



RAPPORT DE STAGE DE FIN D'ÉTUDES

Pour l'obtention du Diplôme de Licence Professionnelle en
Développement et Administration de l'Internet et de l'Intranet

Conception et Entraînement d'un Modèle de Détection des Deepfakes Basé sur les Réseaux de Neurones Convolutifs (CNNs)

Réalisé par :
Mme. RAMLA Babaha

Encadré par :
Dr. ELVETH Sidi (FST)
Dr. RIFAA Sadegh
(MTNIMA)

Dédicace

“

*À mes parents, véritables sources de motivation dans
chaque étape de ma vie,
qui m'ont guidé à chaque étape de ce parcours.*

*À mon frère et mes sœurs, pour leurs encouragements et
nos moments partagés,
qui m'ont ancré et inspiré.*

*Ce travail est le reflet de votre soutien,
et je vous en suis profondément reconnaissant.*

”

Ramla

Remerciements

Tout d'abord, je remercie Allah le tout puissant de m'avoir donné le courage et la patience nécessaires à mener ce travail à son terme.

Je tiens à remercier tout particulièrement mon encadrant académique **Dr. Elveth Sidi**, pour l'aide compétente qu'il m'a apporté, pour sa patience et son encouragement. Son œil critique m'a été très précieux pour structurer le travail et pour améliorer la qualité des différentes sections.

Je tiens à remercier également mon encadrante professionnelle **Dr. Rifaa Sadegh**.

Je tiens à adresser mes sincères remerciements et toute ma reconnaissance à **Dr. Elbennany Med Mahmoud**, notre coordinateur à la faculté, pour son accompagnement constant, ses conseils avisés et son engagement tout au long de cette année.

Je tiens à exprimer ma profonde gratitude à l'ensemble des enseignants qui m'ont encadrée au cours de cette année universitaire, pour la qualité de leur enseignement, leur disponibilité et leur accompagnement tout au long de mon parcours académique. Je remercie également toutes les personnes qui ont contribué de près ou de loin à mon apprentissage cette année.

Que les membres de jury trouvent, ici, l'expression de mes sincères remerciements pour l'honneur qu'ils me font en prenant le temps de lire et d'évaluer ce travail.

Pour finir, je souhaite remercier toute personne ayant contribué de près ou de loin à la réalisation de ce travail.

Résumé

Le présent rapport a été élaboré suite à un projet de fin d'études effectué au sein du (MTNIMA). Il s'inscrit dans le cadre de notre formation pour l'obtention du Diplôme de Licence Professionnelle en Développement et Administration de l'Internet et de l'Intranet.

La technologie Deepfake a rapidement évolué, produisant des images truquées très réalistes et difficiles à détecter. Ce projet propose une méthode basée sur le deep learning, combinant des réseaux de neurones convolutionnels (CNN) , pour la détection des deep-fakes. Le système se compose de trois étapes principales : prétraitement (extraction des images, détection et alignement des visages, recadrage des caractéristiques), détection et prédiction. Une méthode de vote majoritaire fusionne les prédictions issues des modèles appliqués à différentes caractéristiques faciales. Le modèle est entraîné sur les jeux de données FaceForensics et DFDC, et évalué selon les métriques précision, rappel, F1-score . Les résultats expérimentaux montrent une efficacité notable avec une précision de 93%.

Ce projet contribue à l'avancement des techniques de deep learning basées sur CNN pour une détection fiable des deepfakes sur les réseaux sociaux.

Mots clés : Convolutional neural network, Deepfake detection, Face recognition

Liste des acronymes

MTNIMA	Ministère de la Transformation Numérique, de l'Innovation et de la Modernisation de l'Administration
MTCNN	Multi-task Cascaded Convolutional Neural Networks
ReLU	Rectified Linear Unit (activation function)
OpenCV	Open Source Computer Vision Library
DL	Deep Learning
ML	Machine Learning
FF++	FaceForensics++
Celeb-DF	Celeb-DeepFake
IA	Artificielle Intelligence
CNN	Convolutional Neural Network
GAN	Generative Adversial Network
PR	Precision-Recall
FNR	False Negative Rate
TNR	True Negative Rate
ANR	Artificial Neural Network

Table des matières

Dédicace	I
Remerciements	II
Résumé	III
1 Introduction générale	1
1.1 Contexte	1
1.2 Problématique	2
1.3 Objectives du projet	2
1.4 Organisation du rapport	3
2 Présentation de la structure d'accueil	4
2.1 Présentation du Ministère	4
2.2 Organisation interne	4
2.3 Secteurs d'activité et missions principales	5
2.4 L'environnement de travail	5
3 Contexte théorique et état de l'art	8
3.1 l'intelligence artificielle	8
3.1.1 Historique de l'intelligence artificielle	8
3.2 Les grandes familles de l'IA	10
3.2.1 IA Symbolique	10
3.2.2 Machine Learning	11
3.2.3 Deep Learning	15
3.3 Les Deepfakes et leurs enjeux	22
3.3.1 Définition des Deepfakes	22
3.3.2 Techniques de génération	22
3.3.3 Menaces associées	23
3.4 Techniques de détection des Deepfakes	23
3.4.1 Présentation des méthodes existantes	23
3.4.2 Focus sur les CNNs	24
3.4.3 Limites et défis de la détection des deepfakes	29
4 Analyse des besoins	30
4.1 Analyse du besoin	30
4.1.1 Exigences fonctionnelles et non fonctionnelles	30
4.2 Cahier des charges	31

4.2.1	Contexte	31
4.2.2	Objectifs	31
4.2.3	Spécifications fonctionnelles	32
4.2.4	Spécifications techniques	32
4.2.5	Planning résumé sur 3 mois	32
4.2.6	Livrables	32
5	Réalisation	33
5.1	Choix technologiques et architecture	33
5.1.1	Environnement et outils & Justification des choix	33
5.1.2	Architecture technique proposée	39
5.2	Mise en œuvre du projet	41
5.2.1	4.2.1 Préparation et traitement des données	41
5.2.2	Conception et entraînement du modèle	42
5.2.3	implémentation technique	43
5.2.4	Développement de l'interface web de test	44
5.2.5	Intégration du modèle avec l'interface	45
5.2.6	Démonstration	48
5.3	Résultats et interprétation	50
5.4	Méthodologie d'évaluation	50
5.4.1	Résultats expérimentaux	51
5.4.2	Analyse critique des performances	52
	Conclusion générale & Perspectives	53

Table des figures

2.1	L'organigramme du Ministere.	7
3.1	IA vs ML vs DL	16
3.2	ML vs DL	16
3.3	Neurone biologique Vs Neurone artificiel	18
3.4	Perceptron Multicouche	20
3.5	An example of CNN architecture for image classification	24
3.6	Les calculs principaux exécutés à chaque étape de la couche de convolution	26
3.7	Operation-three-on-the-pooling-method	28
3.8	Couche fully-connected	28
5.1	Diagramme d'architecture	40
5.2	Affichage des résultats d'analyse avec indication visuelle et score de confiance	46
5.3	Affichage des résultats d'analyse avec indication visuelle et score de confiance	46
5.4	Exemple d'image analysée en taille réelle	49
5.5	Exemple d'analyse d'une image deepfake via l'interface	49
5.6	Scan ME :)	54

Chapitre 1

Introduction générale

1.1 Contexte

La falsification et la manipulation des contenus multimédias, en particulier des images et des vidéos faciales, ont connu une progression fulgurante ces dernières années, notamment grâce aux avancées majeures dans le domaine de l'intelligence artificielle et de l'apprentissage profond. Les techniques dites *DeepFake*, reposent sur des architectures sophistiquées combinant des réseaux de neurones convolutifs (CNN) pour l'analyse et l'extraction de caractéristiques visuelles, ainsi que des modèles génératifs adversariaux (GAN) qui permettent de créer des contenus visuels synthétiques d'une qualité et d'un réalisme jusqu'alors inégalés. Ces technologies, apparues publiquement vers la fin de l'année 2017, ont rapidement suscité l'intérêt non seulement des chercheurs et développeurs, mais également des acteurs malveillants cherchant à exploiter leur potentiel destructeur.

L'émergence des deepfakes soulève des enjeux cruciaux à plusieurs niveaux. D'un point de vue sociétal, la prolifération de contenus falsifiés menace directement la crédibilité des médias et la confiance du public dans les informations diffusées. Sur le plan juridique et éthique, elle pose des questions complexes liées à la protection de la vie privée, à la réputation des individus, ainsi qu'à la responsabilité des plateformes hébergeant ces contenus. Les deepfakes sont ainsi utilisés pour la diffusion de fausses informations susceptibles d'influencer l'opinion publique et les processus démocratiques, la création de vidéos difamatoires ou à caractère pornographique sans consentement, ainsi que pour commettre des fraudes financières sophistiquées.

La facilité d'accès à ces technologies et la rapidité avec laquelle ces contenus peuvent être partagés sur les réseaux sociaux aggravent les risques d'une propagation massive et incontrôlée. Face à cette menace croissante, il est devenu impératif de développer des méthodes de détection robustes, capables d'identifier rapidement et avec fiabilité les manipulations faciales, afin de protéger les individus, les institutions et la société dans son ensemble.

Dans ce contexte, la criminalistique numérique s'est considérablement renouvelée, orientant ses efforts vers l'élaboration de solutions innovantes fondées sur le Machine Learning et le Deep Learning, notamment via l'utilisation des réseaux de neurones convolutifs. Ces travaux sont essentiels pour lutter contre la désinformation.

1.2 Problématique

Dans le cadre de ma formation pour l'obtention de la Licence Professionnelle en Développement et Administration d'Internet et Intranet (DA2I), j'ai eu l'opportunité d'effectuer un stage de quatre mois au sein du Ministère de la Transformation Numérique, de l'Innovation et de la Modernisation de l'Administration (MTNIMA). Intégré à une équipe spécialisée dans la transformation numérique, j'ai contribué au développement d'un modèle de Machine Learning reposant sur des réseaux de neurones convolutifs (CNN) dont l'objectif est la détection des falsifications dites *DeepFake*.

Ce projet s'inscrit dans un contexte où la multiplication et la sophistication croissante des techniques de manipulation faciale représentent une menace sérieuse pour la sécurité de l'information et la confiance accordée aux contenus numériques. En effet, la facilité avec laquelle ces contenus truqués peuvent être créés et diffusés, notamment via les réseaux sociaux, engendre des risques majeurs tels que la désinformation, l'atteinte à la réputation des individus, et la manipulation de l'opinion publique.

Ainsi, la détection automatique et fiable des deepfakes constitue un enjeu fondamental, tant pour les institutions publiques que pour les acteurs privés, dans le but de préserver l'intégrité des échanges numériques. Par ailleurs, ce projet m'a permis d'acquérir et de renforcer des compétences techniques en intelligence artificielle et en traitement d'images, tout en développant des aptitudes organisationnelles et collaboratives essentielles à la conduite de projets innovants dans un environnement professionnel.

La problématique centrale de ce travail consiste donc à concevoir et implémenter une solution technique performante, capable d'identifier avec précision les contenus manipulés, en s'appuyant sur des approches avancées de deep learning, tout en tenant compte des contraintes réelles liées à la diversité et à la volumétrie des données multimédias traitées.

1.3 Objectives du projet

Le projet réalisé dans le cadre de ce stage a permis d'apporter plusieurs contributions significatives tant sur le plan technique que méthodologique. Parmi les principales contributions, on peut citer :

- **Développement d'un modèle performant de détection de deepfakes** : Conception et entraînement d'un réseau de neurones convolutifs (CNN) adapté à la classification des contenus multimédias falsifiés, permettant d'identifier avec une précision satisfaisante les manipulations faciales.
- **Intégration d'un pipeline complet de traitement d'images** : Mise en place d'une chaîne automatisée incluant la prétraitement des données, la détection des visages, l'extraction des caractéristiques pertinentes, ainsi que l'évaluation des résultats.

- **Acquisition et consolidation de compétences** : Le travail effectué a permis de développer des compétences avancées en intelligence artificielle, traitement d'images, gestion de projet, ainsi qu'en travail collaboratif dans un contexte professionnel.
- **Documentation et transfert de connaissances** : Rédaction d'un rapport détaillé et création de supports de présentation pour assurer la pérennité du projet et faciliter son évolution future.

1.4 Organisation du rapport

Le présent rapport est structuré en cinq chapitres principaux :

- **Chapitre 1 : Introduction générale**
Ce premier chapitre a pour objectif d'introduire le sujet du projet , définit clairement la problématique abordée, précise les objectifs visés, puis souligne les contributions majeures apportées au cours du travail réalisé.
- **Chapitre 2 : Présentation de l'entreprise**
Ce chapitre présente l'entreprise dans laquelle le projet a été réalisé. Il décrit son secteur d'activité, ses missions principales , en lien avec le thème de la détection des deepfakes.
- **Chapitre 3 : Contexte théorique et état de l'art**
Ce chapitre propose une revue détaillée des concepts fondamentaux nécessaires à la compréhension du projet. Il présente les notions clés liées à l'intelligence artificielle, aux réseaux de neurones convolutifs (CNN), ainsi qu'aux méthodes de détection des deepfakes.
- **Chapitre 4 :Analyse des besoins**

Ce chapitre est dédié à l'identification et à la formalisation des besoins du projet. Il comprend l'élaboration du cahier des charges, qui définit les objectifs fonctionnels et techniques à atteindre, ainsi que les contraintes à respecter.

- **Chapitre 5 :Réalisation**

Cette partie présente la mise en œuvre concrète du projet, en détaillant les différentes étapes de développement du modèle . Elle inclut la description des choix technologiques opérés, l'architecture logicielle adoptée, ainsi que les méthodologies utilisées pour l'entraînement, la validation et l'évaluation du modèle.

Le rapport se termine par une **conclusion générale et des perspectives**, qui synthétisent les résultats obtenus, mettent en lumière les limites rencontrées, et ouvrent la voie à des travaux futurs.

Chapitre 2

Présentation de la structure d'accueil

2.1 Présentation du Ministère

MTNMIA (Ministère de la Transformation Numérique, de l'Innovation et de la Modernisation de l'Administration) est Un département ministériel dédié à la Transition Numérique, l'Innovation et la Modernisation de l'Administration a été a créé le 26 mai 2021 dont le principal objectif est d'assurer une transition numérique rapide et sûre, de promouvoir l'innovation et de développer le e-Gouvernement. Le Ministre de la Transition Numérique, de l'Innovation et de la Modernisation de l'Administration, assure le suivi des activités de régulation dans les domaines relevant de ses compétences. Le Ministre de la Transition Numérique de l'Innovation et de la Modernisation de l'Administration représente l'Etat auprès des institutions régionales et internationales spécialisées dans les domaines de compétence du Département, gère les rapports de l'Etat avec celles-ci et préside le Conseil Consultatif Stratégique et le Conseil d'Administration.

2.2 Organisation interne

L'administration centrale du Ministère de la Transition Numérique, de l'Innovation et de Modernisation de l'Administration comprend :

- Le Cabinet du Ministre
- Le Secrétariat Général
- Les Directions centrales

2.3 Secteurs d'activité et missions principales

Missions clés du Ministère de la Transition Numérique, de l'Innovation et de la Modernisation de l'Administration :

- Transition Numérique : Politiques TIC, infrastructures numériques, cybersécurité, interopérabilité et accès universel aux services
- Innovation : Soutien aux Start-Ups, financement de projets innovants et partenariats technologiques internationaux.
- Modernisation : Transformation numérique de l'administration, simplification des services publics et réformes institutionnelles.

2.4 L'environnement de travail

Direction du Développement et de l'Interopérabilité (DDI)

Missions principales

- Maîtrise d'ouvrage des projets informatiques transversaux et sectoriels de l'Administration
- Promotion d'un dispositif cohérent de traitement/diffusion de l'information selon normes internationales (sécurité, performance, disponibilité)
- Appui aux structures pour :
 - Identification des besoins d'informatisation
 - Conception de projets et veille sur les offres du marché
- Développement et gestion des portails, sites web, intranet, bases de données et systèmes d'information

Structure organisationnelle

Trois services clés sous l'autorité d'un Directeur et Directeur Adjoint :

- Service des Études & Développement
- Service des Bases de Données
- Service de l'Interopérabilité

1. Service des Études & Développement

Missions :

- Élaboration des cahiers de charges des applications
- Conception de l'architecture des SI (topologie, sécurité, fonctionnalités)
- Plan d'intégration avec les anciens systèmes
- Coordination des traitements informatiques (qualité/coûts/délais)
- Développement des services Internet/Intranet gouvernementaux
- Sécurisation des transactions et mise à jour des sites

Composition :

- Division des Études
- Division du Développement

2. Service des Bases de Données

Missions :

- Optimisation de la production informatique et règles de sauvegarde
- Gestion des procédures d'exploitation et assistance aux utilisateurs
- Validation des produits finis + suivi des volumes de données
- Sécurité et confidentialité des informations

Composition :

- Division de l'Administration des Bases de Données
- Division de l'Exploitation

3. Service de l'Interopérabilité

Missions :

- Élaboration de normes communes pour les SI publics
- Support technique interministériel sur l'interopérabilité
- Déploiement d'une plateforme d'échange de données (hub)

Composition :

- Division des Normes
- Division du Hub Interministériel

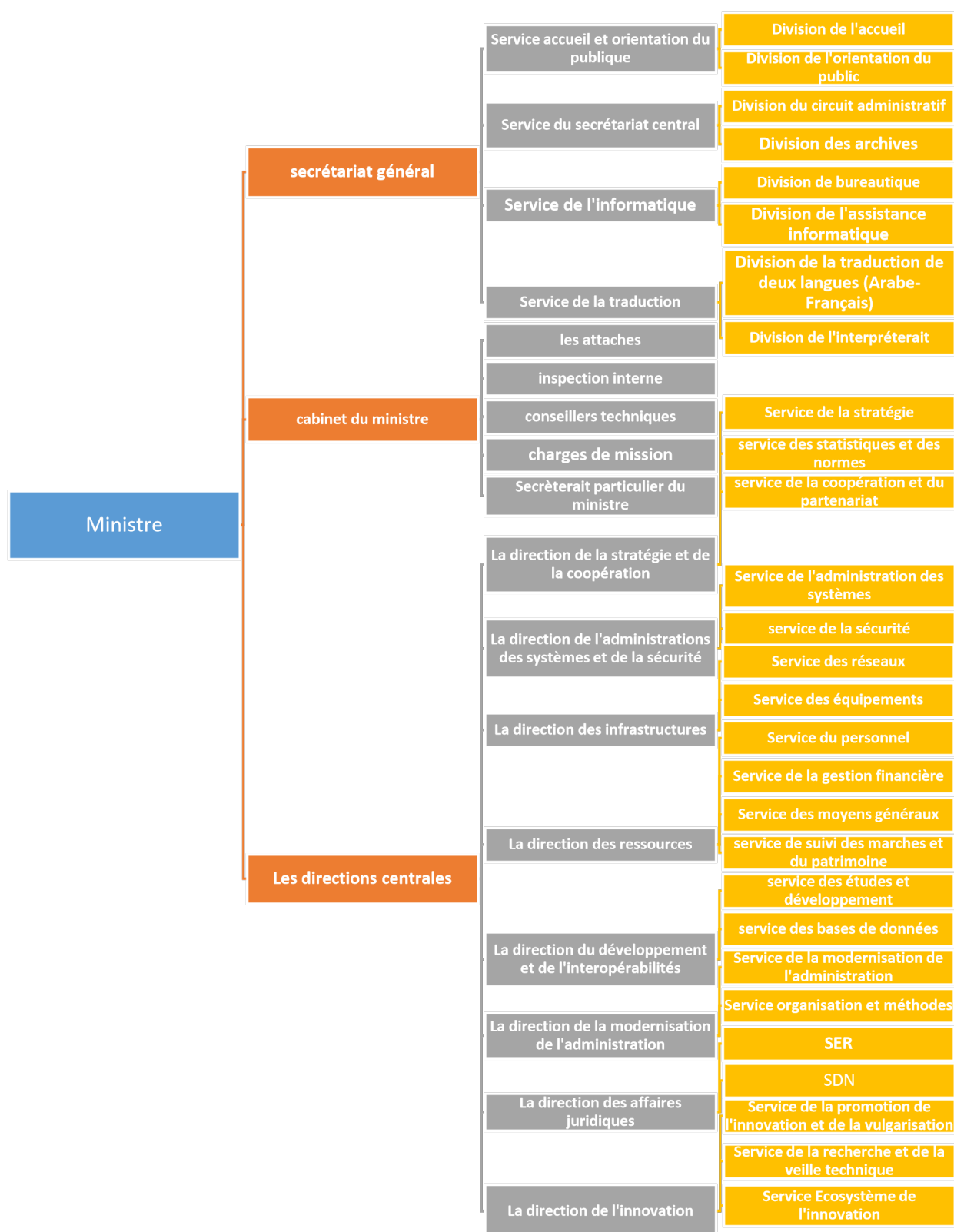


FIGURE 1 - ORGANIGRAMME DU MTNIMA

FIG. 2.1 : L'organigramme du Ministère.

Source : MTNIMA, 2025.

Chapitre 3

Contexte théorique et état de l’art

3.1 l’intelligence artificielle

L’intelligence artificielle (IA) désigne la simulation de l’intelligence humaine dans des machines programmées pour penser et agir comme des humains. Elle implique le développement d’algorithmes et de programmes informatiques capables d’effectuer des tâches nécessitant généralement l’intelligence humaine, telles que la perception visuelle, la reconnaissance vocale, la prise de décision et la traduction linguistique.

Depuis leur développement dans les années 1940, les ordinateurs numériques ont été programmés pour accomplir des tâches très complexes — telles que la découverte de démonstrations pour des théorèmes mathématiques ou jouer aux échecs — avec une grande maîtrise. Malgré les progrès continus de la vitesse de traitement et de la capacité mémoire des ordinateurs, il n’existe pas encore de programmes capables d’égaler la flexibilité humaine complète sur des domaines larges ou dans des tâches nécessitant une grande connaissance du quotidien.

En revanche, certains programmes ont atteint les niveaux de performance d’experts et de professionnels humains dans l’exécution de certaines tâches spécifiques, de sorte que l’intelligence artificielle, dans ce sens limité, se retrouve dans des applications aussi diverses que le diagnostic médical, les moteurs de recherche informatiques, la reconnaissance vocale ou manuscrite, et les chatbots.[1]

3.1.1 Historique de l’intelligence artificielle

L’idée de créer des objets intelligents remonte à l’Antiquité. Les Grecs anciens ont imaginé des créatures mécaniques, tandis que des ingénieurs chinois et égyptiens ont conçu des automates fonctionnant à l’eau ou à la vapeur. [2]

Au XIX^e siècle, Charles Babbage a conçu une machine mécanique capable de simuler des comportements intelligents, posant ainsi les premières bases de l’IA moderne. En 1936, Alan Turing a introduit le concept de la machine de Turing, démontrant qu’une machine pouvait résoudre tout problème représentable sous forme algorithmique. En 1950, il propose le célèbre “test de Turing” afin d’évaluer l’intelligence d’une machine [3].

L'intelligence artificielle est officiellement née en 1956, lors de la conférence de Dartmouth aux États-Unis, organisée par John McCarthy, Marvin Minsky, Claude Shannon et Nathaniel Rochester. McCarthy est d'ailleurs souvent considéré comme le père de l'IA [2].

Parmi les jalons historiques les plus importants :

- **1950** : Test de Turing, fondation de l'IA
- **1951** : Premier réseau de neurones (SNARC)
- **1952** : Programme d'apprentissage autonome (jeu de dames)
- **1956** : Terme « intelligence artificielle » introduit
- **1958** : Perceptron ; langage Lisp
- **1959** : Terme « machine learning » ; modèle Pandemonium
- **1964** : Traitement automatique du langage (STUDENT)
- **1965** : Premier système expert (Dendral)
- **1966** : Chatbot Eliza ; robot Shakey
- **1968** : Système multimodal SHRDLU
- **1969** : Rétropropagation ; limites des perceptrons
- **1973** : Rapport Lighthill, réduction financement IA
- **1980** : Commercialisation machines Lisp
- **1981** : Ordinateurs parallèles (précurseurs GPU)
- **1984** : Concept « hiver de l'IA »
- **1985** : Réseaux bayésiens
- **1988** : Traduction automatique statistique
- **1989** : CNN pour reconnaissance de caractères
- **1997** : LSTM ; Deep Blue bat Kasparov
- **2000** : Modèle probabiliste neuronal du langage
- **2006** : Lancement ImageNet ; IBM Watson
- **2009** : Apprentissage profond avec GPU
- **2011** : CNN « superhumain » ; Siri lancé
- **2012** : CNN profonde, explosion du deep learning

- **2013** : Tianhe-2 ; apprentissage par renforcement ; Word2vec
- **2014** : GAN ; autoencodeurs variationnels ; DeepFace
- **2016** : AlphaGo bat Lee Sedol ; voiture autonome Uber
- **2017** : Modèles de diffusion ; transformers ; alertes Hawking
- **2018** : Robot Cimon ; OpenAI GPT ; robot Lovot
- **2019** : Turing NLG ; IA dépasse radiologues
- **2020** : IA COVID-19 ; GPT-3 ; Omniverse ; AlphaFold
- **2021** : Dall-E ; robot souple quadrupède
- **2022** : Licenciement Lemoine ; AlphaTensor ; FakeCatcher ; ChatGPT
- **2023** : GPT-4 multimodal ; Bing + ChatGPT ; Bard ; appel à pause IA
- **2024** : Améliorations IA générative ; Bard GenAI controversé.[4]

Malgré des avancées prometteuses, l’IA a connu plusieurs périodes de stagnation appelées “hivers de l’IA” (dans les années 1970 et 1980) causées par des attentes irréalistes et des limites technologiques. La discipline connaîtra un renouveau à partir des années 2000 avec l’émergence du Deep Learning et des réseaux neuronaux performants.[3]

3.2 Les grandes familles de l’IA

3.2.1 IA Symbolique

L’intelligence artificielle symbolique est une approche classique de l’IA qui repose sur la représentation explicite des connaissances sous forme de symboles, et sur l’utilisation de règles logiques pour raisonner et prendre des décisions. Elle s’appuie sur des systèmes de logique formelle (comme les systèmes experts ou les moteurs d’inférence) pour simuler des formes de raisonnement humain. Cette méthode est particulièrement efficace dans les domaines où les règles sont bien définies et connues à l’avance.

L’IA symbolique proposait d’exploiter les connaissances par l’application de relations logiques entre des propositions unitaires (« si on est en hiver alors il fait froid dehors » par exemple). Le postulat est que l’intelligence est une affaire de raisonnement. Le but est alors de découvrir et de modéliser les règles permettant de raisonner justement. Dans le dernier quart du XX^{ème} siècle, elle prétendait la venue de robots autonomes qui remplaceraient l’humain dans ses tâches les plus complexes. La promesse n’a pas été tenue. Les « systèmes experts » qu’elle a générés sont bien encore utilisés pour quelques applications, mais ne bénéficient plus d’aucune reconnaissance publique. [5]

3.2.2 Machine Learning

L'apprentissage automatique ou Machine Learning (ML) est une sous-section du domaine de l'intelligence artificielle ou IA en informatique, qui vise à apprendre aux machines à effectuer une tâche sans être explicitement programmé, et cela en utilisant un des algorithmes qu'on appellera modèles et de données.

L'apprentissage automatique est apparu pour la première fois au milieu du XX^e siècle. Alan Turing, l'un des pionniers du domaine, a soulevé des questions fondamentales sur la capacité des machines à imiter certains aspects de l'intelligence humaine dans son article fondateur intitulé **Computing Machinery and Intelligence**. [6]

L'origine de l'apprentissage automatique (machine learning) au sens moderne est généralement associée au nom du psychologue Frank Rosenblatt, de l'Université Cornell, qui, s'appuyant sur des idées relatives au fonctionnement du système nerveux humain, a créé un groupe ayant construit une machine capable de reconnaître les lettres de l'alphabet Rosenblatt (1957, 1959, 1960). Cette machine, baptisée « **perceptron** » par son créateur, utilisait à la fois des signaux analogiques et discrets, et comprenait un élément seuil qui convertissait les signaux analogiques en signaux discrets. Elle est devenue le prototype des réseaux de neurones artificiels modernes (RNA), et son modèle d'apprentissage était proche des modèles d'apprentissage animal et humain développés en psychologie, voir Bush & Mosteller (1951). Rosenblatt lui-même a réalisé les premières études mathématiques sur le perceptron Rosenblatt (1959). Cependant, le théorème de Novikoff Novikoff (1962), qui établit les conditions de convergence de l'algorithme d'apprentissage du perceptron en un nombre fini d'étapes, est devenu plus largement connu. [7]

2.2.2.1 Méthodes d'apprentissage :

Il y a de nombreux sous domaines ou sous sections. ses dernières dépendent du type de problèmes que l'on voudrait traiter, on retrouve :

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning
- Semi-Supervised Learning

2.2.2.1.1 Apprentissage supervisé

L'Apprentissage supervisé est un paradigme central de l'apprentissage automatique basé sur des données d'entraînement étiquetées. Il permet aux modèles d'apprendre à associer des entrées à des sorties connues en minimisant les erreurs de prédiction via un processus itératif. Les techniques incluent la classification (ex. : détection de spam) et la régression (ex. : prédiction du prix des maisons). Il est largement utilisé dans des domaines comme la reconnaissance d'images ou les systèmes de recommandation [8], Les problèmes dans l'apprentissage supervisé sont 2 types : Classification, Régression.

-> Classification :

C'est une tâche d'apprentissage supervisé dont la sortie a des étiquettes définies (valeur discrète) ce qu'on appelle classes ou catégories.

On peut avoir des classifications Binaire ou alors Multiclasse.

La classification binaire prédit pour un modèle (0 ou 1) vrai ou faux, dans le cas de multiclassés le modèle va prédire plus d'une classe.

Le but ici reste le même quelque soit le nombre de classes, prédire si les valeurs discrètes appartiennent à une classe particulière.

Exemple : Gmail qui est en mesure de prédire si un mail donné est un spam ou mail normal.

-> Régression :

La régression est une tâche d'apprentissage supervisé dont l'objectif est de prédire une valeur continue à partir de données d'entrée. Contrairement à **la classification** qui prédit des catégories discrètes, la régression cherche à estimer une quantité numérique, comme le prix d'une maison ou la température. Pour évaluer la performance du modèle, on calcule généralement une mesure d'erreur (comme l'erreur quadratique moyenne), qui quantifie l'écart entre les valeurs prédites et les valeurs réelles. Plus cette erreur est faible, plus le modèle est considéré comme précis et performant.

2.2.2.1.2 Unsupervised Learning

L'apprentissage non supervisé est une catégorie de l'apprentissage automatique dans laquelle l'algorithme a pour tâche d'extraire des motifs, des relations ou des structures à partir de données non étiquetées, c'est-à-dire des données qui ne possèdent pas de labels ou de valeurs cibles explicites. Contrairement à l'apprentissage supervisé, où l'on entraîne un modèle à prédire une sortie connue à partir d'entrées annotées, l'apprentissage non supervisé cherche à comprendre la structure intrinsèque des données sans indication préalable.

Parmi les techniques les plus courantes de l'apprentissage non supervisé, on retrouve le **clustering** (regroupement), qui permet de segmenter les données en groupes homogènes selon leurs similarités, mais aussi la réduction de dimensionnalité, la détection d'anomalies, ou encore l'extraction de caractéristiques.

Ce type d'apprentissage est particulièrement utilisé lorsque l'on souhaite analyser des ensembles de données vastes et non labellisés, afin de découvrir des tendances, des patterns ou des regroupements naturels, ce qui peut servir de base pour des analyses plus poussées ou pour guider des prises de décision.

Ce type d'apprentissage est principalement utilisé quand on souhaite étudier un ensemble de données non labellisées.

-> Clustering :

Le clustering est une technique fondamentale en apprentissage non supervisé qui consiste à regrouper un ensemble de données en plusieurs sous-ensembles appelés clusters ou groupes. Chaque cluster regroupe des données partageant des caractéristiques ou des traits similaires, ce qui permet de mieux comprendre la structure interne des données sans avoir besoin d'étiquettes préalables.

Contrairement à la classification supervisée, où l'on dispose d'exemples annotés (avec des classes connues à l'avance), le clustering ne nécessite pas de données étiquetées. L'algorithme découvre par lui-même les regroupements naturels présents dans les données. Ce processus est particulièrement utile lorsque les catégories ou classes ne sont pas définies ou inconnues, et qu'on souhaite explorer ou segmenter les données de manière autonome.

L'algorithme de clustering va analyser les attributs des données (par exemple, la distance ou la similarité entre points) afin de former des groupes cohérents. Selon la méthode utilisée, le nombre de clusters peut être fixé à l'avance ou déterminé automatiquement par l'algorithme.

2.2.2.1.3 Reinforcement Learning

Entraînement par renforcement est une technique dans laquelle on immerge un agent dans un environnement où celui-ci interagit avec son environnement dans le but d'apprendre.

Un agent dans un environnement est muni de sensors pour capter les informations de son environnement, ainsi que d'actionneurs qui lui permettront d'agir dans son environnement.

Les agents observent l'entrée, puis ils effectuent une action en prenant des décisions. Une fois l'action réalisée, l'agent reçoit des récompenses en conséquence, ce qui renforce le modèle en stockant ses informations dans une base de données. La récompense peut être positive, négative ou nulle selon les actions effectuées.

2.2.2.1.4 Semi-Supervised Learning

Nous avons préalablement vu l'**apprentissage supervisé** et **non supervisé**, dont la principale différence réside dans le fait que les données soient étiquetées ou non. À cela s'ajoutent les méthodes adéquates utilisées pour traiter ces données.

L'apprentissage semi-supervisé regroupe ces deux principes. Il consiste à utiliser un ensemble réduit de données étiquetées accompagné d'un autre ensemble plus large de données non étiquetées du même type.

L'avantage de ce type d'apprentissage réside principalement dans le fait que le processus d'étiquetage des données est souvent long et coûteux. Le semi-supervisé permet donc de réduire les coûts tout en exploitant efficacement un grand volume de données.

-> Différence entre apprentissage supervisé et non supervisé :

Caractéristique	Apprentissage supervisé	Apprentissage non supervisé
Type de données	Données étiquetées (entrée + sortie)	Données non étiquetées (seulement l'entrée)
Objectif	Prédire les résultats à partir des caractéristiques d'entrée	Regrouper ou organiser les données sans étiquettes prédéfinies
Algorithmes courants	Régression linéaire, SVM, Arbres de décision, CNNs	K-Means, Classification hiérarchique, ACP, Autoencodeurs
Tâches principales	Classification, Régression	Regroupement (clustering), Réduction de dimension
Évaluation	Plus facile (comparaison des prédictions avec les étiquettes connues)	Plus difficile (pas de vérité terrain pour validation)
Cas d'usage réels	Diagnostic médical, détection de fraude, reconnaissance d'image	Segmentation de marché, détection d'anomalies, compression de données
Processus d'apprentissage	Nécessite des données annotées par des humains	Apprend directement à partir des données brutes
Précision des résultats	Résultats généralement précis et fiables	Précision modérée, dépend de la qualité du regroupement
Complexité computationnelle	Souvent élevée, dépend de la taille et des labels	Variable, parfois plus rapide sans labels
Besoin en données	Nécessite un grand nombre d'exemples étiquetés	Peut fonctionner avec peu ou pas d'étiquettes
Interprétabilité	Souvent plus simple à interpréter	Parfois plus difficile à interpréter
Adaptabilité	Moins flexible aux données non vues	Plus flexible pour découvrir de nouvelles structures
Robustesse au bruit	Peut être sensible aux erreurs d'étiquetage	Peut être robuste si bien conçu
Objectifs d'apprentissage	Minimisation de l'erreur entre sortie prédite et réelle	Découverte de la structure intrinsèque des données

Tableau 2 : Comparaison entre l'apprentissage supervisé et non supervisé

Ce tableau résume la différence entre apprentissage supervisé et non supervisé.

3.2.3 Deep Learning

Deep Learning ou apprentissage en profondeur ou DL est une branche du Machine Learning entièrement basée sur des réseaux de neurones artificiels.

Le concept d'apprentissage en profondeur existe depuis plusieurs années, mais il a été laissé à l'abandon faute de moyens nécessaires.

Dans le milieu des années 2000, le Machine Learning fait rage dans les compétitions de reconnaissance visuelle, en 2012 DeepMind, une startup dans le domaine de l'IA, arrive dans la compétition avec un algorithme de deep learning qui bat largement tous les autres compétiteurs. L'année suivante, tous les compétiteurs se sont tournés vers le deep learning au vu des résultats obtenus.

L'avancée du DL est due à l'augmentation en exponentiel qu'ont connu les machines en capacités de calculs, et de stockages, ainsi que la disponibilité de données de masses (big data). Ces 3 ingrédients étaient nécessaires pour exploiter le potentiel du DL, qui fût chose impossible dans les années 90.

Les pionniers qui ont soutenu le DL tels que Geoffrey Hinton, ou alors Yoshua Bengio qui a développé les réseaux GAN (Generative Adversarial Networks), Yann LeCun qui est au cœur d'une avancée fulgurante dans le domaine de reconnaissance d'images avec les réseaux Convolutionnels CNN, et son architecture LeNet. Geoffrey Hinton a prouvé que l'apprentissage profond pouvait résoudre des problèmes insolubles par d'autres approches.

2.2.2.1 Machine Learning vs Deep Learning :

La majeure différence qu'on note entre ces 2 concepts provient de la manière dont les données sont présentées au système (modèle).

Les algorithmes de ML nécessitent presque toujours des données structurées, alors que les réseaux d'apprentissage approfondis reposent sur des couches de réseaux de neurones artificiels (RNA). On voit aussi une différence au sein de l'architecture des modèles qui les composent, on note que les modèles type DL sont plus profonds que les modèles type ML.

Deep learning n'utilise que les réseaux de neurones, alors que pour le ML, les réseaux de neurones ne sont qu'une approche de conception des modèles parmi tant d'autres.

En considérant le fait que le DL est la prochaine étape de l'évolution du ML, inculquant aux machines la manière de prendre leurs décisions de façon précise sans l'intervention de l'expert humain.

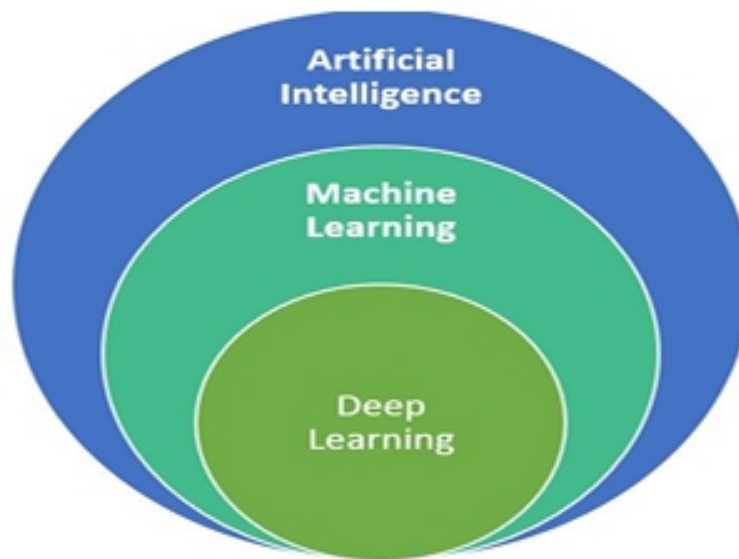


FIG. 3.1 : IA vs ML vs DL

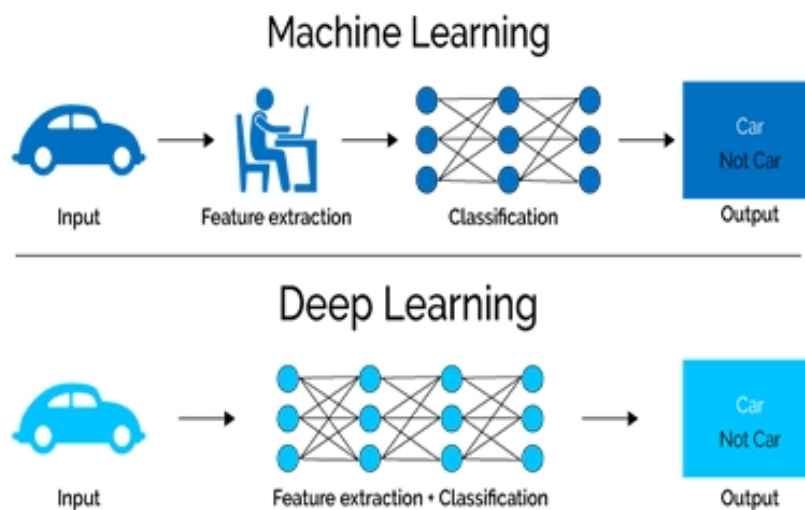


FIG. 3.2 : ML vs DL

2.2.2.2 Pourquoi Deep learning :

Il s'agit d'une combinaison de facteurs, dont :

- **L'omniprésence des données** : Nous sommes dans l'ère de l'informatisation (Internet of Things), et le propre du Deep Learning est de tirer parti d'une grande quantité de données pour en estimer une représentation abstraite et en tirer profit.
- **La puissance de calcul** : La théorie des réseaux de neurones existe depuis quelques décennies, mais c'est grâce à la puissance de calcul accessible aujourd'hui qu'elle se démocratise, notamment depuis que les GPUs sont devenus la plateforme de choix pour le Deep Learning.
- **Des besoins croissants dans le domaine de l'IA** : vision par ordinateur, reconnaissance vocale, traitement du langage, etc.
- **Un effet de mode** : On a tendance à vouloir appliquer le Deep Learning partout, alors que cela reste un moyen et non une fin. Certains problèmes sont tout à fait solubles par d'autres méthodes d'apprentissage statistique. Cela dit, si beaucoup de gens sont prêts à investir dans le Deep Learning, il est normal qu'il devienne si populaire.
- **La capacité de stockage** : qui est devenue beaucoup plus accessible à un prix raisonnable.
- **L'apparition de plateformes et de communautés fortes** : elles encouragent l'évolution de ce domaine, ainsi que sa démocratisation.

2.2.3 Réseau Neurones ou ANN

2.2.3.1 Neurone Biologique

Un neurone ou une cellule nerveuse est l'unité fonctionnelle de base qui constitue un système nerveux. Il assure la transmission d'un signal bioélectrique appelé influx nerveux.

Le neurone biologique reçoit en entrée des signaux transmis par d'autres neurones, grâce à une interaction qui est faite entre les dendrites et les synapses .

Le signal reçu est analysé et traité en effectuant une sommation des signaux en entrée. Le résultat est par la suite comparé à un seuil d'activation : s'il est supérieur, une décharge est alors envoyée le long de son axone vers d'autres neurones.

2.2.3.2 Perceptron :

En 1943, Walter Pitts et Warren McCulloch ont inventé le tout premier neurone artificiel qu'ils ont nommé Perceptron à l'époque. Ces deux chercheurs en neurosciences et sciences cognitives se sont inspirés du fonctionnement du cortex visuel des mammifères .

Le Perceptron était en mesure de résoudre certains problèmes, mais il présentait des limitations. C'est alors qu'ils eurent l'idée de mettre plusieurs perceptrons ensemble, ce qui donna un multi-perceptron, communément appelé réseau de neurones artificiels.

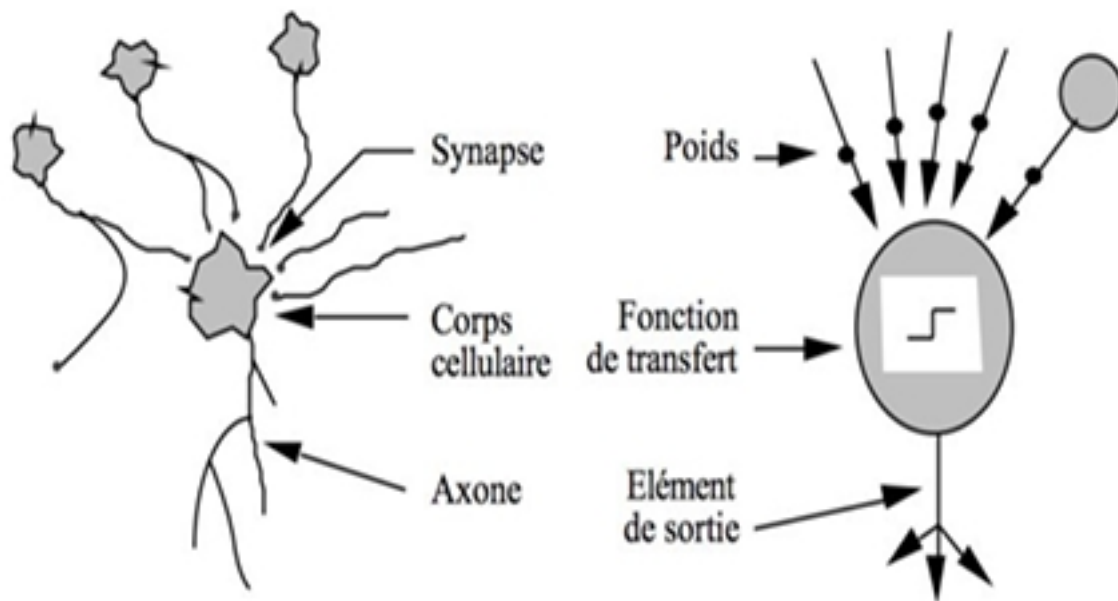


FIG. 3.3 : Neurone biologique Vs Neurone artificiel

2.2.3.3 Neurone Formel (Artificiel) :

On retrouve les poids synaptiques, la fonction d'activation, ainsi que les éléments de sortie, qui sont similaires aux fonctions occupées par les synapses, le corps cellulaire et l'axone chez le neurone biologique.

Le **neurone artificiel** reçoit lui aussi des **stimulis** (des **entrées** X_i , vecteur d'entrées) via ses **poids synaptiques**, que l'on appelle **entrées pondérées**. La notation la plus répandue est W_{ij} (**matrices de poids**). Chaque stimulus subit une **opération mathématique** $X_i \times W_{ij}$. Il est ensuite **analysé et traité** par le neurone, qui transmet le **résultat** à une **fonction d'activation**. Si le résultat est **supérieur** à un certain seuil, un **1** est renvoyé.

2.2.3.4 Fonction d'activation :

Il existe plusieurs **fonctions d'activation**. Le choix de cette dernière dépend de la **fonction du modèle** que l'on souhaite **modéliser**.

Voici un tableau récapitulatif des principales fonctions d'activation utilisées dans les réseaux de neurones :










Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
ReLU		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
PReLU		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
ELU		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

Tableau 1 : Fonctions d'activation et leurs dérivées.

La fonction d'activation à utiliser sont généralement non-linéaire, comme la fonction ReLU ou la tangente hyperbolique. Le but de ces fonctions, c'est de faire varier le résultat entre un intervalle précis. Exemple : pour la sigmoïde, le résultat varie entre 0 et 1.

2.2.3.5 Les couches d'un Réseau de neurone :

Le perceptron est organisé en trois couches :

- **Couche Entrée (Input Layer)** : C'est l'ensemble de neurones qui porte le signal d'entrée du réseau, et par la suite tous les neurones de cette couche sont reliés à la couche suivante.
- **Couche cachée (Hidden Layers)** : Elles peuvent être une ou plusieurs, c'est ici où les relations entre les variables vont être mises en exergue. Le choix du nombre de couches et de neurones est intuitif et nécessite de l'expérience venant de l'expert.
- **Couche sortie (Output Layer)** : Elle représente le résultat du réseau de neurones, c'est ce qu'on appelle la prédiction.

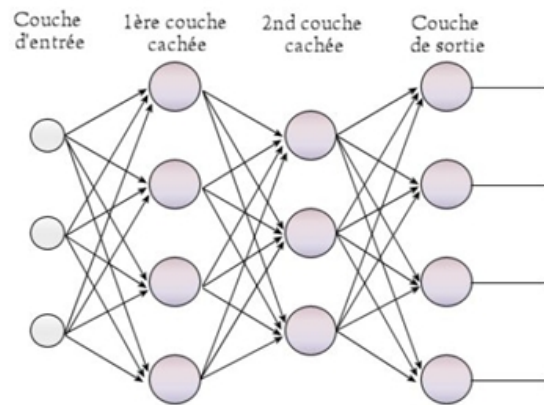


FIG. 3.4 : Perceptron Multicouche

2.2.3.5 Traitement des données

Les données sont le nerf de la guerre, surtout quand il s'agit de Machine Learning, car ces dernières occupent une importance capitale pour construire un bon modèle.

Avant de découper ou fragmenter les données afin de procéder à l'apprentissage, une étape préalable est nécessaire : le **prétraitement des données** ou *data preprocessing*.

Durant cette étape, on va traiter les données en effectuant des transformations. Le but est de transformer des données d'un état brut à un ensemble de données vierges exploitables par un modèle.

Il y a certains aspects à prendre en compte, comme le fait que les données prises en entrée de chaque modèle sont différentes, mais aussi la nécessité de généraliser afin qu'on puisse tester plusieurs algorithmes pour effectuer une comparaison.

Nous allons voir trois méthodes de prétraitement des données pour le *machine learning* :

- **Rescale Data** : remettre à l'échelle la donnée, car il arrive que certaines données au sein même du dataset aient des échelles différentes. Le redimensionnement des données en entrée est utile pour l'utilisation d'algorithmes d'optimisation (ex : *gradient descent*).
- **Binarize Data** : ou binarisation des données. C'est une méthode qui permet de transformer des entrées quelconques en valeurs binaires. Cela est possible grâce au seuil binaire (*binary threshold*) à partir duquel on compare si la valeur de nos données est supérieure au seuil, alors on marque un 1 ; dans le cas contraire, on marque un 0.
- **Standardize Data** : standardisation ou normalisation des données. C'est une méthode utile pour transformer des attributs avec des distributions gaussiennes différentes (moyennes et écarts-types variés) en une distribution gaussienne standard avec une moyenne de 0 et un écart-type de 1.

2.2.3.6 Découpage des données

L'ensemble des données (ou *dataset*) à disposition est généralement découpé en deux, voire trois parties :

- **Données d'apprentissage** (*Training set*) : représentent environ 80 % des données mises à disposition. Elles servent à entraîner notre modèle.
- **Données de test** (*Test set*) : représentent environ 20 % des données globales. Elles sont utilisées durant la phase de test du modèle pour évaluer ses performances finales.
- **Données de validation** (*Validation set*) : représentent une portion (environ 15 à 20 %) des données, utilisées pour tester le modèle durant l'entraînement, afin de valider les résultats obtenus. Ces données ne servent pas à évaluer l'efficacité finale d'un modèle, mais plutôt à comparer les performances entre différents algorithmes.

2.2.3.7 Fonction de perte

Également appelée **fonction d'erreur** (*Loss Function*), son rôle de base est simple : évaluer la qualité des prédictions effectuées sur le dataset. Si les prédictions sont mauvaises ou erronées, la fonction génère une valeur élevée. À l'inverse, si les prédictions sont bonnes, elle produit une valeur faible.

Il existe de nombreuses fonctions permettant de calculer la perte (ou erreur). Voici quelques-unes utilisées en classification :

- **Mean Square Error (MSE)** : c'est l'une des fonctions les plus utilisées, et la plus simple à mettre en œuvre. Elle consiste à prendre la différence entre les prédictions et la vérité terrain, à la corriger, puis à calculer la moyenne de cette erreur sur l'ensemble du jeu de données.
- **Log Loss (Cross Entropy Loss)** : il s'agit également d'une fonction de perte populaire. Son utilisation principale se trouve dans les problèmes de classification, en particulier ceux faisant intervenir les logarithmes (voir Section ??).

3.3 Les Deepfakes et leurs enjeux

3.3.1 Définition des Deepfakes

Les avancées notables des technologies basées sur les réseaux de neurones artificiels (ANN) jouent un rôle essentiel dans la falsification du contenu multimédia. Par exemple, des outils logiciels activés par l'IA comme *FaceApp* et *FakeApp* ont été utilisés pour des échanges de visages réalistes dans des images et des vidéos. Ce mécanisme de substitution permet à n'importe qui de modifier l'apparence du visage, la coiffure, le genre, l'âge, et d'autres attributs personnels. La propagation de ces vidéos truquées suscite de nombreuses inquiétudes et est devenue célèbre sous le terme de *Deepfake*. [9]

Le terme *Deepfake* est dérivé de *Deep Learning* (DL) et *Fake*, et il décrit un contenu vidéo ou image spécifique, photoréaliste, créé avec le support du DL. Ce mot a été nommé d'après un utilisateur anonyme de Reddit à la fin de 2017, qui a appliqué des méthodes d'apprentissage profond pour remplacer une personne.

Ces contenus ont pour but de reproduire de manière réaliste des visages, des voix ou des comportements humains, donnant l'illusion que des individus disent ou font des choses qu'ils n'ont jamais réellement dites ou faites.

3.3.2 Techniques de génération

La génération des deepfakes repose sur des modèles de deep learning, principalement les réseaux antagonistes génératifs (*GANs* - Generative Adversarial Networks).

Les GANs (*Generative Adversarial Networks*) se composent de deux réseaux de neurones : le générateur et le discriminateur, qui s'engagent dans un processus adversarial continu. Le générateur crée des données fictives imitant les données réelles, tandis que le discriminateur tente de distinguer les données réelles des données fictives et fournit un retour d'information au générateur. Ce processus adversarial se poursuit jusqu'à ce que le générateur produise des données indiscernables des données réelles à l'œil humain. [10]

Les *deepfakes* générés par les GANs offrent un grand potentiel en raison de leur créativité et de leur caractère innovant. Ils ont été utilisés dans l'industrie cinématographique pour les effets visuels, dans les jeux vidéo pour créer des personnages réalistes, dans l'éducation pour simuler des événements historiques et rendre l'apprentissage plus ludique et accessible, et même dans l'art pour générer de nouvelles formes d'expression créative. [10]

Ces derniers fonctionnent selon un principe de confrontation entre deux réseaux :

- **Le générateur** : crée de fausses images ou vidéos à partir de données d'entraînement.
- **Le discriminateur** : évalue si les contenus produits sont réels ou synthétiques.

Au fil de l'entraînement, ces réseaux s'améliorent mutuellement, permettant la production de contenus de plus en plus réalistes. D'autres techniques incluent l'auto-encodeur, le face swapping, et la synthèse vocale basée sur les modèles neuronaux.

3.3.3 Menaces associées

Les deepfakes posent de nombreux enjeux éthiques, juridiques et sécuritaires. Parmi les principales menaces :

- **La désinformation** : diffusion de fausses vidéos pour manipuler l'opinion publique, notamment en période électorale.
- **Le chantage et l'usurpation d'identité** : utilisation de visages ou de voix dans des contenus compromettants.
- **La déstabilisation sociale et politique** : création de fausses déclarations de dirigeants ou de figures publiques.
- **Les atteintes à la vie privée** : insertion de visages réels dans des contenus pornographiques ou violents.

Ces menaces soulignent l'importance de développer des outils de détection automatisée des deepfakes, ainsi qu'un cadre réglementaire adapté pour protéger les individus et les sociétés contre leurs usages malveillants.

Les deepfakes peuvent être utilisés pour vous tromper au point que vous remettiez quasiment toutes vos données personnelles et professionnelles. Ils peuvent être détournés à des fins financières ; de nombreux cas ont été rapportés où des directeurs d'entreprises et des PDG ont été usurpés, et où des cadres ont été sollicités pour effectuer des transferts d'argent. [11]

3.4 Techniques de détection des Deepfakes

3.4.1 Présentation des méthodes existantes

Pour faire face aux risques liés aux deepfakes, de nombreuses méthodes de détection ont été développées. Ces approches peuvent être classées en deux grandes catégories :

- **Méthodes basées sur des artefacts visuels** : elles analysent les imperfections générées par les algorithmes de synthèse, telles que des clignements d'yeux non naturels, des incohérences d'éclairage, ou des défauts au niveau des transitions faciales.
- **Méthodes basées sur l'apprentissage automatique** : elles utilisent des modèles d'intelligence artificielle entraînés à reconnaître les caractéristiques subtiles des contenus falsifiés. Les réseaux de neurones convolutifs (CNN), les réseaux récurrents (RNN), ou encore les architectures hybrides sont souvent employés dans ce cadre.

Certaines approches s'appuient également sur l'analyse de la fréquence (Fourier) ou sur des métadonnées (informations cachées dans les fichiers multimédia). D'autres cherchent à détecter des incohérences biométriques ou comportementales (expressions faciales, mouvements de tête, etc.).

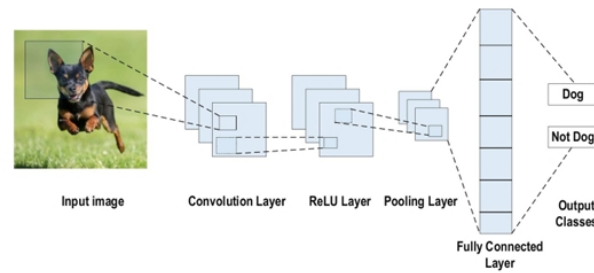


Fig. 7 An example of CNN architecture for image classification

FIG. 3.5 : An example of CNN architecture for image classification

Source : [13].

3.4.2 Focus sur les CNNs

Convolutional Neural Networks (CNN) Les CNN sont des réseaux de *deep learning* spécialisés en vision par ordinateur, capables de reconnaître et de classifier les caractéristiques d'une image. L'architecture des CNN a été influencée par l'organisation et les fonctions du cortex visuel. Elle est conçue pour ressembler aux connexions entre les neurones du cerveau humain.[12]

Les *Convolutional Neural Networks* (CNNs) sont largement utilisés dans la détection des deepfakes en raison de leur capacité à extraire automatiquement des caractéristiques visuelles complexes à partir d'images ou de vidéos.

Les avantages de l'utilisation des **CNN** par rapport aux autres réseaux de neurones traditionnels dans le domaine de la vision par ordinateur sont les suivants :

1. La principale raison de considérer les CNN est la fonctionnalité de partage des poids, qui réduit le nombre de paramètres entraînables du réseau et, par conséquent, aide le réseau à améliorer la généralisation et à éviter le surapprentissage.
2. L'apprentissage simultané des couches d'extraction de caractéristiques et de la couche de classification rend la sortie du modèle à la fois hautement organisée et fortement dépendante des caractéristiques extraites.
3. La mise en œuvre d'un réseau à grande échelle est beaucoup plus facile avec les CNN qu'avec d'autres réseaux de neurones.[13]

Un CNN est constitué de plusieurs couches :

- **Convolutives** : qui détectent des motifs locaux dans les images (bords, textures, formes).

Dans l'architecture des CNN, le composant le plus important est la couche de convolution. Elle se compose d'un ensemble de filtres convolutionnels (appelés noyaux). L'image d'entrée, exprimée sous forme de métriques N-dimensionnelles, est convoluée avec ces filtres pour générer la carte de caractéristiques (feature map) de sortie.

-> **Définition du noyau :** Une grille de nombres ou de valeurs discrètes décrit le noyau. Chaque valeur est appelée poids du noyau. Des nombres aléatoires sont affectés pour agir en tant que poids du noyau au début du processus d'entraînement du CNN. En outre, plusieurs méthodes différentes sont utilisées pour initialiser les poids. Ensuite, ces poids sont ajustés à chaque époque d'entraînement ; ainsi, le noyau apprend à extraire des caractéristiques significatives.

-> **Opération de convolution :** Initialement, le format d'entrée du CNN est décrit. Le format vecteur est l'entrée du réseau de neurones traditionnel, tandis que l'image multicanal est l'entrée du CNN. Par exemple, une image en niveaux de gris est à canal unique, tandis que le format RGB est à trois canaux.

Pour comprendre l'opération de convolution, prenons un exemple d'une image en niveaux de gris 4×4 avec un noyau 2×2 initialisé avec des poids aléatoires. D'abord, le noyau glisse sur toute l'image horizontalement et verticalement. Le produit scalaire entre l'image d'entrée et le noyau est déterminé, où leurs valeurs correspondantes sont multipliées puis additionnées pour créer une valeur scalaire unique. Ce processus est répété jusqu'à ce qu'aucun autre déplacement ne soit possible. Les valeurs ainsi calculées représentent la carte de caractéristiques en sortie.

La Figure 3.6 illustre graphiquement les calculs principaux exécutés à chaque étape. La zone en vert clair représente le noyau 2×2 , et la zone en bleu clair représente la même portion de l'image d'entrée. Les deux zones sont multipliées ; le résultat après sommation des produits (en orange clair) constitue une valeur dans la carte de caractéristiques en sortie.

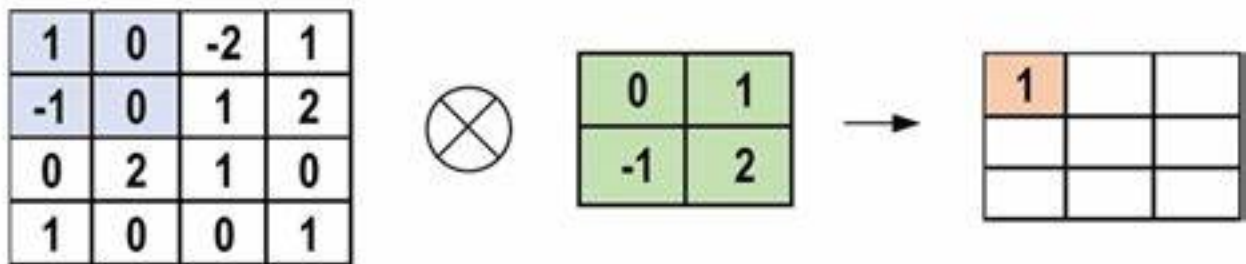
Dans cet exemple, aucun *padding* n'est appliqué, mais un *stride* de 1 (pas de déplacement) est utilisé. D'autres valeurs de stride peuvent être choisies. Une augmentation du stride réduit les dimensions de la carte de caractéristiques. Inversement, appliquer un padding permet de conserver des informations sur les bords de l'image d'entrée.

-> **Taille de la sortie :** En appliquant un padding, la taille de l'image d'entrée augmente, et par conséquent, la taille de la carte de caractéristiques en sortie augmente également.

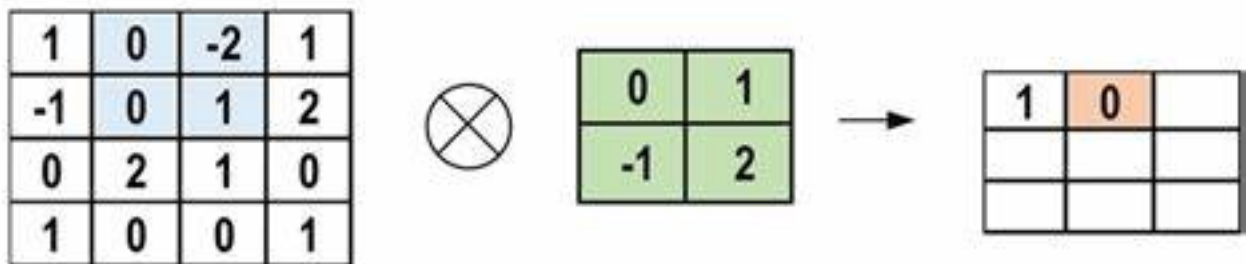
Avantages clés des couches de convolution

- **Connectivité clairsemée (Sparse Connectivity) :** Chaque neurone dans un réseau de neurones entièrement connecté est lié à tous les neurones de la couche suivante. En revanche, dans un CNN, seuls quelques poids sont utilisés entre deux couches adjacentes. Cela réduit le nombre total de connexions et la mémoire nécessaire, les opérations matricielles sont plus coûteuses que les produits scalaires utilisés dans les CNN.
- **Partage des poids (Weight Sharing) :** Tous les poids sont partagés à travers l'ensemble de l'entrée. Cela réduit considérablement le temps d'apprentissage et le nombre de paramètres à apprendre.

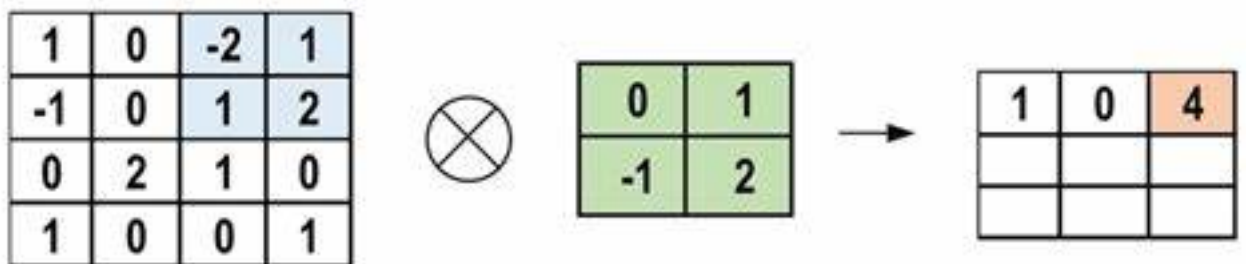
Step-1



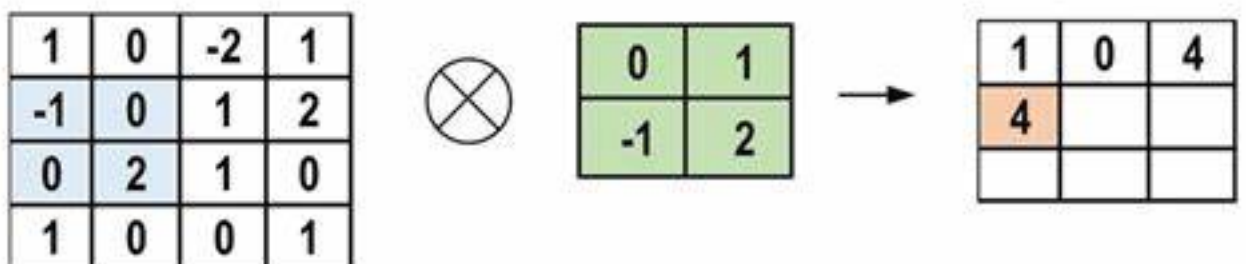
Step-2



Step-3



Step-4



Step-5

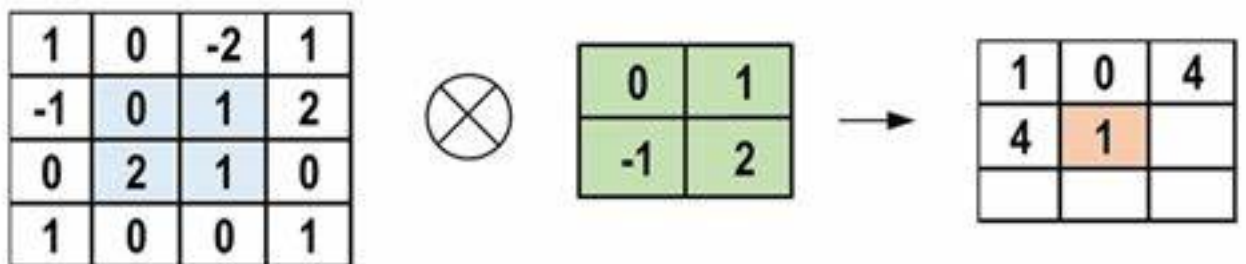


FIG. 3.6 : Les calculs principaux exécutés à chaque étape de la couche de convolution
Source :[13].

- **ReLU et pooling** : qui introduisent la non-linéarité et réduisent la dimensionnalité.

-> Couche de pooling

La tâche principale de la couche de pooling est le sous-échantillonnage des cartes de caractéristiques. Ces cartes sont générées à la suite des opérations de convolution. En d'autres termes, cette approche réduit la taille des cartes de caractéristiques volumineuses pour créer des cartes plus petites. Parallèlement, elle maintient la majorité des informations dominantes (ou caractéristiques) à chaque étape du processus de pooling.

De manière similaire à l'opération de convolution, la taille du *stride* et du noyau est d'abord définie avant l'exécution de l'opération de pooling. Plusieurs types de méthodes de pooling sont disponibles pour être utilisés dans différentes couches de pooling. Ces méthodes incluent le pooling arborescent, le pooling à porte, le pooling moyen, le pooling minimum, le pooling maximum, le pooling moyen global (GAP), et le pooling maximum global. Les méthodes de pooling les plus courantes et fréquemment utilisées sont le *max pooling*, le *min pooling* et le *GAP pooling*. La Figure 3.7 illustre ces trois opérations de pooling.

Parfois, la performance globale du CNN en est diminuée ; cela représente la principale limite de la couche de pooling, car cette couche aide le CNN à déterminer si une certaine caractéristique est présente ou non dans une image d'entrée donnée, mais elle se concentre exclusivement sur la localisation correcte de cette caractéristique. Ainsi, le modèle CNN peut manquer des informations pertinentes.

-> Fonction d'activation (non-linéarité)

La fonction principale de toute fonction d'activation dans tous types de réseaux de neurones est de mapper l'entrée vers la sortie. La valeur d'entrée est déterminée en calculant la somme pondérée des entrées du neurone ainsi que son biais (le cas échéant). Cela signifie que la fonction d'activation décide si un neurone doit être activé ou non en fonction d'une entrée donnée, en produisant la sortie correspondante.

Les couches d'activation non linéaires sont utilisées après toutes les couches avec des poids (appelées couches apprenantes, telles que les couches entièrement connectées et les couches convolutionnelles) dans l'architecture CNN. Cette non-linéarité des couches d'activation signifie que le passage de l'entrée à la sortie sera non linéaire ; de plus, ces couches donnent au CNN la capacité d'apprendre des choses plus complexes. La fonction d'activation doit également être différentiable, ce qui est une caractéristique extrêmement importante, car cela permet d'utiliser la rétropropagation de l'erreur pour entraîner le réseau.

Comme indiqué dans le tableau ??, ces fonctions d'activation sont parmi les plus couramment utilisées dans les CNN et autres réseaux neuronaux profonds.

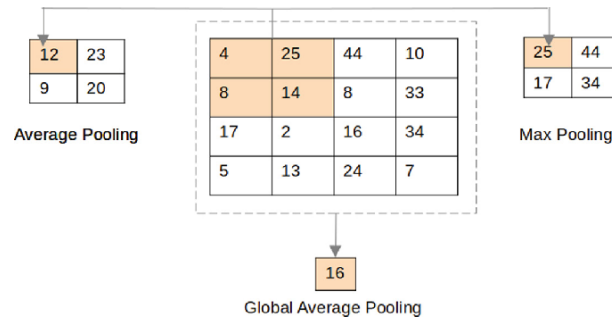


FIG. 3.7 : Operation-three-on-the-pooling-method
Source :[13].

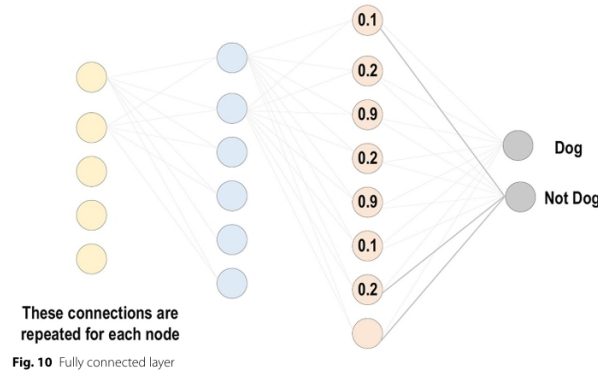


FIG. 3.8 : Couche fully-connected
Source :[13].

- **Couches fully-connected** : qui assurent la classification finale (réel vs faux).

Cette couche est généralement située à la fin de chaque architecture CNN. Dans cette couche, chaque neurone est connecté à tous les neurones de la couche précédente, c'est ce qu'on appelle l'approche entièrement connectée (Fully Connected, FC). Elle est utilisée comme classifieur du CNN. Elle suit la méthode de base du réseau neuronal perceptron multicouche conventionnel, étant un type de réseau neuronal à propagation avant. L'entrée de la couche FC provient de la dernière couche de pooling ou de convolution. Cette entrée est sous forme de vecteur, créé à partir des cartes de caractéristiques après aplatissement. La sortie de la couche FC représente la sortie finale du CNN, comme illustré dans la Fig 3.8. [13].

En entraînant un CNN sur un grand nombre d'exemples de contenus authentiques et de deepfakes, le modèle apprend à repérer les différences statistiques souvent invisibles à l'œil humain. Des architectures populaires comme VGG, ResNet ou EfficientNet sont souvent utilisées pour cette tâche.

3.4.3 Limites et défis de la détection des deepfakes

Malgré les avancées notables dans les techniques de détection des deepfakes, ce domaine reste confronté à de nombreux défis qui freinent encore son déploiement à grande échelle. En effet, la nature même des deepfakes, combinée à l’évolution constante des technologies, pose des obstacles importants que les chercheurs et praticiens doivent relever.

- **Évolution rapide des techniques de génération** : Les méthodes de création de deepfakes progressent à une vitesse impressionnante grâce aux avancées en intelligence artificielle et en réseaux antagonistes génératifs (GAN). Cette évolution continue permet de produire des contenus de plus en plus sophistiqués, avec une qualité visuelle et sonore quasi indiscernable de la réalité. Par conséquent, les systèmes de détection doivent constamment s’adapter pour reconnaître ces nouvelles formes de falsification, ce qui constitue un défi majeur en termes de mise à jour des algorithmes.
- **Généralisation limitée des modèles de détection** : Un autre problème clé réside dans la capacité des modèles à généraliser leur apprentissage. Souvent, un détecteur est entraîné sur un corpus spécifique de deepfakes, issus d’une méthode ou d’un contexte particulier. Lorsqu’il est confronté à des deepfakes générés par une technique différente ou sur un autre type de données, sa performance peut chuter drastiquement. Cette limitation réduit l’efficacité pratique des détecteurs dans des environnements réels, variés et non contrôlés.
- **Problèmes liés aux données d’entraînement** : Les bases de données utilisées pour entraîner les modèles de détection présentent souvent un biais important. Elles peuvent ne pas couvrir suffisamment la diversité des visages (âge, genre, origine ethnique), des expressions faciales, des conditions d’éclairage, ou encore des environnements variés. Ce manque de représentativité entraîne un risque d’inefficacité sur des cas non vus ou peu représentés durant l’entraînement, ce qui peut mener à des faux négatifs ou positifs problématiques.
- **Contre-attaques adversariales** : Les deepfakes ne sont pas les seuls à évoluer : les attaques dites adversariales ciblent directement les détecteurs en introduisant des modifications subtiles mais intentionnelles dans les images ou vidéos. Ces perturbations sont conçues pour tromper les algorithmes de détection sans altérer visuellement le contenu pour un observateur humain. La robustesse face à ces attaques est un enjeu crucial, car elles peuvent compromettre la fiabilité des systèmes, surtout dans des contextes sensibles comme la justice ou la sécurité.

Ces limites techniques et méthodologiques illustrent la complexité du combat contre la désinformation numérique. Pour y répondre efficacement, il est indispensable de continuer à investir dans la recherche multidisciplinaire, en combinant des approches algorithmiques robustes, la constitution de jeux de données plus diversifiés et représentatifs, ainsi qu’une réflexion éthique approfondie. Cette dernière est essentielle pour garantir que les technologies développées respectent la vie privée, les droits fondamentaux et ne renforcent pas les biais sociaux.

Chapitre 4

Analyse des besoins

Ce chapitre regroupe l'analyse des besoins et le cahier des charges du projet. Il permet de définir clairement les objectifs, le public cible, les fonctionnalités attendues, ainsi que les contraintes techniques et temporelles. Cette étape est essentielle pour cadrer le développement du système de détection de deepfakes et assurer sa réussite.

4.1 Analyse du besoin

Une analyse approfondie des besoins constitue une étape essentielle dans la réussite de tout projet technologique. Elle permet de définir clairement les objectifs, d'identifier les utilisateurs cibles, de recenser les fonctionnalités attendues et de prendre en compte les contraintes techniques et temporelles.

Ce projet s'inscrit dans le domaine en pleine expansion de l'intelligence artificielle, plus précisément dans la détection de deepfakes à l'aide de techniques d'apprentissage profond (*deep learning*). Il est destiné principalement à deux catégories d'utilisateurs :

- Les chercheurs et praticiens en intelligence artificielle, souhaitant expérimenter ou évaluer des modèles de détection de falsifications visuelles.
- Les étudiants en informatique ou en science des données, à la recherche d'une plateforme pédagogique pour comprendre les mécanismes de traitement d'images et de classification binaire.

L'objectif principal est de développer un système robuste, modulaire et évolutif, permettant la détection automatique de deepfakes à partir d'images faciales, via des architectures de réseaux de neurones convolutifs (CNN).

4.1.1 Exigences fonctionnelles et non fonctionnelles

Le système doit répondre aux fonctionnalités suivantes :

- Prétraitement des données : importation, nettoyage, détection et recadrage automatique des visages, normalisation et augmentation d'images.

- Entraînement du modèle : support des modèles préentraînés (ex. EfficientNetB0), gestion des hyperparamètres et callbacks (early stopping, checkpoints).
- Évaluation : calcul de métriques standardisées (accuracy, précision, rappel, F1-score) et visualisation des résultats.
- Interface utilisateur : interface Web ergonomique (Flask + HTML) pour soumettre des images et visualiser les résultats.

En termes de qualité, le système doit être performant (temps d'entraînement et d'inférence réduits), portable (multi-plateformes, y compris Google Colab), sécurisé (protection des données utilisateur), extensible et maintenable (code modulaire et documenté).

Contraintes

Les principales contraintes du projet sont les suivantes :

- Ressources matérielles limitées, notamment l'absence de GPU haut de gamme, ce qui impose le recours à des solutions comme Google Colab pour bénéficier d'un environnement avec GPU gratuit.
- Délai de réalisation fixé à trois mois, couvrant l'ensemble des phases : analyse, conception, implémentation, expérimentation et rédaction du rapport final.
- Gestion efficace du stockage des volumineux jeux de données, via l'utilisation de plateformes de stockage en ligne telles que Google Drive et Kaggle.
- Maintenabilité du code, avec une architecture modulaire et une documentation claire pour faciliter les évolutions futures.

4.2 Cahier des charges

4.2.1 Contexte

Les deepfakes sont des images et vidéos générées par intelligence artificielle qui imitent des visages réels de manière quasi indétectable. Ces contenus sont source de risques liés à la désinformation, à l'usurpation d'identité et à la cybersécurité. Le projet vise à concevoir et entraîner un modèle de détection basé sur des CNN.

4.2.2 Objectifs

- Étudier les techniques de génération de deepfakes (GANs).
- Collecter et prétraiter les données (extraction d'images, détection de visages).
- Développer et entraîner un modèle CNN avec TensorFlow/Keras.
- Tester et évaluer la précision du modèle sur des données non vues.

4.2.3 Spécifications fonctionnelles

- Entrée : image ou vidéo.
- Sortie : classification (réelle ou deepfake).
- Entraînement sur datasets open-source (FaceForensics++, DFDC, Celeb-DF).
- Évaluation via précision, rappel et F1-score.

4.2.4 Spécifications techniques

- Langage : Python.
- Frameworks : TensorFlow, Keras, OpenCV.
- Comparaison de plusieurs architectures CNN.

4.2.5 Planning résumé sur 3 mois

- Mois 1 : étude des deepfakes et préparation des données.
- Mois 2 : implémentation et entraînement des modèles CNN.
- Mois 3 : comparaison, amélioration et finalisation.

4.2.6 Livrables

- Modèle de détection fonctionnel (exploitable sur Google Colab).
- Rapport détaillé méthodologique et résultats.
- Présentation finale du projet et perspectives d'amélioration.

Chapitre 5

Réalisation

Ce chapitre présente la mise en œuvre concrète du projet, depuis le choix des technologies et de l'architecture jusqu'à l'implémentation technique. Il détaille l'environnement et les outils utilisés, la préparation et le traitement des données, ainsi que la réalisation du modèle de détection. Enfin, les résultats obtenus sont exposés et analysés pour en interpréter la pertinence et les performances.

5.1 Choix technologiques et architecture

5.1.1 Environnement et outils & Justification des choix



Google Colab ^a est une plateforme cloud gratuite de Google offrant un environnement Jupyter Notebook avec GPU gratuits. Elle est idéale pour le prototypage et l'entraînement de modèles de deep learning lourds.

^a<https://colab.research.google.com/>



Kaggle ^a est une plateforme collaborative pour accéder à des jeux de données, exécuter des notebooks et participer à des compétitions de machine learning avec un support GPU.

^a<https://www.kaggle.com/>



Jupyter Notebook ^a est un environnement interactif combinant code Python, visualisations et documentation. Il facilite l'expérimentation et la visualisation des résultats en data science.

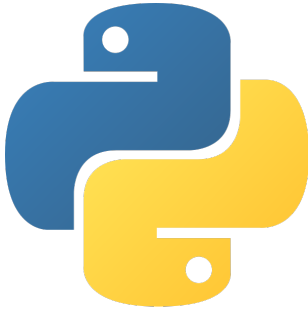
^a<https://jupyter.org/>



Visual Studio Code

Visual Studio Code ^a est un éditeur de code source léger et puissant, utilisé dans ce projet pour écrire, déboguer et gérer le code Python et les fichiers front-end.

^a<https://code.visualstudio.com/>



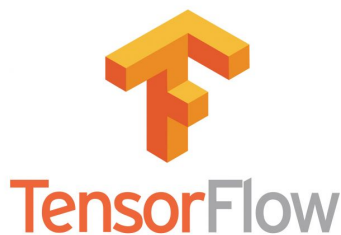
Python ^aest un langage de programmation interprété, simple et polyvalent. Il est utilisé dans ce projet pour le traitement des images, le chargement du modèle de détection de deepfakes et la mise en place du serveur via Flask.

^a<https://www.python.org/>



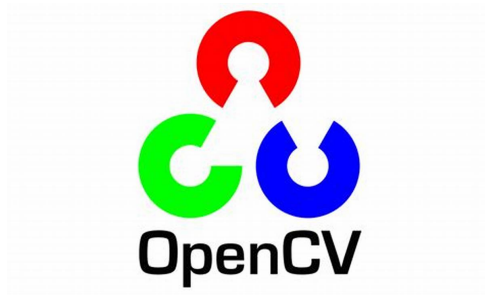
Github ^a est une plateforme d'hébergement de code basée sur Git. Dans ce projet, elle est utilisée pour versionner, sauvegarder et partager le code source de l'application web de détection de deepfakes.

^a<https://www.Github.org/>



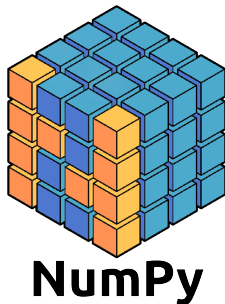
TensorFlow ^a est une bibliothèque open-source de Google dédiée au calcul numérique et au deep learning, permettant la création, l'entraînement et le déploiement de réseaux de neurones profonds.

^a<https://www.tensorflow.org/>



OpenCV ^a est une bibliothèque de traitement d'images et vidéos permettant la lecture, la conversion, le recadrage et la sauvegarde des images nécessaires au prétraitement des données.

^a<https://www.opencv.org/>



NumPy ^a est une bibliothèque essentielle pour la manipulation efficace des tableaux multidimensionnels et les calculs mathématiques sur les données image.

^a<https://www.numpy.org/>



Pandas ^a facilite la gestion et l'analyse des données tabulaires, notamment pour le traitement et la visualisation des résultats des prédictions.

^a<https://www.pandas.pydata.org/>



Flask

Flask est un micro-framework web en Python utilisé dans ce projet pour gérer la logique serveur. Il permet de charger le modèle de détection de deepfakes et de traiter les fichiers envoyés via l'interface web de manière locale.

HTML



HTML est un langage de balisage utilisé pour structurer le contenu de l'interface web du projet.

CSS



CSS (Cascading Style Sheets) est un langage de feuille de style utilisé pour décrire l'apparence et la mise en forme des pages web. Dans le cadre de ce projet, CSS est utilisé pour concevoir une interface utilisateur esthétique, responsive et cohérente.



JS est l'un des langages de base du développement web, utilisé dans ce projet pour rendre l'interface utilisateur interactive et dynamique. Il permet notamment de gérer les interactions avec l'utilisateur (comme la sélection et le pré-chargement des fichiers médias), d'améliorer l'expérience sur la page, et d'assurer une communication fluide avec le serveur Flask en local.



Overleaf est une plateforme en ligne collaborative qui facilite la rédaction et la mise en forme de documents en **LaTeX**. Dans le cadre de ce projet, Overleaf a été utilisé pour écrire et structurer le rapport final. Cette plateforme a permis de bénéficier d'une gestion efficace de la mise en page, de l'intégration des figures, des tableaux, ainsi que des références bibliographiques, tout en assurant une collaboration aisée et un accès facile au document depuis n'importe quel poste connecté à Internet.

5.1.2 Architecture technique proposée

L'architecture technique du système **DeepScan** a été conçue pour assurer une détection efficace et interactive des deepfakes tout en offrant une expérience utilisateur fluide et réactive. Cette architecture repose sur une organisation modulaire combinant une interface web dynamique, un serveur backend performant, ainsi qu'un modèle d'intelligence artificielle optimisé pour la classification des images et vidéos.

-> Présentation générale

Le système se compose de trois couches principales :

- **Interface utilisateur (Front-end)** : développée avec des technologies web modernes (HTML, CSS, JavaScript), elle permet à l'utilisateur d'importer des médias (images), de lancer l'analyse, et de visualiser les résultats de manière interactive.
- **Serveur d'application (Back-end)** : implémenté en Python avec le framework Flask, il gère la réception des fichiers, la communication avec le modèle de détection, et le retour des résultats à l'interface utilisateur.
- **Modèle d'intelligence artificielle** : un réseau de neurones convolutifs (CNN) basé sur EfficientNetB0 pré-entraîné, adapté à la classification binaire (deepfake / réel). Le modèle est chargé dans le backend et traite les médias pour générer des prédictions.

-> Flux de données

Le processus de traitement suit le schéma suivant :

1. L'utilisateur charge un fichier média via l'interface web.
2. Le média est envoyé au serveur Flask via une requête HTTP POST.
3. Le serveur prétraite le média (extraction de frames, détection des visages).
4. Le modèle CNN analyse les visages extraits pour détecter les deepfakes.
5. Les résultats (scores de confiance, annotations des visages) sont renvoyés au front-end.
6. L'interface affiche dynamiquement les résultats, incluant des cadres colorés autour des visages et des scores associés.

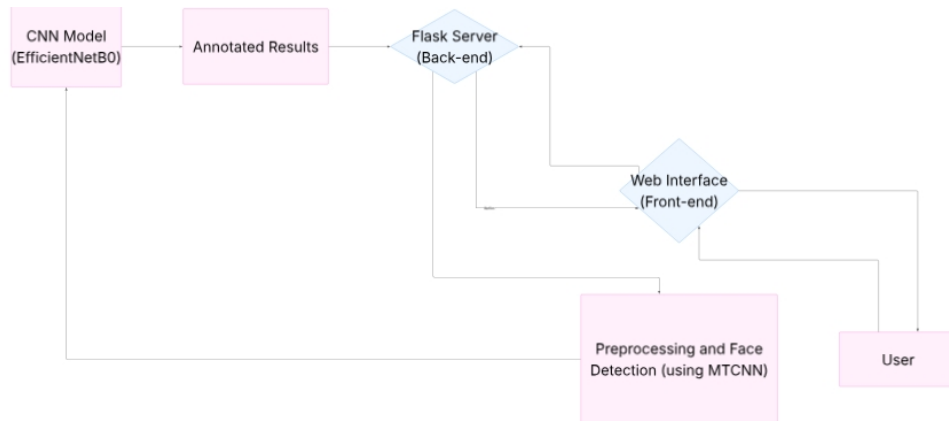


FIG. 5.1 : Diagramme d'architecture

-> Diagramme d'architecture

Cette architecture modulaire garantit une maintenance facilitée, une évolutivité du système, ainsi qu'une expérience utilisateur optimale.

5.2 Mise en œuvre du projet

5.2.1 4.2.1 Préparation et traitement des données

La performance d'un modèle d'apprentissage profond dépend fortement de la qualité, de la diversité et du volume des données utilisées lors de l'entraînement. Dans le cadre de la détection de deepfakes, il est essentiel de disposer d'un corpus riche en manipulations faciales, générées à l'aide de multiples techniques de synthèse, afin de garantir une capacité de généralisation optimale.

a) Constitution du corpus

Plusieurs jeux de données open source de référence ont été localement téléchargés, analysés et fusionnés. Les bases suivantes ont été utilisées :

- **DeepFake-TIMIT**
- **FaceForensics++**
- **Google Deep Fake Detection (DFD)**
- **Celeb-DF**
- **Facebook Deepfake Detection Challenge (DFDC)**

La fusion de ces jeux de données a permis de constituer un corpus complet de **100k vidéos**, représentant environ **1 140 identités distinctes**, et couvrant près de **20 méthodes de falsification** différentes (face swapping, reenactment, synthesis, etc.). Ce méga-dataset a été nettoyé, converti en images, structuré, puis téléversé sur mon compte **Kaggle** sous forme de dataset privé pour un accès facilité depuis l'environnement d'entraînement.

b) Prétraitement des données

Les vidéos ont été converties en images fixes à travers l'extraction de frames significatives. Les étapes suivantes ont ensuite été réalisées :

1. **Détection et recadrage de visages** : l'algorithme MTCNN a permis de localiser et de recadrer automatiquement les visages détectés dans chaque image.
2. **Nettoyage** : les images floues, partiellement détectées ou incorrectes ont été supprimées.
3. **Équilibrage des classes** : un sous-échantillonnage a été effectué afin d'obtenir un ensemble équilibré de **25 000 images réelles** et **25 000 images falsifiées**.
4. **Prétraitement** : les images ont été redimensionnées à 224x224, normalisées dans l'intervalle $[0, 1]$, et préparées sous forme de tenseurs exploitables par les modèles de deep learning.

Ce pipeline de traitement a été conçu pour garantir la qualité, la diversité et l'équilibre des données en vue de l'entraînement du modèle.

5.2.2 Conception et entraînement du modèle

La détection des deepfakes repose sur la capacité du modèle à identifier des artefacts subtils dans les visages. Pour cela, un pipeline d'entraînement robuste, automatisé et reproductible a été mis en œuvre dans Google Colab, en utilisant le framework **TensorFlow**.

a) Gestion du pipeline et suivi de progression

Un mécanisme de suivi a été implémenté via un fichier JSON hébergé sur Google Drive. Ce système permet d'enregistrer l'étape actuelle du pipeline (montage du drive, téléchargement des données, préparation, entraînement, etc.) et de reprendre à tout moment sans tout réexécuter.

b) Téléchargement du dataset et préparation locale

Le dataset unifié a été téléversé sur mon compte Kaggle. Il a été téléchargé via l'API Kaggle dans l'environnement Colab, puis converti en un ensemble de 50 000 images équilibrées.

c) Découpage du dataset

Le dataset a été divisé automatiquement à l'aide de la bibliothèque **split-folders** selon la répartition suivante : **80%** pour l'entraînement, **10%** pour la validation, et **10%** pour le test.

d) Architecture du modèle

Le modèle repose sur l'architecture **EfficientNetB0**, préentraînée sur ImageNet et utilisée comme base d'extraction de caractéristiques. À cette base ont été ajoutées les couches suivantes :

- **GlobalAveragePooling2D**
- **Dense(512)** avec ReLU et Dropout(0.5)
- **Dense(128)** avec ReLU et Dropout(0.3)
- **Dense(1)** avec activation **sigmoïde** pour la classification binaire

e) Générateurs de données

L'augmentation de données a été réalisée via **ImageDataGenerator**, avec des transformations telles que rotations, translations, zooms et inversions horizontales, afin d'améliorer la robustesse du modèle.

f) Entraînement et sauvegarde du modèle

Le modèle a été compilé avec l'optimiseur **Adam** (taux d'apprentissage = 0,0001) et la fonction de perte **binary_crossentropy**. Deux callbacks ont été utilisés :

- **EarlyStopping** : pour arrêter l'entraînement en cas de stagnation.

- **ModelCheckpoint** : pour sauvegarder automatiquement le meilleur modèle sur Google Drive.

Le modèle a été entraîné sur 20 époques avec validation continue.

g) Évaluation et prédictions

À l'issue de l'entraînement, le modèle a été évalué sur le jeu de test. Les prédictions ont été sauvegardées dans un fichier CSV, contenant :

- Le nom du fichier
- Le score de prédiction (probabilité entre 0 et 1)
- Le label prédit (réel ou falsifié)

Des statistiques globales telles que la moyenne des scores, la répartition réelle/falsifiée, et le nombre total de prédictions ont également été générées.

5.2.3 implémentation technique

L'implémentation du système repose sur une architecture modulaire combinant apprentissage profond, traitement d'image et interface web. Elle a été réalisée à l'aide de plusieurs technologies complémentaires, détaillées ci-dessous.

a) Environnement de développement

Le développement a été effectué principalement dans l'environnement **Google Colab**, qui offre un accès gratuit à des GPU (NVIDIA Tesla T4), facilitant l'entraînement des modèles profonds. Le stockage des données et des modèles a été assuré via **Google Drive**, tandis que les jeux de données ont été hébergés sur **Kaggle**.

b) Langages et bibliothèques utilisés

Le cœur du projet repose sur les technologies suivantes :

- **Python** : langage principal pour le développement de l'algorithme.
- **TensorFlow / Keras** : pour la conception, l'entraînement et l'évaluation du modèle EfficientNetB0.
- **OpenCV** : pour le traitement d'images (lecture, conversion, manipulation).
- **MTCNN** : pour la détection automatique des visages dans les images.
- **Split-Folders** : pour la division automatique du dataset en ensembles d'entraînement, validation et test.
- **Flask** : pour la création de l'API backend permettant le chargement d'images et l'inférence en ligne.
- **HTML/CSS/JavaScript** : pour le développement de l'interface utilisateur graphique (front-end).

c) Organisation du code

Le projet est structuré en plusieurs modules :

- **model/** : contient le script de construction et d'entraînement du modèle Efficient-NetB0.
- **preprocessing/** : regroupe les fonctions de traitement des images (détection, recadrage, nettoyage).
- **app/** : contient le serveur Flask, les routes d'API, et le chargement du modèle pour l'inférence.
- **frontend/** : interface web en React permettant à l'utilisateur de téléverser une image et de consulter le résultat.

d) Déroulement de l'exécution

Le pipeline d'exécution suit les étapes suivantes :

1. L'utilisateur charge une image via l'interface.
2. L'image est transmise au serveur Flask via une requête HTTP.
3. Le backend applique la détection de visage avec **MTCNN**, redimensionne l'image, la normalise et la transmet au modèle entraîné.
4. Le modèle retourne une probabilité indiquant si le visage est réel ou falsifié.
5. Le résultat est affiché à l'utilisateur avec une boîte délimitant le visage analysé.

e) Exécution locale

L'application peut être exécutée localement en important le modèle sauvegardé au format **.h5**. Une simple commande `python app.py` permet de lancer le serveur Flask. L'interface est accessible via le navigateur à l'adresse `http://localhost:5000`.

5.2.4 Développement de l'interface web de test

L'interface web, développée en HTML, CSS et JavaScript, constitue la couche de présentation du système DeepScan, permettant à l'utilisateur d'interagir facilement avec le modèle de détection CNN.

-> Présentation de l'interface L'interface se compose d'une section principale (*hero section*) offrant un espace clair pour le chargement et l'analyse des médias (images et vidéos). Les utilisateurs peuvent importer des fichiers au format JPEG, PNG, avec une limite de taille fixée à 50 Mo.

Un design moderne et immersif est obtenu grâce à un fond animé de particules et une grille subtile en arrière-plan, garantissant une expérience utilisateur engageante.

-> Fonctionnalités clés

- **Zone de dépôt et de sélection de fichiers** : L'utilisateur peut glisser-déposer un média ou cliquer pour sélectionner un fichier local.
- **Bouton d'analyse** : Apparaît automatiquement dès qu'un fichier valide est chargé, permettant de lancer le traitement.
- **Retour visuel** : Pendant l'analyse, une animation de chargement indique le traitement en cours.
- **Affichage des résultats** : Une fois l'analyse terminée, les résultats sont affichés avec un indicateur visuel (icône, couleurs) et des détails sur la classification et le score de confiance.
- **Aperçu modal** : Les images analysées peuvent être ouvertes en plein écran dans une fenêtre modale pour une inspection détaillée.

-> Aspects techniques L'interface utilise :

- Une structure HTML sémantique et accessible.
- Un style CSS moderne, avec animations et design responsive garantissant une bonne expérience sur tous types d'écrans.
- La gestion des événements (chargement, clic, glisser-déposer) est assurée par JavaScript (dans `static/script.js`).

-> **Illustrations** Les figures 5.2 et 5.3 montrent respectivement la zone de téléchargement de fichier et l'affichage des résultats d'analyse.

Modal d'aperçu L'interface inclut une fenêtre modale permettant d'afficher en grand format l'image sélectionnée, améliorant ainsi l'expérience utilisateur lors de la consultation des résultats.

5.2.5 Intégration du modèle avec l'interface

L'intégration du modèle de détection basé sur un réseau de neurones convolutionnels (CNN) avec l'interface web constitue une étape clé du projet, assurant une interaction fluide entre la couche frontend et le backend de traitement.

Le modèle, entraîné préalablement sur un large ensemble d'images réelles et manipulées, est chargé localement côté serveur à l'aide d'une application Flask. Cette application traite directement les fichiers médias sélectionnés par l'utilisateur via l'interface web. L'analyse se fait en local à l'aide du modèle enregistré dans le dossier du projet.



FIG. 5.2 : Affichage des résultats d'analyse avec indication visuelle et score de confiance



FIG. 5.3 : Affichage des résultats d'analyse avec indication visuelle et score de confiance

-> Flux d'intégration

- L'utilisateur télécharge une image ou une vidéo via l'interface web.
- Le fichier est envoyé par requête HTTP POST au serveur Flask.
- Le serveur effectue un prétraitement adapté (extraction des visages, redimensionnement).
- Le modèle CNN réalise la classification (deepfake ou authentique).
- Les résultats, comprenant la prédiction et la confiance associée, sont renvoyés en réponse JSON.
- L'interface affiche les résultats avec une visualisation claire et intuitive.

-> **Gestion des fichiers et traitement** Le backend gère la réception des fichiers via des routes sécurisées, assure leur stockage temporaire, et utilise des bibliothèques spécialisées (par exemple, MTCNN pour la détection faciale) afin d'isoler les régions d'intérêt avant analyse.

Aspects techniques L'utilisation de JSON pour la communication client-serveur garantit une interopérabilité simple et une mise à jour dynamique de l'interface sans rechargement complet de la page. De plus, la modularité du backend permet d'intégrer facilement de nouveaux modèles ou fonctionnalités dans le futur.

Cette intégration harmonieuse entre modèle et interface web facilite ainsi l'utilisation du système DeepScan par des utilisateurs non techniques, tout en conservant une architecture robuste et évolutive.

5.2.6 Démonstration

Cette section illustre le fonctionnement concret du système **DeepScan** à travers une démonstration détaillée et progressive de son utilisation, permettant de mieux comprendre les différentes étapes impliquées.

-> **Chargement du média** L'utilisateur commence par importer un fichier image via une interface web intuitive et conviviale. Le système prend en charge plusieurs formats courants tels que JPEG, PNG, et JPG, garantissant ainsi une grande flexibilité. Une limite de taille maximale de 50 Mo est imposée afin d'assurer un traitement rapide et efficace, évitant ainsi les lenteurs liées à des fichiers trop volumineux ou incompatibles. L'interface propose également des messages clairs en cas d'erreur ou de non-respect des contraintes.

-> **Analyse et traitement** Une fois le fichier téléchargé avec succès, l'utilisateur peut lancer l'analyse en cliquant sur un bouton dédié. Cette action déclenche l'envoi sécurisé du média vers le serveur local où est hébergé le modèle de réseau de neurones convolutifs (CNN). Le modèle procède à une extraction approfondie des caractéristiques faciales présentes dans l'image ou la vidéo. Cette étape inclut la détection précise des visages, leur alignement, puis l'évaluation minutieuse des indices susceptibles de révéler une falsification deepfake.

-> **Affichage des résultats** Les résultats de la prédiction sont immédiatement affichés dans l'interface de manière claire et dynamique. Pour chaque visage détecté, un cadre coloré est superposé, accompagné d'une annotation indiquant le score de confiance du modèle, exprimé en pourcentage. Ce retour visuel permet à l'utilisateur d'identifier rapidement les zones suspectes, tout en disposant d'une mesure quantitative de la fiabilité de la détection. En outre, des messages explicatifs accompagnent les résultats pour faciliter leur interprétation, même par des utilisateurs non experts.

-> **Interactivité** Le système propose également une expérience utilisateur enrichie grâce à une fenêtre modale interactive qui permet de visualiser en taille réelle les images analysées. Cette fonctionnalité favorise une inspection détaillée des visages et des zones suspectes, renforçant la transparence des résultats. De plus, l'utilisateur peut naviguer aisément entre les différents visages détectés, zoomer, ou encore exporter les images annotées pour un usage ultérieur. Ces options contribuent à une meilleure appropriation des résultats et à une utilisation pratique du système DeepScan dans des contextes variés.

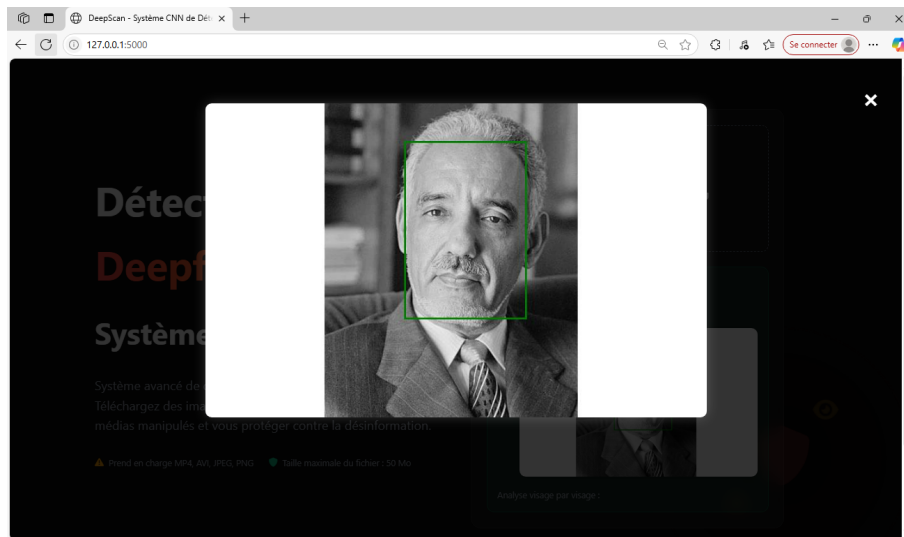


FIG. 5.4 : Exemple d'image analysée en taille réelle

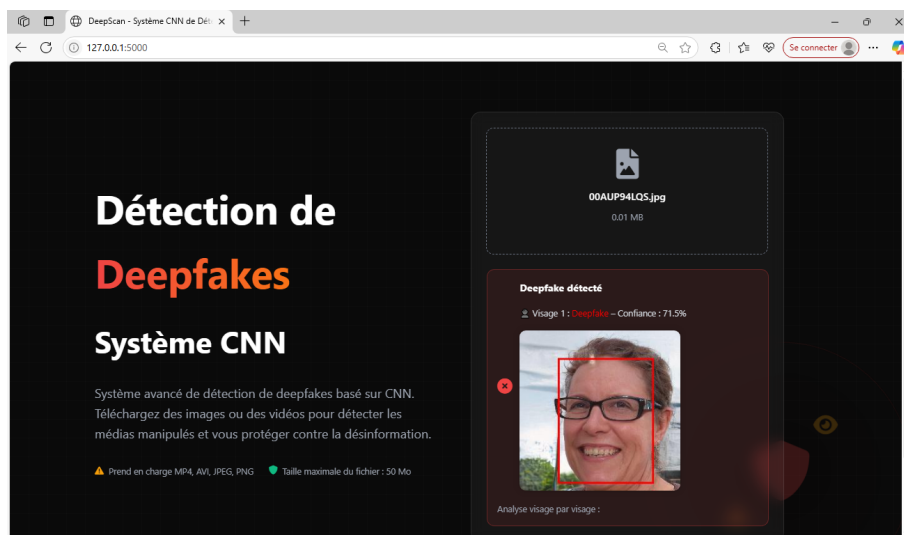


FIG. 5.5 : Exemple d'analyse d'une image deepfake via l'interface

Exemple d'utilisation : des capture d'écran montre l'analyse des images détectées comme deepfake, réelle avec les annotations et la confiance affichées.

5.3 Résultats et interprétation

5.4 Méthodologie d'évaluation

L'évaluation du modèle **DeepScan** a été réalisée à l'aide d'un jeu de test composé de 5000 images, réparties équitablement entre 2500 images réelles et 2500 deepfakes. Cette répartition équilibrée permet d'obtenir une évaluation fiable et représentative des performances du modèle.

Les métriques d'évaluation utilisées sont les suivantes :

- **Exactitude (Accuracy)** : proportion de bonnes prédictions sur l'ensemble des exemples.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Précision (Precision)** : proportion des prédictions positives correctes parmi toutes les prédictions positives.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Rappel (Recall)** : proportion des exemples positifs correctement identifiés parmi tous les exemples positifs réels.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **F1-score** : moyenne harmonique entre la précision et le rappel, donnant un équilibre entre ces deux métriques.

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

où :

- TP (True Positives) : nombre de deepfakes correctement détectés.
- TN (True Negatives) : nombre d'images réelles correctement classifiées.
- FP (False Positives) : nombre d'images réelles incorrectement classifiées comme deepfakes.
- FN (False Negatives) : nombre de deepfakes non détectés (classifiés comme réels).

Le modèle repose sur l'architecture **EfficientNetB0**, pré-entraînée sur le jeu de données ImageNet, qui a été adaptée en remplaçant les couches finales pour permettre une classification binaire (réel vs deepfake).


Les images ont été prétraitées par redimensionnement à une résolution de 128×128 pixels, suivie d'une normalisation des valeurs des pixels afin d'améliorer la convergence du modèle. La classification finale est réalisée à l'aide d'un seuil de décision fixé à 0.5,

c'est-à-dire que les sorties supérieures ou égales à ce seuil sont classées comme deepfakes, et les autres comme réelles.

Cette méthodologie rigoureuse permet d'évaluer précisément les capacités de détection du modèle dans des conditions proches d'une utilisation réelle.

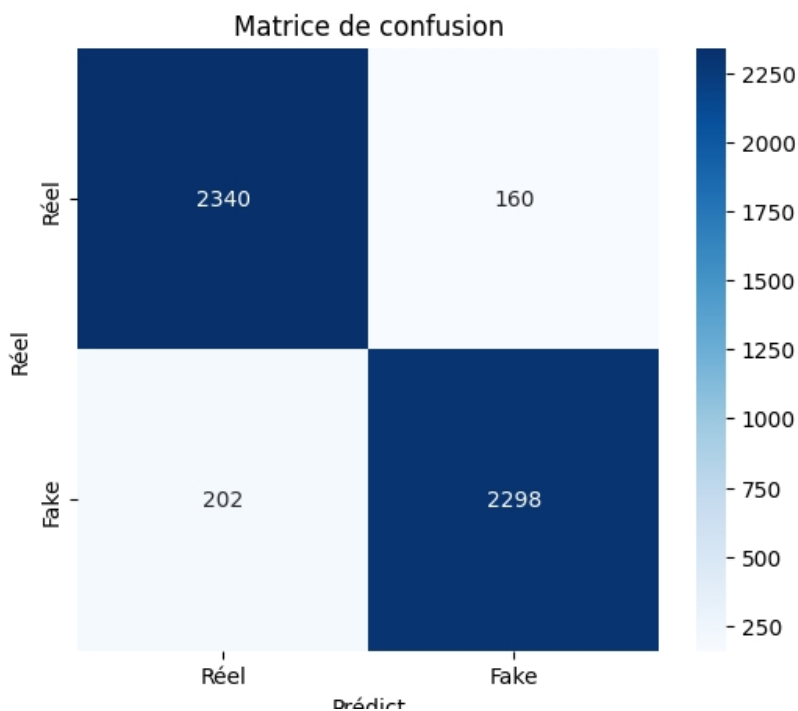
5.4.1 Résultats expérimentaux

Le rapport de classification obtenu est présenté ci-dessus.

 Rapport de classification :

	precision	recall	f1-score	support
réel	0.92	0.94	0.93	2500
fake	0.93	0.92	0.93	2500
accuracy			0.93	5000
macro avg	0.93	0.93	0.93	5000
weighted avg	0.93	0.93	0.93	5000

Exactitude (accuracy) : 0.9276



La matrice de confusion correspondante permet de visualiser la répartition des prédictions correctes et erronées.

Ces résultats correspondent à une exactitude globale de **92.76%**, avec un F1-score de **93%** sur les deux classes. Le modèle présente un bon équilibre entre détection des vrais deepfakes et limitation des faux positifs.

5.4.2 Analyse critique des performances

Le modèle développé pour la détection de deepfakes démontre une efficacité globale satisfaisante, avec des performances équilibrées entre les classes réelles et falsifiées. Plus précisément :

- **Taux de faux positifs modéré** : 160 images authentiques ont été incorrectement classées comme falsifiées. Ce résultat indique que le modèle conserve une bonne capacité à reconnaître les images réelles, tout en limitant les fausses alertes.
- **Taux de faux négatifs faible** : 202 deepfakes n’ont pas été détectés, ce qui suggère que le modèle parvient globalement à capturer les caractéristiques distinctives des images falsifiées.

Les **scores de précision et de rappel** atteints sont compatibles avec une utilisation en contexte réel de cybersécurité, où il est crucial de maintenir un équilibre entre la **réduction des fausses alertes** et la **détection fiable des contenus malveillants**.

Cependant, certaines **limitations structurelles** subsistent :

- Le modèle se limite à l’analyse d’*images statiques*, sans intégrer les dynamiques temporelles propres aux vidéos, ce qui restreint son champ d’action.
- Les jeux de données utilisés, bien que riches et variés, peuvent ne pas refléter toutes les *méthodes de génération les plus récentes*, réduisant ainsi la capacité de généralisation du modèle.
- La complexité du phénomène des deepfakes rend la tâche de détection particulièrement sensible aux *subtilités visuelles* que certains modèles légers peuvent ne pas capter de manière optimale.

En conclusion, le modèle mis en œuvre constitue une **base fiable et cohérente**, adaptée à des cas d’usage réels. Sa robustesse actuelle offre une marge d’exploitation concrète, tout en laissant entrevoir des pistes techniques pour consolider davantage ses performances en production. ““

Souhaites-tu que je l’intègre dans un style de rapport complet (avec section numérotée, page de titre, etc.) ou que je t’aide à l’intégrer dans un chapitre existant ?

Conclusion générale & Perspectives

Ce projet de fin d'études nous a permis une immersion dans le domaine passionnant de l'intelligence artificielle, à travers la détection de deepfakes. En partant de zéro, nous avons conçu un pipeline de détection basé sur des réseaux neuronaux convolutifs et une interface utilisateur permettant d'analyser des images pour détecter d'éventuelles manipulations. Ce travail, réalisé en partenariat avec le Ministère de la Numérisation, nous a permis de développer des compétences techniques solides, tout en nous confrontant à des enjeux éthiques majeurs liés à l'IA.

Implications éthiques

La lutte contre les deepfakes soulève des questions fondamentales :

- Comment éviter les faux positifs/négatifs dans la détection ?
- Qui est responsable en cas d'erreur de classification ?
- Comment protéger la vie privée des personnes dont les visages ou voix sont analysés ?

Ces interrogations montrent que l'IA doit être encadrée par des principes d'éthique, de transparence et de responsabilité.

Perspectives d'évolution

Plusieurs pistes d'amélioration peuvent être envisagées pour faire évoluer ce projet :

- **Fusion multi-modale** : combiner image, vidéo et son pour une détection plus robuste et complète.
- **Amélioration des performances** : utiliser des architectures de pointe comme EfficientNetV2 ou Vision Transformers, avec des techniques avancées d'entraînement.
- **Interface utilisateur évoluée** : ajout de visualisations interactives, score explicatif, historique d'analyse.
- **Déploiement web** : hébergement sur le cloud avec une API sécurisée accessible aux institutions.

En somme, cette expérience nous a offert non seulement une base technique solide, mais aussi une ouverture vers des problématiques sociétales et éthiques majeures. Elle constitue un tremplin vers d'éventuelles futures recherches ou projets professionnels dans le domaine de la cybersécurité, de l'IA ou de la lutte contre la désinformation.

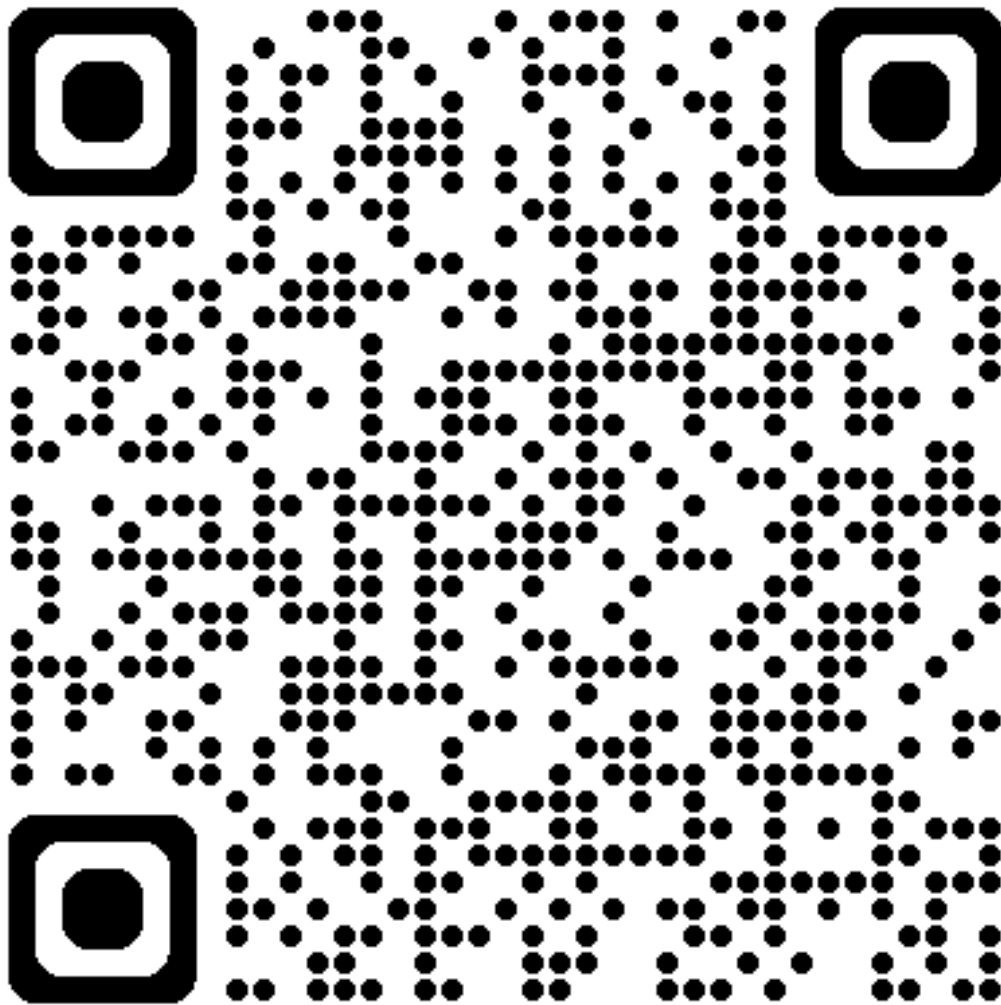


FIG. 5.6 : Scan ME :)

Bibliographie

- [1] B. J. COPELAND. “History of Artificial Intelligence (AI)”. In : *Encyclopedia Britannica* (juin 2025). Consulté le 28 juin 2025. URL : <https://www.britannica.com/science/history-of-artificial-intelligence>.
- [2] L. TSANG et al. “The Impact of Artificial Intelligence on Medical Innovation in the European Union and United States”. In : *Intellectual Property & Technology Law Journal* 29.8 (août 2017), p. 3-10.
- [3] Maad M. MIJWIL. “History of Artificial Intelligence”. In : (2015). Consulté sur ResearchGate. URL : https://www.researchgate.net/publication/322234922_History_of_Artificial_Intelligence.
- [4] Ron KARJIAN. “The history of artificial intelligence : Complete AI timeline”. In : (2025). Sous la dir. d’Industry Editor RON KARJIAN. Consulté le 28 juin 2025.
- [5] Arnaud KOHLER. “Relation entre IA symbolique et IA forte”. In : *HAL Archives Ouvertes* (2020). Version 2, consultée sur HAL. URL : <https://hal.science/hal-02444894v2>.
- [6] Umberto MICHELUCCI. “Machine Learning : History and Terminology”. In : *Fundamental Mathematical Concepts for Machine Learning in Science* (2024). Consulté le 28 juin 2025. DOI : [10.1007/978-3-031-56431-4_2](https://doi.org/10.1007/978-3-031-56431-4_2). URL : https://doi.org/10.1007/978-3-031-56431-4_2.
- [7] Alexander FRADKOV. “Early History of Machine Learning”. In : *IFAC-PapersOnLine* 53.2 (jan. 2020). Consulté via ScienceDirect, p. 1385-1390. DOI : [10.1016/j.ifacol.2020.12.1888](https://doi.org/10.1016/j.ifacol.2020.12.1888). URL : <https://doi.org/10.1016/j.ifacol.2020.12.1888>.
- [8] Naseem ZAIDI, Brijendra SINGH et Sunil YADAV. “The Evolution of Machine Learning Algorithms : A Comprehensive Historical Review”. In : *International Journal of Applied Research (IJAR)* 4.9 (2018). Consulté le 28 juin 2025, p. 49-55. ISSN : 2394-7500. DOI : [10.22271/allresearch.2018.v4.i9a.11451](https://www.allresearchjournal.com/archives/?year=2018&vol=4&issue=9&part=A&ArticleId=11451). URL : <https://www.allresearchjournal.com/archives/?year=2018&vol=4&issue=9&part=A&ArticleId=11451>.
- [9] Md Shohel RANA et al. “Deepfake Detection : A Systematic Literature Review”. In : *IEEE Access* 10 (2022), p. 25220-25245. DOI : [10.1109/ACCESS.2022.3154404](https://doi.org/10.1109/ACCESS.2022.3154404).
- [10] Jhanvi JHEELAN et Sameerchand PUDARUTH. “Using Deep Learning to Identify Deepfakes Created Using Generative Adversarial Networks”. In : *Computers* 14 (2025), p. 60. DOI : [10.3390/computers14020060](https://doi.org/10.3390/computers14020060).

- [11] Johan BRANDQVIST. “The Cybersecurity Threat of Deepfake”. In : (Spring term 2024). Supervisor : Martin Lundgren, Examiner : Marcus Nohlberg. URL : <https://www.diva-portal.org/smash/get/diva2:1880041/FULLTEXT01.pdf>.
- [12] D. BHATT et al. “CNN Variants for Computer Vision : History, Architecture, Application, Challenges and Future Scope”. In : *Electronics* 10.20 (2021), p. 2470. DOI : [10.3390/electronics10202470](https://doi.org/10.3390/electronics10202470). URL : <https://doi.org/10.3390/electronics10202470>.
- [13] Laith ALZUBAIDI et al. “Review of Deep Learning : Concepts, CNN Architectures, Challenges, Applications, Future Directions”. In : *Journal of Big Data* 8.1 (2021), p. 14-74. DOI : [10.1186/s40537-021-00444-8](https://doi.org/10.1186/s40537-021-00444-8). URL : <https://doi.org/10.1186/s40537-021-00444-8>.