature maps et met en œuvre l'opération de pooling sur chacun d'eux. L'action de pooling vise à diminuer la dimension des images tout en conservant leurs traits essentiels. Pour ce faire, nous segmentons l'image en cellules uniformes, puis nous conservons la valeur la plus élevée présente dans chaque cellule.

La technique employée fait appel à la création d'une fenêtre de 2 ou 3 pixels qui se déplace sur une image, à l'instar de la convolution. Cependant, cette fois-ci, nous avançons par étapes de 2 pour une fenêtre de taille 2, et par étapes de 3 pour une fenêtre de taille 3. On désigne la dimension de la fenêtre par le terme « kernel size », tandis que les déplacements sont nommés « strides ». À chaque phase, nous sélectionnons la valeur maximale parmi celles présentes dans la fenêtre, et cette valeur devient un pixel dans une nouvelle image. On appelle cela le Max Pooling.[28]



La couche de pooling contribue à diminuer le nombre de paramètres et la quantité de calculs au sein du réseau. Cela améliore donc l'efficacité du réseau tout en prévenant le surajustement. Il est en réalité souvent bénéfique que les valeurs maximales soient identifiées de façon moins précise dans les cartes de caractéristiques post-pooling que dans celles qui sont en entrée. Effectivement, quand on tente d'identifier un chien par exemple, il n'est pas nécessaire de localiser précisément ses oreilles : savoir qu'elles sont à peu près de la tête est suffisant .De ce fait, la couche de pooling rend le réseau moins affecté par la localisation des caractéristiques : le fait qu'une caractéristique soit légèrement plus haute ou plus basse, ou même qu'elle présente une orientation un peu différente ne devrait pas entraîner une modification majeure dans la catégorisation de l'image.

3.3.4. Couche fully-connected :

La dernière couche d'un réseau de neurones est toujours une couche entièrement connectée. Cette catégorie de couche reçoit un vecteur comme entrée et génère un nouveau vecteur en tant que sortie. À cet effet, elle utilise un modèle linéaire suivi éventuellement d'une fonction d'activation sur les valeurs entrantes.

La couche finale entièrement connectée sert à catégoriser l'image d'entrée du réseau : elle produit un vecteur de dimension N, N correspondant au nombre de classes dans notre tâche de classification d'images.

Chaque composante du vecteur représente la probabilité que l'image d'entrée soit classée dans une catégorie spécifique.

Par exemple, si la tâche est de différencier les chats des chiens, le vecteur final aura une dimension de 2 : le premier élément (et le second) indique respectivement la probabilité d'être un « chat » (ou un « chien »). Donc, le vecteur [0.9 0.1] indique que l'image à 90% de probabilité d'être celle d'un chat.

Ainsi, pour estimer les probabilités, la couche entièrement connectée effectue une multiplication de chaque élément d'entrée par un poids, réalise une addition et applique ensuite une fonction d'activation (logistique si N=2, softmax si N>2) : Cette opération consiste à effectuer une multiplication entre le vecteur d'entrée et la matrice qui contient les poids. L'expression « fully-connected » provient du fait que chaque valeur d'entrée est reliée à toutes les valeurs de sortie.[28]

3.4. L'architecture de CNN :

Il est clair que l'image a aujourd'hui une importance considérable dans de nombreux domaines liés à l'IA : détection d'objets, design, jeux, sécurité, santé, recherche et développement...

Habituellement, les images sont analysées à l'aide de réseaux de neurones convolutionnels (CNN), qui sont spécifiquement développés pour le traitement et la reconnaissance d'images en manipulant des données pixelisées.

Dans cette partie, nous allons explorer les couches principales du CNN, puis nous aborderons les diverses transformations et évolutions de sa structure.

L'architecture du CNN peut varier d'une version à l'autre, mais certaines caractéristiques demeurent toujours reconnaissables. Voici les principales :

1. Couche de convolution (Convolutional layer)

Son objectif principal est d'identifier les divers éléments et propriétés de l'image donnée en entrée. Le concept consiste à utiliser le Kernel (également appelé filtre : il détermine l'étendue du champ de vision de la convolution) sur les pixels de l'image, en le déplaçant grâce à un élément de mesure nommé Stride (Stride : il définit la distance parcourue par le noyau lors de son déplacement sur l'image). Le Kernel est utilisé à plusieurs reprises pour déceler davantage de caractéristiques associées à notre entrée. Afin de conserver la taille de la convolution, il est parfois nécessaire d'utiliser le Padding. (Padding : c'est l'action d'entourer l'image de zéros tout autour pour autoriser le Kernel à se déplacer dans toutes les directions, garantissant ainsi une convolution uniforme). L'étape finale consiste à intégrer le biais et à utiliser la fonction Relu pour une meilleure régulation de l.

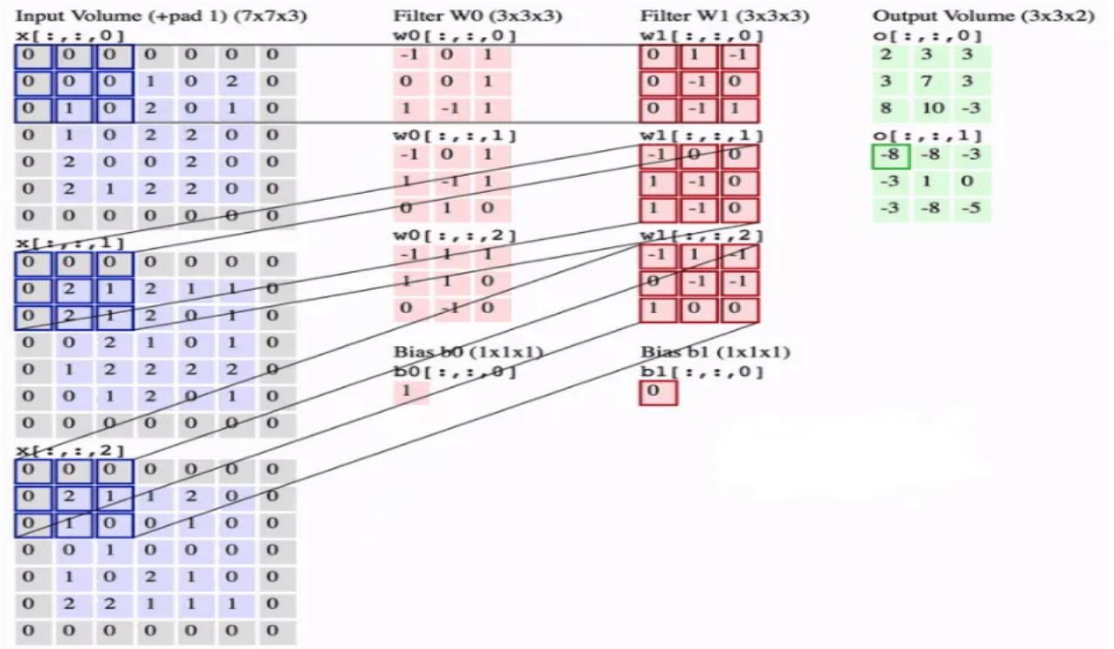


Figure 3.4 : Une démonstration du Conv layer avec 2 Kernels, chaqu'un avec taille Kernel = 3 pixels, Stride= 2 pixels, Padding = 1pixel.

2. Couche de mise en commun (Pooling layer) :

Plusieurs buts sont poursuivis ici : éliminer toutes les fonctionnalités et propriétés redondantes captées lors de la convolution tout en conservant les plus significatives, diminuer graduellement la dimension spatiale de la représentation pour minimiser le nombre d'éléments et le travail computationnel au sein du réseau. Afin de diminuer la taille de l'image, on distingue deux sortes de couches de pooling : le Max Pooling où l'on définit un espace voisinage (un filtre) qui se déplace sur l'entrée et on retient l'élément le plus significatif de la zone couverte par le filtre, et le Average Pooling qui, comme son nom l'indique, conserve la moyenne des valeurs rencontrées dans le filtre.

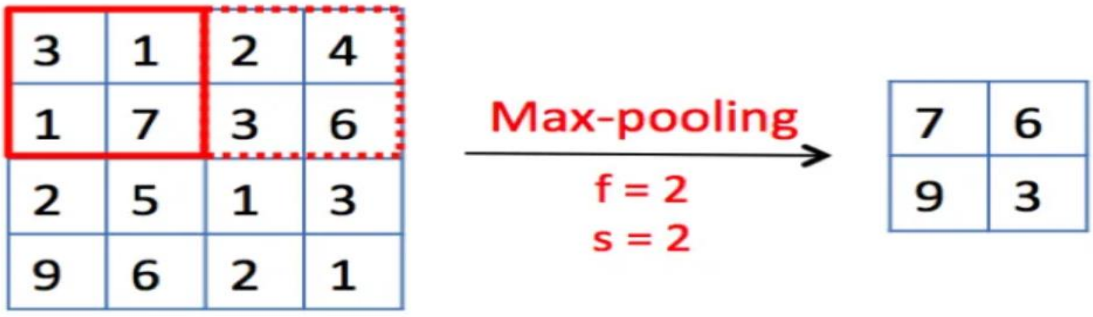


Figure 3.5 : Exemple de Max Pooling.

3. Couche entièrement connectée (Fully-Connected Layer) :

Son objectif est de déterminer la classe de sortie. La couche totalement connectée établit une connexion entre chaque neurone d'une couche et chaque neurone d'une autre couche, tout en diminuant le nombre d'unités de la seconde couche. La couche finale entièrement connectée se sert d'une fonction d'activation nommée Softmax pour catégoriser les caractéristiques produites à partir de l'image d'entrée en fonction des diverses classes disponibles, leur attribuant des probabilités décimales proportionnelles à l'entraînement réalisé avec les ensembles de données d'apprentissage.[29]

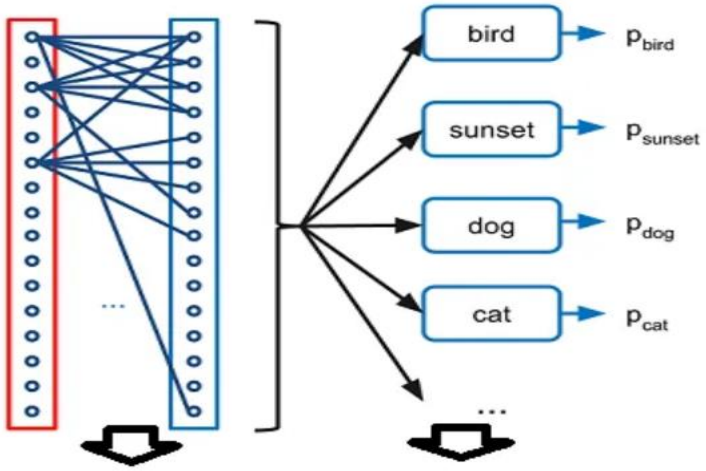


Figure 3.6: Example de couche entièrement connectée

3.5. Les Modèles CNN :

3.5.1. LeNet-5 :

1. Historique :

LeNet a d'abord été employé pour l'identification des chiffres écrits à la main dans les documents bancaires. LeNet-5, dévoilé en 1998 par Yann LeCun et ses collaborateurs, est l'une des premières structures CNN à démontrer l'efficacité de l'apprentissage profond pour résoudre des problématiques complexes de reconnaissance d'images.

2. Objectifs et Innovations

LeNet-5 avait pour but principal de révolutionner la reconnaissance des caractères écrits à la main. Plus précisément, il avait pour objectif de relever les défis liés à la lecture automatique des chiffres présents sur les chèques et les codes postaux. LeNet-5, avec son architecture avant-gardiste, a non seulement perfectionné la précision des systèmes de reconnaissance, mais a aussi ouvert des perspectives pour diverses applications dans des secteurs tels que la numérisation de documents et l'identification d'adresses.

Le modèle LeNet-5 a apporté de nombreuses améliorations significatives en matière d'architecture des réseaux de neurones convolutifs :

- L'application de convolutions pour le repérage de caractéristiques locales.
- L'emploi du pooling pour diminuer la dimensionnalité des cartes de caractéristiques.
- L'arrangement des caractéristiques dans des couches successives.
- LeNet est un CNN classique avec des couches convolutionnelles et des sous-échantillonnages.

Son fonctionnement général peut se résumer ainsi :

$$\boldsymbol{x^{(l)} = \sigma\left(w^{(l)} * x^{(l-1)} + b^{(l)}\right)} \qquad (\mathbf{3.1})$$

- x^l : sortie de la couche l
- $w^{(l)}$: Poids du filtre convolutionnel
- $*$: opération de convolution
- $b^{(l)}$: Biais
- σ: fonction d'activation (ex : tanh ou ReLU)

3. Architecture :

LeNet-5 est constitué de sept couches, comprenant des couches de convolution ainsi que des couches de pooling. Ces couches facilitent l'extraction et le classement des caractéristiques cruciales pour la classification d'images.

Couches de convolution (C1, C3) : Ces couches se chargent de la détection des propriétés à partir des images d'entrée. C1 traite une image de 32x32 pixels et produit 6 cartes de caractéristiques ayant une dimension de 28x28 pixels.

Couches de sous-échantillonnage (S2, S4) : Ces couches diminuent la dimension des cartes de caractéristiques, ce qui permet de condenser l'information tout en préservant les données les plus cruciales.

Couches entièrement connectées (C5, F6) : Ces couches sont utilisées pour la classification. La couche finale produit une sortie qui représente la probabilité que chaque classe (comme les chiffres de 0 à 9) soit la bonne.[30]

Tableau 3.1 : Les couches de LeNet-5.[31]

Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	32×32	-	-	-
1	Convolution	6	28×28	5×5	1	tanh
2	Average Pooling	6	14×14	2×2	2	tanh
3	Convolution	16	10×10	5×5	1	tanh
4	Average Pooling	16	5×5	2×2	2	tanh
5	Convolution	120	1×1	5×5	1	tanh
6	FC	-	84	-	-	tanh
Output	FC	-	10	-	-	softmax

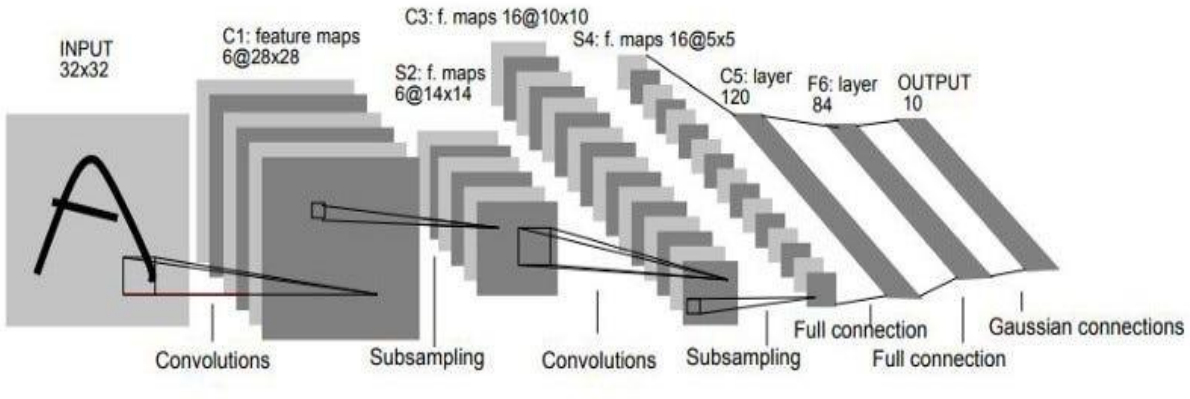


Figure 3.7 : Architecture de LeNet

4. Avantages

Ce réseau est une bonne introduction au monde des réseaux neuronaux et est très simple à comprendre. Il fonctionne bien pour la reconnaissance d'images de caractères.

5. Limitations

Bien que LeNet ait connu du succès, il présente des contraintes concernant sa capacité à gérer des images plus complexes et à des résolutions supérieures. LeNet est spécifiquement conçu pour traiter des images de résolution basse, généralement de 32x32 pixels, ce qui limite son utilisation dans des domaines où des résolutions plus élevées sont requises.

3.5.2. ResNet :

1. Historique

Le Réseau Neuronal Résiduel, plus connu sous l'acronyme ResNet, a été présenté en 2015 par Kaiming He et son équipe de Microsoft Research. Ce modèle a été élaboré pour relever les défis liés à l'apprentissage de réseaux neuronaux d'une grande profondeur, en particulier la question des gradients qui disparaissent. En 2015, ResNet a remporté du concours ImageNet Large Scale Visual Recognition Challenge, prouvant ainsi son efficacité et sa capacité à acquérir

des représentations complexes à partir d'immenses ensembles de données. Depuis sa création, ResNet s'est imposé comme un standard dans le champ du deep learning et a donné naissance à plusieurs versions telles que ResNet-18, ResNet-50, ResNet-101 et ResNet-152.

2. Objectifs

Parmi les objectifs majeurs de ResNet, on trouve :

- Faciliter l'apprentissage de réseaux extrêmement profonds : En intégrant des connexions résiduelles, ResNet offre la possibilité d'ajouter plus de couches tout en évitant les problèmes classiques associés à la profondeur des réseaux.
- Affiner la précision dans des tâches complexes : En formant le réseau à apprendre des fonctions résiduelles, il acquiert la capacité de saisir des caractéristiques plus abstraites et élaborées dans les données.
- Favoriser la capacité à généraliser : ResNet contribue à améliorer les performances sur divers ensembles de données en minimisant le surajustement grâce à une architecture plus robuste.

3. Architecture

L'architecture de ResNet se base sur l'insertion de « skip connections » (ou connexions directes), facilitant le passage de l'information et des gradients à travers plusieurs couches sans déformation.

Chaque bloc résiduel peut s'écrire sous la forme suivante :

$$\boldsymbol{y = F(x, \{w_i\}) + x} \qquad \textbf{(3.2)}$$

Où $F(x)$ représente une fonction correspondant à deux ou trois couches de convolution, et x désigne l'entrée initiale [32].

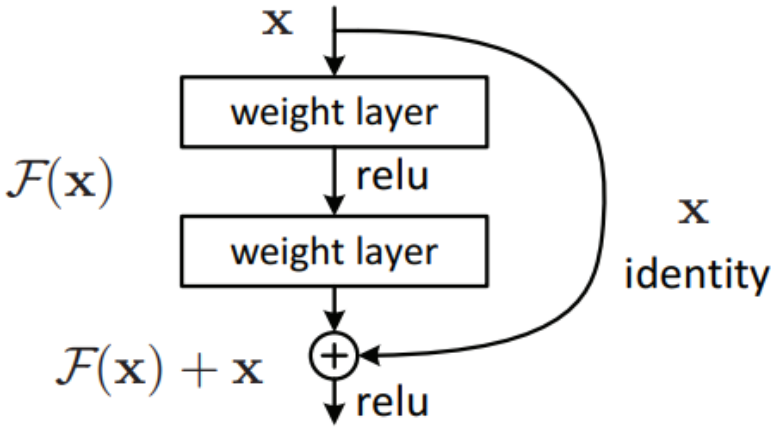


Figure 3.8 : Connexions en saute

La version ResNet-18 compte 18 couches, la version ResNet-50 compte 50 couches, celle de ResNet-101 en comporte 101, tandis que la ResNet-152 en possède 152.

- **Comparaison des modèles ResNet : ResNet18, ResNet50 et ResNet101**
  - ResNet18 : s'appuie sur des blocs simples comportant deux couches convolutionnelles, ce qui en favorise une exécution plus rapide et plus légère, idéale pour les missions moins sophistiquées ou les systèmes aux ressources restreintes.
  - ResNet50 et ResNet101 : mettent en œuvre des blocs bottleneck plus profonds (3 couches de convolution) afin d'optimiser la faculté d'apprentissage tout en maîtrisant le coût computationnel.
  - Avec une profondeur considérablement plus importante, ResNet101 a la capacité de tirer des caractéristiques très précises, cependant cela nécessite plus de temps et de ressources pour le calcul.

Voici le tableau d'analyse comparative des modèles Resnet en fonction de la profondeur et des blocs utilisés.

Tableau 3.2 : Analyse comparative des modèles ResNet

Modèle	Profondeur	Nombre de couches résiduelles	Type de bloc utilisé	Nombre de paramètres (approx.)
ResNet-18	18 couches	8 blocs résiduels simples.	Basic Bloc (2 couches conv)	~11 millions
ResNet-50	50 couches	16 blocs résiduels	Bottleneck Block (3 couches conv)	~25 millions
ResNet-101	101 couches	33 blocs résiduels	Bottleneck Block (3 couches conv)	~44 millions

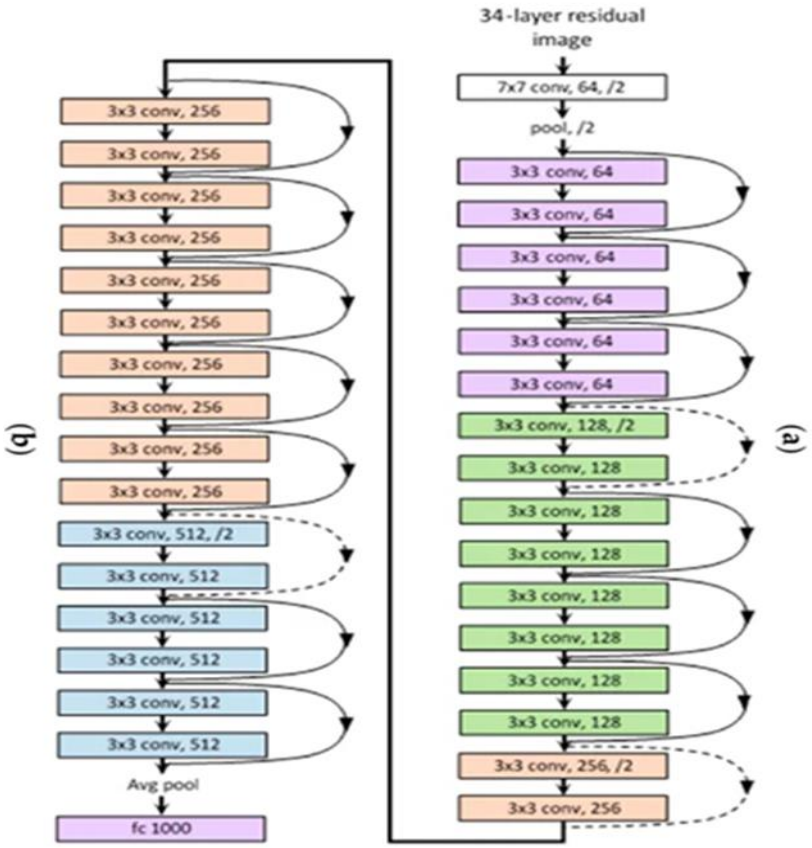


Figure 3.9 : Architecture de ResNet

4. Nouveautés apportées par ResNet
    - Apprentissage facilité dans les réseaux neuronaux profonds.
    - Stabilisation du gradient de descente.
    - Éradication du surapprentissage approfondi.
    - Excellentes performances sur ImageNet, COCO et CIFAR.

Ces améliorations ont fait de ResNet une référence dans les architectures contemporaines dédiées à la vision par ordinateur.

5. Fonctionnement des blocs résiduels

Un bloc résiduel typique comprend souvent deux ou trois couches de convolution, suivi d'une opération d'addition avec l'entrée du bloc. Cela facilite la conservation de l'information à travers les différentes strates. Des variantes telles que ResNet-Bottleneck emploient une combinaison de 1×1, 3×3, et 1×1 pour minimiser les calculs.[33]

3.5.3. Xception :

1. Historique et Contexte

François Chollet a présenté le modèle Xception (Extreme Inception) en 2016. C'est une amélioration du modèle Inception v3 qui repose sur l'idée que les convolutions traditionnelles peuvent être efficacement substituées par des convolutions séparables en profondeur (depthwise separable convolutions). L'idée se base sur une supposition robuste : il est possible que la corrélation spatiale et la corrélation des canaux soient totalement indépendantes l'une de l'autre.

2. Objectifs et innovation

Diminuer la quantité de paramètres sans compromettre ou en améliorant les performances.

Optimiser la performance du traitement d'images en diminuant la complexité de calcul. Faciliter un apprentissage plus rapide et des modèles plus approfondis via une séparation améliorée des données spatiales et des canaux.

Xception repose sur la convolution séparée en profondeur (depthwise separable convolution).

Equation (Xception Block) :

$$\boldsymbol{y = Pointwise\big(Depthwise(x)\big)} \qquad (\mathbf{3.3})$$

- Depthwise : convolution indépendante sur chaque canal.
- Pointwise : convolution 1×1pour combiner les canaux.

3. Architecture

➢ Bloc d'entrée (Entry Flow)

Convolutions traditionnelles avec un sous-échantillonnage maximal (max pooling) Préparation des caractéristiques pour le deep learning.

➢ Bloc central (Flux central)

Constitué de plusieurs blocs identiques.

Chaque bloc est composé de trois niveaux de convolutions séparables par profondeur.

Chaque couche est suivie d'une normalisation par lot et d'une activation ReLU.

C'est la section la plus profonde du réseau.

➢ Flux de sortie (Exit Flow)

Dernière réduction et classification

Intègre des convolutions traditionnelles pour le passage à la sortie.[34]

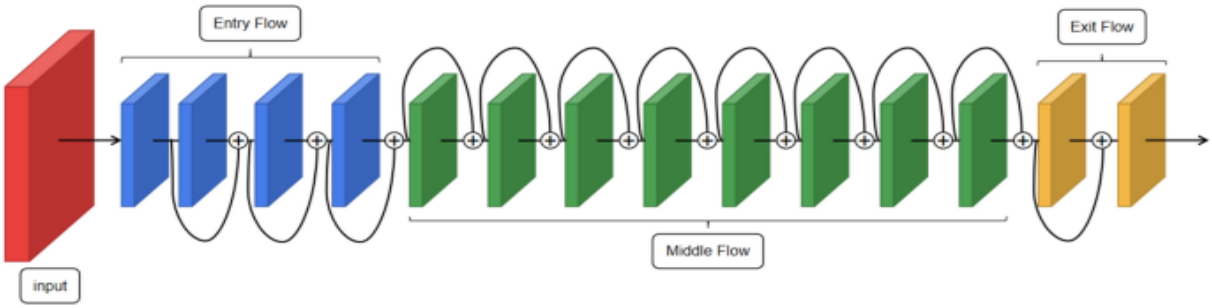


Figure 3.10 : Architecture de Xception

4. Nouveauté Apportée par Xception
   - Première structure architecturale entièrement fondée sur les convolutions séparables.
   - Constitue une version fortement simplifiée du modèle Inception, éliminant les voies de convolution parallèles.
   - Performance supérieure à Inception v3 sur ImageNet et divers autres ensembles de données.

3.5.4. DenseNet :

1. Historique

En 2017, Gao Huang, Zhuang Liu, Laurens van der Maaten et Kilian Q. Weinberger ont suggéré le modèle DenseNet (Réseaux de Convolution Densément Connectés). C'est une avancée des architectures profondes (à l'instar de ResNet) destinée à optimiser la réutilisation des caractéristiques (features) en créant des liaisons directes entre toutes les couches d'un bloc dense.

2. Objectifs et innovation
   - Optimiser la diffusion des gradients dans les réseaux de neurones profonds.
   - Diminuer l'overfitting par le biais d'un effet de régularisation.
   - Réduire le nombre de paramètres en réemployant les caractéristiques acquises.
   - Faciliter l'apprentissage de réseaux très profonds.
   - DenseNet relie chaque couche à toutes les précédentes via concaténation.[35]

Sa formule:

$$\boldsymbol{x^l = H^{(l)}\left(\left[x^{(l)}, x^{(1)}, \ldots, x^{(l-1)}\right]\right)} \qquad \boldsymbol{(3.4)}$$

Où : $\boldsymbol{\left[x^{(l)}, x^{(1)}, \ldots, x^{(l-1)}\right]}$: concaténation des sorties précédentes

$\boldsymbol{H^{(l)}}$: BN → ReLU → Conv.[36]

3. Architecture

Un schéma simplifié de l'architecture typique d'un modèle DenseNet.

Tableau 3.3 : Un schéma simplifié de l'architecture DenseNet

Composant	Description
Entrée	Image d'entrée (par exemple, 224 × 224 pixels).
Blocs Denses	Composés de plusieurs couches convolutionnelles avec connexions denses. Chaque couche utilise les sorties de toutes les couches précédentes comme entrées.
Couches de Normalisation	Intègre souvent la normalisation par lots (Batch Normalization) pour stabiliser l'apprentissage.
Couches Entièrement Connectées	À la fin du réseau, ces couches produisent une distribution de probabilités pour la classification finale.

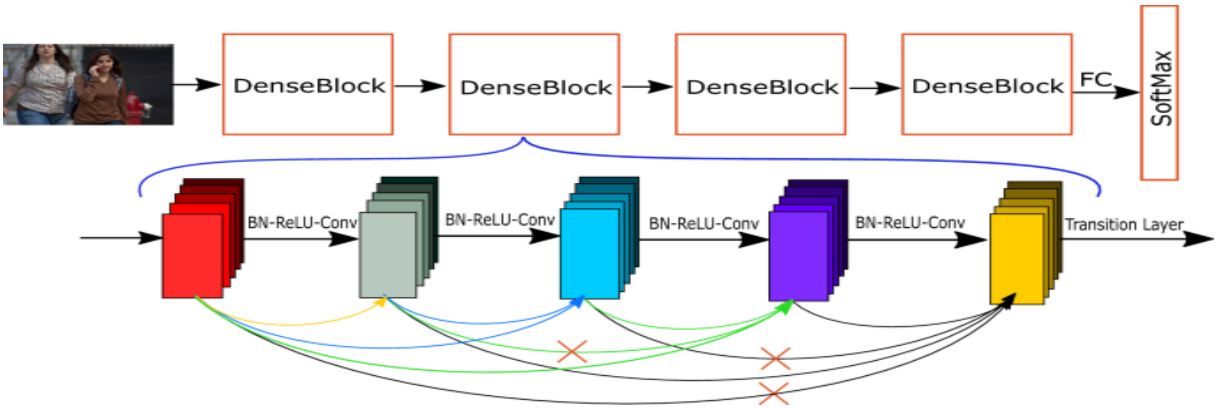


Figure 3.11 : Architecture de DenseNet

4. Nouveautés Apportées par DenseNet
   - Connexions denses : chaque couche reçoit toutes les sorties de la couche précédente en entrée.
   - Pas d'addition (comme ResNet), mais plutôt une concaténation des caractéristiques.
   - Chaque couche ne contribue que légèrement à l'ajout de nouvelles caractéristiques. Modèle plus compact et performant.
5. Avantages et Performances
   - Propagation efficace du gradient
   - Réutilisation des caractéristiques
   - Moins de paramètres que ResNet pour des performances similaires, voire supérieures
   - Moins de surapprentissage sur petits jeux de données

- Bonne généralisation

3.5.5. AlexNet :

1. Historique

AlexNet appartient parmi les modèles les plus marquants de l'histoire des réseaux de neurones convolutifs. Conçu par Alex Krizhevsky, Ilya Sutskever et Geoffrey Hinton, ce modèle a triomphé au concours ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 avec une large marge. Ce modèle a ravivé l'engouement pour l'apprentissage profond (deep learning) dans le domaine de la vision par ordinateur.

2. Architecture

AlexNet comprend 8 couches : cinq couches de convolution (Conv) et trois couches complètement reliées (FC).

Tableau 3.4 : Architecture détaillée du modèle AlexNet

Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	227×227×3	-	-	-
1	Convolution	96	55×55×96	11×11	4	relu
	Max Pooling	96	27×27×96	3×3	2	relu
2	Convolution	256	27×27×256	5×5	1	relu
	Max Pooling	256	13×13×256	3×3	2	relu
3	Convolution	384	13×13×384	3×3	1	relu
4	Convolution	384	13×13×384	3×3	1	relu
5	Convolution	256	13×13×256	3×3	1	relu
	Max Pooling	256	6×6×256	3×3	2	relu
6	FC	-	9216	-	-	relu
7	FC	-	4096	-	-	relu
8	FC	-	4096	-	-	relu
Output	FC	-	1000	-	-	softmax

AlexNet suit une architecture traditionnelle de convolution + ReLU + pooling + FC (fully connected). Sa nouveauté inclut l'usage massif de ReLU et de dropout.

Sa formule est :

$$\boldsymbol{x^{(l)} = ReLU\left(w^{(l)} * x^{(l-1)} + b^{(l)}\right)} \qquad (\mathbf{3.5})$$

Même forme que LeNet, mais avec ReLU et dropout dans les couches entièrement connectées [36]

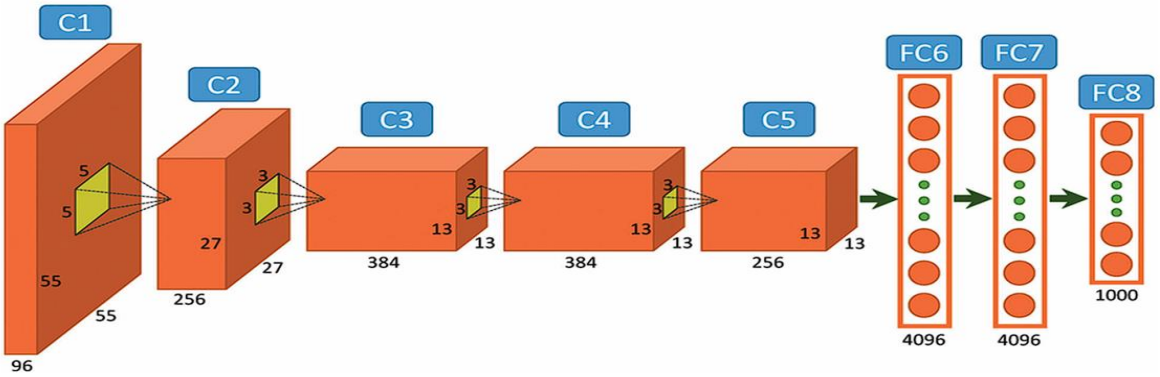


Figure 3.12 : Architecture de AlexNet

3. Objectifs et Contributions
    - Faciliter l'apprentissage des réseaux neuronaux profonds en utilisant des GPU pour la formation
    - Utiliser des méthodes telles que le Dropout pour minimiser le surapprentissage.
    - L'utilisation de ReLU pour améliorer la vitesse d'entraînement par rapport aux fonctions d'activation traditionnelles (tanh, sigmoid) est recommandée.
    - Optimiser les performances sur des bases de données volumineuses telles qu'ImageNet.
4. Nouveautés apportées par AlexNet
    - Utilisation intensive de GPU pour l'apprentissage profond.
    - ReLU comme fonction d'activation rapide et efficace.
    - Dropout introduit dans les couches entièrement connectées.
    - Data augmentation (recadrage, inversion d'images).
    - LRN (Local Response Normalization) pour augmenter le contraste.

3.5.6.VGG :

1. Historique

VGG est un modèle de réseau de neurones convolutionnels développé par K. Simonyan et A. Zisserman à l'Université d'Oxford. Il a été présenté lors de la compétition ILSVRC (ImageNet Large Scale Visual Recognition Challenge) en 2014, où il a atteint une précision remarquable de 92,7 % dans la classification d'images, surpassant de nombreux modèles précédents. Ce modèle a été conçu pour tirer parti de l'architecture profonde tout en maintenant une simplicité dans sa structure, ce qui lui a permis de devenir un standard dans le domaine de la vision par ordinateur.

2. Objectifs

L'objectif principal du modèle VGG était d'améliorer la performance des réseaux de neurones dans la reconnaissance d'images en utilisant une architecture plus profonde et plus uniforme. En introduisant des noyaux de convolution de petite taille (3x3), VGG a permis d'apprendre des caractéristiques plus complexes tout en maintenant un nombre réduit de paramètres. Ce modèle a également été conçu pour être facilement adaptable à d'autres tâches de classification d'images, ce qui en fait un choix populaire pour le transfert learning.

3. Architecture

L'architecture de VGG se caractérise par sa profondeur et sa simplicité. Les deux variantes les plus connues sont VGG16 et VGG19, qui contiennent respectivement 16 et 19 couches. Voici un aperçu détaillé de l'architecture typique de VGG16 :

tableau 3.5 :Architecture détaillée du réseau VGG16 :couches, dimensions et fonctions

Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	224×224×3	-	-	-
1	2 × Convolution	64	224×224×64	3×3	1	relu
	Max Pooling	64	112×112×64	3×3	2	relu
3	2 × Convolution	128	112×112×128	3×3	1	relu
	Max Pooling	128	56×56×128	3×3	2	relu
5	2 × Convolution	256	56×56×256	3×3	1	relu
	Max Pooling	256	28×28×256	3×3	2	relu
7	3 × Convolution	512	28×28×512	3×3	1	relu
	Max Pooling	512	14×14×512	3×3		relu
10	3 × Convolution	512	14×14×512	3×3	1	relu
	Max Pooling	512	7×7×512	3×3	2	relu
13	FC	25088	25088	-	-	relu
14	FC	4096	4096	-	-	relu
15	FC	4096	4096	-	-	relu
Output	FC	1000	1000	-	-	softmax

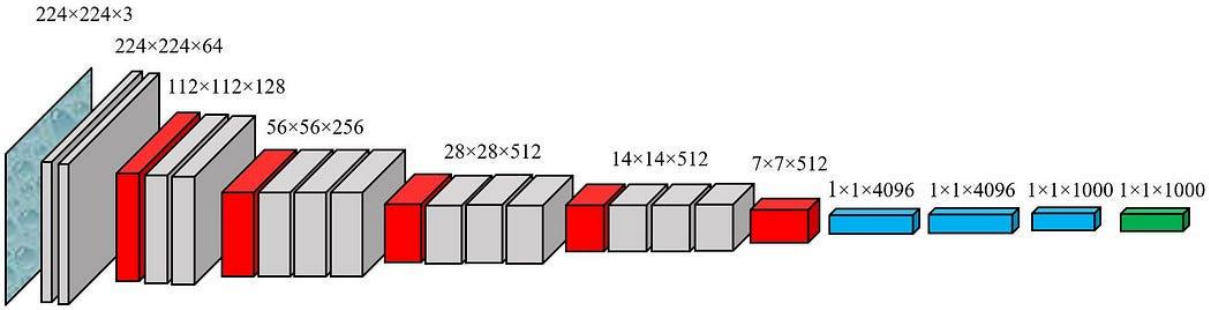


Figure 3.13 : architecture de VGG

4. Caractéristiques Clés
    - Noyaux de Convolution Petit (3x3) : Permettent une meilleure extraction des caractéristiques tout en réduisant le nombre total de paramètres.
    - Max-Pooling (2x2) : Utilisé après certaines couches convolutionnelles pour réduire la taille des représentations tout en conservant les informations essentielles.
    - Couches Entièrement Connectées : Les dernières couches sont entièrement connectées pour effectuer la classification finale.

3.5.7. MobileNet

1. Historique

MobileNet est une série de modèles de réseaux de neurones convolutifs conçus pour être utilisés sur des appareils mobiles et dans des systèmes intégrés, élaborée par Google. Le modèle initial, MobileNetV1, a fait son apparition en 2017, suivi par MobileNetV2 en 2018 et MobileNetV3 en 2019. Ces modèles ont été élaborés pour minimiser la latence, la consommation d'énergie et l'empreinte mémoire, tout en préservant des performances solides en matière de classification et de détection.

2. Objectifs
    - Proposer un réseau allégé pour les dispositifs mobiles.
    - Conserver des performances concurrentielles dans la classification d'images.
    - Diminué le nombre de paramètres et la complexité de calcul.
    - Capacité d'adaptation aisée à d'autres missions : identification d'objets, segmentation, reconnaissance de visages.
3. Architecture

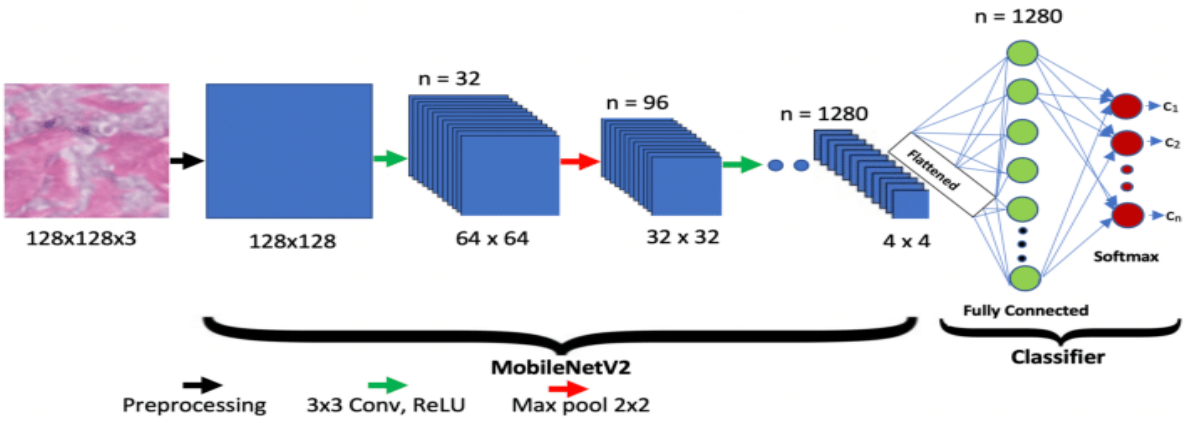


Figure 3.14 : Architecture de MobileNet

4. Nouveautés apportées par MobileNet

Convolutions séparables par profondeur : une méthode qui décompose une convolution standard en deux phases distinctes :

4.1. Depthwise :

C'est une architecture améliorant un CNN. Afin de réduire la complexité de calcul élevée requise par le CNN, Google utilise une nouvelle structure de calcul dans MobileNet, visant à diminuer le coût de calcul d'un modèle CNN.

MobileNet est une architecture CNN plus rapide et un modèle plus compact intégrant une nouvelle couche convolutionnelle appelée DSC (Depthwise Separable Convolution). Grâce à leur taille compacte, ces modèles sont particulièrement adaptés à une implémentation sur appareils mobiles et embarqués. Cette approche divise principalement la convolution initiale en deux parties distinctes : DWC (Depth Wise Convolution) et PWC (Point Wise Convolution), permettant de réduire les besoins de calcul sans compromettre la structure sous-jacente.

La différence entre le DSCNN et le CNN conventionnel réside dans l'approche DWC. Dans DWC, le noyau de convolution est divisé en une forme monocanale (une convolution appliquée à chaque canal d'entrée individuellement), comme illustré à **la figure (3.15).** Pour chaque canal des données d'entrée, un filtre de taille k est établi, et des opérations de convolution distinctes sont effectuées sur chaque canal sans modifier la profondeur de l'image d'entrée. Cependant, ce processus limite l'extension de la carte de caractéristiques, restreignant ainsi sa dimensionnalité. Par conséquent, l'utilisation de PWC devient nécessaire pour combiner les résultats du DWC et générer une nouvelle carte de caractéristiques.

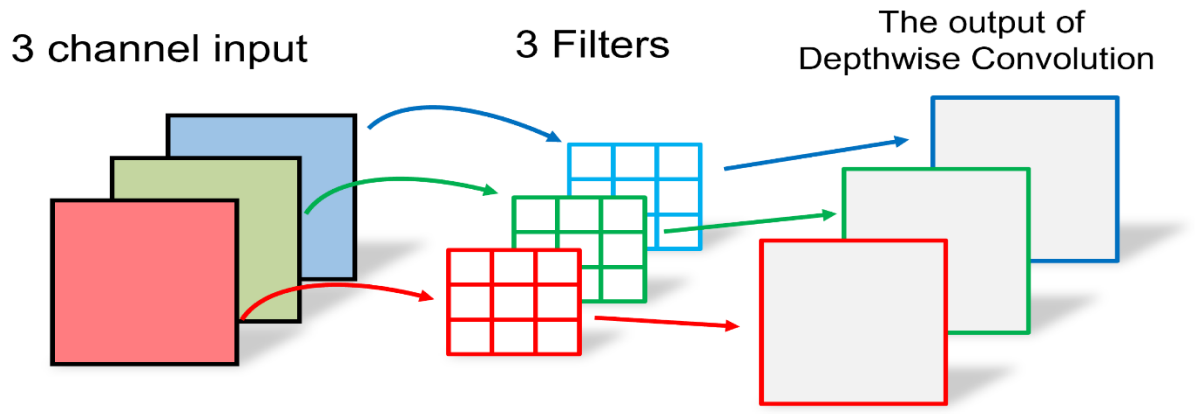


Figure 3.15: Illustration de convolutions séparables en profondeur. [37]

4.2. Pointwise :

Est un type de convolution utilisant un noyau 1x1 : un noyau itérant sur chaque point. La profondeur de ce noyau est égale au nombre de canaux de l'image d'entrée. Elle peut être utilisée

conjointement avec les convolutions en profondeur pour produire une classe efficace de convolutions appelées convolutions séparables en profondeur.

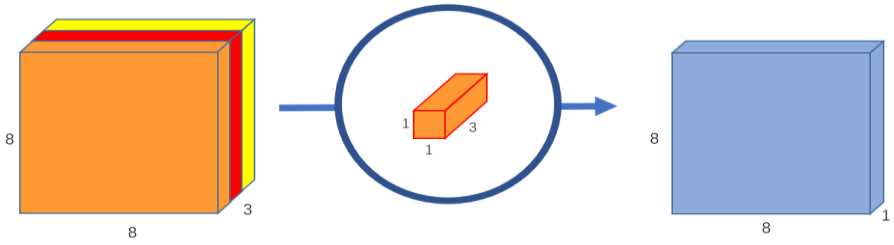


Figure 3.16 : Réseau neuronal convolutif ponctuel

La formule de MobileNetV1 :

$$\boldsymbol{y = Pointwise\big(Depthwise(x)\big)} \qquad \boldsymbol{(3.6)}$$

La formule de MobileNetV2 :

$$\boldsymbol{y = x + Pw_{linear}Dw\left(Pw_{expand}(x)\right)} \qquad \boldsymbol{(3.7)}$$

- PW = pointwise, DW = depthwise
- Résidu ajouté si dimensions compatibles

5. Hyperparamètres ajustables

Multiplicateur de largeur (α) : destiné à diminuer le nombre de canaux.

Multiplicateur de Résolution (ρ) : utilisé pour diminuer la taille de l'image d'entrée.

ReLU6 est utilisé, étant optimisé pour le calcul à faible précision sur les dispositifs mobiles.[38]

3.5.8. EfficientNet :

1. Historique

Mingxing Tan et Quoc V. Le, chercheurs de Google AI, ont présenté EfficientNet en 2019. Ce modèle a été élaboré pour répondre au besoin d'architectures de réseaux neuronaux qui soient précisément efficaces tout en étant aussi économes en ressources. EfficientNet a rapidement gagné en notoriété, en dépassant les performances des modèles antérieurs pour des tâches liées à la vision par ordinateur tout en diminuant significativement les besoins en puissance de calcul. Il a été élaboré dans le contexte d'une étude visant à améliorer les structures des réseaux neuronaux, en se servant de méthodes sophistiquées telles que la recherche d'architecture neuronale (Neural Architecture Search, NAS).

2. Objectifs et innovation

Parmi les objectifs fondamentaux d'EfficientNet apparaissent :

- Optimisation de l'efficience : Élaborer un modèle qui garantit une grande précision tout en exigeant moins de puissance de calcul par rapport aux structures antérieures.

- Uniformité de la scalabilité : Mettre en place une technique d'échelonnement qui modifie simultanément la profondeur, la largeur et la résolution du réseau, favorisant une utilisation plus efficace des ressources.
- Efficacité dans les tâches de vision par ordinateur : Viser à égaler ou surpasser les performances des modèles actuels sur des références comme ImageNet, tout en étant plus léger et plus rapide.
- EfficientNet repose sur le Compound Scaling : il augmente simultanément la profondeur d, la largeur w et la résolution d'entrée r.

Sa formule:

$$\boldsymbol{depth: d = \alpha^{\emptyset}\ , width: w = \beta^{\emptyset}\ , resolution: r = \gamma^{\emptyset}} \qquad (3.8)$$

Sous la contrainte :

$$\boldsymbol{\alpha * \beta^2 * \gamma^2 \approx 2\ et\ \alpha, \beta, \gamma > 0} \qquad (3.9)$$

- ϕ : coefficient de scaling global
- α, β, γ: constantes déterminées par grid search.[39]

3. Structure Générale

Un schéma simplifié de l'architecture typique d'un modèle EfficientNet :

Tableau 3.6: Composants principaux de l'architecture EfficientNet et leurs descriptions

Composant	Description
Entrée	Images ajustées (par exemple, 224 × 224 pixels).
Couches Convolutionnelles	L'extraction des caractéristiques se fait à l'aide de convolutions 3×3 et 1×1.
Blocs MBConv	Constitués de plusieurs couches associées par des connexions résiduels dans le but d'améliorer l'efficacité.
Couches Entièrement Connectées	Ces couches, en fin de réseau, génèrent une distribution de probabilités pour la classification finale.

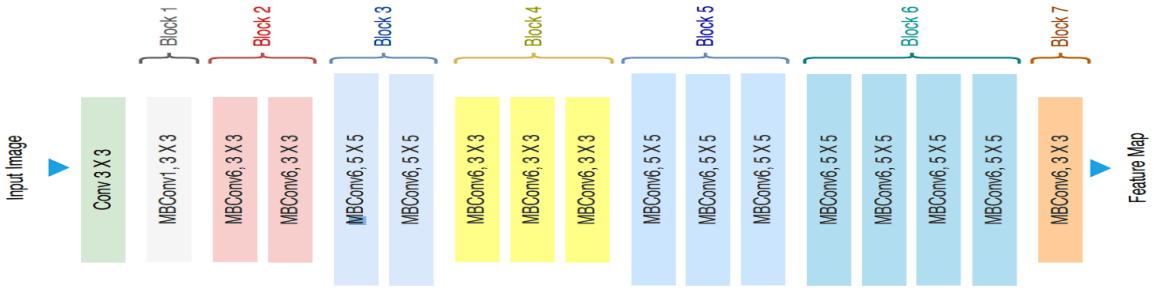


Figure 3.17 : Architecture de EfficientNet

3.6. Conclusion :

Ce chapitre a permis de poser les bases essentielles relatives aux réseaux de neurones convolutifs (CNN), en définissant leur structure fondamentale à travers l'étude de leurs différentes couches et de leur architecture globale. Nous avons vu comment les CNN exploitent la convolution, le pooling, l'activation, et les connexions entre couches pour extraire efficacement des caractéristiques hiérarchiques à partir des données visuelles.

Par ailleurs, l'analyse des principaux modèles CNN, allant de LeNet-5, précurseur historique, jusqu'à des architectures récentes et sophistiquées telles que ResNet, Xception, DenseNet, VGG, MobileNet, AlexNet et EfficientNet, a illustré l'évolution constante des approches visant à améliorer la précision, la profondeur, la rapidité et la légèreté des réseaux.

Chapitre 4 :
Implémentation

4.1.Introduction :

Ce chapitre se propose de comparer l'efficacité de divers modèles profonds fondés sur les réseaux de neurones convolutifs (CNN), en les appliquant à la classification d'images. Plus précisément, nous traiterons des images de chiffres manuscrits, d'images de chiffres présentes sur des photos de panneaux de maisons extraites de Google Street View et d'images de vêtements. Ce travail vise à comparer divers modèles de réseaux de neurones convolutifs (CNN) telle que AlexNet, Resnet18, Resnet50, Resnet101, xception et mobileNet pour identifier celui qui présente les meilleures performances dans le domaine de la classification d'images. Ces modèles seront formés sur les données identiques et leurs rendements seront jugés à l'aide de divers critères, y compris la moyenne de précision, la précision globale, ainsi que le coefficient Kappa. L'objectif final est de déterminer le modèle le plus efficace de façon objective.

4.2.Présentation des modèles CNN utilisés :

- AlexNet : est parmi les premiers modèles de réseaux de neurones convolutifs profonds qui ont marqué l'histoire dans le domaine de la vision par ordinateur. Lancé en 2012, il a remporté le concours ImageNet cette année-là avec une précision largement supérieure à celle des autres modèles. Il comprend 8 couches (5 de convolution et 3 entièrement connectées) et exploite des fonctions d'activation ReLU pour rendre l'apprentissage plus rapide. AlexNet utilise aussi des méthodes telles que le dropout pour prévenir le surapprentissage et la normalisation locale (LRN).
- ResNet18 : fait partie de la famille ResNet (au sens de Residual Network), apparue en 2015. Ce modèle est constitué de 18 couches avec des blocs résiduels, qui est un type de module permettant de contourner le problème du gradient qui disparait dans les réseaux de neurones profonds. Les blocs résiduels se construisent par une connexion directe (ou skip connection) entre l'entrée et la sortie du bloc, ce qui stabilise l'apprentissage. ResNet18 est léger, rapide, et propice à une utilisation dans des applications temps réel ou des bases de données de taille modérée.
- ResNet50 : ResNet50 est une version plus profonde de la même architecture ResNet (50 couches), qui utilise des bottleneck blocks, une version plus compacte et efficace des blocs résiduels. Cela contribue à améliorer les performances sans trop alourdir le réseau, sa profondeur et sa structure optimisée lui permettent d'apprendre des représentations plus complexes et d'obtenir de très bons résultats avec de grandes bases d'images comme ImageNet. C'est un excellent compromis entre précision et efficacité.

- ResNet101 : va encore plus loin avec un réseau dont la profondeur est de 101 couches. Il s'inspire de ResNet50, mais prévoit d'intégrer plusieurs blocs pour améliorer sa capacité d'apprentissage. On se sert généralement de ce modèle pour des missions complexes requérant une grande précision, comme la détection d'objets ou la segmentation d'images. Cependant, les calculs nécessitent plus de ressources et le processus d'entraînement ainsi que l'inférence prend également plus de temps.
- Xception : est un modèle introduit en 2017 par Google. Ce modèle s'inspire de l'architecture Inception tout en la perfectionnant par le recours aux convolutions séparables en profondeur (depthwise separable convolutions), une technique qui permet d'examiner chaque canal de l'image individuellement avant de fusionner les résultats. Cette solution permet au réseau de bénéficier d'une plus grande vitesse et efficacité, sans nuire aux performances. Xception est également reconnu pour produire d'excellentes performances avec un nombre restreint de paramètres. Ce qui en fait un candidat idéal pour des systèmes efficaces, mais moins gourmands.
- MobileNet : est une autre architecture conçue par Google, pour les appareils mobiles. Elle utilise aussi la convolution séparable en profondeur pour diminuer la complexité en calculs à réaliser et la taille du modèle construit. MobileNet est un modèle très léger et rapide qui nécessite peu de ressources, ce qui le rend parfaitement adapté aux applications en temps réel sur les smartphones, les drones ou les objets connectés. Bien que ce modèle soit moins précis que les plus grands modèles, il demeure efficace pour accomplir des tâches de classification standards.

4.3.Logiciels utilisés : MATLAB

MATLAB est un outil de programmation et de calcul numérique employé par des millions d'ingénieurs et scientifiques pour l'analyse de données, le développement d'algorithmes et la création de modèles.

Il combine un espace de travail conçu pour les processus d'analyse itérative et de conception, avec un langage de programmation qui permet d'exprimer directement les mathématiques à travers des tableaux et des matrices.

4.4. Description des bases de données :

4.4.1. La base de données MNIST :

La base de données MNIST modifiée du National Institute of Standards and Technology est une importante collection de chiffres écrits à la main, fréquemment employée pour former différents systèmes d'analyse d'images et modèles d'apprentissage automatique.

La base de données MNIST dédiée aux chiffres écrits à la main comprend un lot d'entraînement de 60 000 exemples, ainsi qu'un lot test comportant 10 000 exemples, de diffèrent chiffre de 0 à 9. Les chiffres sont normalisés en termes de taille et positionnés au centre d'une image de dimensions fixes. Chaque image dans la base de données MNIST est en niveaux de gris, avec une résolution de 28x28 pixels.

Voici **(Figure 4.1)** une représentation d'image de chiffres provenant de notre base de données :

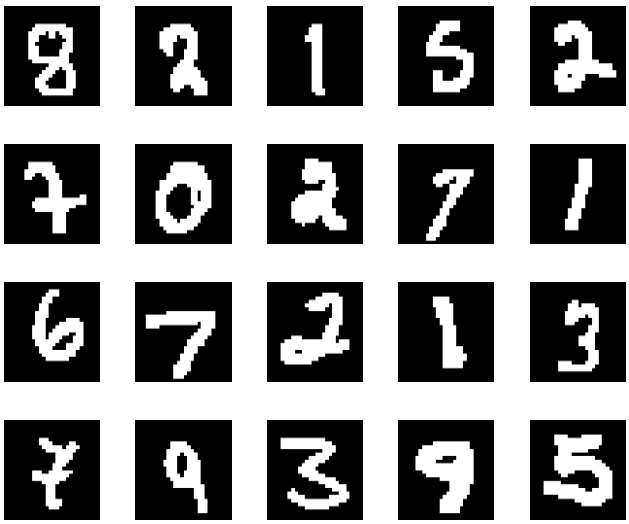


Figure 4.1 : Représentation d'image de chiffres provenant de notre base de données.

4.4.2. La base de données Fashion-MNIST :

L'ensemble de données Fashion-MNIST est une base de données d'images d'articles Zalando (une entreprise allemande spécialisée dans la vente de vêtements, chaussures et accessoires en ligne). Il se compose d'un ensemble d'apprentissage de 60 000 exemples et d'un ensemble de test de 10 000 exemples. Chaque exemple est une image en niveaux de gris de 28 x 28, associée à une étiquette de 10 catégories. Voici la **Figure (4.2)** des exemples de jeu de données Fashion-MNIST.

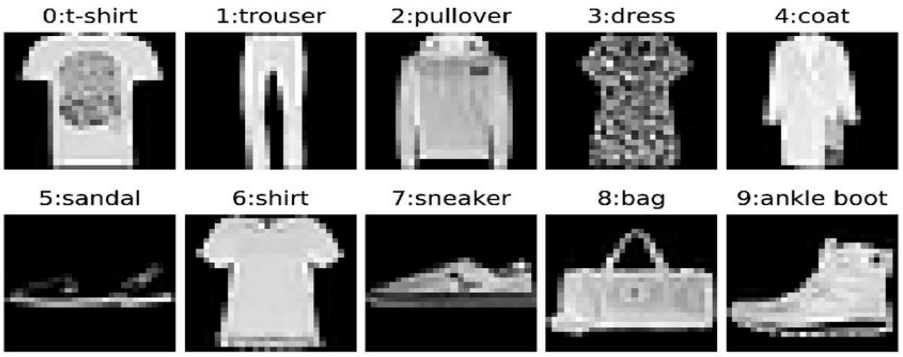


Figure 4.2 : Exemples de jeu de données Fashion-MNIST

4.4.3. La base de données (SVHN) :

Street View House Numbers (SVHN) est un jeu de données de référence pour la classification de chiffres. Il contient 600 000 images RGB de taille 32×32 représentants des chiffres imprimés (de 0 à 9), extraits de photos de plaques de numéros de maisons. Les images sont recadrées de manière à centrer le chiffre principal, mais les chiffres voisins ainsi que d'autres éléments perturbateurs sont conservés dans l'image.

Le jeu de données SVHN est divisé en trois ensembles : un ensemble d'entraînement, un ensemble de test, et un ensemble supplémentaire (extra set) contenant 530 000 images moins complexes, qui peuvent être utilisées pour faciliter le processus d'apprentissage.

Voici La **figure (4.3)** représente des échantillons de l'ensemble de données Street View House Numbers.

Figure 4.3: Echantillons de l'ensemble de données Street View House Number

4.5. L'algorithme d'apprentissage pour les modèles CNN :

Ce schéma présente les étapes clés du processus de formation d'un modèle de réseau de neurones convolutif (CNN) dans le but de réaliser la classification d'images. Le processus débute par le chargement et le prétraitement des données, poursuit avec la définition de l'architecture du modèle, sa compilation, l'apprentissage par rétropropagation, l'évaluation sur les données de test et conclut par la comparaison des performances entre différents modèles.

1). Chargement de prétraitement des données
2). Définition de l'architecture du CNN
3). Compilation de modèle (Perte, optimisateur, métriques)
4). Entraînement du modèle (Par rétropropagation)
5). Evaluation sur les données de test
6). Comparaison des modèles

Figure 4.4 : Les étapes de l'algorithme d'apprentissage pour les différents modèles CNN.

4.6. Optimisateur et paramètres utilisés :

Dans ce contexte, nous avons utilisé empiriquement l'optimiseur Adam sans étude préalable dans la tâche de classification d'images. Le taux d'apprentissage initial a été fixé à 1e-3 pour les trois bases de données MNIST, Fashion-MNIST, et SVHN. On fixe le nombre maximal d'époques d'entraînement à 10 pour MNIST et Fashion-MNIST, et à 5 pour SVHN afin d'assurer une convergence rapide du modèle, sans surapprentissage, en étant conscient qu'un nombre raisonnable pourrait être adéquat, aussi en tenant compte de la puissance de calcul limitée de notre PC. La taille du mini-batch est de 128 permet d'exploiter la puissance de calcul tout en garantissant une optimisation stable. Afin d'améliorer la capacité d'adaptation du modèle, les données sont mélangées à chaque époque. Pour finir, le modèle est évalué en temps réel durant l'apprentissage à l'aide du jeu de test afin de vérifier ses performances.

4.7. Métriques d'évaluation :

4.7.1. Rappel sur la définition des performances :

- Définition de la précision globale : c'est une métrique cruciale en apprentissage automatique qui détermine la proportion de prédictions justes réalisées par un modèle, sans tenir compte de la catégorie.
- Définition de la précision moyenne : est une mesure utilisée pour évaluer la performance des modèles, notamment en détection d'objets et systèmes de recherche d'information. Elle calcule l'exactitude moyenne par classe en prenant la moyenne de l'exactitude de chaque classe individuelle

- Définition de coefficient Kappa : Est une statistique robuste utilisée pour déterminer le degré de concordance entre deux évaluateurs ou observateurs qui classent des éléments dans des catégories exclusives les unes aux autres.

4.8. Résultats obtenus par bases de données :

4.8.1. Résultats sur MNIST :

4.8.1.1.Tableau comparatif des performances :

Tableau 4.1 : Comparaison des performances des différents réseaux CNN sur la base de données MNIST

Réseaux	Précision globale	Précision moyenne	Kappa
Alexnet	98.76	98.76	98.62
Resnet18	99.21	99.22	99.12
Resnet50	98.83	98.84	98.70
Resnet101	98.51	98.51	98.34
Xception	99.20	99.20	99.11
Mobilenetv2	97.85	97.91	97.61

4.8.1.2. Observation et comparaison des performances des modèles CNN :

Le tableau présente les résultats de six modèles de réseaux de neurones convolutifs (CNN) telle que AlexNet, ResNet18, ResNet50, ResNet101, Xception et MobileNetV2 effectués sur le jeu de données MNIST en se basant sur trois indicateurs : précision globale, précision moyenne et coefficient de kappa.

On remarque que les taux de précision varient légèrement d'un modèle à l'autre, cependant tous obtiennent des scores très hauts (>97%).

- Précision globale :

Observation : La précision la plus élevée est obtenue avec le modèle ResNet18, qui atteint le score de 99,21 %, suivi par le modèle Xception avec une précision de 99,20 %, Puis viennent les modèles ResNet50 avec une précision de 98,83 % et AlexNet avec 98,76 %, ce dernier étant toujours compétitif compte tenu de son ancienneté, alors que le modèle MobileNetV2 présente le score le plus faible de 97,85 %.

Comparaison : Ce contraste peut s'expliquer par le fait que les modèles ResNet et Xception sont plus profonds et récents, offrant ainsi une capacité supérieure à extraire des caractéristiques discriminantes dans l'image, contrairement à AlexNet et MobileNetV2 qui sont plus simples.

➢ Précision moyenne :

Observation :

- ResNet18 et Xception affichent de hauts niveaux de précision (99.22 % et 99.20 %), avec une bonne stabilité entre les classes.
- ResNet101 et AlexNet sont plus constants (98.51 % et 98.76 %), tandis que MobileNetV2 doit se contenter d'une précision médiocre (97.91 %).

Comparaison : La précision moyenne démontre la robustesse de ResNet18 et Xception qui parviennent à bien classifier toutes les classes malgré une architecture de profondeur différente.

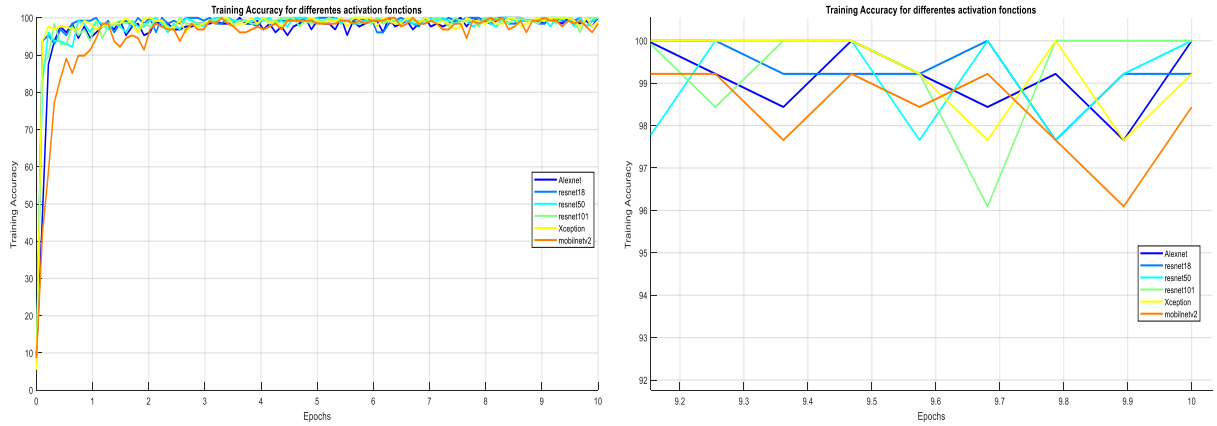
➢ Coefficient de Kappa :

Observation :

- Le meilleur coefficient a été obtenu par ResNet18 (99.12%), suivi de Xception (99.11%), ce qui montre qu'il y a une très bonne concordance entre les étiquettes prédites et les étiquettes de vérité.
- Les plus mauvais résultats sur cette métrique ont été obtenus par MobileNetV2 (97.61%) et ResNet101 (98.34%).

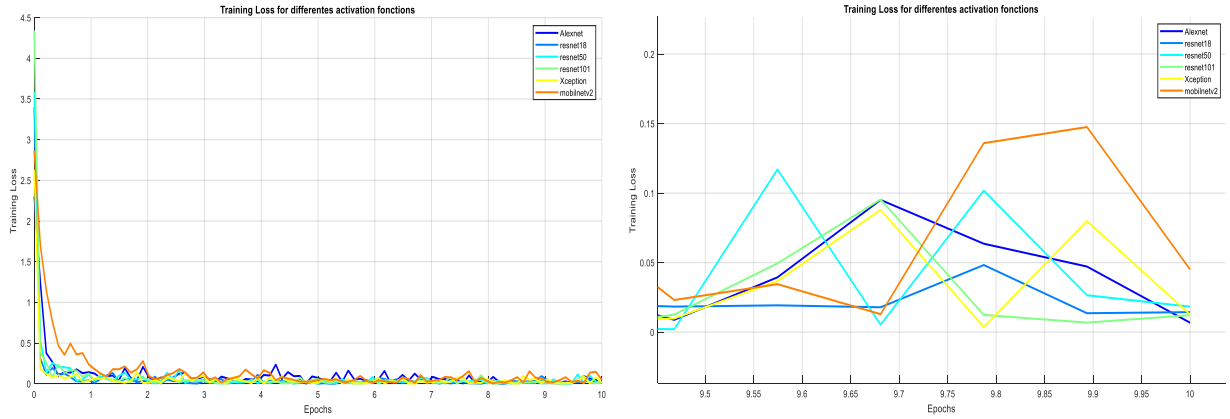
Comparaison : Le coefficient Kappa, qui permet d'évaluer la fiabilité des prédictions en les randomisant, montre que ResNet18 et Xception sont les plus fiables dans leurs classifications.

4.8.1.3. Analyse graphique des performances d'apprentissage et de validation des différents réseaux convolutifs :



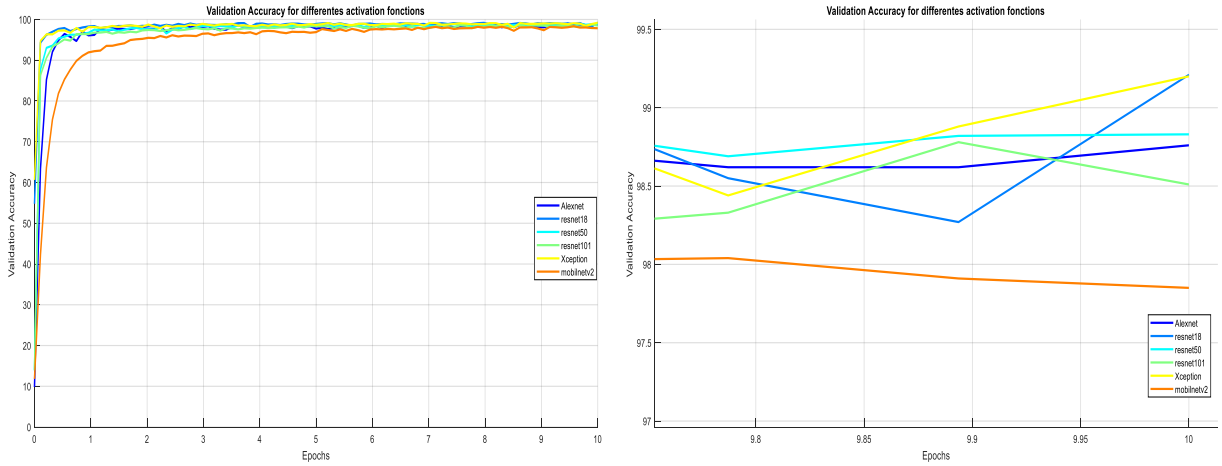
(a) (b)

Figure 4.5 : (a) Précision d'apprentissage pour diffèrent réseaux convolutif appliqué à la base de données MNIST, (b) Zoom de (a).



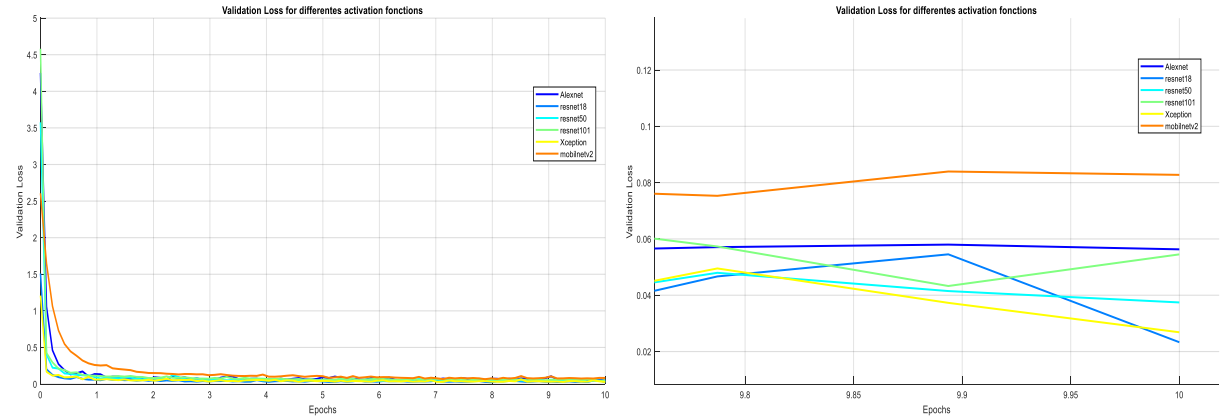
(c) (d)

Figure 4.6 : (c) Erreur (perte) d'apprentissage pour diffèrent réseaux convolutif appliqué à la base de données MNIST, (d) Zoom de (c).



(e) (f)

Figure 4.7 : (e) Précision de validation pour diffèrent réseaux convolutif appliqué à la base de données MNIST, (f) Zoom de (e).



(g) (h)

Figure 4.8 : (g) Erreur (perte) de validation pour diffèrent réseaux convolutif appliqué à la base de données MNIST, (h) Zoom de (g).

4.8.1.4. Conclusion :

On peut faire le constat que **ResNet18** est le meilleur compromis entre précision, robustesse et fiabilité même si sa profondeur est inférieure à celle de **ResNet50** ou **ResNet101**. On note que **Xception** est une bonne alternative qui permet de faire un bon compromis d'efficacité. Il est également observé que des architectures plus légères, comme **MobileNetV2**, entraînent une petite diminution de la performance dans notre situation.

On notera néanmoins qu'**AlexNet**, bien que plus ancien, reste performant sur MNIST mais est surpassé par les architectures modernes.

4.8.2. Resultat sur Fashion MNIST:

4.8.2.1.Tableau comparatif des performances:

Tableau 4.2 : Comparaison des performances des différents réseaux CNN sur la base de données Fashion MNIST.

Réseaux	Précision globale	Précision moyenne	Kappa
Alexnet	98.60	98.60	98.44
Resnet18	99.02	99.02	98.91
Resnet50	98.97	98.98	98.86
Resnet101	98.82	98.83	98.69
Xception	99.17	99.17	99.08
Mobilenetv2	97.66	97.70	97.40

4.8.2.2.Observation et comparaison des performances des modèles CNN :

On constate que les taux de précision varient légèrement d'un modèle à l'autre, néanmoins, tous les réseaux convolutifs atteignent des scores très élevés (supérieurs à 97 %), indiquant leur efficacité sur le jeu de données Fashion MNIST.

- Précision globale :

Observation :

- Le modèle Xception a atteint l'exactitude la plus élevée avec 99,17 % suivi très près par le modèle ResNet18 avec 99,02% de précision.
- Les modèles ResNet50 et ResNet101 obtiennent respectivement 98,97 % et 98,82 %.
- Malgré son ancienneté, AlexNet demeure compétitif avec un taux de 98,60 %.
- Enfin, le modèle MobileNetV2 fournit des résultats moins satisfaisants avec une précision de 97,66%.

Comparaison : Cette différence pourrait être due au fait que Xception et les architectures ResNet sont plus modernes et profondes, incorporant des techniques comme les connexions

résiduelles ou les convolutions séparables, favorisant une extraction plus efficace des caractéristiques. En revanche, bien qu'AlexNet et MobileNetV2 soient plus légers, ils révèlent leurs limites quand on les compare à des architectures actuelles.

- Précision moyenne :

Observation :

- Xception et ResNet18 présentent une précision moyenne exellente avec des scores de 99,17 % et 99,02 % respectivement, témoignant d'une performance homogène à travers toutes les classes.
- ResNet50 (98,98 %) et ResNet101 (98,83 %) affichent aussi de bonnes performances.
- AlexNet a obtenu un score de 98,60 %, alors que MobileNetV2 est un peu en dessous avec 97,70 %.

Comparaison : Une précision moyenne élevée indique que les modèles Xception et ResNet18 sont capables de bien classer l'ensemble des classes sans biais important, traduisant leur robustesse inter-classes. Par ailleurs, ResNet50 et ResNet101 présentent aussi une cohérence notable. Malgré son ancienneté, AlexNet maintient un bon équilibre. En ce qui concerne MobileNetV2, il paraît moins performant, probablement en raison de sa structure optimisée pour des environnements légers, souvent au prix d'une précision détaillée.

- Coefficient de Kappa :

Observation :

- Xception a le meilleur coefficient de Kappa. Il présente un taux de 99,08%, suivie de ResNet18 98,91%.
- ResNet50 est de 98,86%, suivi par AlexNet 98,44%, montrant que les deux modèles sont fiables.
- ResNet101 a un légèrement plus faible, de 98,69%, tandis que MobileNetV2 obtient le score le plus bas de 97,40%.

Comparaison : Les modèles Xception et ResNet18 présentent la meilleure fiabilité avec un taux d'erreurs aléatoires insuffisant. ResNet50 et AlexNet ont également une cohérence suffisante, tandis que ResNet101, bien que très profond, a l'air un peu moins stable**.** MobileNetV2 reste en retrait, ce qui est justifié par leur plus grande vulnérabilité dans des tâches de classification plus difficiles.

4.8.2.3. Analyse graphique des performances d'apprentissage et de validation des différents réseaux convolutifs :

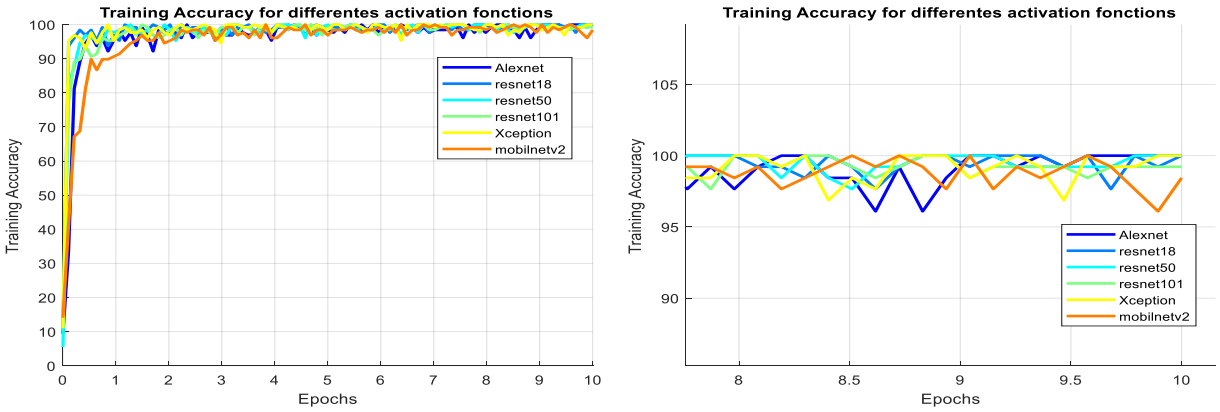


Figure 4.9 : (a) Précision d'apprentissage pour diffèrent réseaux convolutif appliqué à la base de données Fashion MNIST, (b) Zoom de (a).

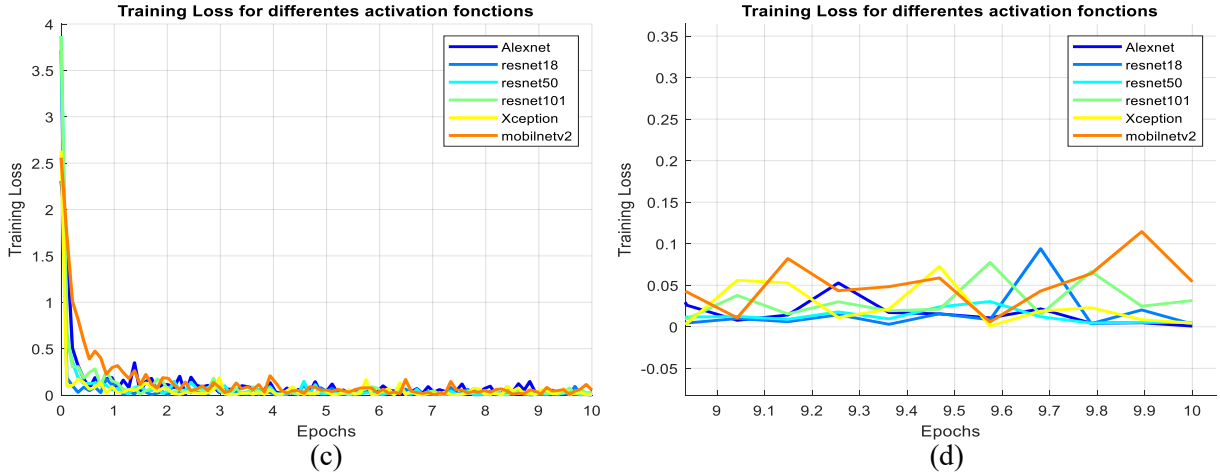


Figure 4.10 : (c) Erreur (perte) d'apprentissage pour diffèrent réseaux convolutif appliqué à la base de données Fashion MNIST, (d) Zoom de (c).

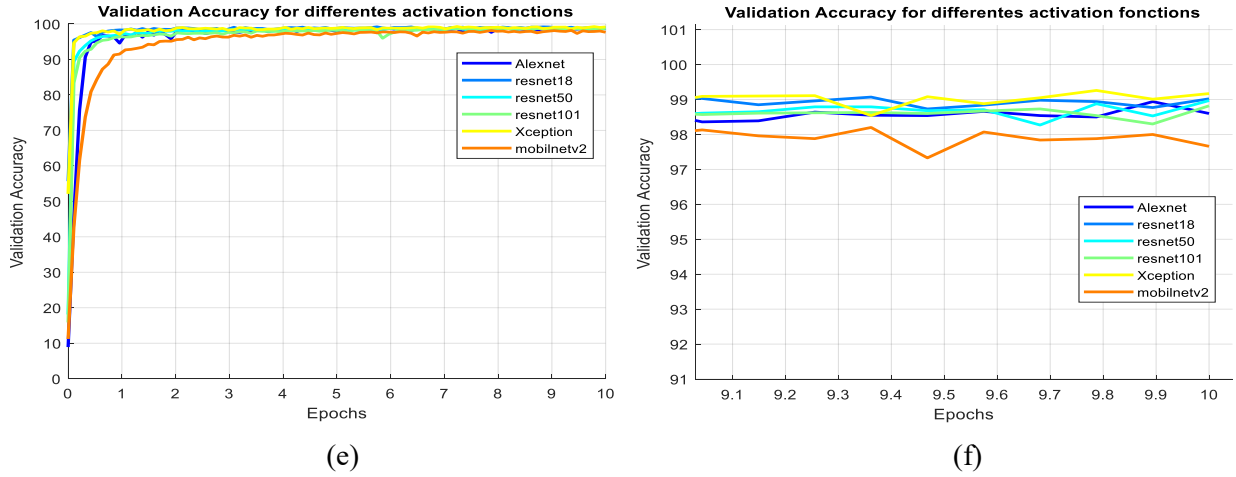
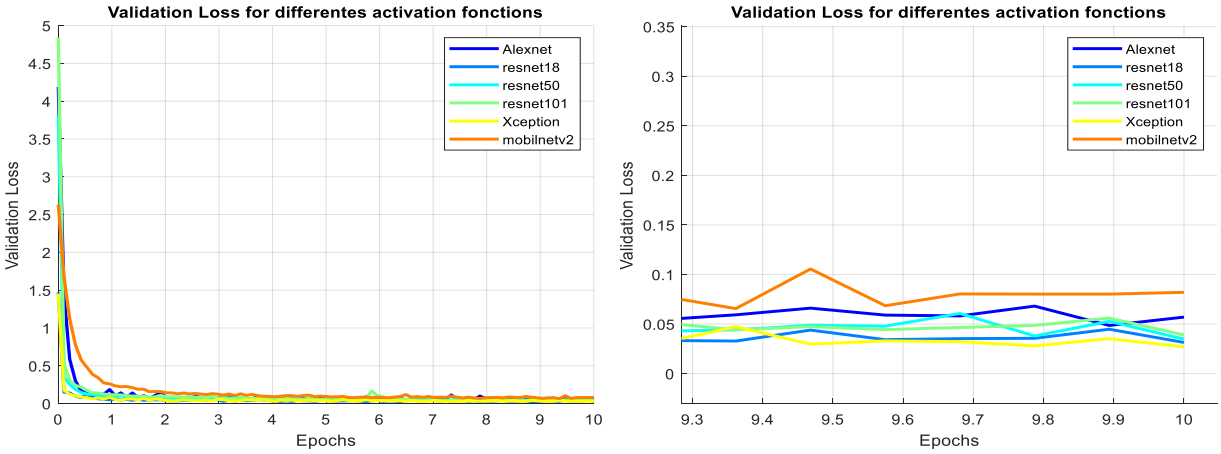


Figure 4.11 : (e) Précision de validation pour diffèrent réseaux convolutif appliqué à la base de données Fashion-MNIST, (f) Zoom de (e).



(g) (h)

Figure 4.12 : (g) Erreur (perte) de validation pour diffèrent réseaux convolutif appliqué à la base de données Fashion-MNIST, (h) Zoom de (g).

4.8.2.4. Conclusion :

Les résultats démontrent que les modèles **Xception** et **ResNet18** sont les plus performant sur Fashion MNIST tant sur leurs scores de précision que sur les scores de moyennes et la fiabilité des résultats. **ResNet50** et **ResNet101** ont également donné des résultats très solides, Bien que **AlexNet** soit relativement ancienne, elle s'en sort plutôt bien.

En revanche, **MobileNetV2** a atteint les performances les moins puissantes, bien que consommant le moins de ressources.

4.8.3. Resultat sur Street View House Numbers (SVHN):

4.8.3.1.Tableau comparatif des performances:

Tableau 4.3 : Comparaison des performances des différents réseaux CNN sur la base de données SVHN.

Réseaux	Précision globale	Précision moyenne	Kappa
Alexnet	85.78	84.17	83.90
Resnet18	91.34	90.60	90.18
Resnet50	88.41	87.54	86.86
Resnet101	61.90	63.21	56.26
Xception	92.62	92.02	91.64
Mobilenetv2	86.32	84.76	84.49

4.8.3.2.Observation et comparaison des performances des modèles CNN :

- Précision globale :

Observation :

- Le modèle Xception obtient la meilleure précision globale avec 92,62 %, suivi de près par ResNet18 avec 91,34 %.
- Les modèles ResNet50 (88,41%) et MobileNetV2 (86,32%) affichent de bonnes performances.
- AlexNet, bien qu'ancien, réussit à atteindre 85,78 %.
- Le modèle ResNet101 affiche un résultat particulièrement faible avec 61,90 %.

Comparaison : L'excellence de Xception et Resnet18 est due à leurs conceptions avancées qui incluent des méthodes efficaces pour identifier des fonctionnalités importantes telles que les convolutions séparables ou les connexions résiduelles. Au contraire, les performances faibles de Resnet101 indiquent un inadapté ou un ajustement inadéquat à l'ensemble de données SVHN. Alexnet reste cohérent pour une version plus ancienne, tandis que MobileNetV2, conçu pour sa légèreté, révèle ses limites par rapport à des systèmes plus complexe.

- Précision moyenne :

Observation :

- Xception affiche une précision moyenne remarquable de 92,02%, tandis que ResNet18 suit avec un score de 90,60%.
- ResNet50 obtient un score de 87,54 %, tandis que MobileNetV2 le suit de près avec 84,76 %.
- AlexNet atteint 84,17 %, un peu en dessous.
- Une fois de plus, le score de ResNet101 est assez bas avec 63,21 %.

Comparaison : Une précision moyenne élevée chez Xception et ResNet18 montre une performance équilibrée sur l'ensemble des classes, sans biais significatif. ResNet50 et MobileNetV2 restent corrects, mais moins homogènes. AlexNet, bien que stable, commence à montrer des limites face aux modèles modernes. ResNet101, malgré sa profondeur, échoue à assurer une généralisation correcte, ce qui peut indiquer une complexité excessive pour ce jeu de données.

- Coefficient de Kappa :

Observation :

- Le coefficient Kappa le plus élevé est obtenu par Xception (91,64 %), suivi par ResNet18 (90,18 %).

- ResNet50 affiche un bon score avec 86,86 %, suivi de MobileNetV2 (84,49 %) et AlexNet (83,90 %).
- Le score le plus faible revient à ResNet101 avec 56,26 %.

Comparaison : Le Coefficient de Kappa chez Xception et ResNet18 indique une grande fiabilité des classifications, comportant peu d'erreurs aléatoires. ResNet50, AlexNet et MobileNetV2 présentent une cohérence qui est satisfaisante, bien qu'un peu inférieure. Cependant, la note très faible de ResNet101 suggère un manque de validité, probablement attribuable à une complexité non justifiée par les propriétés du jeu de données SVHN.

4.8.3.3. Analyse graphique des performances d'apprentissage et de validation des différents réseaux convolutifs :

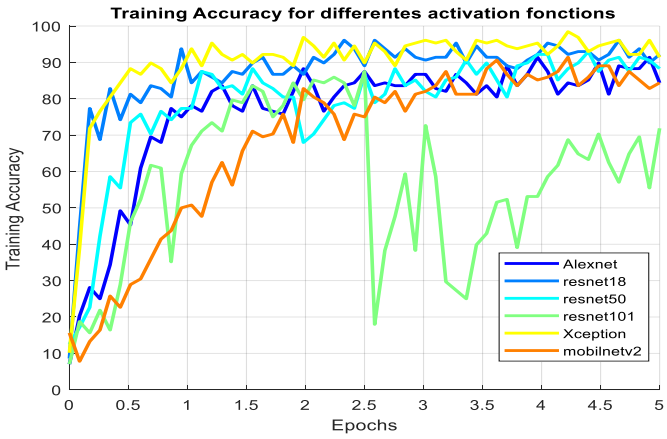


Figure 4.13 : Précision d'apprentissage pour différent réseaux convolutif appliqué à la base de données SVHN.

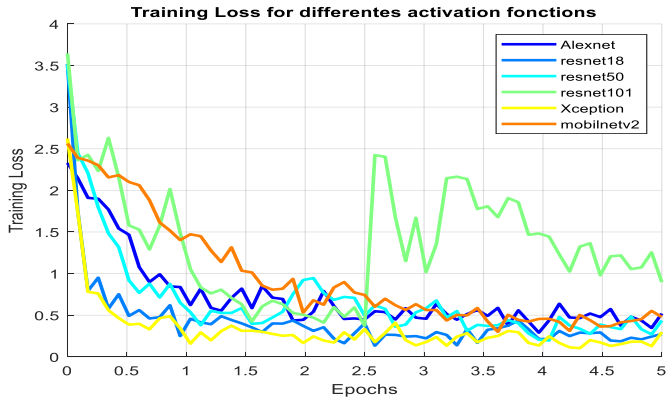


Figure 4.14 : Erreur (perte) d'apprentissage pour différent réseaux convolutif appliqué à la base de données SVHN

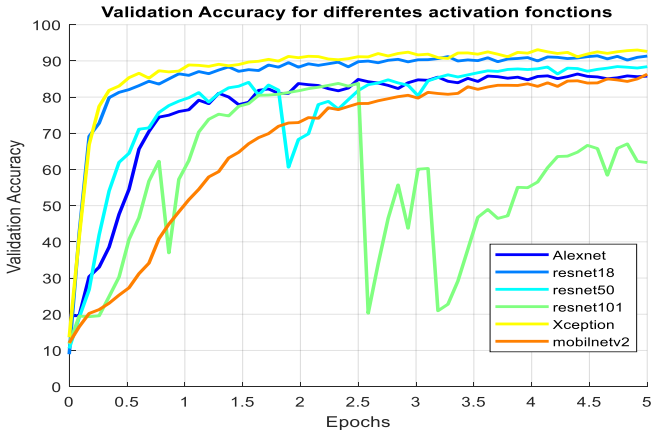


Figure 4.15 : Précision de validation pour diffèrent réseaux convolutif appliqué à la base de données SVHN.

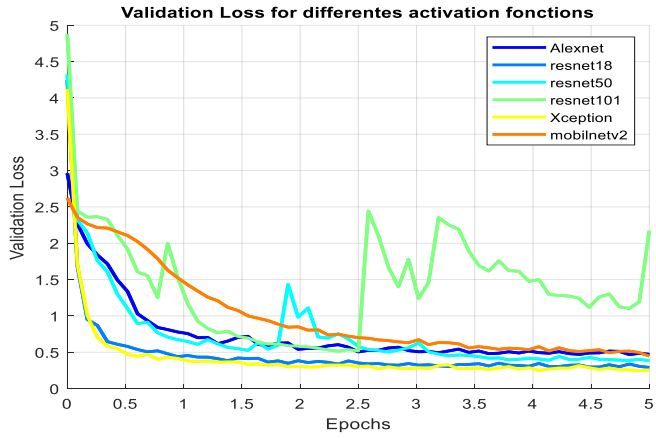


Figure 4.16 : Erreur (perte) de validation pour diffèrent réseaux convolutif appliqué à la base de données SVHN.

4.8.3.4. Conclusion :

Les résultats sur SVHN montrent que les modèles **Xception** et **ResNet18** offrent les meilleures performances globales, alliant précision et efficacité. **ResNet50**, **AlexNet** et **MobileNetV2** restent acceptables, mais moins performants. À l'inverse, **ResNet101** se montre inadapté, probablement en raison de sa complexité excessive. Ainsi, les architectures modernes mais légères comme **Xception** ou **ResNet18** s'avèrent les plus efficaces pour ce type de données.

4.9. Comparaison des performances entre différents modèles CNN sur les 3 bases de données :

4.9.1. Comparaison de la difficulté des bases de données :

- MNIST et Fashion-MNIST sont les plus facile à classifier pour tous les modèles.
- SVHN est le plus complexe en raison de ses images en couleur, bruyantes et présentant des variations de luminosité.

4.9.2. Conclusion générale :

En considérant la précision globale, la précision moyenne et le coefficient Kappa, le modèle Xception émerge effectivement comme le plus performant parmi les cinq autres modèles qu'ils sont : ResNet18, ResNet50**,** AlexNet, ResNet101 et MobileNetV2.

1. Le modèle **Xception** se révèle être le meilleur parmi tous les réseaux évalués. Il présente une précision élevée sur divers ensembles de données, qu'ils soient simples ou complexes. Sa cohérence sur l'ensemble des métriques en fait un modèle particulièrement robuste et fiable, approprié pour une multitude de tâches de classification.
2. Le **ResNet18** est un modèle qui trouve un équilibre parfait entre performance et simplicité. Il propose des performances solides sur l'ensemble des jeux de données, en présentant une cohérence notable dans la classification. Sa structure moins complexe lui permet d'être plus léger et rapide à entrainer tout en préservant une haute précision.
3. Le modèle **ResNet50** présente de solides performances, en particulier sur des ensembles de données simples comme MNIST et Fashion MNIST. Cependant, il montre une performance moins constante sur des bases plus complexes telles que SVHN, particulièrement en ce qui concerne la précision moyenne.
4. Le modèle **ResNet101** donne d'excellents scores sur des jeux de données simples, mais présente une grande instabilité lorsqu'il s'agit de données complexes telles que SVHN. Cette dégradation de performance indique un risque d'overfitting (surapprentissage) ou une mauvaise généralisation du modèle.
5. Malgré son ancienneté, le modèle **AlexNet** conserve sa fiabilité pour des ensembles de données simples tels que MNIST ou Fashion MNIST. L'architecture simplifiée de ce modèle rend l'entraînement aisé, néanmoins, elle montre ses faiblesses sur des ensembles de données plus sophistiqués tels que SVHN.
6. Le **MobileNetV2** se caractérise par sa légèreté et son faible besoin en ressources, ce qui le rend approprié pour les systèmes intégrés et les applications sur mobiles. Cependant, ses performances demeurent moins bonnes comparées à d'autres modèles sur l'ensemble des jeux de données.

4.9.3. Tableau récapitulatif:

	Bases de données				
Réseaux	**MNIST**	**Fashion-MNIST**	**SVHN**	**Temps de calcul**	**Analyse générale des performances**
Xception	Performance suréminent	Performance suréminent	Performance suréminent	Très lent car il contient beaucoup de couches séquentielles à exécuter	Performance exceptionnelle, solide généralisation inter-domaine
Resnet18	Trés bonne performance	Trés bonne performance	Trés bonne performance	Rapide à exécuter, à entrainer et à tester grâce à sa simplicité	Trés bonne performance, offrant un bon équilibre entre rapidité et exactitude.
Resnet50	Trés bonne performance	Trés bonne performance	Performance moyenne	Durée moyenne de calcul (plus massif que ResNet18, mais moins que ResNet101)	Stabilité réduite sur données complexe malgré de bonne performance sur bases simple
Resnet101	Trés bonne performance	Performance moyenne	performance trés faible	Lent à entraîner et à exécuter à cause de sa profondeur (101 couches)	Détérioration a été observée sur SVHN, potentiellement due à un surentrainement
AlexNet	Performance médiane	Performance médiane	Performance faible	Rapide grâce à son mécanisme simplifié	Moins performant en comparaison avec les architectures modernes, notamment pour SVHN
MobilenetV22	Performances faible	Performances faible	Performances moyenne	Plus rapide à cause de la convolution séparable en profondeur	Modèle léger et rapide. Il présente un bon équilibre entre vitesse et taille, mais sa précision est inférieure à celle des modèles plus complexe

Tableau 4.4 : Analyse des performances et du temps de calcul de différents modèles CNN sur trois bases de données.

4.10. Conclusion :

En tenant compte de tous les paramètres, le modèle Xception s'impose comme le meilleur en termes de précision globale, de précision moyenne et du coefficient Kappa. Cette solidité sur divers ensembles de données en fait le choix préféré pour notre tâche de classification.

Conclusion générale

Conclusion générale

Ce mémoire a présenté une étude comparative de plusieurs modèles de réseaux de neurones convolutifs appliqués à la classification d'images, en utilisant les bases de données MNIST, Fashion-MNIST et SVHN. L'analyse des performances a mis en évidence que le modèle Xception offre les meilleurs résultats globaux, en termes de précision, de capacité de généralisation, tout en maintenant un bon équilibre entre profondeur et rapidité.

Néanmoins, cette étude se limite à des bases de données standard et à des conditions d'entraînement spécifiques. Des analyses plus poussées, incluant des jeux de données plus complexes, des réglages d'hyperparamètres optimisés, des évaluations sur des plateformes matérielles variées, le calcul de la moyenne des précisions sur plusieurs essais, et l'augmentation du nombre d'époque permettraient d'approfondir les résultats obtenus.

Bibliographie

Bibliographie

[1] Salesforce, « Qu'est-ce qu'un réseau de neurones ? », *Salesforce*. [En ligne]. Disponible : https://www.salesforce.com/fr/resources/definition/reseau-de-neurones/#topic1

[2] GRETISI, « Historique des réseaux de neurones », *GRETISI*. [En ligne]. Disponible : https://www.gretsi.fr/historique/reseaux-neurones

[3] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2e éd., Sebastopol, CA, USA : O'Reilly Media, 2019.

[4] Data Bird, « Réseaux de neurones », *Data Bird Blog*. [En ligne]. Disponible : https://www.data-bird.co/blog/reseaux-de-neurones

[5] I. Goodfellow, Y. Bengio et A. Courville, *Deep Learning*, MIT Press, 2016, Section 6.3 – Feedforward Computation

[6] S. Ruder, « An overview of gradient descent optimization algorithms », *arXiv preprint*, arXiv:1609.04747, 2016.

[7] MonCoachData, « Comprendre les réseaux de neurones », *MonCoachData Blog*, [En ligne]. Disponible : https://moncoachdata.com/blog/comprendre-les-reseaux-de-neurones .

[8] D. P. Kingma et J. Ba, « Adam: A Method for Stochastic Optimization », *arXiv preprint*, arXiv:1412.6980, 2014. [En ligne]. Disponible : https://arxiv.org/abs/1412.6980

[9] S. N. Gupta, « Understanding the RMSprop Optimizer: A Guide », *Built In*, 26 février 2025. [En ligne]. Disponible : https://builtin.com/articles/rmsprop-optimizer

[10] Y. LeCun, Y. Bengio et G. Hinton, « Deep learning », *Nature*, vol. 521, pp. 436–444, 2015

[11] SERIDI, Merwan."Application des réseaux de neurones pour la classification des données." Dirigé par Mr Debeche Mehdi,Université 08 mai 1945 Guelma, Octobre 2020.

[12] M.-R. Bouguelia, Classification et apprentissage actif à partir d'un flux de données évolutif en présence d'étiquetage incertain, Thèse de doctorat, Université de Lorraine, France, 2015.

[13] T. Keldenich, "Qu'est-ce que la Classification Binaire ? – Meilleur Guide" et "Qu'est-ce que la Classification Multi-Classes ? – Meilleur Guide," Inside Machine Learning, juillet 2023

[14] Keylabs, "Types of Classification Models: Binary, Multiclass, and Multilabel," Keylabs Blog, 2 septembre 2024. [En ligne]. Disponible sur : https://keylabs.ai/blog/types-of-classification-models-binary-multiclass-and-multilabel/

[15] P.-N. Tan, M. Steinbach, et V. Kumar, Introduction to Data Mining, 2e éd., Pearson, 2018

[16] IBM, « Qu'est-ce que l'apprentissage supervisé ? », *IBM*, [En ligne]. Disponible : https://www.ibm.com/fr-fr/topics/supervised-learning. [Consulté le : 16 mai 2025].

[17] P. Besse, “Machines à vecteurs supports,” Wikistat, Université Toulouse III – Paul Sabatier, 2012. [En ligne]. Disponible sur : https://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-m-app-svm.pdf

[18] Spot Intelligence, “Support Vector Machines (SVM) Made Simple & How To Tutorial,” Spot Intelligence, 6 mai 2024

[19] R. Kassel, “Algorithme de classification : Définition et principaux modèles,” DataScientest, 22 novembre 2022

[20] Zilliz, “What Is K-Nearest Neighbors (KNN) Algorithm in ML?,” Zilliz Blog, 2 mars 2025

[21] DATAtab, “Tutoriel : Corrélation,” DATAtab, [En ligne]. Disponible sur : https://datatab.fr/tutorial/correlation. [Consulté le : 17 mai 2025].

[22] JMP Statistical Discovery, “Coefficient de corrélation : Qu'est-ce que le coefficient de corrélation ?” JMP – Statistical Knowledge Portal, [En ligne]. Disponible sur : https://www.jmp.com/fr/statistics-knowledge-portal/what-is-correlation/correlation-coefficient.

[23] K. Tannir, “L’algorithme A-priori,” Blog de Khaled Tannir, 3 mai 2011

[24] GeeksforGeeks, “How to Check the Accuracy of Your Machine Learning Model?,” GeeksforGeeks, 9 janv. 2025.

[25] Nanobaly, “Practical Guide to the Confusion Matrix for AI,” Innovatiana, 29 mars 2024.

[26] GeeksforGeeks, « How to Check the Accuracy of Your Machine Learning Model? », *GeeksforGeeks*, 9 janv. 2025. [En ligne]. Disponible : https://www.geeksforgeeks.org/how-to-check-the-accuracy-of-your-machine-learning-model/. [Consulté le : 17 mai 2025].

[27] DeepAI, « Convolutional Neural Network », *DeepAI Glossary*. [En ligne]. Disponible : https://deepai.org/machine-learning-glossary-and-terms/convolutional-neural-network

[28] Merzougui, D., Réseau de neurones convolutionnel CNN, Chapitre 3, Université de Batna 2, Algérie, 2023

[29] M. Adbib, “L’architecture CNN – 1 : Bases de l’architecture CNN,” Medium, 24 juin 2022

[30] Y. LeCun, L. Bottou, Y. Bengio et P. Haffner, « Gradient-Based Learning Applied to Document Recognition », *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, nov. 1998. [En ligne]. Disponible : http://vision.stanford.edu/cs598_spring07/papers/Lecun98.pdf

[31] Bangar, "LeNet-5 Architecture Explained," Medium, Jun. 22, 2022. [Online]. Available: https://medium.com/@siddheshb008/lenet-5-architecture-explained-3b559cb2d52b

[32] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” *in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.

[33] S. Zagoruyko et N. Komodakis, « Wide Residual Networks », arXiv preprint arXiv:1605.07146, 2016

[34] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” in CVPR, 2017.

[35] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” in CVPR, 2017

[36] S. Jégou, M. Drozdzal, D. Vázquez, A. Romero, and Y. Bengio, “The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation,” *arXiv preprint arXiv:1611.09326*, 2016. [Online]. Available: https://arxiv.org/abs/1611.09326

[37] J.-J. Shang, N. Phipps, I.-C. Wey et T. H. Teo, « A-DSCNN: Depthwise Separable Convolutional Neural Network Inference Chip Design using an Approximate Multiplier », *Preprint*, 2023. [En ligne]. Disponible sur : https://www.preprints.org/manuscript/202306.1435/v1

[38] A. G. Howard et al., “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” arXiv, 2017.

[39] M. Tan and Q. V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” in ICML, 2019.