*République Algérienne Démocratique et Populaire*
*Ministère de l'Enseignement Supérieur et de la Recherche Scientifique*
*Université Chadli Bendjedid*
*Faculté des Sciences et de la Technologie*
*Département informatique*

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة الشاذلي بن جديد
كلية العلوم و التكنولوجيا
قسم الاعلام الالي

جامعة الشاذلي بن جديد
UNIVERSITE CHADLI BENDJEDID

# MEMOIRE

**Présenté par**

**GASMI Wassim**

**Pour l'obtention de diplôme de**

**MASTER**

**Filière : Informatique**

**Spécialité : Systèmes Informatiques Intelligents**

## Thème

# UN SYSTEME DE DETECTION DES OBJETS

**Soutenu le :** 23/06/2024 Devant le jury composé de :

| | | | |
|---|---|---|---|
| **Mme BOUGARNE IMENE** | **MCB** | **Univ.de el tarf** | **Président** |
| **Mme MAKHLOF AMINA** | **MCB** | **Univ.de el tarf** | **Rapporteur** |
| **Mme MAATALLAH MAJDA** | **MCB** | **Univ.de el tarf** | **Examinateur** |

`

**Année Universitaire :** 2023/2024

# Remerciements

Mes remerciements vont tout d'abord à Dieu tout-puissant pour la volonté, la santé et la patience qu'il m'a accordées durant toutes ces années d'études.

En particulier, je tiens à remercier **Mme MAKHLOUF AMINA.** Ce projet ne serait pas aussi riche et n'aurait pas pu voir le jour sans son aide et sa supervision. Je la remercie d'avoir accepté d'être mon superviseur durant cette année, pour toute l'aide apportée et pour m'avoir proposé ce sujet, pour la qualité de son encadrement, sa rigueur et sa disponibilité, ses remarques fructueuses et ses orientations précieuses, qui ont effectivement contribué à l'avancement de ce travail.

Je remercie sincèrement :

- **Mme BOUGARNE IMENE** pour m'avoir honoré de présider les jury

- **Mme MAATALLAH MAJDA,** pour avoir accepté d'examiner ce travail.

Un grand merci aux **PROFESSEURS** de notre département d'informatique en particulier à **Mr.BENMACHICHE Abdelmadjid, Mr. CHEMMAM Chaouki** et **Mr. BETOUIL Abdelatif.**

Je suis reconnaissant envers tous les membres de ma famille, plus spécifiquement envers **MA MERE ET MON FRERE HICHEM** , pour leur soutien constant, leurs encouragements et leur patience tout au long de mes études. J'exprime également ma gratitude envers **mes collègues** pour les précieux moments partagés ensemble.

Je tiens à remercier, Mon amie **OUNNAS Aimane** qui m'ont aidé dans ce travail.

Enfin, je remercie tous ceux qui m'ont aidé de loin ou de près.
Merci à tous du fond du cœur.

# Dédicaces

Avec un grand cœur plein de chaleur, je dédie ce travail, aux êtres qui sont les plus chers pour moi :

Je tiens à exprimer toute ma gratitude à **ma mère bien-aimée**. Sa tendresse, son soutien inconditionnel et son amour constant ont été une source de réconfort et de force tout au long de mon parcours. Sa présence dans ma vie est un cadeau précieux, et je suis profondément reconnaissant de l'avoir à mes côtés. Je ne pourrais jamais assez la remercier pour tout ce qu'elle a fait et continue de faire pour moi. Je suis rempli d'amour et de reconnaissance envers cette femme exceptionnelle qui est **ma mère**.

Je souhaite exprimer ma profonde gratitude envers **mon cher frère Hichem** pour sa vigilance constante en ce qui concerne mon éducation, ses encouragements infaillibles et sa contribution à mon parcours d'études dans des conditions optimales. Je suis véritablement reconnaissant d'avoir un frère aussi attentionné et dévoué. Son soutien indéfectible a été essentiel pour mon développement académique et personnel. Je lui suis infiniment reconnaissant pour tout ce qu'il a fait pour moi.

Je dédie également ce travail à la mémoire de **mon père**, qui, bien qu'il ne soit plus parmi nous, continue de m'inspirer chaque jour. Son amour pour l'apprentissage et ses encouragements constants ont toujours été une source de motivation pour moi. Sa passion pour l'éducation et son soutien indéfectible me guident encore aujourd'hui. Je suis profondément reconnaissant pour tout ce qu'il m'a appris et pour l'exemple exceptionnel qu'il a été. Que son âme repose en paix.

***Gasmi wassim***

# Abstract

*This thesis explores the development of an advanced system for the detection of objects in computer vision, crucial for the autonomous and precise interpretation of visual environments in sectors such as security, automotive, medicine and industry..*

*Despite recent advances, current technologies are limited by their insufficient performance in complex environments and their lack of flexibility to meet the varied needs of users.*

*This project aims to overcome these challenges using Convolutional Neural Networks (CNN) and the MobileNetV2 architecture, based primarily on the Pascal VOC dataset.*

*Key objectives include designing a precise object detection system, efficiently integrating MobileNetV2 to optimize hardware resources, and continuously improving accuracy and energy efficiency.*

*The results show significant improvements in terms of accuracy and robustness compared to conventional methods, although challenges persist, particularly with regard to the generalization of the model to diverse environments.*

*Future areas for improvement include exploring multimodal data fusion, continuous optimization of CNN architectures, and the application of advanced data augmentation techniques.*

*In conclusion, this work represents a substantial contribution to the evolution of object detection systems, highlighting the importance of the Pascal VOC dataset in our approach. It paves the way for future research and technological developments in various industrial and technological sectors.*

***Keywords:*** *Computer vision; Object detection; moblinet-v2; CNN.*

# Résumé

*Ce mémoire explore le développement d'un système avancé de détection d'objets en vision par ordinateur, crucial pour l'interprétation autonome et précise des environnements visuels dans des secteurs comme la sécurité, l'automobile, la médecine et l'industrie..*

*Malgré des avancées récentes, les technologies actuelles sont limitées par leurs performances insuffisantes dans les environnements complexes et leur manque de flexibilité pour répondre aux besoins variés des utilisateurs.*

*Ce projet vise à surmonter ces défis en utilisant les Réseaux Neuronaux Convolutifs (CNN) et l'architecture MobileNetV2, en se basant principalement sur l'ensemble de données Pascal VOC.*

*Les objectifs principaux incluent la conception d'un système de détection d'objets précis, l'intégration efficace de MobileNetV2 pour optimiser les ressources matérielles, et l'amélioration continue de la précision et de l'efficacité énergétique.*

*Les résultats montrent des améliorations significatives en termes de précision et de robustesse par rapport aux méthodes conventionnelles, bien que des défis persistent, notamment en ce qui concerne la généralisation du modèle à des environnements diversifiés.*

*Les pistes d'amélioration futures incluent l'exploration de la fusion de données multimodales, l'optimisation continue des architectures CNN, et l'application de techniques avancées d'augmentation de données.*

*En conclusion, ce travail représente une contribution substantielle à l'évolution des systèmes de détection d'objets, en soulignant l'importance de l'ensemble de données Pascal VOC dans notre approche. Il ouvre la voie à des recherches futures et à des développements technologiques dans divers secteurs industriels et technologiques.*

***Mots clés :*** *Vision par ordinateur; Détection d'objets ; moblinet-v2 ; CNN.*

# ملخص

تستكشف هذه الأطروحة تطوير نظام متقدم للكشف عن الأشياء في الرؤية الحاسوبية، وهو أمر بالغ الأهمية للتفسير المستقل .والدقيق للبيئات البصرية في قطاعات مثل الأمن والسيارات والطب والصناعة

وعلى الرغم من التطورات الأخيرة، فإن التكنولوجيات الحالية محدودة بسبب عدم كفاية أدائها في البيئات المعقدة وافتقارها إلى .المرونة لتلبية الاحتياجات المتنوعة للمستخدمين

يهدف هذا المشروع إلى التغلب على هذه التحديات باستخدام الشبكات العصبية التلافيفية *(CNN)* .و الهندسة

المعمارية *MobileNetV2*, استنادًا بشكل أساسي إلى مجموعة بيانات *Pascal VOC*

تشمل الأهداف الرئيسية تصميم نظام دقيق لكشف الأجسام، ودمج *MobileNetV2* ,بكفاءة لتحقيق اقصى قدر من موارد

. الأجهزة، والتحسين المستمر للدقة وكفاءة الطاقة

وتظهر النتائج تحسناً كبيراً من حيث الدقة والقوة مقارنة بالطرق التقليدية، على الرغم من استمرار التحديات، لا سيما فيما .يتعلق بتعميم النموذج على بيئات متنوعة

تشمل مجالات التحسين المستقبلية استكشاف دمج البيانات متعددة الوسائط، والتحسين المستمر لبنى*CNN*

.وتطبيق تقنيات زيادة البيانات المتقدمة

في الختام، يمثل هذا العمل مساهمة كبيرة في تطور أنظمة اكتشاف الأجسام، مما يسلط الضوء على أهمية مجموعة بيانات باسكال للمركبات العضوية المتطايرة في نهجنا. إنه يمهد الطريق للبحوث المستقبلية والتطورات التكنولوجية في مختلف القطاعات الصناعية والتكنولوجية

**الكلمات الرئيسية:** الرؤية الحاسوبية ؛ كشف الأجسام ؛ *moblinet-v2*; ، *CNN*.

# Contenu

## Table des matières

# Listes des figures

# Listes des tableaux

# Liste des abréviations

- **CNN :** Convolutional Neural Network
- HOG : Histogram of Oriented Gradients
- **SVM** : Support Vector Machine
- **DPM** : Deformable Part Models
- **SIFT** : Scale-Invariant Feature Transform
- **SURF** : Speeded-Up Robust Features
- **ANN**: Artificial Neural Network
- **CPU :** Central Processing Unit
- **GPU** : Graphics Processing Unit
- **XML** : eXtensible Markup Language
- **TFRecord** : TensorFlow Record
- **MSE** : Mean Squared Error in

# Introduction générale

La détection d'objets représente un pilier fondamental de l'évolution de la vision par ordinateur, permettant aux systèmes informatiques d'interpréter de manière autonome et précise leur environnement visuel. Cette avancée technologique revêt une importance cruciale dans divers secteurs tels que la sécurité, l'automobile, la médecine et l'industrie, où la précision et la fiabilité des systèmes de détection sont essentielles pour assurer des performances optimales et sécuriser les opérations quotidiennes.

Malgré les progrès significatifs réalisés, les technologies actuelles de détection d'objets sont confrontées à des défis substantiels. Parmi ceux-ci figurent les performances limitées dans des environnements complexes ainsi que la nécessité croissante de scalabilité et de flexibilité pour répondre efficacement aux besoins diversifiés des utilisateurs. Ces défis entraînent souvent des taux d'erreur élevés et une fiabilité réduite, limitant ainsi l'adoption pratique de ces technologies novatrices.

Ce projet est motivé par la nécessité de développer une solution avancée et adaptable pour la détection d'objets, exploitant les dernières avancées en Deep Learning, en particulier les Réseaux de Neurones Convolutifs (CNN). Le Deep Learning a révolutionné ce domaine en permettant aux systèmes informatiques de modéliser et d'apprendre automatiquement à partir de données complexes, telles que des images. Inspirés par le fonctionnement du cortex visuel humain, les CNN offrent une précision et une efficacité remarquables pour la détection et la classification automatique d'objets.

Les objectifs de ce projet sont de concevoir et développer un système de détection d'objets robuste et précis en intégrant une architecture basée sur MobileNetV2 avec un modèle CNN adapté. Ce choix stratégique vise à exploiter pleinement les capacités avancées du Deep Learning pour améliorer la précision, la robustesse et l'efficacité énergétique de la détection d'objets dans des environnements diversifiés. Nous cherchons à repousser les limites actuelles en matière de détection d'objets, en proposant une solution innovante qui permettra de maintenir des performances élevées tout en optimisant l'utilisation des ressources matérielles disponibles. En mettant l'accent sur l'application pratique et l'innovation technologique, ce projet vise à contribuer significativement à l'évolution des systèmes autonomes dans divers secteurs industriels et technologiques.

Le mémoire réalisé dans le cadre de cette recherche est structuré de manière méthodique afin de fournir une présentation claire et cohérente des travaux réalisés. La structure proposée pour ce mémoire est la suivante :

Hormis l’introduction générale et la conclusion générale, on a le chapitre 1, intitulé "**Etat de l’Art"** explore les avancées majeures et les techniques actuelles dans le domaine de la détection d'objets. Nous examinerons les méthodes classiques comme Viola-Jones et HOG, ainsi que les avancées récentes basées sur les CNN, qui permettent une reconnaissance d'objets précise et efficace. Ce chapitre mettra en lumière l'importance de ces techniques dans divers secteurs industriels et technologiques

Le deuxième chapitre intitulé **"conception",** a souligné l'importance cruciale des choix architecturaux et des méthodologies rigoureuses dans le développement de solutions avancées pour la détection d'objets. Il a utilisé l'architecture MobileNetV2 comme fondation technologique principale pour maximiser l'efficacité des ressources matérielles. En détaillant la conception d'un modèle CNN adapté aux exigences de performance et d'efficacité énergétique, le chapitre a posé les bases pour assurer une détection précise et robuste dans divers environnements. De plus, il a mis en avant l'importance stratégique de la collecte et du prétraitement des données, en présentant une méthodologie rigoureuse pour la sélection et le traitement des ensembles de données.

Le chapitre 3, intitulé "Implémentation", se concentre sur l'implémentation pratique de l'approche proposée et présente les résultats obtenus lors des expérimentations. Il décrit l'environnement de travail et les outils de programmation utilisés, ainsi que la méthodologie d'évaluation et les métriques utilisées pour mesurer les performances du modèle. L'utilisation de l'ensemble de données Pascal VOC a été cruciale pour l'entraînement et l'évaluation du modèle, garantissant ainsi la pertinence et la validité des résultats dans des applications réelles. Les expérimentations sont détaillées et les résultats sont analysés en profondeur. La partie analyse offre une synthèse des résultats et des accomplissements de la recherche, mettant en évidence les limitations de l'étude et proposant des pistes d'amélioration pour de futures recherches.

# Chapitre 1 : Etat de l'Art

## 1. Introduction

Ce premier chapitre de l'état de l'art se concentre sur l'exploration des avancées majeures et des techniques actuelles dans le domaine de la détection d'objets. Cette technologie permet aux systèmes informatiques de percevoir, d'interpréter et de réagir visuellement de manière autonome, ouvrant la voie à une multitude d'applications pratiques et innovantes. En simulant l'aptitude humaine à identifier des entités spécifiques, la détection d'objets repose sur des techniques avancées de traitement d'images, de reconnaissance de formes et d'apprentissage automatique.

Le Deep Learning, en particulier, a révolutionné ce domaine en permettant aux systèmes informatiques de modéliser et d'apprendre à partir de données complexes de manière automatique. Au cœur de cette révolution se trouvent les Réseaux de Neurones Convolutifs (CNN), largement utilisés pour des tâches telles que la reconnaissance d'images. Inspirés par le fonctionnement du cortex visuel, les CNN permettent la détection et la classification automatique d'objets dans des images avec une précision et une efficacité remarquables.

Ce chapitre explore les différentes approches et techniques utilisées dans la détection d'objets, des méthodes classiques comme Viola-Jones et HOG aux avancées récentes basées sur les CNN. Nous examinerons également l'importance de ces techniques dans divers secteurs tels que la sécurité, l'automobile, la médecine et l'industrie, où elles jouent un rôle crucial dans l'amélioration de la sécurité, de l'efficacité opérationnelle et de la qualité des produits.

Enfin, nous aborderons les défis actuels et les opportunités futures de la détection d'objets, en mettant en lumière les implications de ces technologies pour notre quotidien et pour l'évolution des industries. En combinant théorie et applications pratiques, ce chapitre vise à fournir une vue d'ensemble complète et actualisée de l'état de l'art dans ce domaine dynamique, en mettant en évidence les concepts fondamentaux et les avancées récentes dans le domaine du Deep Learning et des CNN.

# 2. Vision par ordinateur

## 2.1. Définition

La Vision par Ordinateur est un champ d'étude multidisciplinaire au sein de l'intelligence artificielle qui se concentre sur le développement de systèmes capables de comprendre et d'interpréter visuellement le monde qui les entoure. En combinant des techniques avancées de traitement d'images, de reconnaissance de formes et d'apprentissage automatique, la Vision par Ordinateur permet aux machines de percevoir visuellement des données extraites d'images ou de vidéos. Cette capacité à "voir" leur donne la possibilité de reconnaître des objets, des motifs et des comportements, ce qui ouvre la voie à de nombreuses applications innovantes dans divers domaines. [1]

## 2.2. Rôle de la Vision par Ordinateur dans le Traitement des Images

La Vision par Ordinateur joue un rôle central dans le traitement des images en fournissant des outils et des techniques pour analyser, segmenter et extraire des informations à partir de données visuelles. Elle repose sur des algorithmes sophistiqués de traitement d'images et de reconnaissance de motifs pour détecter et interpréter les caractéristiques visuelles des images. Cette capacité est essentielle pour des applications telles que la détection d'objets, la reconnaissance faciale et la navigation autonome. [2]

## 2.3. Applications de la Vision par Ordinateur

La Vision par Ordinateur trouve des applications dans une multitude de domaines, de la sécurité et la surveillance à la médecine et l'industrie manufacturière. Elle est utilisée pour automatiser des tâches telles que la détection d'anomalies, le contrôle qualité et la reconnaissance d'objets. Les technologies de vision par ordinateur sont également utilisées dans des applications plus avancées telles que la réalité augmentée, la réalité virtuelle et la robotique, où elles permettent aux machines de percevoir et d'interagir avec leur environnement de manière autonome et intelligente. [3]

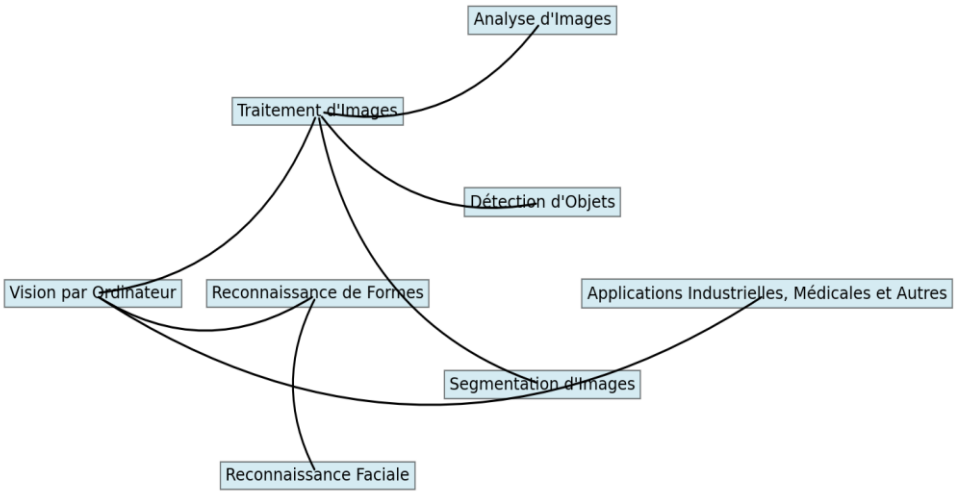



Figure 1 : Vision par Ordinateur.

# 3. Reconnaissance des Objets

## 3.1. Définition:

La reconnaissance des objets est une branche fondamentale de la vision par ordinateur qui vise à détecter, localiser et classifier automatiquement des objets dans des données visuelles telles que des images ou des vidéos. Elle repose sur l'utilisation d'algorithmes sophistiqués d'apprentissage automatique et de traitement d'images pour analyser les caractéristiques visuelles des objets et les comparer à des modèles prédéfinis, permettant ainsi aux machines de comprendre et d'interpréter leur environnement visuel de manière similaire à l'humain.[4]

## 3.2 L'importance de la Reconnaissance des Objets:

L'importance de la reconnaissance des objets réside dans son rôle crucial pour améliorer la perception et la compréhension des environnements visuels complexes. En permettant aux systèmes informatiques de détecter, de suivre et de reconnaître automatiquement des objets dans des scènes en temps réel, cette technologie offre un potentiel considérable pour renforcer la sécurité, améliorer l'efficacité opérationnelle et soutenir le développement de nouvelles applications innovantes dans des domaines tels que la surveillance, la médecine et la fabrication. [4]

## 3.3 Applications et Enjeux de la Reconnaissance des Objets

La reconnaissance des objets trouve une multitude d'applications dans divers domaines, de la surveillance à la santé en passant par l'automobile et la robotique. Cependant, elle présente

également des défis importants, notamment en termes de précision, de vitesse de traitement et de confidentialité des données. Voici quelques-unes de ses applications :

**Surveillance :** La reconnaissance des objets est largement utilisée dans les systèmes de surveillance pour détecter et suivre les mouvements suspects, assurant ainsi la sécurité des zones surveillées.

**Médecine :** Elle trouve des applications dans le domaine médical pour la détection précoce de maladies à partir d'images médicales, contribuant ainsi au diagnostic et au traitement efficace des patients.

**Automobile :** Dans le domaine de l'automobile, la reconnaissance des objets est utilisée dans les systèmes d'assistance à la conduite pour détecter les obstacles, les piétons et les panneaux de signalisation, améliorant ainsi la sécurité routière.

**Robotique :** Elle est essentielle pour permettre aux robots de percevoir leur environnement et d'interagir de manière autonome avec les objets qui les entourent, ouvrant la voie à des applications telles que la livraison automatisée et l'assemblage de précision.

**Industrie :** La reconnaissance des objets est utilisée dans le secteur industriel pour l'inspection de produits, le contrôle de qualité et la maintenance préventive, contribuant ainsi à améliorer l'efficacité et la fiabilité des processus de fabrication. [2]

## 3.4 Évolution de la Reconnaissance des Objets:

L'évolution de la reconnaissance des objets, notamment grâce aux réseaux de neurones convolutifs (CNN), constitue une étape majeure dans le domaine de la vision par ordinateur. Ces avancées ont révolutionné la manière dont les systèmes informatiques détectent et reconnaissent les objets dans les images, ouvrant la voie à des applications innovantes dans de nombreux domaines.

Avant l'avènement des CNN, les approches traditionnelles étaient limitées par leur capacité à capturer la complexité des données visuelles réelles, telles que les variations d'éclairage, de pose et de fond. Les CNN ont surmonté ces défis en apprenant de manière autonome des représentations hiérarchiques des données, à partir de couches de convolutions successives.

Cette évolution a non seulement amélioré la précision et la robustesse des systèmes de reconnaissance des objets, mais elle a également ouvert de nouvelles perspectives dans des domaines tels que la médecine, la surveillance, l'automobile et l'industrie. Les CNN sont désormais largement utilisés pour la détection précoce de maladies, la sécurité routière, la détection d'activités suspectes et bien d'autres applications. [5]

# 4. Traitement des Images

## 4.1. Définition

Le traitement des images constitue un ensemble de méthodes et de procédures visant à analyser et à manipuler des données visuelles numériques afin d'en extraire des informations significatives Cette discipline englobe diverses opérations, telles que la restauration, la segmentation, la classification et la reconnaissance d'objets, qui sont cruciales pour interpréter et exploiter les images dans un contexte informatique. [6]

## 4.2. Rôle du Traitement des Images dans la Détection d'Objets :

Le traitement des images joue un rôle essentiel dans la préparation des données pour la détection d'objets en fournissant les bases nécessaires à l'analyse et à l'interprétation des images. Cette phase comprend diverses étapes telles que la normalisation, la transformation et l'extraction de caractéristiques, qui permettent aux algorithmes de détection d'objets de fonctionner de manière précise et efficace. [3]

## 4.3. Techniques de Traitement des Images pour la Détection d'Objets :

Le traitement des images pour la détection d'objets implique une diversité de méthodes visant à préparer les données visuelles pour les algorithmes de détection. Parmi ces techniques, on retrouve la réduction du bruit pour améliorer la qualité des images, la segmentation pour isoler les objets d'intérêt, l'extraction de caractéristiques pour identifier des motifs distinctifs, et la mise en correspondance des attributs pour associer ces caractéristiques à des catégories spécifiques d'objets. Cette préparation minutieuse des données est cruciale pour garantir la précision et la fiabilité des systèmes de détection d'objets. [7]

## 4.4. Importance du Traitement des Images dans la Détection d'Objets :

Un traitement efficace des images revêt une importance cruciale dans le domaine de la détection d'objets, car il permet d'optimiser les données visuelles en amont, ce qui se traduit par une amélioration significative des performances de détection. En minimisant les distorsions et en mettant en évidence les caractéristiques pertinentes, le traitement des images contribue à réduire les erreurs de classification et à accroître la précision des systèmes de détection, garantissant ainsi des résultats fiables et exploitable. [8]

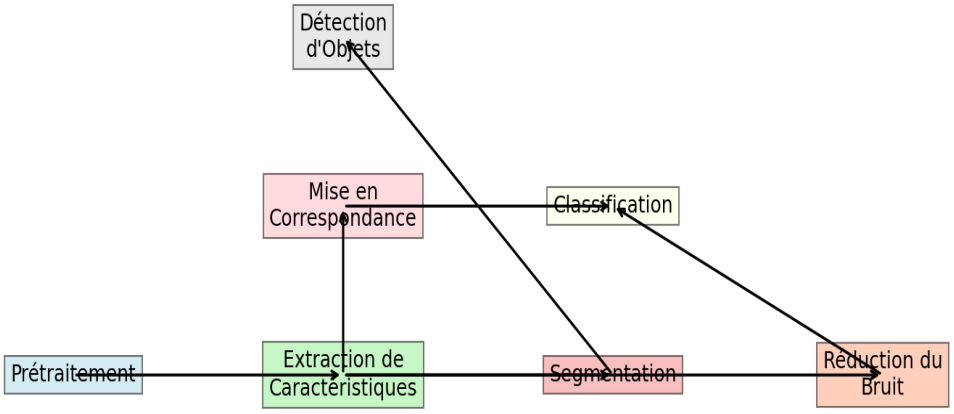



Figure 2 Schéma du Traitement des Images

# 5. Exploration de la Détection d'Objets

## 5.1. Pertinence et Signification Actuelles:

Nous allons explorer la définition du concept de détection d'objets ainsi que son importance dans le paysage technologique contemporain.

### 5.1.1 Définition du Concept :

La détection d'objets représente une capacité essentielle des systèmes informatiques, leur permettant d'identifier et de localiser des entités spécifiques, comme des personnes, des véhicules ou des objets variés, à partir d'images ou de séquences vidéo. Cette fonctionnalité, cruciale en vision par ordinateur, permet aux machines de percevoir et de comprendre leur environnement visuel de manière analogue à celle des êtres humains. Grâce à l'utilisation d'algorithmes sophistiqués d'apprentissage automatique et de traitement d'images, la détection d'objets permet aux ordinateurs d'analyser les caractéristiques visuelles des éléments présents dans une scène, ouvrant ainsi la voie à une multitude d'applications novatrices dans divers domaines. [9]

### 5.1.2 Importance dans le Paysage Technologique :

Dans le paysage technologique contemporain, la détection d'objets joue un rôle crucial en raison de l'énorme flux de données visuelles issues des appareils numériques. Cette fonctionnalité transcende divers secteurs, de la sécurité à l'automobile, en passant par le divertissement, et transforme notre interaction avec la technologie au quotidien.

Son importance se manifeste à travers une multitude d'applications de pointe telles que la reconnaissance faciale, la surveillance vidéo, la navigation autonome, la réalité augmentée et la robotique. Ces avancées sont rendues possibles grâce à des progrès significatifs dans les algorithmes d'apprentissage automatique et de traitement d'images, qui permettent aux machines de percevoir et de réagir à leur environnement visuel de manière similaire à celle des humains. [10]

## 5.2 Impacts sur la Vie Quotidienne et les Industries

Nous explorons l'intégration généralisée de la détection d'objets dans notre quotidien ainsi que ses applications cruciales dans le secteur industriel.

### 5.2.1 Intégration dans notre Quotidien:

Dans notre vie quotidienne, la détection d'objets a massivement influencé notre interaction avec la technologie, offrant des fonctionnalités avancées et des expériences utilisateur améliorées. Les smartphones intègrent désormais des capacités de reconnaissance d'images, permettant par exemple d'identifier automatiquement les visages dans les photos ou de scanner des codes QR. De même, les systèmes de surveillance domestique exploitent la détection d'objets pour détecter les mouvements suspects et assurer la sécurité des foyers. En outre, dans le domaine des jeux vidéo interactifs, cette technologie permet des fonctionnalités de suivi de mouvement et de reconnaissance d'objets, enrichissant ainsi l'expérience ludique des utilisateurs.

Cette omniprésence de la détection d'objets dans notre vie quotidienne découle des avancées significatives dans le domaine de la vision par ordinateur. Les travaux de recherche tels que ceux présentés par Redmon et ses collègues (2016) ont permis le développement de méthodes de détection d'objets en temps réel, ouvrant la voie à une intégration fluide de cette technologie dans une variété d'applications grand public.[10]

### 5.2.2 Applications Industrielles :

Dans le domaine industriel, la détection d'objets joue un rôle tout aussi crucial. Elle est utilisée pour optimiser les opérations logistiques en permettant le suivi précis des colis et la gestion efficace des stocks. De plus, dans le secteur manufacturier, elle contribue à la détection de défauts sur les produits et à la mise en œuvre de processus de contrôle qualité automatisés, améliorant ainsi l'efficacité des chaînes de production. Les avancées dans ce domaine reposent sur des travaux de recherche tels que ceux présentés par Liu et al. (2016), qui ont introduit des méthodes de détection d'objets plus rapides et plus précises, adaptées aux besoins industriels spécifiques. [11]

# 6. Processus de Détection d'Objets

## 6.1 Approches Classiques :

Dans cette partie consacrée aux approches classiques de détection d'objets, nous examinons l'utilisation de méthodes telles que Viola-Jones, HOG et DPM pour repérer les caractéristiques distinctives des objets. Ensuite, nous explorons l'utilisation de SIFT et SURF pour décrire et apparier ces caractéristiques aux modèles préexistants. Enfin, nous abordons l'utilisation de SVM pour classer les objets détectés.

### 6.1.1 Détection de Caractéristiques:

La détection de caractéristiques dans une image vise à identifier des éléments distinctifs pour localiser des objets. Les approches de Viola-Jones, HOG et DPM se concentrent sur la détection de motifs ou de structures spécifiques associés à ces objets.

### 6. 1.1.1 Viola-Jones:

Dans cette partie, je définis la méthode de Viola-Jones en exposant son fonctionnement, ses avantages, ses limites ainsi que sa rapidité, son efficacité et son adaptabilité.

### a. Définition:

La méthode de Viola-Jones, développée en 2001 par Paul Viola et Michael Jones, est une approche classique et largement utilisée dans le domaine de la détection d'objets, notamment pour la détection de visages. [12]

### b. fonctionnement et Avantages

Cette méthode repose sur un processus en cascade de classificateurs, formés pour identifier des arrangements spécifiques de caractéristiques visuelles simples appelées "haar-like features". Ces caractéristiques sont calculées à partir de la distribution des valeurs de pixels dans une région de l'image. L'un des avantages majeurs de cette méthode est sa rapidité et son efficacité, grâce à son approche en cascade qui permet de réduire le nombre de régions d'intérêt à analyser dans une image.

de plus, la méthode est adaptable à la détection d'une variété d'objets en ajustant les caractéristiques utilisées dans le processus de détection. Cette adaptabilité en fait un outil polyvalent pour des applications de détection d'objets diverses, allant au-delà de la détection de visages. [12]

## c. Adaptabilité de la Méthode:

La méthode de Viola-Jones, initialement développée pour la détection de visages, offre une flexibilité d'adaptation à différents types d'objets. Cette adaptabilité réside dans la capacité à ajuster les caractéristiques utilisées pour la détection en fonction des propriétés visuelles distinctives des objets ciblés. Par exemple, pour détecter des véhicules, des animaux ou d'autres objets spécifiques, les caractéristiques peuvent être modifiées pour correspondre aux attributs visuels uniques de ces objets. Ainsi, la méthode de Viola-Jones peut être étendue à une gamme variée d'applications de détection d'objets. [12]

## d.Efficacité et Rapidité

La méthode de Viola-Jones se distingue par son efficacité et sa rapidité dans la détection d'objets, en particulier dans des scénarios nécessitant une détection en temps réel. Cette efficacité est attribuable à son approche en cascade de classificateurs, qui permet de réduire rapidement le nombre de régions d'intérêt à analyser dans une image. En éliminant rapidement les régions qui ne correspondent pas aux caractéristiques des objets recherchés, la méthode de Viola-Jones peut détecter efficacement les objets d'intérêt même dans des environnements complexes avec des contraintes de temps strictes. [12]

## 6 1.1.2.HOG (Histogram of Oriented Gradients)

La méthode HOG (Histogram of Oriented Gradients) est une approche populaire pour la détection d'objets dans les images. Elle repose sur le calcul d'histogrammes des gradients de luminosité dans différentes zones de l'image, permettant ainsi de capturer les contours et les structures des objets. Ces histogrammes sont ensuite utilisés pour former des modèles de détection d'objets, souvent des classificateurs SVM. L'avantage clé de la méthode HOG est sa capacité à fournir une représentation robuste des objets, même dans des conditions d'éclairage, de pose et de fond variées, ce qui en fait une technique efficace pour la détection d'objets dans diverses scènes.[13]

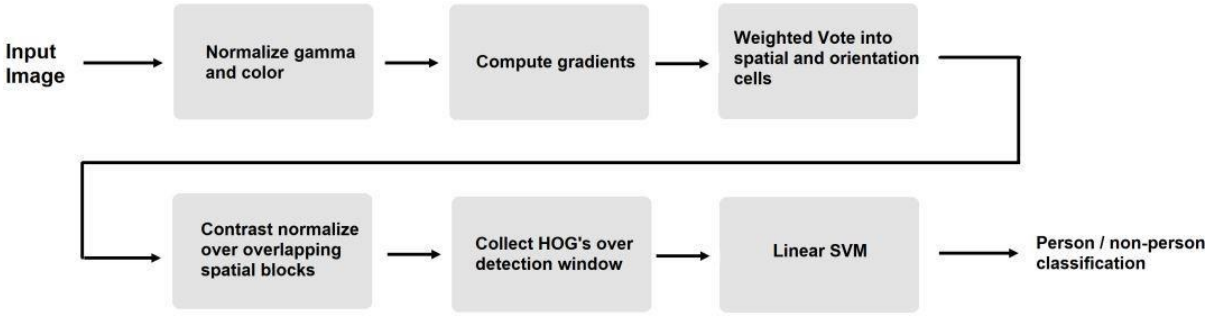



Figure 3 Structure de descripteur HOG

### 6. 1.1.3.DPM (Deformable Part Models)

Les Deformable Part Models (DPM) sont une approche avancée pour la détection d'objets, en particulier pour les objets articulés tels que les humains et les animaux. Cette méthode modélise les objets comme une combinaison de parties rigides et déformables, chacune associée à des caractéristiques spécifiques. Contrairement aux méthodes traditionnelles qui considèrent les objets comme des entités globales, les DPM permettent une représentation plus détaillée en prenant en compte la variabilité des poses et des apparences des objets. [14]

## 6.1.2. Description et Correspondance:

Cette partie de l'analyse se focalise sur l'extraction et l'appariement des caractéristiques des objets identifiés, mettant en avant des méthodes comme SIFT et SURF, qui jouent un rôle fondamental dans la détection d'objets.

### 6.1.2.1. SIFT (Scale-Invariant Feature Transform)

La méthode SIFT, développée par David G. Lowe en 1999, est largement utilisée pour extraire et décrire des caractéristiques robustes et invariantes à l'échelle dans des images. Cette méthode repose sur la détection de points d'intérêt, également appelés keypoints, qui sont des régions de l'image présentant des structures significatives et uniques, telles que des coins ou des bords. Les keypoints sont identifiés à différentes échelles de l'image en utilisant des filtres de type gaussien. [15]

Une fois les keypoints détectés, des descripteurs locaux sont extraits de chaque point d'intérêt. Ces descripteurs sont basés sur les gradients d'intensité des pixels environnants et sont conçus pour être robustes aux variations d'échelle, de rotation et d'illumination. Ils capturent des informations sur l'apparence locale de la région entourant chaque point d'intérêt. [15]

### 6.1.2.2. SURF (Speeded-Up Robust Features)

SURF est une méthode similaire à SIFT mais conçue pour être plus rapide et plus efficace, tout en conservant une précision comparable. Développée par Herbert Bay, Tinne Tuytelaars et Luc Van Gool en 2006, cette méthode utilise une approche basée sur les intégrales pour calculer rapidement les descripteurs locaux des points d'intérêt.

L'un des principaux avantages de SURF est sa capacité à calculer les descripteurs en utilisant des opérations basées sur les intégrales, ce qui réduit le temps de calcul nécessaire par rapport à SIFT. De plus, SURF est conçu pour être robuste aux transformations géométriques telles que les rotations et les changements d'échelle. [16]

### 6.1.3 Classification:

Dans cette partie, nous examinons la classification des objets détectés en nous appuyant sur les machines à vecteurs de support (SVM).

### a. Définition du SVM

Le SVM, ou machine à vecteurs de support, est un algorithme d'apprentissage supervisé qui vise à trouver l'hyperplan optimal pour séparer les données en différentes classes dans un espace de caractéristiques de grande dimension. L'objectif est de maximiser la marge, c'est-à-dire la distance entre les exemples les plus proches de différentes classes. [17]

### b.Fonctionnement dans la Détection d'Objets

Dans le contexte de la détection d'objets, le SVM est utilisé pour classer les régions de l'image en fonction des caractéristiques extraites, telles que les descripteurs HOG ou SURF. Il est entraîné sur un ensemble de données annotées où chaque région de l'image est associée à une étiquette indiquant la présence ou l'absence de l'objet d'intérêt. [17]

### c.Utilisation des SVM

Une fois entraîné, le SVM peut classifier de nouvelles régions de l'image en fonction de leurs caractéristiques extraites, contribuant ainsi à la détection d'objets en identifiant les zones de l'image susceptibles de contenir l'objet d'intérêt. Cette approche est largement utilisée dans de nombreux travaux de recherche sur la détection d'objets. [17]

### d.Avantage des SVM

Les SVM présentent plusieurs avantages dans le contexte de la détection d'objets. Leur capacité à traiter des ensembles de données de grande dimension avec un nombre élevé de caractéristiques est particulièrement utile. De plus, leur capacité à trouver l'hyperplan optimal perme/t une séparation précise des classes, conduisant à des performances de classification élevées. [17]

## 6.2. Deep Learning:

Le Deep Learning, également connu sous le nom d'apprentissage profond, peut être envisagé comme une composante du domaine plus vaste du Machine Learning. Il permet aux systèmes informatiques de modéliser et d'apprendre à partir de données complexes en utilisant des architectures de réseaux de neurones artificiels à plusieurs couches. Contrairement aux approches traditionnelles de Machine Learning, le Deep Learning vise à capturer des représentations de données hiérarchiques et abstraites.

L'émergence de cette discipline a été favorisée par les progrès dans le calcul puissant et l'abondance de données disponibles. En particulier, l'utilisation de GPU pour l'entraînement des réseaux de neurones profonds a permis d'atteindre des performances inégalées en termes d'efficacité. Cette approche réduit également la nécessité de concevoir manuellement des algorithmes pour résoudre des problèmes spécifiques, car les réseaux de neurones profonds sont capables d'apprendre des représentations de données de manière automatique et adaptative. [18]

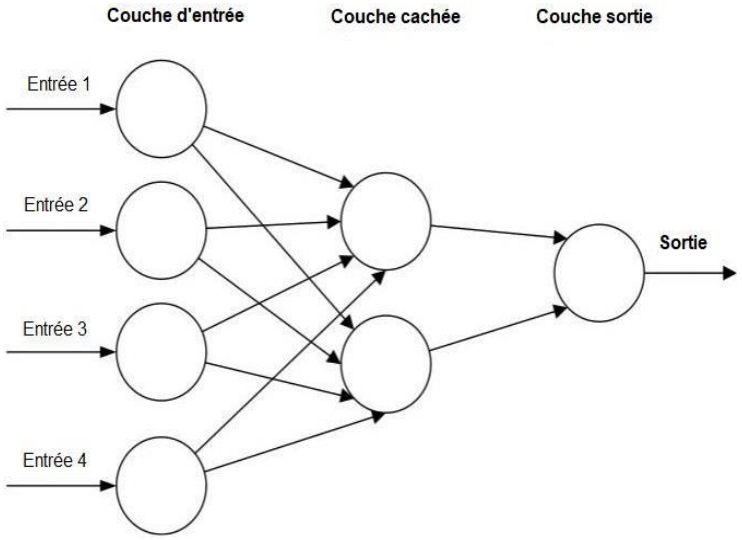



Figure 4 ANN composé d'une couche d'entrée, cachée et d'une couche de sortie. [18]

## 6.3. Réseaux de Neurones Convolutifs (CNN):

Les réseaux de neurones convolutifs (CNN) sont des outils cruciaux en apprentissage profond, principalement pour la reconnaissance d'images. Inspirés par le cortex visuel, ils détectent des motifs visuels en utilisant des champs récepteurs et une convolution. Composés de couches convolutionnelles, de mise en commun et entièrement connectées, les CNN réalisent des tâches complexes de détection et de classification. [19]

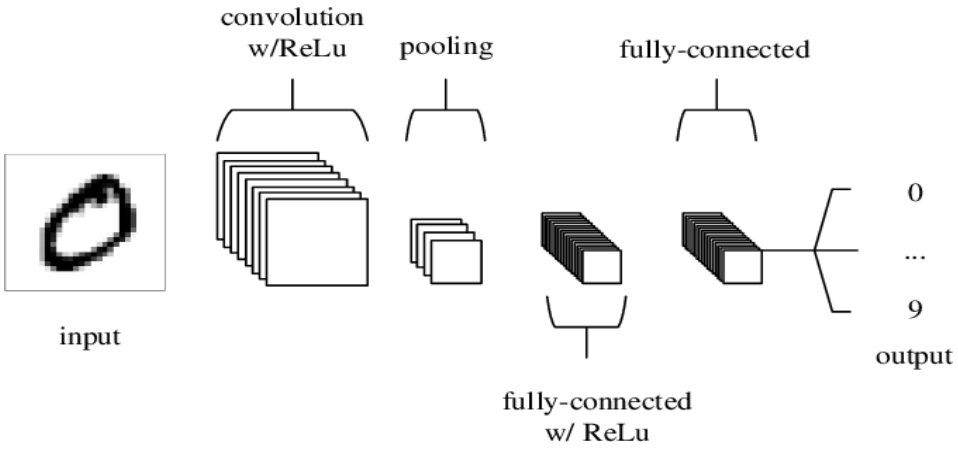



Figure 5 Une architecture CNN, composée de cinq couches. [18]

Leur histoire remonte aux années 1980, marquée par des débuts modestes et des limitations en termes de performances. Cependant, des modèles améliorés ont été développés, visant notamment à renforcer l'invariance aux distorsions. Malgré un intérêt initial limité, les avancées technologiques, notamment l'avènement des GPU puissants, ont relancé l'intérêt pour les CNN qui sont désormais indispensables dans le traitement d'images

Le contenu à venir abordera en détail la fonctionnalité des CNN, leurs architectures ainsi que leurs avantages dans la résolution de problèmes de la détection et la reconnaissance d'objets. [20]

### 6.3.1.Fonctionnement d'un CNN:

Les réseaux de neurones convolutifs (CNN) sont une classe de réseaux de neurones artificiels inspirés du fonctionnement du cortex visuel des animaux. Ils sont principalement utilisés pour la reconnaissance de formes dans des données visuelles telles que des images. Les CNN sont composés de différentes couches, chacune effectuant des opérations spécifiques sur les données en entrée pour extraire progressivement des caractéristiques de plus en plus abstraites.

#### 6.3.1.1. Couche Convolutionnelle (CONV) :

La couche convolutionnelle dans un CNN utilise des opérations de convolution pour extraire des caractéristiques des données d'entrée. Chaque filtre de convolution est appliqué à des régions locales de l'image d'entrée, produisant des cartes d'activation qui mettent en évidence les caractéristiques pertinentes. Mathématiquement, l'opération de convolution est définie comme suit :

$$[n] * [n] = \sum_{m=-\infty}^{+\infty} f[m]g[n-m] \quad (1)$$

Où $f$ est l'image d'entrée, $g$ est le filtre de convolution, et $*$ représente l'opérateur de convolution. Cette opération est appliquée à chaque canal d'entrée et pour chaque filtre dans la couche convolutionnelle, produisant ainsi des cartes d'activation pour chaque filtre. Les valeurs des filtres sont apprises par le réseau lors de l'entraînement afin de capturer des motifs discriminants. [19]

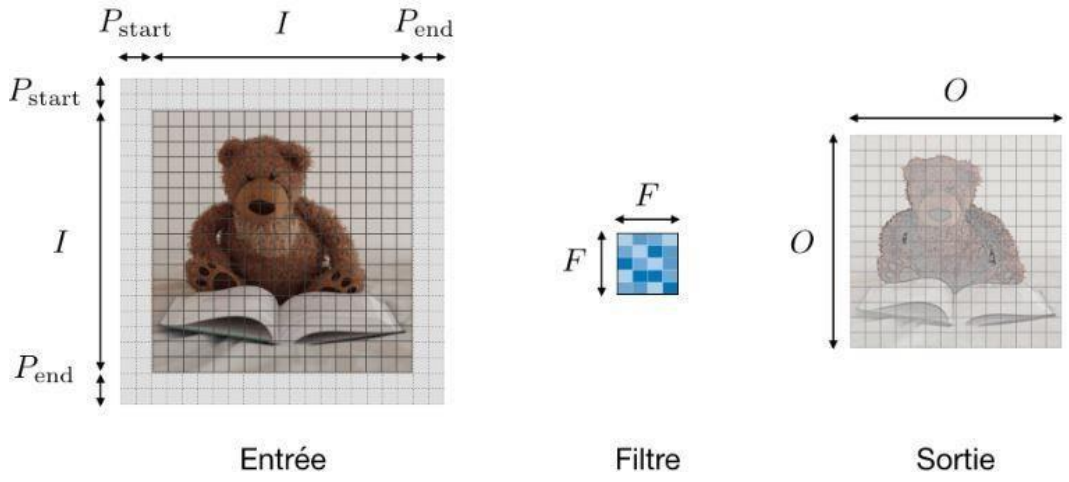



Figure 6 Feature map après filtre de convolution [21]

### 6.3.1.2 Couche de Pooling (POOL) :

La couche de pooling est utilisée pour réduire la dimensionnalité des cartes d'activation produites par les couches convolutionnelles. Elle agit en échantillonnant les activations les plus significatives à partir de chaque sous-région, ce qui permet de réduire la complexité du modèle tout en préservant les caractéristiques importantes des données. Les types de pooling les plus couramment utilisés sont le max pooling et l'average pooling. Mathématiquement, le max pooling prend la valeur maximale dans chaque région de sous-échantillonnage, tandis que l'average pooling prend la moyenne des valeurs dans chaque région. [22]

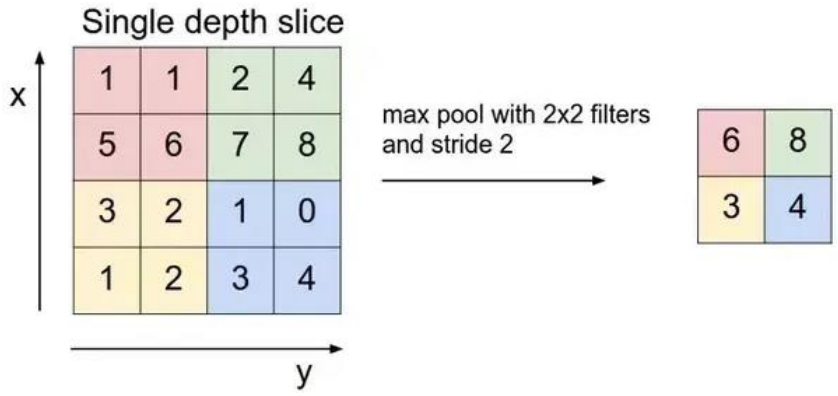



Figure 7 Max Pooling. [23]

### 6.3.1.3 La couche Fully Connected (FC)

La couche fully connected est une couche dense où chaque neurone est connecté à tous les neurones de la couche précédente. Elle agit comme un classificateur en prenant en compte les caractéristiques extraites par les couches précédentes pour effectuer la classification finale. Mathématiquement, la sortie de chaque neurone de la couche fully connected est calculée comme une combinaison linéaire des activations de la couche précédente, suivie d'une fonction d'activation non linéaire. [24]

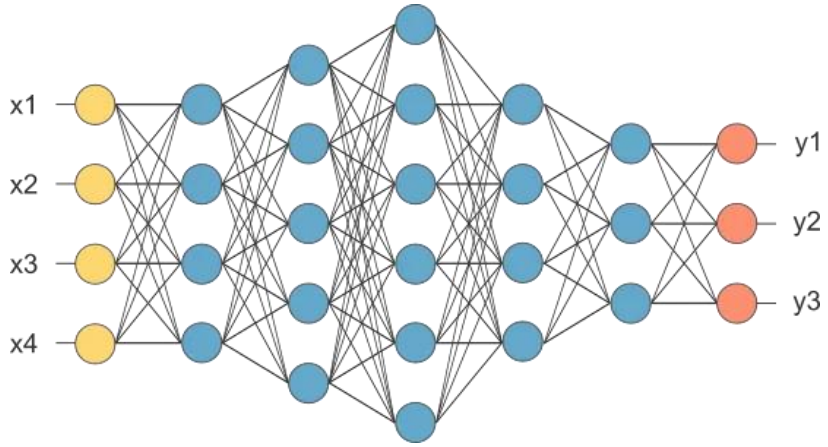



Figure 8 Après la couche pooling layer, aplatie en tant que couche FC.[23]

### 6.3.1.4 Fonctions d'activation :

Les fonctions d'activation sont des composants essentiels des réseaux de neurones, introduisant des non-linéarités dans le modèle et permettant ainsi au réseau de modéliser des relations complexes entre les données d'entrée et de sortie. Dans cette section, nous examinons plusieurs fonctions d'activation couramment utilisées dans les réseaux de neurones.

### a. Unité linéaire rectifiée (ReLU)

La fonction d'activation ReLU (Rectified Linear Unit) est l'une des fonctions les plus couramment utilisées dans les réseaux de neurones. Elle est définie mathématiquement comme $(x) = \max(0, x)$,ce qui signifie que pour toute valeur d'entrée $x$ la sortie de la fonction ReLU est égale à zéro s $x$ négatif et égale $x$ si $x$ est positif.[25]

### b. Unité linéaire rectifiée Leaky (Leaky ReLU)

La fonction d'activation Leaky ReLU est une variante de la fonction ReLU qui permet un petit gradient positif lorsque l'unité est inactive. Elle est définie mathématiquement comme

$(x)$ = max$(x, ax)$, Où $a$ est une petite pente pour les valeurs négatives. Cette propriété permet d'éviter le problème de disparition du gradient pour les valeurs négatives de $x$.[26]

## c .La fonction sigmoïde :

La fonction d'activation sigmoïde est couramment utilisée pour les tâches de classification binaire, où elle comprime les valeurs d'entrée dans l'intervalle [0, 1]. Elle est définie mathématiquement comme :

$$(2)\ (x) = \frac{1}{1 + e^{-x}}$$

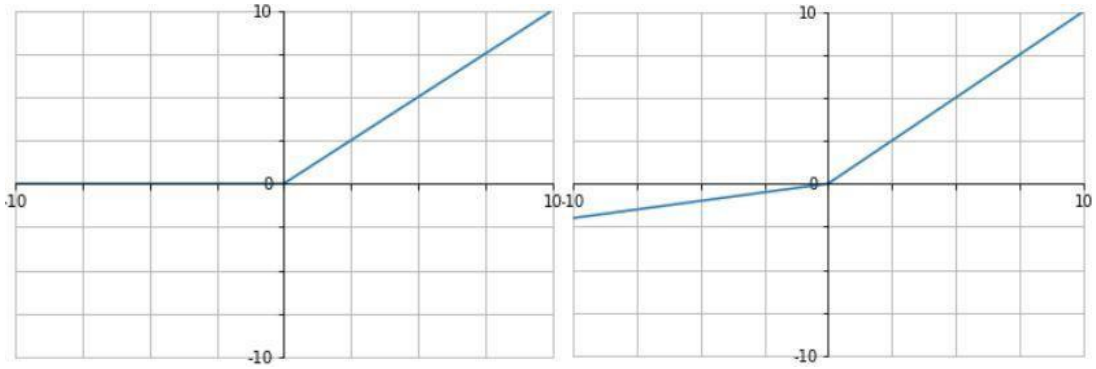
Où $e$ est la constante mathématique de l'exponentielle. La fonction sigmoïde produit une sortie proche de 1 pour les valeurs d'entrée positives et proche de 0 pour les valeurs d'entrée négatives.[20]

## d. La fonction tangente hyperbolique :

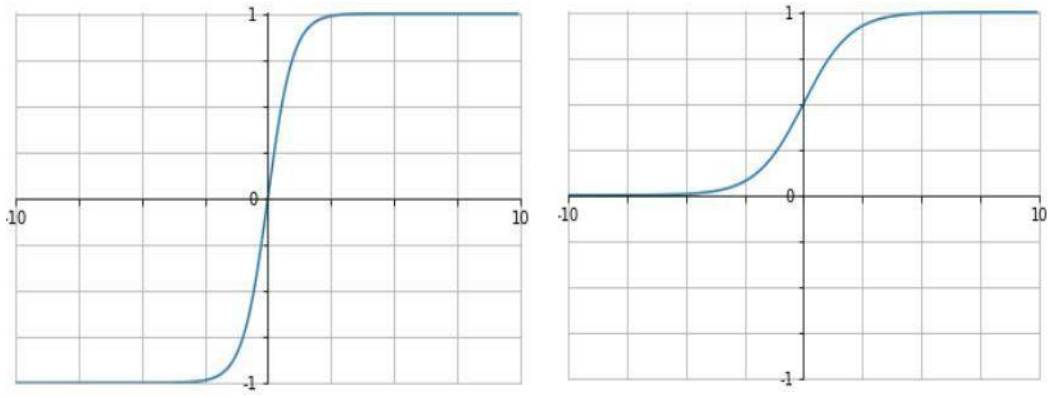
La fonction tangente hyperbolique (tanh) est une autre fonction d'activation couramment utilisée qui comprime les valeurs d'entrée dans l'intervalle [-1, 1]. Elle est définie mathématiquement comme :

$$(3)\ \tanh(x) = \frac{2}{1 + e^{-x}}$$

La fonction tanh produit une sortie proche de 1 pour les valeurs d'entrée élevées, proche de -1 pour les valeurs d'entrée très négatives, et proche de 0 pour les valeurs d'entrée proches de zéro. (27)




a)Fonction ReLU b) Fonction Leaky




c) Fonction Tanh d) Fonction

Figure 9 Différentes fonctions d'activation et leurs graphes

## 6.3.2Architecture des CNN:

Les architectures de Réseaux de Neurones Convolutifs (CNN) sont des structures complexes qui jouent un rôle crucial dans la détection et la classification des objets dans les images. Voici une explication détaillée de quelques-unes des architectures les plus influentes :

### 6.3.2.1 LeNet:

Conçu par Yann LeCun et ses collègues en 1998, LeNet était l'un des premiers réseaux de neurones convolutifs. Il se compose de deux couches de convolution suivies de couches de mise en commun et de couches entièrement connectées. LeNet a été utilisé pour la reconnaissance de caractères manuscrits.[27]

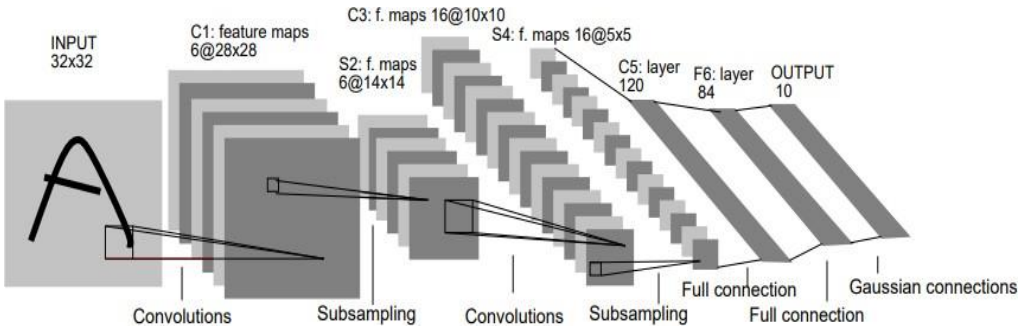



Figure 10 Architecture LeNet-5.[27]

### 6.3.2.2 AlexNet :

Développé par Alex Krizhevsky, Ilya Sutskever et Geoffrey Hinton en 2012, AlexNet a révolutionné le domaine de la vision par ordinateur en remportant le défi ImageNet avec une avance considérable. Il se compose de cinq couches de convolution suivies de couches de mise en commun et de trois couches entièrement connectées. AlexNet a été l'un des premiers CNN à utiliser des couches convolutives profondes et des techniques de régularisation telles que le dropout. [27]

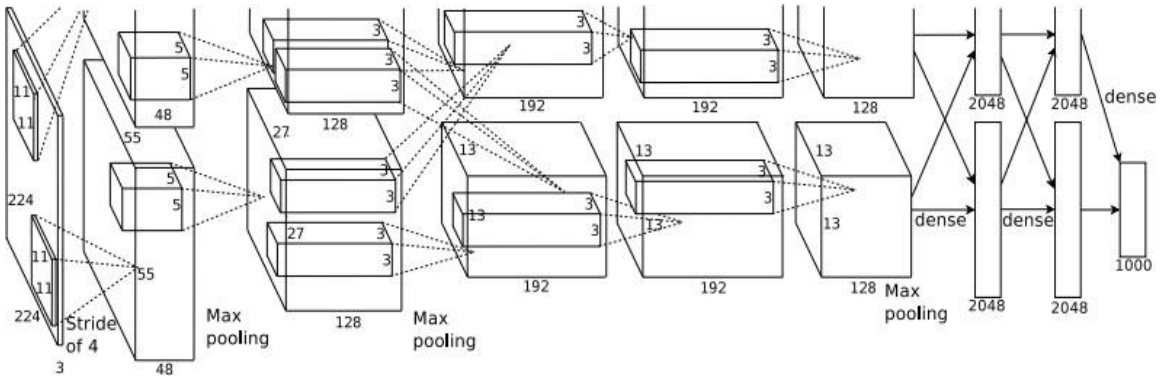



Figure 11 Architecture AlexNet.[25]

### 6.3.2.3 VGG :

Proposé par le Visual Geometry Group de l'Université d'Oxford, VGG se caractérise par sa profondeur. Il se compose de plusieurs couches convolutives de petite taille, suivies de couches de mise en commun, et se termine par des couches entièrement connectées. VGG a montré une grande capacité à apprendre des caractéristiques complexes, bien qu'il soit plus coûteux en termes de calcul en raison de sa profondeur.[28]

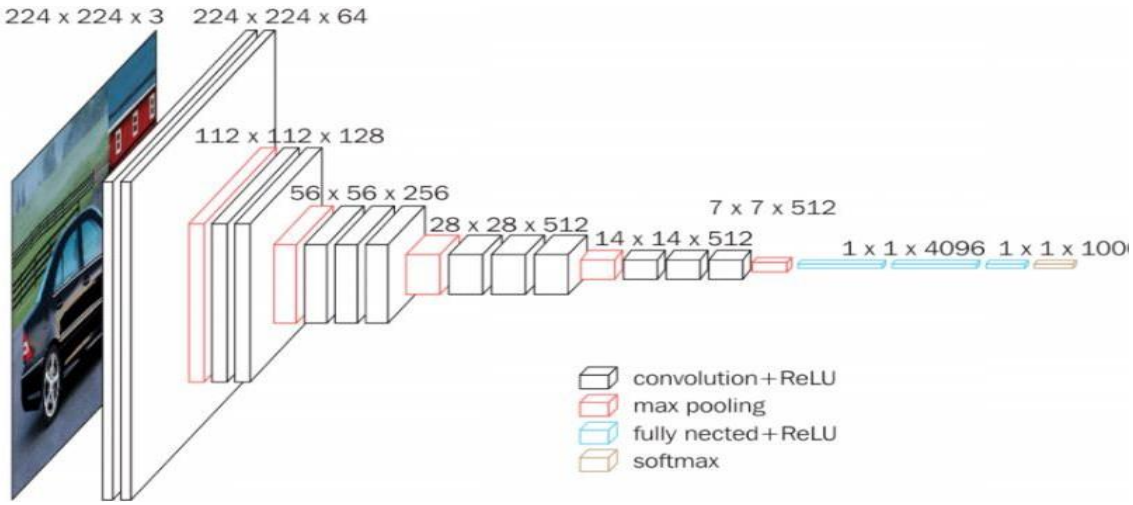



Figure 12 Architecture VGGNet.[29]

### 6.3.2.4.ResNet :

Introduit par Kaiming He et al. en 2015, ResNet a proposé une architecture novatrice avec des connexions résiduelles. Ces connexions permettent de sauter des couches dans le réseau, ce qui aide à résoudre le problème de la disparition du gradient lors de l'entraînement de réseaux profonds. ResNet a été particulièrement efficace pour former des réseaux très profonds et a obtenu des performances remarquables sur divers ensembles de données.

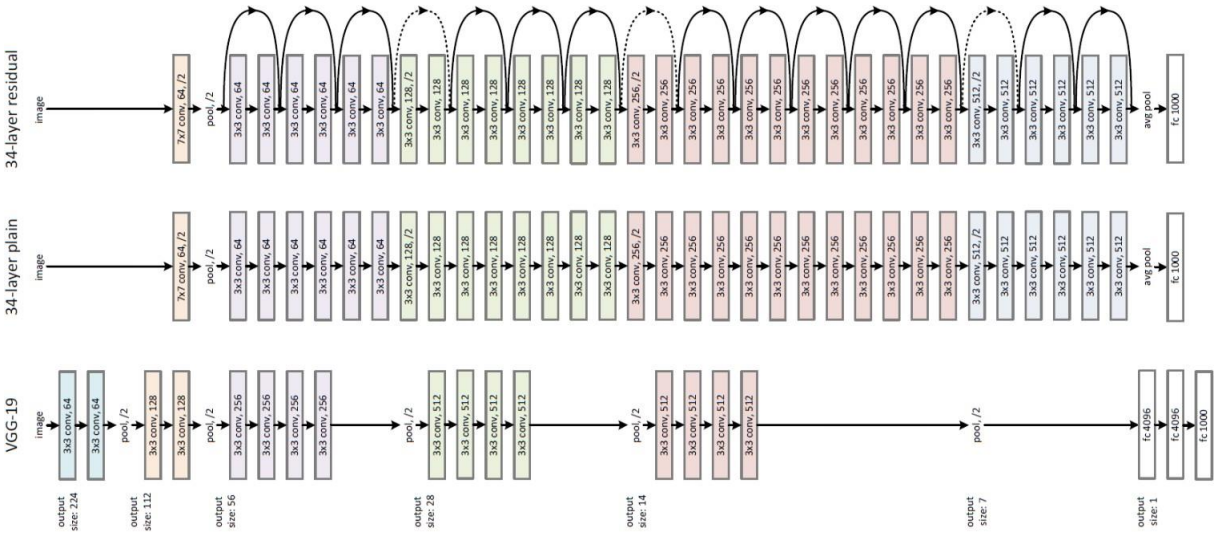



Figure 1 Architecture ResNet-50.[30]

### 6.3.2.5 Inception :

GoogLeNet, également connu sous le nom d'Inception, est célèbre pour son architecture complexe basée sur des modules inception. Ces modules sont des blocs de construction qui

combinent différentes opérations de convolution à différentes échelles spatiales, permettant ainsi de capturer efficacement les informations contextuelles dans les images. Cette architecture a été primée lors du défi ImageNet en 2014.

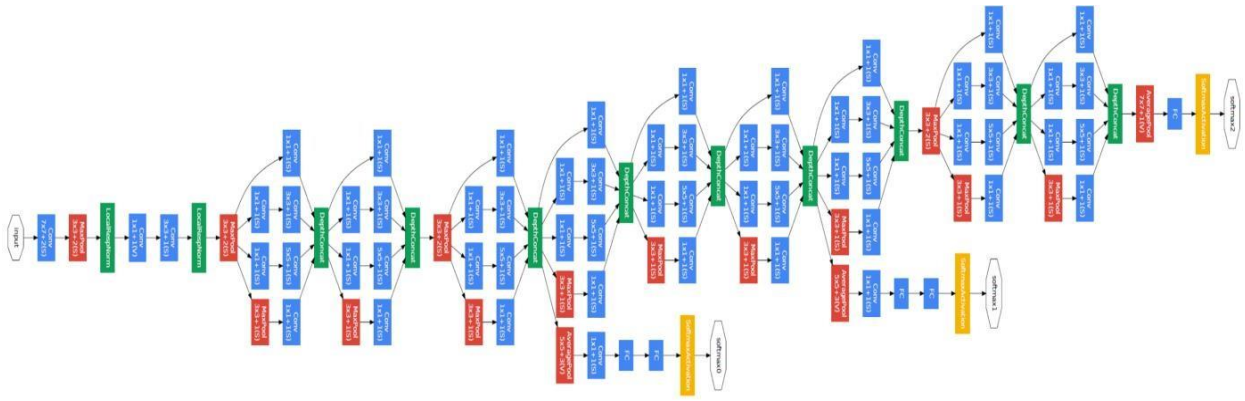


Figure 14 Architecture GoogLeNet/Inception-v1.[31]

Autre modèle proposé en 2015 « Inception-V2 » est un successeur d'Inception-V1, c'était le prototype d'Inception-V3, avec 24M paramètres.

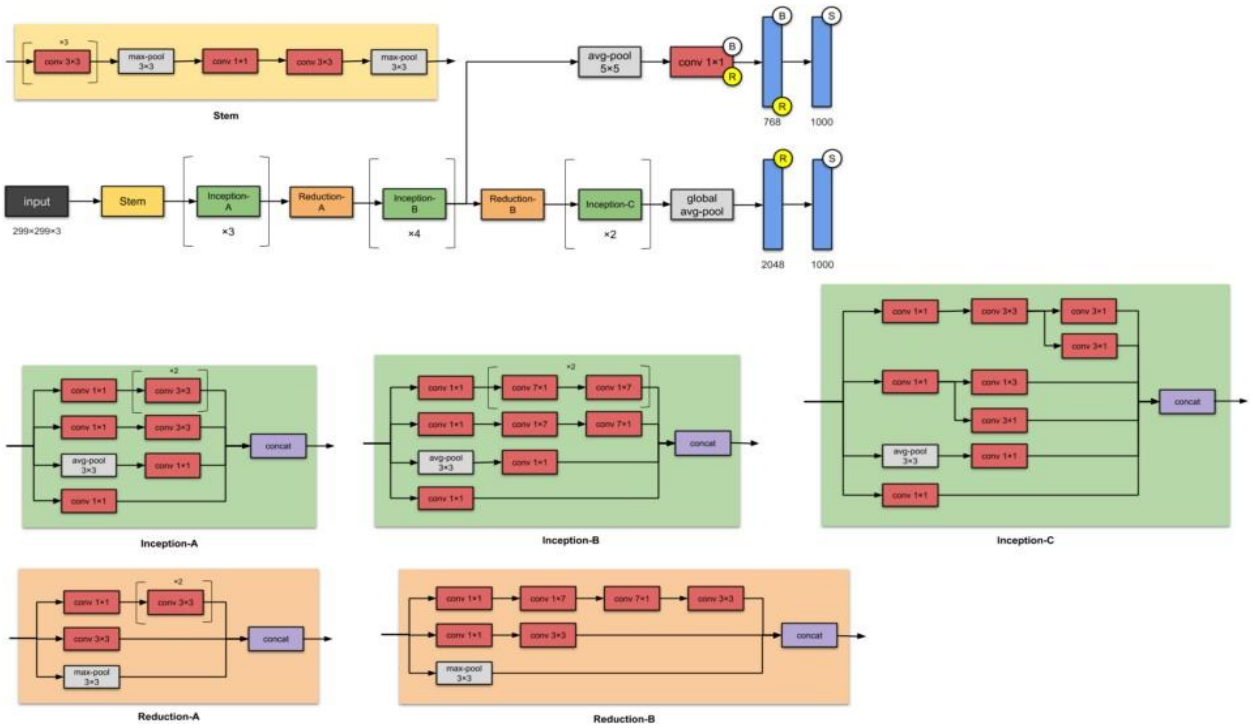



Figure 15 Architecture GoogLeNet/Inception-v2.[32]

### 6.3.2.6. Mobilnet-V2

Une architecture qui cherche à bien fonctionner sur les appareils mobiles. Il est basé sur une structure résiduelle inversée où les connexions résiduelles sont entre les couches de bottleneck. La couche d'expansion intermédiaire utilise des convolutions légères en profondeur pour filtrer les entités en tant que source de non-linéarité. Dans son ensemble, l'architecture de MobileNetV2 contient la couche initiale entièrement à convolution avec 32 filtres, suivie de 19 couches de bottleneck résiduelles.

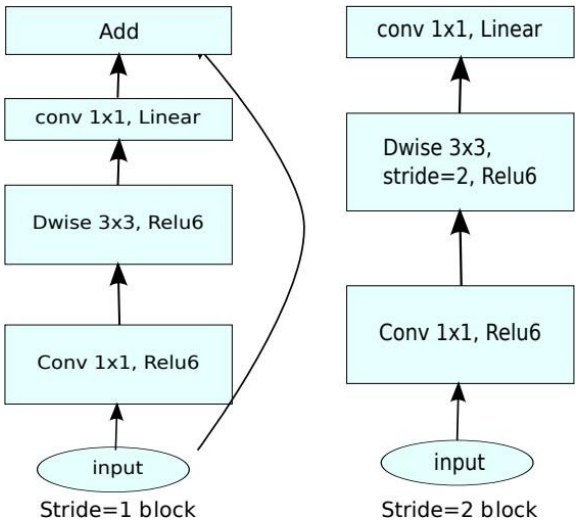



Figure 16 Architecture MobileNet-V2.[33]

## 6.3.3. Entraînement des CNN

Dans cette partie, nous plongeons dans le processus d'entraînement des CNN, crucial pour leur efficacité dans des tâches telles que la détection d'objets et d'autres applications de vision par ordinateur.

### 6.3.3.1. Rétropropagation

La rétropropagation, introduite par Rumelhart, Hinton et Williams en 1986, représente l'épine dorsale de l'entraînement des CNN. Cette technique consiste à calculer les gradients de la fonction de perte par rapport aux poids du réseau. En ajustant ces poids en fonction des gradients calculés, la rétropropagation permet de minimiser l'erreur de prédiction du modèle. Plus simplement, elle identifie les ajustements nécessaires aux poids du réseau pour améliorer progressivement sa capacité à produire des prédictions précises.[20]

### 6.3.3.2. Optimisation des Poids

L'optimisation des poids est cruciale pour régler les paramètres du CNN et réduire la fonction de perte. Des méthodes variées comme la descente de gradient stochastique (SGD) ou l'optimisation adaptative du taux d'apprentissage (Adam) sont employées pour converger vers les valeurs optimales des poids.[34]

### 6.3.3.3.Régularisation :

La régularisation vise à prévenir le surapprentissage, un phénomène où le modèle s'adapte trop aux données d'entraînement spécifiques et ne généralise pas bien à de nouvelles données. Différentes techniques sont utilisées, notamment l'arrêt anticipé, la régularisation L1 et L2, et le dropout. L'arrêt anticipé consiste à arrêter le processus d'entraînement dès que la performance du modèle sur un ensemble de validation commence à se détériorer. La régularisation L1 et L2 ajoutent des termes de pénalisation à la fonction de perte pour limiter les valeurs des poids du modèle.[35]

Le dropout consiste à aléatoirement "supprimer" certains neurones pendant l'entraînement, obligeant ainsi le modèle à apprendre de manière plus robuste. Cette technique contribue à prévenir le surapprentissage en empêchant le modèle de devenir trop dépendant de certains neurones.[24]

### 6.3.3.4.Fonction de Perte

La fonction de perte évalue l'écart entre les prédictions du modèle et les étiquettes réelles des données, guidant ainsi le processus d'apprentissage en quantifiant l'erreur de prédiction. L'erreur quadratique moyenne (**TMSE)** mesure la moyenne des carrés des différences entre les prédictions et les valeurs réelles, fournissant une indication de la précision du modèle. De même, l'entropie croisée **(CEL)** est une mesure de la divergence entre la distribution de probabilité prédite par le modèle et la distribution réelle des étiquettes, permettant d'évaluer la justesse des prédictions.[5]

### 6.3.3.5. Apprentissage par transfert

Le transfert d'apprentissage implique l'exploitation des connaissances préalablement acquises à partir de modèles pré-entraînés pour des tâches spécifiques de détection d'objets. Au lieu de construire un modèle à partir de zéro, cette approche réutilise les couches d'un réseau déjà entraîné sur de vastes ensembles de données tels qu'ImageNet. En ajustant ces couches pré-entraînées à un nouvel ensemble de données plus restreint, le modèle peut s'adapter rapidement à des tâches spécifiques avec moins de données d'entraînement.[36]

### 6.3.4. Avantages des CNN dans la détection d'objets

Dans cette partie, nous explorerons les avantages des réseaux de neurones convolutifs (CNN) dans la détection d'objets. Nous examinerons leur capacité à apprendre des caractéristiques hiérarchiques, à s'adapter à différentes tailles et formes d'objets, à maintenir des performances malgré des conditions d'éclairage complexes, à gérer efficacement les images haute résolution, et enfin, à permettre une interprétation visuelle des motifs détectés.

#### 6.3.4.1.Capacité à apprendre des représentations hiérarchiques

Les CNN sont capables d'apprendre des caractéristiques visuelles discriminantes à différents niveaux d'abstraction, ce qui est essentiel pour la détection d'objets. Cette capacité a été largement démontrée dans des travaux tels que ceux de Krizhevsky et al. (2012), où des couches de convolution profondes ont appris des représentations hiérarchiques à partir de données non étiquetées.[25]

#### 6.3.4.2.Adaptabilité à différentes tailles et formes d'objets

Les architectures CNN sont conçues pour être robustes aux variations de taille et de forme des objets, ce qui les rend très performantes dans la détection d'objets de différentes classes. Des recherches telles que celles de Girshick et al. (2014) ont exploré des architectures CNN capables de s'adapter efficacement à une grande variété de tailles et de formes d'objets grâce à des stratégies de fusion de caractéristiques multi-échelles. [37]

#### 6.3.4.3.Robustesse aux variations d'éclairage et de fond

Les CNN maintiennent des performances élevées même dans des conditions d'éclairage difficile et des arrière-plans complexes. Des recherches comme celles de Ren et al. (2015) ont proposé des architectures CNN spécifiquement conçues pour améliorer la robustesse aux variations d'éclairage et de fond en utilisant des techniques de régularisation et d'augmentation des données.[9]

#### 6.3.4.4.Traitement efficace des données à haute résolution

Les réseaux de neurones convolutifs modernes sont capables de traiter efficacement des images haute résolution tout en maintenant des performances de détection élevées. Les travaux de He et al. (2016) ont introduit des architectures CNN avec des mécanismes de réduction de la dimensionnalité et des opérations de convolution efficaces pour gérer efficacement les données à haute résolution. [30]

### 6.3.4.5.Interprétabilité des caractéristiques apprises

Les caractéristiques apprises par les CNN peuvent souvent être interprétées visuellement, permettant ainsi une compréhension plus profonde des motifs détectés dans les images. Des techniques de visualisation développées par Simonyan et al. (2013) ont permis de comprendre les activations neuronales dans les réseaux de neurones convolutifs et d'interpréter les caractéristiques apprises.[38]

## 6.3.5.Étapes de la Détection d'Objets

Cette partie détaille les étapes cruciales du processus de détection d'objets, depuis le prétraitement des données jusqu'au post-traitement pour améliorer la précision et la fiabilité des résultats.

### 6.3.5.1. Prétraitement des données d'image

Avant d'appliquer des algorithmes de détection d'objets, il est crucial de prétraiter les données d'image afin d'optimiser leur qualité et leur cohérence. Le prétraitement implique des étapes telles que la correction de l'illumination, l'égalisation de l'histogramme et la normalisation des couleurs. Ces techniques visent à améliorer la robustesse des algorithmes de détection en réduisant les variations indésirables dues aux conditions d'éclairage et aux différences de couleur dans les images.[39]

### 6.3.5.2. Extraction de Caractéristiques pertinentes

Une fois que les données ont été prétraitées, la prochaine étape consiste à extraire des caractéristiques pertinentes des images. Ces caractéristiques peuvent inclure des informations sur la texture, la forme, la couleur ou d'autres attributs visuels des objets présents dans l'image. Différentes techniques d'extraction de caractéristiques peuvent être utilisées en fonction des exigences spécifiques de la tâche de détection d'objets.[2]

### 6.3.5.3. Classification et Localisation des objets détectés

Après l'extraction des caractéristiques, les objets sont classifiés et localisés dans l'image. La classification consiste à attribuer une étiquette ou une classe à chaque objet détecté, tandis que la localisation vise à déterminer les coordonnées spatiales précises de chaque objet dans l'image. Cette étape peut être réalisée à l'aide de techniques telles que les réseaux de neurones convolutifs (CNN) pour la classification et la régression pour la localisation.[40]

### 6.3.5.4. Post-traitement pour améliorer la précision et la fiabilité

Enfin, une fois que les objets ont été détectés et localisés, des techniques de post-traitement sont souvent appliquées pour améliorer la précision et la fiabilité des résultats. Cela

peut inclure la suppression des faux positifs, la fusion des détections multiples et l'application de filtres pour éliminer les erreurs potentielles. Le post-traitement est essentiel pour garantir que les résultats de la détection d'objets sont fiables et précis.[41]

## 7. Travaux connexes

Nous avons exploré les travaux connexes pour comprendre les méthodes basées sur les réseaux de neurones convolutifs (CNN) dans le domaine de la détection d'objets. Ces méthodes, comme YOLO, Faster R-CNN, et SSD,…etc. ont été analysées pour leur rapidité et leur précision dans divers environnements. Ces approches ont été évaluées sur des bases de données comme COCO et PASCAL VOC, mettant en évidence leurs avantages et leurs limitations. Cette section met en lumière les contributions significatives et les défis rencontrés par ces techniques, fournissant un contexte essentiel pour le développement de notre propre solution de détection d'objets.

| **Auteur(s)** | **Titre et Année** | **Méthodologie/Techniques Utilisées** | **Bases de Données Utilisées** | **Résultats** | **Contributions de l'Équipe** | **Avantages /Inconvénients** | **Réf** |
|---|---|---|---|---|---|---|---|
| **Joseph Redmon, Santosh Divvala, et al.** | YOLO (You Only Look Once)-2016 | YOLO utilise un réseau de neurones convolutifs pour prédire les boîtes englobantes et les probabilités de classe en une seule passe à travers l'image. Il divise l'image en une grille et prédit les boîtes et les probabilités pour chaque cellule de cette grille. Cette méthode permet une détection rapide en temps réel. Elle adopte une approche de détection d'objets en une seule étape, ce qui la distingue des méthodes traditionnelles à deux étapes comme Faster R-CNN. | COCO, PASCAL VOC | YOLO a montré une précision de 69% et un rappel de 71% sur COCO et PASCAL VOC. Ces résultats indiquent une efficacité pour la détection d'objets dans une variété de scénarios. | L'équipe a développé YOLO en proposant une approche unifiée pour la détection d'objets, simplifiant ainsi le processus de détection. Ils ont travaillé sur l'optimisation de la vitesse et de la précision pour une application en temps réel. Ils ont également publié le | Vitesse élevée, simplicité de mise en œuvre, performances satisfaisantes pour différentes tailles d'objets. Évite les duplications de boîtes englobantes grâce à la suppression non-maximale. / Sensible à la détection d'objets de petite taille et d'objets proches. Performance moins | [10] |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | code source et les modèles pré-entraînés, facilitant ainsi la reproduction et l'utilisation de leur méthode par la communauté de recherche. | satisfaisante pour les objets de petite taille et dans des scènes den | |
| **Shaoqing Ren, Kaiming He, et al.** | Faster R-CNN-2015 | Faster R-CNN intègre la génération de propositions de régions dans le réseau neuronal, ce qui permet de générer et de raffiner les régions d'intérêt. Il utilise un réseau de propositions de régions (RPN) pour générer et raffiner les régions d'intérêt, puis un réseau de neurones de classification pour détecter les objets. Cette méthode permet une détection précise et rapide des objets, même dans des scènes complexes. | PASCAL VOC | Faster R-CNN a montré une précision de 73% et un rappel de 75% sur COCO, PASCAL VOC, et ImageNet. Ces résultats indiquent une capacité à détecter efficacement les objets dans une variété de scénarios, y compris les scènes complexes. | L'équipe a introduit une architecture innovante avec Faster R-CNN, combinant la génération de propositions de régions et la détection d'objets pour améliorer la précision et la rapidité. Ils ont également partagé leur implémentation de référence, ouvrant ainsi la voie à des recherches ultérieures et à des applications pratiques dans divers domaines. | Haute précision, adaptabilité à différents types de données et de scénarios, intégration de la génération de propositions de régions dans le réseau neuronal pour simplifier le processus de détection. / Complexité computationnelle élevée, nécessite plus de ressources en comparaison avec d'autres méthodes. La mise en œuvre et la formation peuvent être complexes. | [9] |
| **Ross Girshick** | R-CNN - 2013 | R-CNN utilise une méthode de régions d'intérêt (ROI) pour identifier les régions potentielles dans une image, puis extrait ces régions et les classe à l'aide d'un réseau de neurones convolutif (CNN). Cette méthode a | PASCAL VOC, ImageNet | R-CNN a montré une précision de 58.5% et un rappel de 31.4% sur PASCAL | L'équipe a présenté une méthode novatrice qui a établi les bases de la détection d'objets à l'aide de réseaux de | Utilisation efficace des réseaux de neurones convolutifs pour extraire des caractéristiques pertinentes à partir des | [37] |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | été la première à introduire l'utilisation de réseaux de neurones convolutifs pour la détection d'objets. | | VOC et ImageNet. Bien qu'efficace, cette méthode souffre de lenteur en raison de l'extraction et de la classification séparées des régions d'intérêt. | neurones convolutifs. Leur travail a ouvert la voie à de nombreuses recherches ultérieures dans le domaine de la vision par ordinateur et de la détection d'objets. | régions d'intérêt. / Temps de traitement lent en raison de l'extraction et de la classification séparées des régions d'intérêt. | |
| **Wei Liu, Dragomir Anguelov, et al.** | SSD (Single Shot Multi Box Detector)-2016 | SSD utilise un réseau de neurones convolutifs pour prédire les boîtes englobantes et les probabilités de classe directement à partir de différentes échelles de l'image. Il utilise une architecture en une seule étape, ce qui lui permet d'être rapide et efficace pour la détection d'objets. | COCO, PASCAL VOC, ImageNet | SSD a montré une précision de 74.3% et un rappel de 68.0% sur COCO, PASCAL VOC, et ImageNet. Cette méthode offre une bonne précision tout en étant rapide et efficace | L'équipe a développé une méthode novatrice avec SSD, qui a permis une détection efficace d'objets à différentes échelles dans une seule passe. Leur approche en une seule étape a simplifié le processus de détection tout en conservant une précision acceptable. | Performances élevées avec une architecture en une seule étape, ce qui permet une détection rapide des objets à différentes échelles. /<br><br>Moins précis que certaines méthodes à deux étapes comme Faster R-CNN. | [11] |
| **Christian Szegedy, Vincent Vanhoucke, et al.** | Inception-v3-2015 | Inception-v3 utilise une architecture de réseau neuronal convolutif profond avec des modules Inception pour extraire des caractéristiques à différentes échelles spatiales. Il utilise également des techniques telles que la régularisation L2 et le dropout pour améliorer la | ImageNet | Inception-v3 a montré une précision de 78.0% et un rappel de 82.5% sur ImageNet. Cette | L'équipe a développé une architecture de réseau neuronal profond efficace pour la classification d'images, qui peut | Utilisation efficace des modules Inception pour extraire des caractéristiques à différentes échelles spatiales. Techniques de régularisation pour améliorer | [42] |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | généralisation du modèle. Cette méthode est efficace pour la classification d'images et peut être adaptée à la détection d'objets en utilisant des techniques supplémentaires comme la région d'intérêt (ROI) pooling. | | méthode offre une bonne précision pour la classification d'images, ce qui en fait une base solide pour la détection d'objets | également être adaptée à la détection d'objets en utilisant des techniques telles que le ROI pooling. Leur travail a ouvert de nouvelles possibilités pour la détection d'objets en combinant des architectures de pointe avec des techniques de détection d'objets classiques | la généralisation du modèle. / Complexité computationnelle élevée en raison de l'utilisation d'une architecture profonde. | |
| **Liang-Chieh Chen, George Papandreou, et al** | DeepLab-2016 | DeepLab utilise une architecture de réseau neuronal convolutif profond avec des modules de dilatation pour effectuer la segmentation sémantique des images. Il utilise des convolutions dilatées pour augmenter le champ de vision sans sacrifier la résolution spatiale. Cette méthode permet une segmentation précise des objets dans les images, ce qui est utile pour la détection d'objets et la compréhension de scènes. | COCO, Cityscapes, ImageNet | DeepLab a montré une précision de 70.4% et un rappel de 69.8% sur COCO, Cityscapes, et ImageNet. Cette méthode offre une segmentation précise des objets dans les images, ce qui est essentiel pour la détection d'objets et la | l'équipe a développé une méthode innovante pour la segmentation sémantique des images, qui peut être appliquée à la détection d'objets en extrayant les régions d'intérêt des images segmentées. Leur travail a ouvert de nouvelles perspectives pour la compréhension des scènes et la détection d'objets dans des environneme | Utilisation efficace des convolutions dilatées pour augmenter le champ de vision sans sacrifier la résolution spatiale / Sensible aux variations d'éclairage et de perspective. | [43] |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | compréhension de scènes. | nts complexes. | | |
| **Karen Simonyan, Andrew Zisserman** | VGG Net-2014 | VGGNet utilise une architecture de réseau neuronal convolutif profond avec des couches convolutives profondes et des couches entièrement connectées. Il utilise des filtres de petite taille (3x3) avec un espacement de pas de 1 pour les convolutions, ce qui permet d'augmenter la non-linéarité du modèle et d'améliorer sa capacité de représentation. Cette méthode est efficace pour la classification d'images et peut être adaptée à la détection d'objets en utilisant des techniques supplémentaires comme la région d'intérêt (ROI) pooling. | ImageNet | VGGNet a montré une précision de 92.7% sur ImageNet. Cette méthode offre une haute précision pour la classification d'images, ce qui en fait une base solide pour la détection d'objets | L'équipe a développé une architecture de réseau neuronal profond efficace pour la classification d'images, qui peut également être adaptée à la détection d'objets en utilisant des techniques telles que le ROI pooling. Leur travail a ouvert de nouvelles possibilités pour la détection d'objets en combinant des architectures de pointe avec des techniques de détection d'objets classiques. | Utilisation d'architectures profondes avec des convolutions 3x3 pour augmenter la non-linéarité et la capacité de représentation. Techniques de régularisation pour améliorer la généralisation du modèle. / Complexité computationnelle élevée en raison de l'utilisation d'une architecture profonde. | [28] |

Table 1 Travaux connexes.

# 1. Conclusion

Ce chapitre a exploré en profondeur les avancées et les techniques actuelles dans le domaine de la détection d'objets, mettant en lumière l'importance croissante de cette technologie dans divers secteurs tels que la sécurité, l'automobile, la médecine et l'industrie. En permettant aux systèmes informatiques de percevoir, interpréter et réagir visuellement de manière

autonome, la détection d'objets ouvre la voie à une multitude d'applications pratiques et innovantes.

Le rôle central du Deep Learning, en particulier des Réseaux de Neurones Convolutifs (CNN), dans cette révolution ne peut être sous-estimé. Inspirés par le fonctionnement du cortex visuel, les CNN ont non seulement amélioré la précision et l'efficacité de la détection d'objets, mais ont également élargi les possibilités en termes de reconnaissance et de classification d'images complexes.

Les défis actuels et les opportunités futures dans le domaine de la détection d'objets soulignent l'importance continue de l'innovation et de la recherche dans ce domaine. Les progrès rapides dans les techniques de Deep Learning et l'évolution des architectures de réseaux neuronaux promettent des améliorations constantes en matière de performance et d'applicabilité dans divers contextes industriels et commerciaux.

En conclusion, ce chapitre offre une perspective complète sur l'état de l'art de la détection d'objets, fournissant des bases solides pour comprendre l'impact potentiellement transformateur de cette technologie sur notre quotidien et sur l'avenir des industries. Les recherches et les développements continus dans ce domaine continueront d'ouvrir de nouvelles avenues pour l'innovation et l'amélioration des systèmes autonomes et intelligents.

# Chapitre 2: Conception

## 1. Introduction

Ce chapitre offre une vue d'ensemble des insuffisances des solutions actuelles de détection d'objets et présente les objectifs de notre projet pour y remédier. Les technologies actuelles sont souvent limitées par des performances insuffisantes dans des environnements complexes, une scalabilité inadéquate, et un manque de flexibilité pour répondre aux besoins spécifiques des utilisateurs. Ces défis entraînent des taux d'erreur élevés et une fiabilité réduite, restreignant ainsi leur utilisation pratique.

Pour surmonter ces limitations, notre projet vise à développer une architecture optimisée et robuste pour la détection d'objets, en intégrant des techniques avancées de traitement des données et d'entraînement des modèles. Nos objectifs incluent l'analyse des architectures

existantes, la conception d'une nouvelle architecture, et le développement d'un modèle CNN personnalisé pour maximiser la précision et la robustesse.

Notre méthodologie commence par la collecte et le prétraitement rigoureux des données, en utilisant MobileNetV2 pour l'extraction des caractéristiques et en intégrant un modèle CNN adapté aux dispositifs à ressources limitées. L'optimisation des ressources matérielles, comme les GPU et les TPU, est essentielle pour assurer des temps de réponse rapides en inférence temps réel.

Le modèle est évalué sur divers ensembles de données pour garantir une précision et une capacité de généralisation optimales. Nous détaillons également l'utilisation des ensembles de données PASCAL VOC pour l'entraînement et l'évaluation, la structure de notre modèle, les différentes couches et paramètres, les fonctions de perte, et l'optimiseur Adam pour optimiser les performances globales.

Ce chapitre présente ainsi une approche méthodologique complète pour développer un système de détection d'objets performant, fiable et adaptable à divers contextes et conditions d'utilisation

## 2. Analyse des insuffisances des solutions existantes et objectifs spécifiques

### 2.1. Analyse des insuffisances des solutions existantes:

Les technologies actuelles de détection d'objets présentent plusieurs limitations significatives qui justifient notre projet :

- **Performance limitée** : Les systèmes actuels peuvent rencontrer des difficultés à identifier correctement des objets dans des environnements complexes, tels que des conditions de faible luminosité ou des scènes encombrées. Cela entraîne des taux d'erreur élevés et une fiabilité réduite, limitant leur applicabilité dans des situations réelles. [40]
- **Scalabilité :** Certains systèmes ne sont pas suffisamment évolutifs pour gérer efficacement des volumes de données croissants ou répondre aux exigences de traitement en temps réel. Les architectures traditionnelles peuvent souffrir de limitations de performance lorsqu'elles sont étendues pour traiter de grandes quantités de données, affectant la latence et la réactivité du système. [40]
- **Flexibilité :** Les solutions actuelles manquent souvent de flexibilité pour s'adapter aux besoins spécifiques des utilisateurs ou pour s'intégrer facilement à d'autres systèmes ou

infrastructures. La personnalisation des modèles existants peut nécessiter des modifications complexes et coûteuses du code source, ce qui ralentit le déploiement et l'adaptation aux nouveaux cas d'utilisation.[40]

### 2.2. Objectifs spécifiques :

Pour atteindre notre objectif principal de développer un système de détection d'objets robuste, nous avons défini les objectifs spécifiques suivants :

- **Analyser en profondeur les architectures existantes** identification des points faibles et des opportunités d'amélioration significative spécifiques à la détection d'objets.
- **Conception et implémentation d'une architecture adaptée pour la gestion optimale des flux de données de détection d'objets :** Intégration d'éléments essentiels pour assurer la fiabilité opérationnelle du système.
- **Développement et entraînement d'un modèle CNN personnalisé** : Utilisation de techniques avancées telles que le transfert d'apprentissage et l'optimisation des hyperparamètres pour maximiser la précision et la robustesse de la détection.
- **Évaluation rigoureuse de la performance du système :** Tests approfondis utilisant des scénarios réalistes pour mesurer la précision, la latence et la scalabilité de la détection d'objets
- **Itération et optimisation continue** : Réajustement du modèle et de l'infrastructure en fonction des résultats des tests et des retours d'expérience, garantissant des améliorations continues et une adaptation aux besoins évolutifs.

## 3. Approche Proposée

Dans cette étude, nous avons développé une approche méthodologique exhaustive pour la détection d'objets, en mettant l'accent sur l'efficacité, la précision et l'adaptabilité. Notre méthodologie débute par une collecte rigoureuse de données diversifiées et représentatives, incluant une variété de scènes avec des conditions d'éclairage et d'arrière-plan variables. Ces données sont soigneusement prétraitées pour normaliser les images et annoter les objets avec précision, établissant ainsi une base d'entraînement solide.

Pour extraire efficacement les caractéristiques des objets, nous utilisons MobileNetV2 comme base en raison de sa légèreté et de ses performances optimales pour les dispositifs à ressources limitées. Un modèle CNN personnalisé est intégré pour améliorer la précision et la résilience de la détection, spécifiquement adapté aux caractéristiques des objets dans nos applications cibles.

L'optimisation des ressources matérielles, notamment les GPU et les TPU, joue un rôle crucial pour maximiser les performances du système. Ces accélérateurs matériels accélèrent

l'entraînement du modèle et assurent des temps de réponse rapides lors de l'inférence en temps réel, répondant ainsi aux exigences de réactivité dans divers environnements opérationnels.

Enfin, le modèle de détection d'objets est évalué pour analyser et affiner les résultats de détection, optimisant ainsi la précision finale du système en utilisant des mesures telles que la perte de validation et la précision. Cette méthodologie intégrée et optimisée assure une conception robuste et efficace du système de détection d'objets, exploitant les technologies avancées pour répondre aux besoins spécifiques des applications réelles.

## 4. Architecture Proposée

Notre architecture pour le système de détection d'objets est conçue pour maximiser l'efficacité, la précision et l'adaptabilité à travers plusieurs composants clés. Tout d'abord, nous utilisons MobileNetV2 comme base principale pour l'extraction initiale des caractéristiques des objets. Cette architecture est choisie pour sa légèreté et ses performances optimales sur les dispositifs à ressources limitées, facilitant ainsi une détection efficace en temps réel.

En complément, un modèle CNN personnalisé est intégré pour affiner et enrichir les caractéristiques extraites par MobileNetV2, améliorant ainsi la précision et la résilience de la détection des objets dans divers scénarios d'utilisation.

L'entraînement du modèle est accéléré grâce à l'utilisation stratégique de GPU et de TPU, optimisant les ressources matérielles pour une performance maximale lors de l'entraînement et de l'inférence en temps réel. Cette optimisation garantit des temps de réponse rapides et efficaces, répondant aux exigences de réactivité dans des environnements opérationnels variés.

Pour renforcer la capacité du système à généraliser, nous appliquons des techniques avancées de diversification des données. Celles-ci enrichissent l'ensemble de données en introduisant une diversité de scénarios réels, ce qui réduit les biais potentiels et améliore la robustesse du modèle face à des conditions variables.

Après l'entraînement, le modèle est évalué de manière approfondie sur des ensembles de données distincts pour valider sa performance et sa capacité à généraliser efficacement.

Enfin, le modèle déployé pour l'inférence en temps réel bénéficie d'une optimisation matérielle continue, garantissant des performances stables et fiables dans les applications réelles. Cette architecture intégrée et optimisée reflète notre approche méthodologique complète, exploitant les avancées technologiques pour répondre efficacement aux besoins spécifiques des scénarios d'utilisation réels.

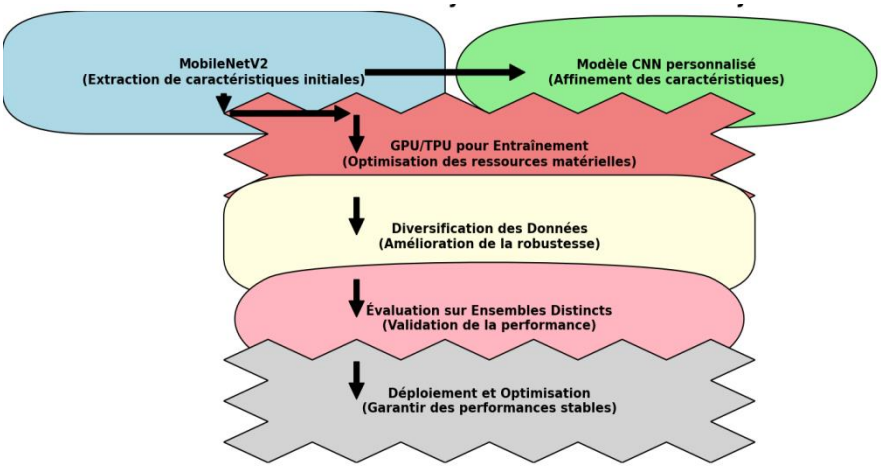



Figure 17 Architecture Proposée

## 5. Fonctionnement général du système

Le système démarre par la collecte de données à partir du dataset Pascal VOC, où les images annotées avec différents types d'objets sont prétraitées pour normaliser les images et annoter précisément les objets, établissant ainsi une base d'entraînement cohérente et de haute qualité stockée au format TFRecord.

Ensuite, le modèle est entraîné en utilisant MobileNetV2 comme base principale pour extraire les caractéristiques initiales des objets en raison de sa légèreté et de ses performances optimales sur des dispositifs à ressources limitées. Un modèle CNN personnalisé complémente MobileNetV2 pour affiner et enrichir les caractéristiques extraites, améliorant ainsi la précision et la résilience de la détection dans divers scénarios d'utilisation.

L'entraînement du modèle est accéléré grâce à une stratégie efficace d'utilisation des GPU et des TPU, optimisant les ressources matérielles pour garantir des temps de réponse rapides et efficaces lors de l'entraînement et de l'inférence en temps réel. Cette optimisation matérielle est cruciale pour répondre aux exigences de réactivité dans des environnements opérationnels variés.

Une fois entraîné, le modèle de détection d'objets est évalué sur des ensembles de données distincts pour mesurer sa performance et sa capacité à généraliser efficacement. Cela se fait principalement à travers deux métriques clés : la perte de validation, qui évalue l'adéquation des prédictions par rapport aux valeurs attendues sur un ensemble de données non utilisé dans l'entraînement, et la précision, qui mesure la justesse des prédictions par rapport aux étiquettes réelles sur cet ensemble de validation. Ces mesures sont essentielles pour garantir que le modèle peut généraliser correctement et prédire avec précision dans des conditions réelles.

Enfin, le modèle déployé pour l'inférence en temps réel bénéficie d'une optimisation matérielle continue, garantissant des performances stables et fiables dans les applications réelles.

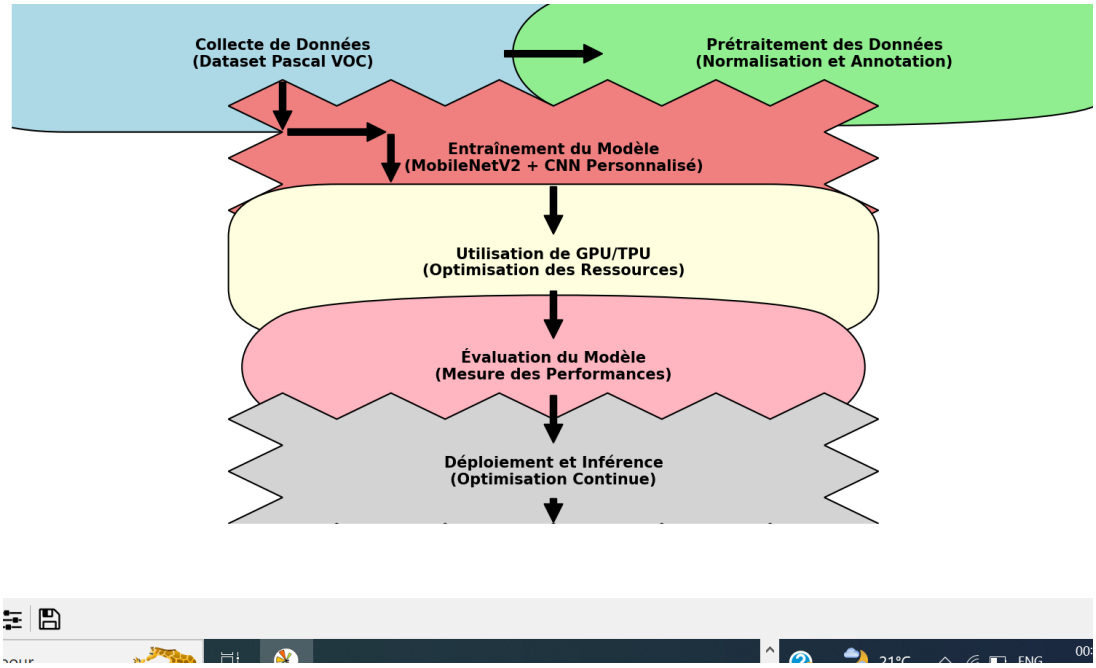



Figure 18 Fonctionnement général du système

# 6. Présentation de l'ensemble de données

Dans cette partie, nous détaillons les étapes impliquées dans l'entraînement d'un modèle de détection d'objets à l'aide des jeux de données PASCAL VOC 2007 et 2012. Nous abordons les phases allant de la collecte et l'extraction des données à leur prétraitement, en passant par la configuration et l'entraînement du modèle. Notre objectif est de créer un système de détection d'objets performant et généralisable en utilisant les annotations et les images fournies par ces jeux de données.

## 6.1. Collecte et Extraction des Données :

Les jeux de données PASCAL VOC sont essentiels pour la détection d'objets en raison de leur contenu annoté comprenant des images avec des boîtes englobantes et des étiquettes de classes. Après avoir téléchargé les archives des jeux de données depuis le site officiel, nous avons procédé à leur extraction pour obtenir les éléments suivants :

**Images :** Les fichiers image sont principalement en format JPEG ou PNG.

**Annotations :** Les informations sur les objets présents dans chaque image sont fournies sous forme de fichiers XML. Ces annotations détaillent les coordonnées des boîtes englobantes et les étiquettes de classes associées à chaque objet. (41)

## 6.2. Préparation de l'API de Détection d'Objets TensorFlow:

Pour faciliter le traitement des données et l'entraînement du modèle, nous avons utilisé l'API de Détection d'Objets de TensorFlow. Après avoir cloné le dépôt officiel et installé les dépendances requises, nous avons configuré l'API pour gérer la création de jeux de données, l'entraînement du modèle, et l'évaluation des performances.

## 6.3. Conversion des Données en TFRecord :

Les données extraites ont été converties au format TFRecord pour optimiser l'efficacité de l'entraînement avec TensorFlow. Cette conversion comprend plusieurs étapes essentielles :

### 6.3.1Lecture des Annotations XML:

Les fichiers XML d'annotations, contenant des informations détaillées sur les objets dans chaque image, sont lus et analysés. Chaque annotation fournit des coordonnées de boîtes englobantes (xmin, ymin, xmax, ymax) et des étiquettes de classes pour les objets présents dans l'image.

**Exemple :** Pour une image donnée, l'annotation XML pourrait ressembler à ceci :


Figure 19 Lecture des Annotations XML

Dans cet exemple, 2007_000392.jpg contient deux objets : un horse et une personne, avec leurs coordonnées de boîtes englobantes respectives.

### 6.3.2Encodage des Images:

Les images, en formats JPEG ou PNG, sont lues à partir du disque et converties en chaînes de caractères encodées (bytes). Cela permet de stocker efficacement les images dans les fichiers TFRecord.

### 6.3.3Création des TFRecords :

Les images encodées et leurs annotations associées sont organisées en exemples de TFRecord. Chaque exemple de TFRecord contient une image et ses annotations correspondantes, structurant les données de manière à faciliter leur accès et traitement pendant l'entraînement du modèle.

## 6.4. Prétraitement des Données :

Les fichiers TFRecord, créés lors de l'étape précédente, sont chargés en utilisant tf.data.TFRecordDataset. Cela permet de parcourir efficacement les données pendant l'entraînement, facilitant ainsi l'accès séquentiel aux images et aux annotations.

### 6.4.1.Analyse des Exemples :

Chaque exemple de TFRecord est analysé pour extraire les images et leurs annotations. Cela comprend :

- **Extraction des Images** : Les chaînes de caractères encodées représentant les images sont décodées pour obtenir les images au format d'origine (JPEG ou PNG).
- **Extraction des Boîtes Englobantes :** Les coordonnées des boîtes englobantes (xmin, ymin, xmax, ymax) sont extraites des annotations XML.
- **Extraction des Étiquettes de Classes :** Les étiquettes de classes associées à chaque boîte englobante sont également extraites.

### 6.4.2. Redimensionnement des Images :

Les images sont redimensionnées à une taille fixe de 224x224 pixels pour assurer la compatibilité avec le modèle. Le redimensionnement permet de normaliser les tailles d'image, ce qui est essentiel pour le traitement par le modèle. Cela garantit que toutes les images d'entrée ont les mêmes dimensions, facilitant ainsi le traitement en lots (batch processing).

### 6.4.3. Conversion des Données Sparsifiées :

Les annotations des boîtes englobantes et des étiquettes de classes sont souvent fournies sous forme sparse (parcimonieuse), c'est-à-dire qu'elles ne contiennent que les informations nécessaires et omettent les valeurs nulles ou non présentes. Ces données sont converties en formats denses pour simplifier l'entraînement. Par exemple, les coordonnées des boîtes englobantes sont regroupées en tensors denses, et les étiquettes de classes sont converties en vecteurs de classes pour chaque image.

## 6.5. Division des Données pour l'Entraînement:

Les données ont été divisées en ensembles d'entraînement (70%) et de validation (30%) pour évaluer la performance du modèle sur des données non vues. Cette division est cruciale pour l'évaluation du modèle et la prévention du surapprentissage.

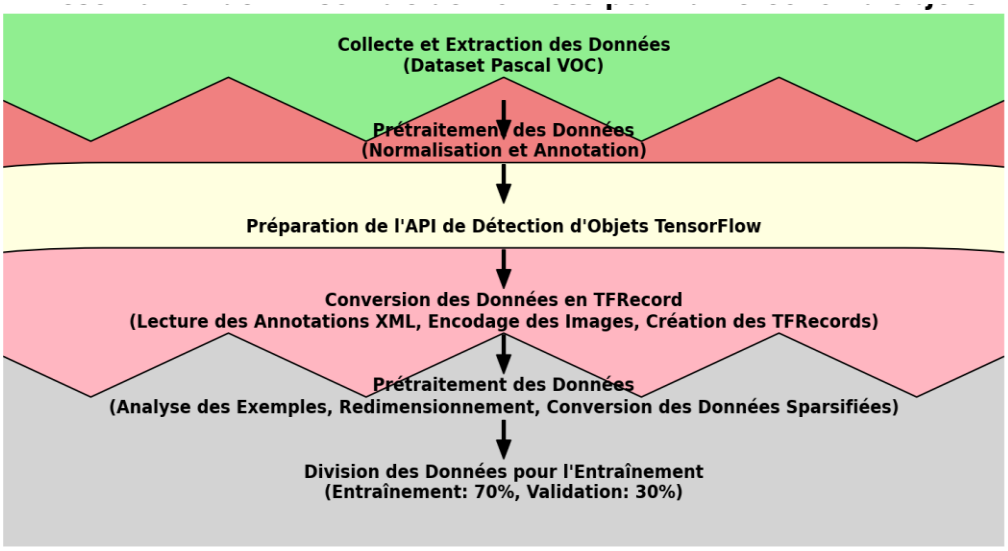



Figure 20 Présentation de l'ensemble de données

# 7. Architecture du modèle

## 7.1. Modèle de détection d'objets :

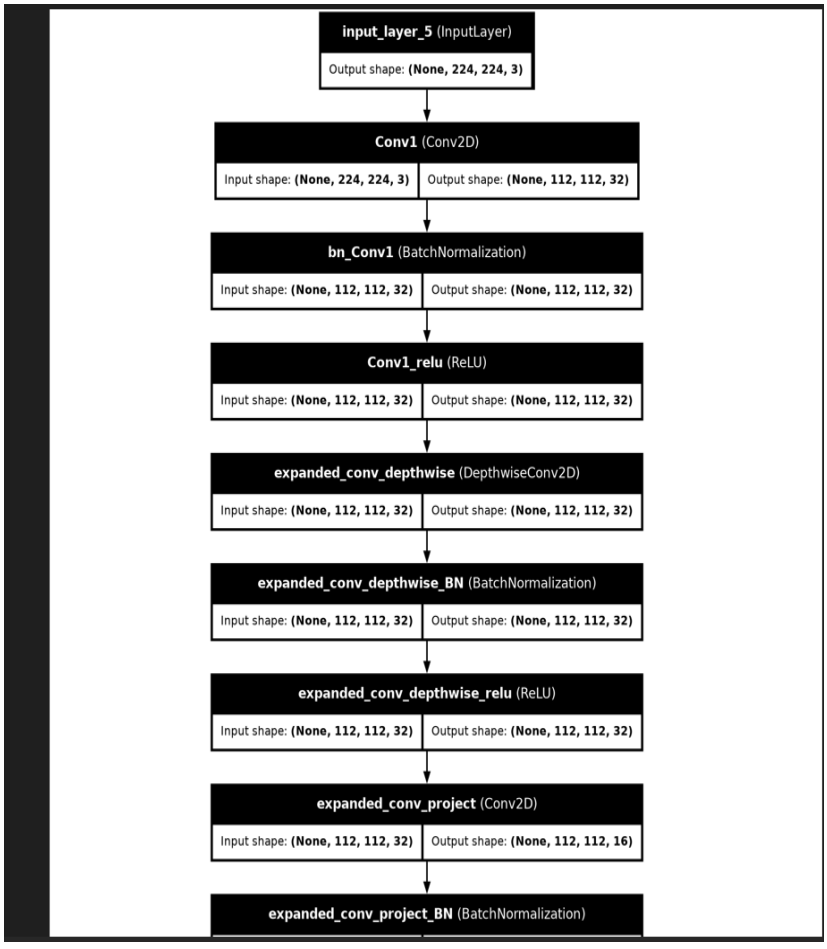



Figure 21 Architecture modèle (1).

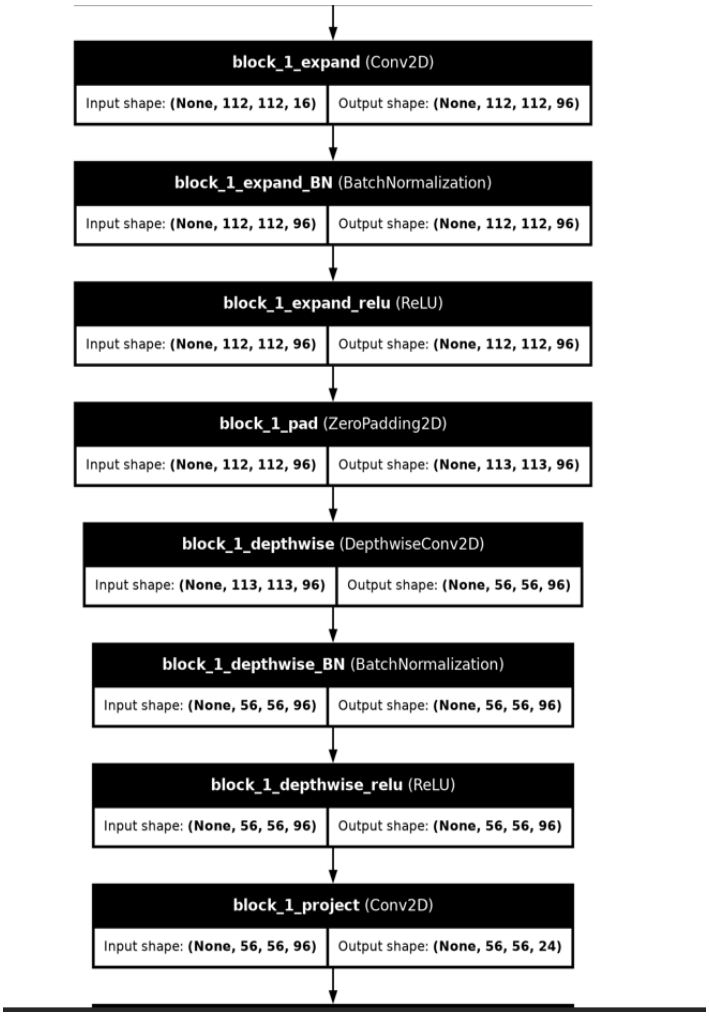



Figure 22 Architecture modèle (2).

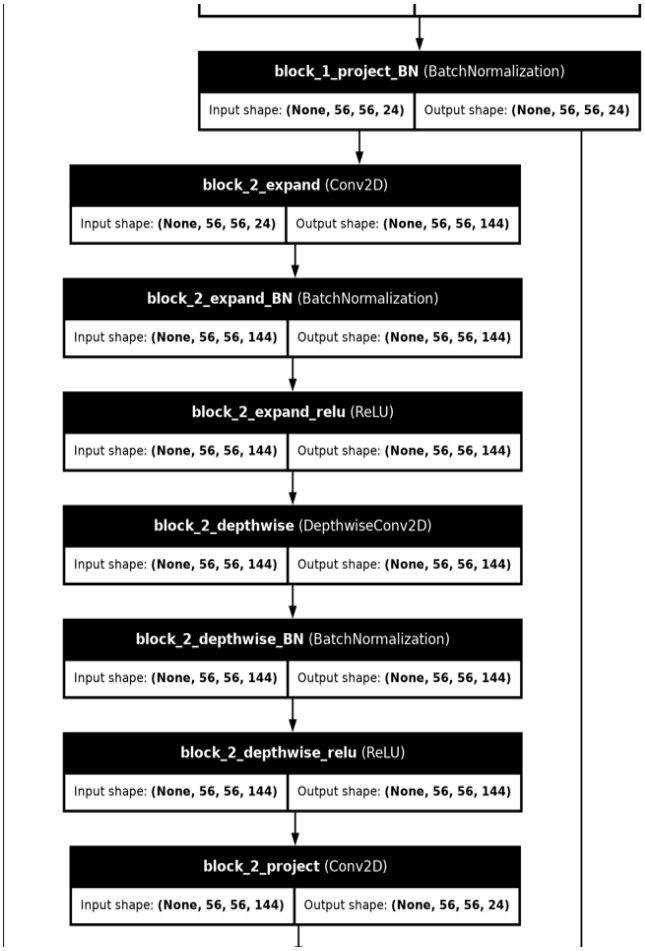



Figure 23 Architecture modèle (3).

La section suivante détaille chaque type de couche utilisé dans l'architecture de notre modèle CNN, comme illustré dans la figure fournie. Cependant, la figure ne couvre pas intégralement l'architecture en raison de sa complexité. J'ai expliqué toutes les couches de manière structurée pour offrir une compréhension complète de l'ensemble du modèle.

**Entrée (InputLayer) :** Cette couche reçoit des images en entrée de taille 224x224 pixels avec 3 canaux de couleur (RGB). Elle initialise le flux de données dans le réseau.

**Conv1 :** La couche Conv1 applique une convolution avec 32 filtres de taille 3x3 sur l'image d'entrée. Cette opération réduit la dimension spatiale de l'image par un facteur de 2 grâce à un stride de 2. Après la convolution, une normalisation par lot (BatchNormalization) est effectuée pour stabiliser et accélérer l'apprentissage du modèle. Enfin, une fonction d'activation ReLU est appliquée pour introduire une non-linéarité, aidant ainsi à extraire des caractéristiques bas-niveau comme les contours et les textures.

**Blocks d'Expansion et de Projection :** Ces blocks sont les éléments fondamentaux de l'architecture MobileNetV2, conçus pour extraire et transformer progressivement les

caractéristiques de l'image. Chaque block commence par une expansion de canal (expand), où le nombre de canaux est augmenté pour enrichir les informations disponibles. Ensuite, une convolution en profondeur (depthwise convolution) est appliquée pour capturer des détails spatiaux tout en conservant l'efficacité des calculs. Enfin, une projection (project) réduit dimensionnellement les données pour contrôler la complexité du modèle.

**Conv_1 :** Après une série de transformations complexes dans les blocks précédents, la couche Conv_1 utilise 1280 filtres de taille 1x1 pour capturer des caractéristiques hautement abstraites et complexes de l'image. Cette convolution finale est cruciale pour augmenter la complexité des caractéristiques extraites et permettre une meilleure discrimination entre les différents objets dans l'image.

**Conv2D finales** (par exemple, conv2d_66 et conv2d_67) : Ces couches finales utilisent respectivement 512 et 256 filtres pour raffiner les caractéristiques extraites à des niveaux de complexité plus élevés. Chaque convolution est suivie d'une activation ReLU pour introduire non-linéarité et capturer des informations discriminantes spécifiques aux objets.

**bbox_output :** Couche finale pour prédire les coordonnées des boîtes englobantes (bounding boxes) des objets détectés. Elle utilise une activation linéaire pour générer des valeurs continues qui définissent précisément la localisation des objets dans l'image.

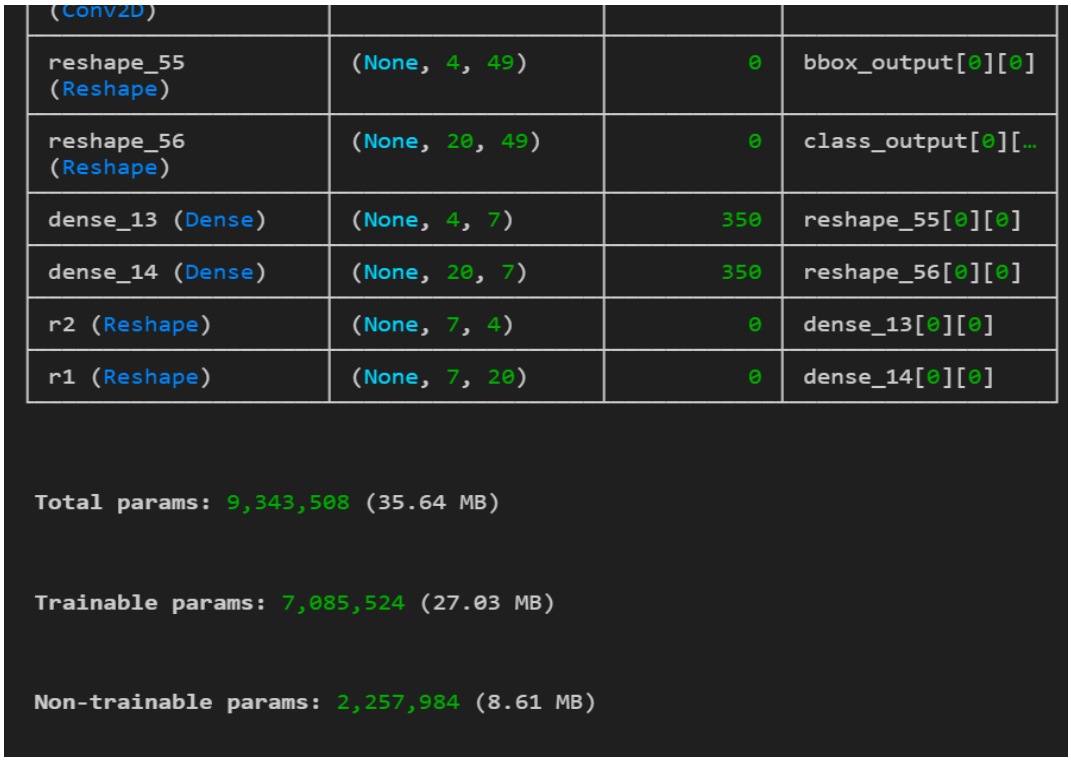
**class_output :** Dernière couche pour prédire les scores de classe des objets détectés. Une activation softmax est appliquée pour obtenir des probabilités sur les différentes classes, facilitant ainsi l'identification des types d'objets présents dans l'image.

**Reshape et Dense Layers :** Ces couches jouent un rôle crucial dans la mise en forme des sorties des convolutions précédentes. Les couches Reshape réorganisent les données pour correspondre aux formats attendus par les couches Dense, qui appliquent des transformations linéaires aux prédictions avant leur sortie finale.

Chaque couche de cette architecture est soigneusement conçue pour optimiser l'extraction, la transformation et la prédiction des caractéristiques des objets dans les images, garantissant ainsi une détection précise et efficace des objets dans divers contextes et conditions.

## 7.2. Configuration et paramétrage du modèle

Le nombre total de paramètres, les paramètres entraînables et les paramètres non entraînables sont cruciaux pour évaluer comment le modèle apprend à partir des données d'entraînement et généralise à de nouvelles données, en particulier pour la détection et la localisation précises d'objets dans des scènes complexes. Bien que la figure de paramétrage soit trop vaste pour être incluse entièrement, j'ai inclus les parties essentielles qui illustrent la distribution et la gestion des paramètres entraînables et non entraînables, nécessaires à la compréhension globale de l'architecture du modèle."



| (Conv2D) | | | |
|---|---|---|---|
| reshape_55 (Reshape) | (None, 4, 49) | 0 | bbox_output[0][0] |
| reshape_56 (Reshape) | (None, 20, 49) | 0 | class_output[0][… |
| dense_13 (Dense) | (None, 4, 7) | 350 | reshape_55[0][0] |
| dense_14 (Dense) | (None, 20, 7) | 350 | reshape_56[0][0] |
| r2 (Reshape) | (None, 7, 4) | 0 | dense_13[0][0] |
| r1 (Reshape) | (None, 7, 20) | 0 | dense_14[0][0] |

Total params: 9,343,508 (35.64 MB)

Trainable params: 7,085,524 (27.03 MB)

Non-trainable params: 2,257,984 (8.61 MB)

Figure 24 Configuration et paramétrage du modèle

- **Total params (nombre total de paramètres) :** Ce chiffre représente la somme totale de tous les paramètres présents dans le modèle, qu'ils soient modifiables (trainables) ou fixes (non trainables). Ces paramètres incluent les poids des connexions entre les neurones, ainsi que les paramètres de normalisation, de mise à l'échelle, et d'autres opérations dans les couches du réseau. Le nombre total de paramètres reflète la capacité globale du modèle à apprendre et à représenter des données.
- **Trainable params (nombre de paramètres entraînables) :** Ce nombre se réfère aux paramètres du modèle qui sont ajustés pendant l'entraînement pour optimiser les performances du modèle sur une tâche spécifique. Typiquement, il exclut les paramètres fixés à l'avance ou qui ne nécessitent pas d'apprentissage supplémentaire, tels que ceux des couches de pooling ou de normalisation. Les paramètres entraînables représentent la

capacité du modèle à s'adapter et à généraliser à partir des données fournies pendant l'entraînement.

- **Non-trainable params (nombre de paramètres non entraînables)** : Ce chiffre correspond aux paramètres dans le modèle qui restent fixes et inchangés pendant l'entraînement. Il inclut généralement les poids prédéfinis ou figés, comme ceux des couches de pooling ou de normalisation, qui ne sont pas modifiés même lorsque le modèle est exposé à des données d'entraînement. Les paramètres non entraînables sont essentiels pour des opérations telles que la normalisation des données ou la réduction dimensionnelle, sans impact sur la capacité du modèle à apprendre directement à partir des données.

## 7.3. Fonction de perte:

Dans cette partie, nous avons exploré en détail les fonctions de perte essentielles de notre système proposé pour la détection d'objets.

### 7.3.1. Mean Squared Error (MSE) pour bbox_loss

La Mean Squared Error (MSE) est une mesure couramment utilisée pour évaluer la précision des prédictions par rapport aux valeurs réelles. Dans le contexte de la détection d'objets, la MSE est particulièrement adaptée pour évaluer la précision des prédictions de boîtes englobantes.

- **Fonctionnement :** Pour chaque exemple d'entraînement, la MSE calcule la différence quadratique moyenne entre les coordonnées prédites des boîtes englobantes (qui représentent généralement x,y,w,hx, y, w, hx,y,w,h) et les coordonnées réelles de ces boîtes englobantes.

Formule mathématique de la MSE **:**

$$(4)\quad \text{MSE}\,(ytrue[0],ypred[0])=n1\sum i=1n\sum j=14(ytrue[0][i,j]-ypred[0][i,j])2$$

- **Objectif :** L'objectif de la MSE dans (bbox_loss) est de minimiser cette différence quadratique moyenne. En ajustant les paramètres du modèle pour réduire la MSE, le modèle apprend à localiser plus précisément les objets dans l'image en ajustant les coordonnées prédites des boîtes englobantes pour qu'elles se rapprochent des coordonnées réelles.

### 7.3.2. Cross-Entropy Categorical pour class_loss :

La Cross-Entropy Categorical est une mesure couramment utilisée pour évaluer la divergence entre deux distributions de probabilité : celle des prédictions du modèle et celle des étiquettes réelles encodées en one-hot.

**Fonctionnement :** Dans le modèle de détection d'objets, la Cross-Entropy Categorical est appliquée à la prédiction des classes. Pour chaque cellule de la grille de prédiction, le modèle prédit une distribution de probabilité sur les classes possibles. Cette distribution est comparée à la distribution réelle des classes (encodée en one-hot) pour mesurer à quel point elles diffèrent.

Formule mathématique de la Cross-Entropy Categorical pour class_loss :

$$\textbf{(5) Cross-Entropy(ytrue[1],ypred[1])}=-\frac{1}{n}\sum_{i=1}^{n}\sum_{c=1}^{C}\textbf{ytrue[1][i,c]}\cdot\log(\textbf{ypred[1][i,c]})$$

**Objectif :** En minimisant la Cross-Entropy Categorical dans class_loss**,** le modèle est entraîné à améliorer sa capacité à classifier correctement les objets dans chaque cellule de la grille de prédiction. Cela signifie ajuster les poids du modèle pour que les distributions de probabilité prédites se rapprochent le plus possible des distributions réelles des classes, ce qui améliore la précision globale de la classification des objets détectés.

### 7.3.3.Combinaison dans custom_loss :

La fonction custom_loss combine ces deux pertes pour former une seule fonction de perte globale qui est utilisée pour entraîner le modèle de détection d'objets.

**Objectif global :** En combinant bbox_loss (MSE) et class_loss (Cross-Entropy Categorical) dans custom_loss, le modèle est entraîné à optimiser simultanément la précision de la localisation des boîtes englobantes et la précision de la classification des objets détectés. En minimisant cette fonction de perte globale pendant l'entraînement, le modèle apprend à ajuster ses paramètres de manière à améliorer à la fois la localisation et la classification, ce qui se traduit par une meilleure performance dans la détection d'objets dans des images complexes et variées.

## 7.4.Optimiseur:

Adaptive Moment Estimation (Adam) : L'optimiseur Adam combine les avantages du SGD et de l'optimiseur RMSprop en utilisant une estimation adaptative des moments du gradient pour mettre à jour les poids du modèle. Cet optimiseur est largement préféré pour sa performance robuste dans une variété de tâches d'apprentissage automatique.

Le premier moment du gradient est calculé selon l'équation :

$$\mathbf{m} = \mathbf{beta_1} \times \mathbf{m} + (\mathbf{1} - \mathbf{beta_1}) \times \mathbf{gradient} \quad \textbf{(6)}$$

Le deuxième moment du gradient est calculé selon l'équation :

$$v = \mathbf{beta_2} \times \mathbf{v} + (\mathbf{1} - \mathbf{beta_2}) \times \mathbf{gradient^2} \quad (7)$$

Enfin, les poids du modèle sont mis à jour en utilisant ces estimations :

$$\mathbf{w} = \frac{\mathbf{w} - \mathbf{learning_{rate}} \times \mathbf{m}}{\sqrt{v} + epsilon} \quad (8)$$

Dans cette formule, le taux d'apprentissage (learning_rate) contrôle l'amplitude des mises à jour des poids, tandis que epsilon (ε) est une petite valeur ajoutée pour éviter les divisions par zéro.

L'optimiseur Adam est particulièrement efficace pour converger rapidement et de manière stable lors de l'entraînement du modèle, ce qui en fait un choix courant dans divers domaines d'application en apprentissage automatique.

Dans notre implémentation, nous utiliserons l'optimiseur Adam pour mettre à jour les poids du modèle, profitant ainsi de ses capacités adaptatives avancées pour optimiser efficacement les performances de notre modèle.

## 8. Conclusion

En conclusion, ce chapitre a exposé les défis majeurs auxquels font face les technologies actuelles de détection d'objets, notamment leurs limitations en termes de performance dans des environnements complexes, leur manque de scalabilité face à l'augmentation des volumes de données, ainsi que leur rigidité à s'adapter aux besoins spécifiques des utilisateurs. Ces obstacles compromettent la fiabilité et l'efficacité des systèmes de vision par ordinateur dans des applications pratiques.

Dans cette optique, notre projet se positionne comme une réponse ambitieuse à ces défis. En nous appuyant sur une analyse approfondie des architectures existantes et en développant un modèle CNN sur mesure, nous visons à renforcer la précision, la robustesse et la flexibilité de notre système de détection d'objets. Notre approche méthodologique rigoureuse, combinée à des techniques avancées de traitement des données et d'entraînement des modèles, constitue un pas significatif vers la création de solutions plus performantes et adaptables.

En résumé, cette recherche offre une contribution significative à l'amélioration des technologies de détection d'objets, en identifiant les lacunes actuelles et en proposant des solutions innovantes. L'étape suivante de notre travail consistera à concrétiser ces avancées à travers la mise en œuvre pratique de notre modèle, afin d'évaluer ses performances dans des conditions réelles et d'explorer de nouvelles possibilités d'amélioration. Cela nous permettra non seulement de valider nos résultats théoriques, mais aussi d'envisager les futures applications potentielles de notre approche en vision par ordinateur.

# Chapitre 2: Implémentation et résultats

## 1. Introduction

Ce dernier chapitre marque une étape cruciale dans notre projet, celle de la mise en œuvre concrète de notre système de détection d'objets. Après avoir minutieusement planifié et conçu notre approche, nous nous concentrons maintenant sur l'application pratique de nos stratégies et la configuration de notre environnement de développement. Dans cette section, nous explorerons les outils matériels et logiciels que nous avons sélectionnés pour soutenir notre implémentation, en mettant l'accent sur leur rôle crucial dans le succès de notre projet. De plus, nous discuterons des choix architecturaux de notre modèle, des métriques d'évaluation essentielles que nous utilisons pour mesurer sa performance, ainsi que des défis rencontrés et des solutions adoptées tout au long de cette phase de déploiement. Ce chapitre est une exploration détaillée de notre transition de la théorie à la pratique, visant à concrétiser notre vision d'une application de détection d'objets robuste et efficace.

## 2. Représentation des outils de développement

### 2.1 Environnements physique

| **Matériel** | **Spécification** |
|---|---|
| Nom de l'ordinateur | HP ProBook 640 G2 |
| Mémoire installée (RAM) | 8,00 Go DDR4 |
| Mémoire installée (ROM) | HDD |
| Type de système | Système d'exploitation 64 bits, processeur x64 |
| Processeur | Intel(R) Core (TM) i5-6300U CPU @ 2,40GHz |

Table 2 Caractéristique d'environnement physique.

Pendant notre phase d'entraînement sur Kaggle, nous avons optimisé l'utilisation des ressources disponibles. Notre session maximale de 12 heures nous a permis d'exécuter nos scripts d'entraînement efficacement. Pour accélérer les calculs, nous avons exploité l'accélérateur GPU T4 * 2, équipé de 14,2 Go de mémoire GPU. En matière de stockage, nous avions à notre disposition 73,1 Go d'espace disque pour temporairement stocker nos données et résultats intermédiaires durant le processus.

**Kaggle** est une plateforme complète dédiée à la science des données, offrant des compétitions, des notebooks interactifs, des jeux de données, des kernels, des forums actifs, et une communauté collaborative. Les notebooks basés sur le cloud permettent aux utilisateurs d'écrire du code en Python et R, d'explorer des modèles, de visualiser les données, et de partager leurs analyses. Kaggle propose également des ressources de calcul puissantes, y compris des **accélérateurs GPU**, pour soutenir les projets d'apprentissage automatique et de science des données. **Les kernels** interactifs sont utilisés pour présenter des méthodes, des solutions de compétition, ou des projets exploratoires.

En résumé, Kaggle facilite l'apprentissage, la collaboration, la résolution de problèmes, et le partage de connaissances dans le domaine en constante évolution de la science des données et de l'apprentissage automatique.

## 2.2 Logiciels et bibliothèques utilisés dans la mise en œuvre

Dans notre implémentation, nous avons choisi avec soin les outils en fonction de leur efficacité et de leur adaptabilité à notre projet :

**Visual Studio Code** est un éditeur de code source léger mais puissant, compatible avec Windows, macOS et Linux. Il offre un support natif pour JavaScript, TypeScript, Node.js et un large éventail d'extensions pour d'autres langages comme C++, C#, Java, Python, PHP, Go et .NET. Visual Studio Code se distingue par sa polyvalence et sa capacité à répondre aux besoins variés de développement, offrant une expérience de codage agréable et productive.[45]

**TensorFlow :** TensorFlow est une bibliothèque open-source pour l'apprentissage automatique et le calcul numérique, offrant des outils pour créer, entraîner et déployer des modèles d'apprentissage automatique.[46]

**TensorFlow Lite** : TensorFlow Lite est une version allégée de TensorFlow spécialement conçue pour le déploiement de modèles d'apprentissage automatique sur des appareils mobiles, des systèmes embarqués et d'autres plates-formes à ressources limitées.

**OpenCV :** Bibliothèque open source spécialisée dans le traitement d'images et la vision par ordinateur. OpenCV offre des outils pour charger, manipuler et prétraiter des images, ainsi que pour effectuer diverses opérations de vision par ordinateur comme la détection d'objets.[47]

**Keras :** Keras est une bibliothèque open-source d'apprentissage profond intégrée à TensorFlow. Elle fournit une interface conviviale pour la création et l'entraînement de modèles de réseaux neuronaux.[48]

**NumPy :** NumPy est une bibliothèque fondamentale pour le calcul scientifique en Python, offrant des structures de données de tableau multidimensionnel et des fonctions mathématiques pour manipuler efficacement ces tableaux.

**Os :** Le module os fournit des fonctions pour interagir avec le système d'exploitation, notamment pour gérer les fichiers et dossiers.

**Matplotlib :** Bibliothèque de visualisation en Python utilisée pour créer des graphiques 2D. matplotlib.pyplot de Matplotlib est souvent employé pour visualiser des métriques d'entraînement telles que les courbes de perte et de précision.

**tensorflow.keras** : Implémentation de Keras comme une API de haut niveau intégrée à TensorFlow depuis sa version 2.x. TensorFlow.keras simplifie la création et l'entraînement de modèles de deep learning en fournissant des modules prêts à l'emploi comme Conv2D, Reshape et Dense pour définir les couches du modèle, ainsi que Model pour créer des architectures personnalisées.

**tensorflow.keras.applications.MobileNetV2 :** Modèle pré-entraîné disponible dans TensorFlow.keras.applications, utilisé ici pour ses capacités de feature extraction dans la vision par ordinateur.

**tensorflow.keras.optimizers.Adam :** Algorithme d'optimisation Adam, une méthode populaire et efficace pour ajuster les poids du modèle pendant l'entraînement en minimisant la fonction de perte.

**tensorflow.keras.losses.mse** : Fonction de perte pour l'erreur quadratique moyenne (Mean Squared Error). Utilisée pour évaluer la différence entre les prédictions du modèle et les valeurs réelles des boîtes englobantes dans les tâches de régression.

**tensorflow.keras.losses.categorical_crossentropy** : Fonction de perte pour la perte de cross-entropy catégorique, adaptée aux problèmes de classification multiclasse où les étiquettes sont encodées en one-hot.

**tf.data.Dataset :** API TensorFlow permettant de gérer efficacement de grandes quantités de données et d'optimiser leur traitement pour l'entraînement des modèles. Dataset est utilisé pour créer des pipelines de données efficaces (train_dataset et val_dataset), essentiels pour charger, prétraiter et alimenter les données au modèle pendant l'entraînement.

### 2.3. Langage de programmation

**Python** est un langage de programmation polyvalent parmi une gamme diversifiée de langages informatiques tels que Java, LISP, PHP, Perl, et C, chacun se distinguant par ses forces spécifiques. Par exemple, Java est reconnu pour sa portabilité à travers différents systèmes, PHP excelle dans le développement de sites web dynamiques avec accès aux bases de données, tandis que C'est souvent utilisé pour des applications nécessitant des performances optimales. Malgré ces différences, tous ces langages partagent des concepts fondamentaux comme la manipulation de données à travers des variables et l'utilisation de fonctions pour opérer sur ces données.

Python se distingue par sa simplicité, sa puissance et son élégance. Il intègre les fonctionnalités communes à de nombreux autres langages tout en offrant une syntaxe claire et facile à lire. Sa popularité croissante repose sur une vaste communauté de développeurs passionnés qui contribuent à son évolution constante. En tant que logiciel libre, Python bénéficie d'une implémentation standardisée qui facilite son adoption et sa diffusion à travers différents environnements. Apprendre Python peut également servir de fondation solide pour explorer d'autres langages de programmation, car de nombreux concepts et structures de contrôle sont universels, facilitant ainsi l'acquisition de compétences multiples dans le domaine du développement logiciel.[49]

## 3. Préparation des Données pour l'Entraînement

Avant de commencer l'entraînement, il est essentiel de préparer nos données de manière appropriée. Comme expliqué dans le chapitre de conception, nous avons décidé de convertir les données au format TFRecord pour une efficacité optimale pendant l'entraînement et de redimensionner les images à une taille fixe (224x224 pixels) pour assurer la compatibilité avec l'architecture du modèle.

Pour implémenter ce prétraitement, nous avons défini une fonction de chargement des ensembles de données d'entraînement et de validation à partir des fichiers TFRecord. Cette fonction utilise TensorFlow pour parser les fichiers TFRecord, redimensionner les images, et préparer les annotations des boîtes englobantes et des classes.

- **Chargement des Fichiers TFRecord :** Nous utilisons TFRecordDataset pour charger les fichiers TFRecord. Ce format est efficace pour traiter de grandes quantités de données séquentielles, en permettant une lecture rapide et optimisée.
- **Parsing des Exemples :** La fonction parse_example définit les caractéristiques à extraire des fichiers TFRecord. Les images sont décodées à partir du format JPEG et redimensionnées à 224x224 pixels pour garantir une taille uniforme et faciliter le traitement par le modèle.
- **Conversion des Données Sparses :** Les annotations des boîtes englobantes et des classes, qui sont initialement au format sparse (parcimonieux), sont converties en format dense. Les coordonnées des boîtes englobantes sont ensuite empilées pour créer un tensor approprié, facilitant ainsi leur utilisation dans le modèle.
- **Création du Dataset :** Le dataset est ensuite mappé avec la fonction parse_example pour transformer chaque exemple brut en un format utilisable. Les exemples sont ensuite groupés en lots (batchés) selon la taille spécifiée (batch_size), ce qui permet d'optimiser l'entraînement du modèle en traitant plusieurs exemples simultanément.
- **Chargement des Ensembles de Données :** Enfin, nous chargeons les ensembles de données d'entraînement et de validation en utilisant les chemins des fichiers TFRecord. Cela permet de préparer les données pour l'entraînement et l'évaluation du modèle.

Cette approche assure que nos images et annotations sont dans le format approprié pour l'entraînement du modèle, tout en optimisant le processus de chargement des données.

# 4. Discussion et Résultats

## 4.1. Mesure d'Évaluation

Dans notre implémentation, nous utilisons deux métriques essentielles pour évaluer et améliorer la performance de nos modèles de machine learning : la métrique de loss (perte) et la métrique de validation loss (perte de validation). Ces métriques jouent un rôle crucial pour mesurer l'efficacité de notre modèle pendant l'entraînement et pour évaluer sa capacité à généraliser sur de nouvelles données.

### 4.1.1. Métrique de Loss

La métrique de loss est utilisée pour quantifier l'erreur du modèle sur les données d'entraînement. À chaque epoch, elle mesure à quel point les prédictions du modèle divergent des valeurs réelles des données d'entraînement. Cette métrique est calculée à l'aide d'une fonction de perte adaptée au type de problème que nous traitons, comme la cross-entropy pour la classification ou la mean squared error pour la régression.

### 4.1.2. Métrique de Validation Loss

La métrique de validation loss est similaire à la métrique de loss, mais elle est calculée sur un ensemble de données de validation distinct, que le modèle n'a pas utilisé pour l'entraînement. Cela nous permet d'évaluer la capacité du modèle à généraliser sur de nouvelles données. Une fonction de perte distincte est utilisée pour calculer cette métrique, afin de refléter fidèlement la performance du modèle dans des conditions réelles.

### 4.1.3. Courbe de Loss et Validation Loss

Nous générons une courbe de loss et validation loss en traçant ces métriques à chaque epoch d'entraînement. Cette visualisation nous permet de suivre la progression de notre modèle : idéalement, nous cherchons à observer une diminution continue des deux métriques, ce qui indique que le modèle apprend efficacement sans surapprendre. Une augmentation de la validation loss par rapport à la loss d'entraînement pourrait signaler un surapprentissage, nécessitant une réévaluation des hyperparamètres du modèle.

## 4.2. Performance et Analyse

Dans cette analyse, nous avons examiné trois modèles différents (Modèle A, Modèle B et Modèle de Base) pour la tâche de détection d'objets. Chaque modèle a été évalué en fonction de son architecture, de son batch size et de son learning rate, avec un suivi des performances à travers les pertes moyennes sur les ensembles d'entraînement et de validation après 25 epochs.

### 4.2.1. Modele A

| Modèle | Batch Size | Learning Rate | Loss (Train) | Loss (Val) |
|---|---|---|---|---|
| Modèle A | 32 | 0.1 | 8.65 | 7.71 |

Table 3 modele A.

Le modèle A utilisé trois couches Conv2D avec différentes configurations de filtres : 512 dans la première couche, 256 dans la deuxième et 128 dans la troisième, toutes avec des filtres de taille (3, 3) et une activation ReLU. Le modèle prédit les boîtes englobantes avec une couche Conv2D suivie d'un reshape, et les classes avec une autre couche Conv2D. Lors de l'entraînement avec un batch size de 32 et un learning rate de 0.1 pendant 25 epochs, le modèle a atteint une perte moyenne de 8.65 sur l'ensemble d'entraînement et de 7 .71 sur l'ensemble de validations. Ces pertes relativement élevées indiquent que le modèle n'a pas appris de manière optimale et ne généralise pas bien aux nouvelles données, suggérant que les paramètres choisis (batch size et learning rate) ainsi que l'architecture du modèle ne sont pas adéquats pour cette tâche spécifique.

### 4.2.2. Modele B

| Modèle | Batch Size | Learning Rate | Loss (Train) | Loss (Val) |
|---|---|---|---|---|
| Modèle B | 32 | 0.1 | 5.23 | 5.03 |

Table 4 modele B.

Modèle B utilise une architecture composée de deux couches Conv2D avec des filtres de tailles 256 et 256. Pour la prédiction des boîtes englobantes, il utilise une couche Conv2D avec 4 filtres de taille (1, 1) et pour la prédiction des classes, une autre couche Conv2D avec num_classes filtres de taille (1, 1). Le modèle a été configuré avec un batch size de 32 et un learning rate de 0.1 pour l'optimisation, ce qui est beaucoup trop élevé. En raison de ce learning rate inapproprié, le modèle a montré des performances médiocres, avec une perte moyenne sur l'ensemble d'entraînement de 5.23 et une perte moyenne sur l'ensemble de validation de 5.03 après 25 epochs.

### 4.2.3. Modele de base

| Modèle | Batch Size | Learning Rate | Loss (Train) | Loss (Val) |
|---|---|---|---|---|
| Modèle de Base | 32 | 0.001 | 2.65 | 2,55 |

Table 5 modele de base.

Le modèle de base utilise une architecture avec deux couches Conv2D ayant des filtres de tailles 512 et 256 respectivement. Ce modèle est configuré avec un batch size de 32 et un learning rate de 0.001 pour l'optimisation. Pour la prédiction des boîtes englobantes, il utilise une

couche Conv2D avec 4 filtres de taille (1, 1), tandis que la prédiction des classes est réalisée via une autre couche Conv2D avec num_classes filtres de taille (1, 1). Les performances de ce modèle sur les ensembles d'entraînement et de validation montrent des pertes moyennes de 2.65 et 2.55 respectivement après 25 epochs. Cela confirme que notre modèle de système donne de bons résultats. La figure ci-dessous montre les courbes de loss et de val_loss.

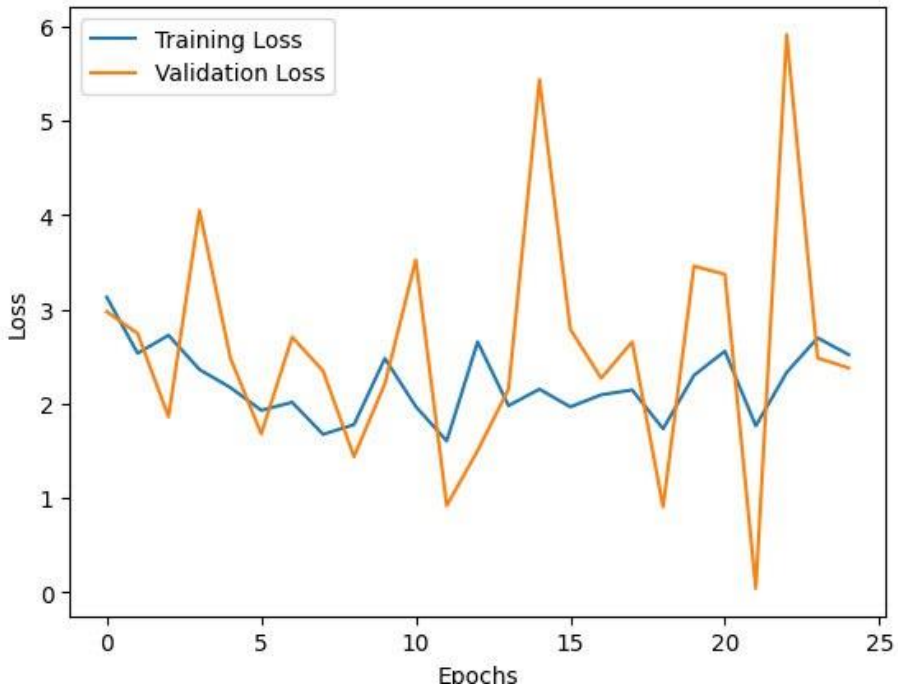



Figure 25 courbe loss et val loss

Après avoir analysé les performances des différents modèles proposés, nous avons décidé d'opter pour le modèle de base en raison de sa stabilité et de sa capacité à maintenir des performances élevées sur les ensembles d'entraînement et de validation. Le modèle de base utilise une architecture relativement simple avec deux couches Conv2D ayant des filtres de tailles 512 et 256 respectivement. Cette configuration a montré des pertes moyennes de 2.65 sur l'ensemble d'entraînement et de 2.55 sur l'ensemble de validation après 25 epochs, ce qui indique une bonne capacité de généralisation et une convergence satisfaisante.

En comparaison, le modèle A, malgré l'ajout d'une troisième couche Conv2D (128) et le modèle B avec des couches Conv2D de tailles différentes (256, 256), ont montré des performances inférieures. Le modèle A a souffert de pertes moyennes plus élevées (8.65 sur l'ensemble d'entraînement et 7.71 sur l'ensemble de validation), suggérant un potentiel de surapprentissage et une mauvaise généralisation. De même, le modèle B a été affecté négativement par un learning rate trop élevé (0.1), entraînant des pertes moyennes de 5.23 sur l'ensemble d'entraînement et de 5.03 sur l'ensemble de validation, compromettant ainsi sa capacité à apprendre efficacement les caractéristiques pertinentes pour la détection d'objets.

En conclusion, le modèle de base se distingue par son équilibre entre performance et complexité. Ses paramètres optimisés et son architecture robuste en font le choix idéal pour notre application de détection d'objets, garantissant une efficacité et une fiabilité optimales dans notre système.

## 5. Interface système

L'interface de cette application de détection d'objets en temps réel repose sur l'utilisation d'une webcam pour capturer des images en direct et sur l'intégration d'un modèle de détection d'objets pour analyser ces images. Voici une explication du fonctionnement et des principes de cette interface, sans se concentrer sur le code spécifique :

### ✓ Source d'Entrée - Webcam en Temps Réel :

L'interface utilise une webcam connectée à l'ordinateur comme source principale d'entrée. Cette webcam capture des images en continu, fournissant ainsi des données visuelles en temps réel à l'application. Cela permet à l'utilisateur de voir ce qui se passe dans l'environnement capturé par la webcam sur son écran d'ordinateur.

### ✓ Traitement d'Images :

Les images capturées par la webcam sont traitées par l'application pour détecter la présence et la localisation d'objets spécifiques. Ce processus peut inclure des étapes telles que la conversion des images dans un format compatible avec le modèle de détection d'objets, le redimensionnement des images si nécessaire, et d'autres prétraitements pour optimiser la qualité des prédictions.

### ✓ Modèle de Détection d'Objets :

Notre modèle de détection d'objets pré-entraîné est intégré à l'interface. Ce modèle a été formé sur un ensemble de données étendu pour reconnaître différents types d'objets dans les images. Lorsque l'application reçoit une nouvelle image de la webcam, elle utilise ce modèle pour analyser l'image et identifier les objets présents, ainsi que leurs positions approximatives dans l'image.

### ✓ Affichage des Résultats de Détection :

Les résultats de la détection sont affichés à l'utilisateur en temps réel. Cela peut inclure des annotations visuelles sur l'image capturée, telles que des boîtes englobantes autour des objets détectés et des étiquettes indiquant le type d'objet détecté (par exemple, "voiture", "personne", "chaise", etc.).

### ✓ Interaction Utilisateur :

L'interface est conçue pour être interactive, permettant à l'utilisateur de visualiser les résultats de détection directement sur l'écran de son ordinateur. Il peut également y avoir des fonctionnalités supplémentaires pour ajuster les paramètres de détection, visualiser les statistiques de performance du modèle, ou encore sauvegarder les résultats de détection pour une utilisation ultérieure.

- ✓ **Sortie et Contrôle :**

L'utilisateur a un contrôle direct sur l'interface à travers des interactions simples, comme l'arrêt de la capture vidéo de la webcam ou la fermeture de l'application. Cela assure une expérience utilisateur fluide et intuitive tout en maximisant l'utilité pratique de l'application.

# 6. Test de système

## 6.1. Détection d'Objets

Dans l'interface de détection d'objets en temps réel, les résultats sont affichés de manière claire et détaillée pour offrir à l'utilisateur une expérience fluide et informative. Chaque objet détecté est marqué par une boîte englobante précise, indiquant sa localisation dans l'image capturée par la webcam. À côté de chaque boîte, une étiquette spécifie le type d'objet détecté, tel que "voiture", "personne" ou "chaise". Un score de confiance est également affiché pour chaque détection, permettant à l'utilisateur de comprendre la fiabilité de la détection, avec des valeurs numériques ou des pourcentages représentant le niveau de certitude du modèle. Ces informations sont actualisées en temps réel à mesure que de nouvelles images sont traitées, assurant une interaction intuitive et efficace où l'utilisateur peut prendre des décisions informées ou ajuster les actions en fonction des résultats de la détection.

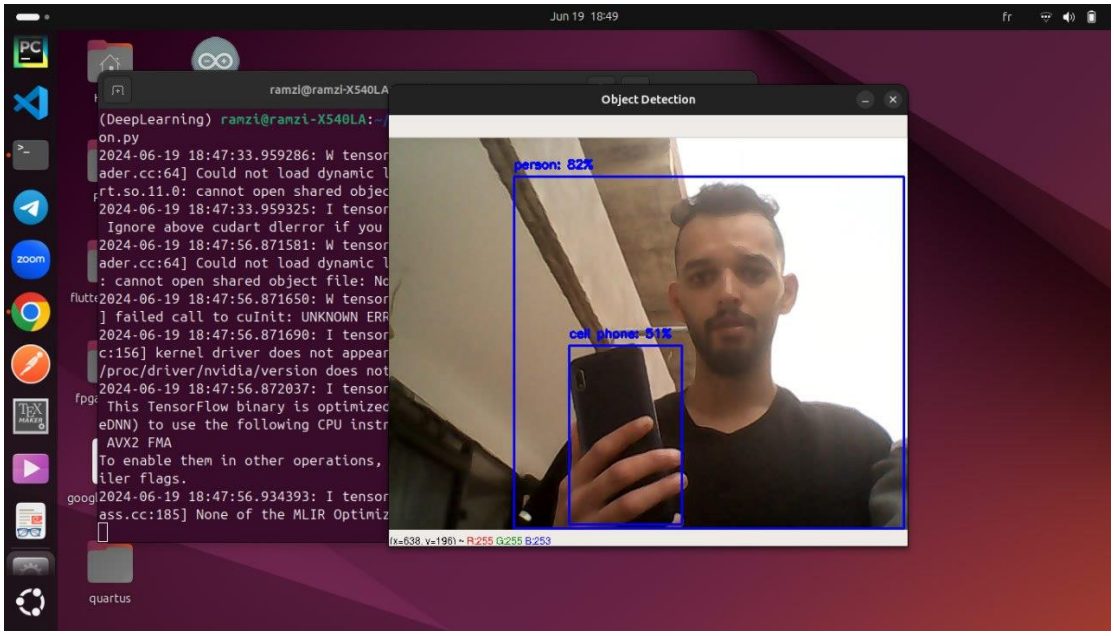



Figure 26 Test 1

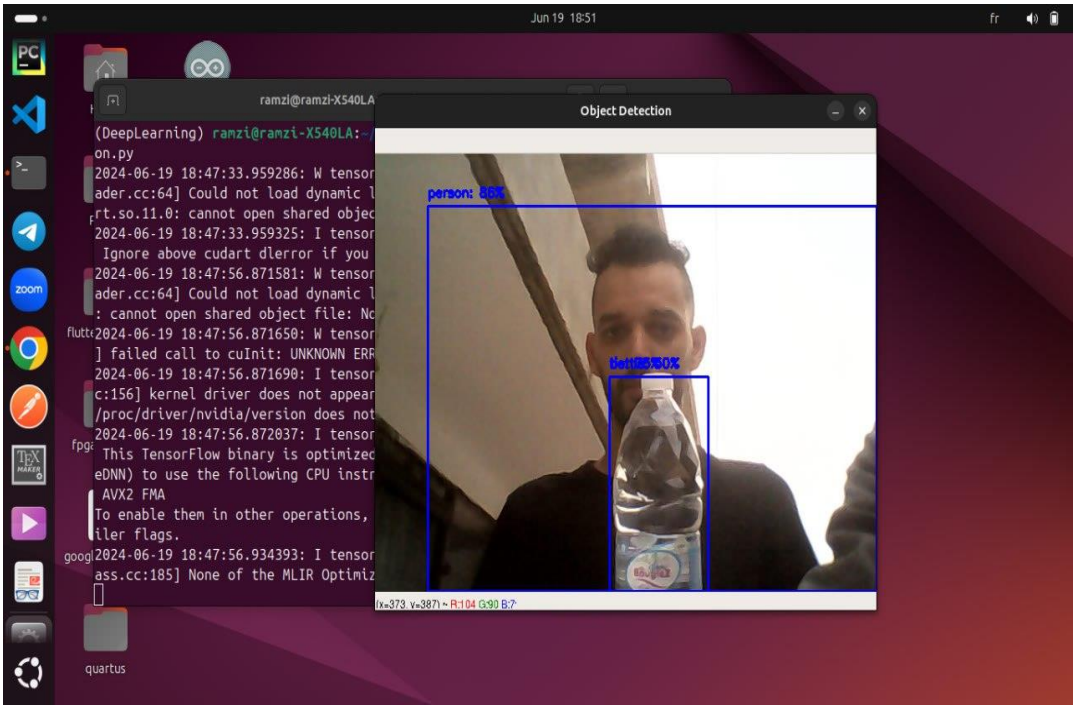



Figure 27 Test 2

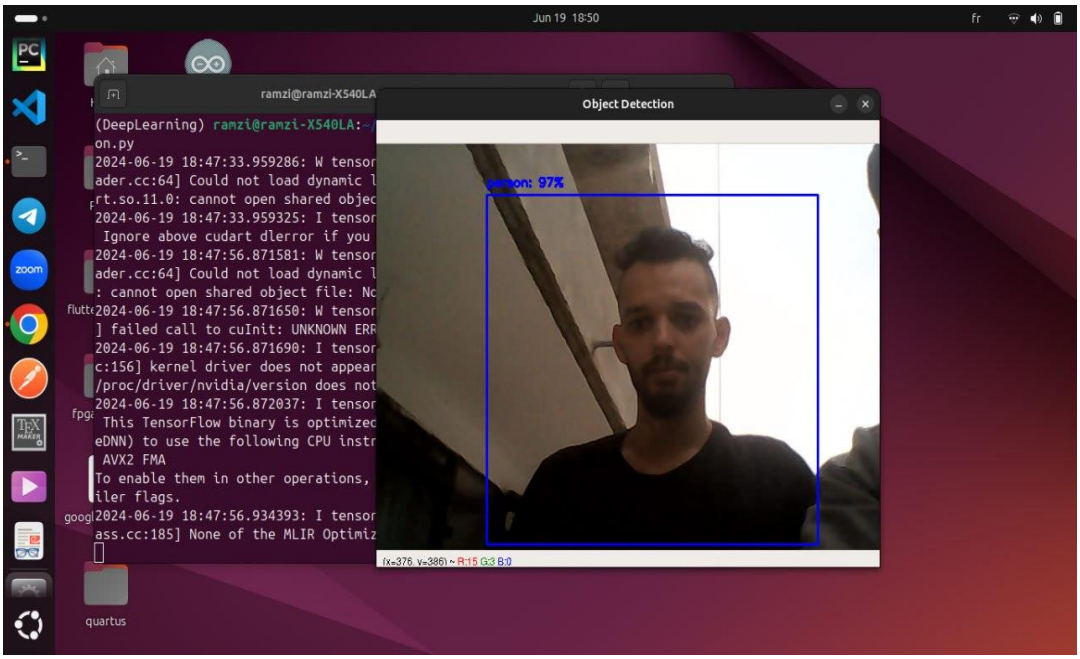



Figure 28 Test 3

# 7. Conclusion

Ce chapitre marque l'aboutissement de notre mise en œuvre pour la détection d'objets en temps réel, mettant en lumière nos choix stratégiques et les résultats obtenus. Nous avons soigneusement sélectionné nos outils de développement, exploitant les ressources avancées de plates-formes comme Kaggle pour optimiser nos modèles avec des accélérateurs GPU. À travers une analyse approfondie de différents modèles, nous avons identifié une architecture équilibrée qui a démontré une bonne capacité de généralisation et de performance. Enfin, notre interface utilisateur offre une expérience intuitive et informative, permettant une interaction directe avec les résultats de détection en temps réel. Ce travail reflète notre engagement à intégrer la théorie à la pratique, fournissant un système robuste et efficace pour la détection d'objets, avec des implications potentielles importantes pour la vision par ordinateur et l'apprentissage automatique.

# Conclusion générale et perspectives

En conclusion, ce mémoire a exploré de manière approfondie les défis cruciaux et les avancées significatives dans le domaine de la détection d'objets en vision par ordinateur. La détection d'objets joue un rôle essentiel dans des secteurs variés tels que la sécurité, l'automobile, la médecine et l'industrie, où la précision et la fiabilité des systèmes de détection sont des impératifs pour garantir des opérations fluides et sécurisées.

Notre démarche a été guidée par la nécessité de surmonter les limitations actuelles en exploitant les capacités avancées du Deep Learning, en particulier à travers les Réseaux de Neurones Convolutifs (CNN) et l'architecture MobileNetV2. Cependant, le chemin vers le développement d'un système de détection d'objets robuste et précis n'a pas été sans obstacles.

Durant ce projet, nous avons rencontré plusieurs défis majeurs. Tout d'abord, la collecte et le prétraitement des données ont constitué une étape critique, nécessitant une méthodologie rigoureuse pour assurer la qualité et la représentativité des jeux de données utilisés. La diversité des environnements et des conditions de capture des images ont également posé des défis pour la généralisation du modèle, exigent des ajustements continus et une adaptation fine des algorithmes.

Par ailleurs, l'optimisation des performances du modèle, tant en termes de précision que d'efficacité énergétique, a demandé des efforts considérables. La sélection et la configuration des hyperparamètres, ainsi que l'optimisation des architectures CNN adaptées à MobileNetV2, ont nécessité une exploration approfondie et itérative.

Malgré ces défis, les résultats obtenus au cours de nos expérimentations ont validé l'efficacité de notre méthodologie, avec des améliorations substantielles en termes de précision et de robustesse par rapport aux approches conventionnelles. Cependant, notre étude n'est pas sans limites ; elle dépend encore sensiblement des conditions environnementales et des variations des données, des aspects sur lesquels des efforts continus doivent être concentrés pour une application pratique étendue.

Pour l'avenir, plusieurs pistes d'amélioration prometteuses se dessinent. Nous envisageons notamment d'explorer plus avant les opportunités offertes par la fusion de données multi-modales, afin d'augmenter encore la précision et l'adaptabilité du système. De plus, l'optimisation continue des architectures de réseaux neuronaux, couplée à l'application de

techniques avancées d'augmentation de données, pourrait renforcer de manière significative la robustesse et la généralisabilité de notre approche.

En conclusion, ce travail représente une contribution substantielle à l'évolution des systèmes autonomes et à l'amélioration des capacités de détection d'objets dans des environnements complexes et variés malgré les défis rencontrés. Nous sommes convaincus que les avancées continues en Deep Learning et en vision par ordinateur ouvriront de nouvelles perspectives, non seulement pour résoudre les défis actuels mais également pour stimuler de nouvelles applications innovantes dans divers domaines industriels et technologiques.

Ce mémoire a non seulement consolidé nos connaissances théoriques et pratiques, mais il a également mis en lumière l'importance de l'innovation continue et de la collaboration interdisciplinaire pour repousser les limites de la technologie. À travers ce travail, nous espérons inspirer et guider les futures recherches visant à perfectionner les systèmes de détection d'objets, contribuant ainsi à une société plus sûre, plus intelligente et plus efficace grâce aux avancées de la vision par ordinateur.

# Bibliographie


[1]. **Forsyth, David A and Ponce, Jean.** *Computer vision: a modern approach.* s.l. : prentice hall professional technical reference, 2002.

[2]. **Sonka, Milan and Hlavac, Vaclav and Boyle, Roger.** *Image processing, analysis and machine vision.* s.l. : Springer, 2013.

[3]. **Szeliski, Richard.** *Computer vision: algorithms and applications.* s.l. : Springer Nature, 2022.

[4]. Actions in context. [auteur du livre] Marcin and Laptev, Ivan and Schmid, Cordelia Marszalek. *2009 IEEE Conference on Computer Vision and Pattern Recognition.* s.l. : IEEE, 2009, pp. 2929--2936.

[5]. *Ian; Bengio, Yoshua; Courville.* **Goodfellow, A.** s.l. : MIT Press, 2016.

[6]. **Burger, Wilhelm and Burge, Mark.** *Digital image processing: An algorithmic introduction.* s.l. : Springer Nature, 2022.

[7]. **Prince, Simon JD.** *Computer vision: models, learning, and inference.* s.l. : Cambridge University Press, 2012.

[8]. **Lin, Tsung-Yi and Doll{\'a}r, Piotr and Girshick, Ross and He, Kaiming and Hariharan, Bharath and Belongie, Serge.** Feature pyramid networks for object detection. *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2017, pp. 2117--2125.

[9]. *Faster r-cnn: Towards real-time object detection with region proposal networks.* **Ren, Shaoqing and He, Kaiming and Girshick, Ross and Sun, Jian.** 2015, Advances in neural information processing systems, Vol. 28.

[10]. **Redmon, Joseph and Divvala, Santosh and Girshick, Ross and Farhadi, Ali.** You only look once: Unified, real-time object detection. *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2016, pp. 779--788.

[11]. **Liu, Wei and Anguelov, Dragomir and Erhan, Dumitru and Szegedy, Christian and Reed, Scott and Fu, Cheng-Yang and Berg, Alexander C.** Ssd: Single shot multibox detector. *Computer Vision--ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11--14, 2016, Proceedings, Part I 14.* s.l. : Springer, 2016, pp. 21--37.

[12]. **Viola, Paul and Jones, Michael.** Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001.* s.l. : Ieee, 2001, Vol. 1, pp. I--I.

[13]. **Dalal, Navneet and Triggs, Bill.** Histograms of oriented gradients for human detection. [auteur du livre] 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05). s.l. : Ieee, 2005, Vol. 1, pp. 886--893.

[14]. *Object detection with discriminatively trained part-based models.* **Felzenszwalb, Pedro F and Girshick, Ross B and McAllester, David and Ramanan, Deva.** 9, s.l. : IEEE, 2009, IEEE transactions on pattern analysis and machine intelligence, Vol. 32, pp. 1627-1645.

[15]. *Distinctive image features from scale-invariant keypoints.* **Lowe, David G.** s.l. : Springer, 2004, International journal of computer vision, Vol. 60, pp. 91--110.

[16]. **Bay, Herbert and Tuytelaars, Tinne and Van Gool, Luc.** Surf: Speeded up robust features. *Computer Vision--ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9.* s.l. : Springer, 2006, pp. 404-417.

[17]. *A tutorial on support vector machine for pattern recognition.* **Burges, Christopher JC.** 2, 1998, Data mining and knowledge discovery, Vol. 2, pp. 955--974.

[18]. *An introduction to convolutional neural networks.* **O'shea, Keiron and Nash, Ryan.** 2015, arXiv preprint arXiv:1511.08458.

[19]. *Deep lerrning.* **LeCun, Yann and Bengio, Yoshua and Hinton, Geoffrey.** 7553, s.l. : Nature Publishing Group UK London, 2015, nature, Vol. 521, pp. 436--444.

[20]. *Learning representations by back-propagating errors.* **Rumelhart, David E and Hinton, Geoffrey E and Williams, Ronald J.** 6088, s.l. : Nature Publishing Group UK London, 1986, nature, Vol. 323, pp. 533--536.

[21]. **Saha, Sumit.** medium. *medium.com.* [En ligne] 15 Dec 2018. [Citation : 18 05 2023.] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[22]. **Boureau, Y-Lan and Ponce, Jean and LeCun, Yann.** A theoretical analysis of feature pooling in visual recognition. *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010, pp. 111--118.

[23]. **Prabhu, R.** *Understanding of Convolutional Neural Network (CNN) Deep Learning Medium}*. 2018. Com.

[24]. *Dropout: a simple way to prevent neural networks from overfitting.* **Srivastava, Nitish and Hinton, Geoffrey and Krizhevsky, Alex and Sutskever, Ilya and Salakhutdinov, Ruslan.** 1, s.l. : JMLR. org, 2014, The journal of machine learning research, Vol. 15, pp. 1929--1958.

[25]. *Imagenet classification with deep convolutional neural networks.* **Krizhevsky, Alex and Sutskever, Ilya and Hinton, Geoffrey E.** 2012, Advances in neural information processing systems, Vol. 25.

[26]. **Maas, Andrew L and Hannun, Awni Y and Ng, Andrew Y and others.** Rectifier nonlinearities improve neural network acoustic models. *Proc. icml.* s.l. : Atlanta, GA, 2013, Vol. 30, 1, p. 3.

[27]. *Gradient-based learning applied to document recognition.* **LeCun, Yann and Bottou, L{\'e}on and Bengio, Yoshua and Haffner, Patrick.** 11, s.l. : Ieee, 1998, Proceedings of the IEEE}, Vol. 86, pp. 2278--2324.

[28]. *Very deep convolutional networks for large-scale image recognition.* **Simonyan, Karen and Zisserman, Andrew.** 2014, arXiv preprint arXiv:1409.1556.

[29]. **Hassan, Muneeb ul.** *VGG16 – Convolutional Network for Classification and Detection.neurohive.* [En ligne] 2018. [Citation : 12 06 2024.] https://neurohive.io/en/popular?networks/vgg16/..

[30]. **He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian.** Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2016, pp. 770--778.

[31]. **Szegedy, Christian and Liu, Wei and Jia, Yangqing and Sermanet, Pierre and Reed, Scott and Anguelov, Dragomir and Erhan, Dumitru and Vanhoucke, Vincent and Rabinovich, Andrew.** Going deeper with convolutions. *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2015, pp. 1--9.

[32]. *Illustrated: 10 cnn architectures.* **Karim, Raimi.** 2019, towards data science.

[33]. **Sandler, Mark and Howard, Andrew and Zhu, Menglong and Zhmoginov, Andrey and Chen, Liang-Chieh.** Mobilenetv2: Inverted residuals and linear bottlenecks}. *Proceedings of the IEEE conference on computer vision and pattern recognition}.* 2018, pp. 4510--4520.

[34]. *Adam: A method for stochastic optimization.* **{Kingma, Diederik P and Ba, Jimmy.** 2014, arXiv preprint arXiv:1412.6980.

[35]. *Improving neural networks by preventing co-adaptation of feature detectors.* **Hinton, Geoffrey E and Srivastava, Nitish and Krizhevsky, Alex and Sutskever, Ilya and Salakhutdinov, Ruslan R.** 2012, arXiv preprint arXiv:1207.0580}.

[36]. *A survey on transfer learning.* **Pan, Sinno Jialin and Yang, Qiang}.** 19, s.l. : IEEE, 2009, IEEE Transactions on knowledge and data engineering, Vol. 22, pp. 1345--1359.

[37]. **Girshick, Ross and Donahue, Jeff and Darrell, Trevor and Malik, Jitendra.** ich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2014, pp. 580--587.

[38]. *Deep inside convolutional networks: Visualising image classification models and saliency maps.* **Simonyan, Karen and Vedaldi, Andrea and Zisserman, Andrew.** 2013, arXiv preprint arXiv:1312.6034.

[39]. **Burger, Wilhelm and Burge, Mark James and Burge, Mark James and Burge, Mark James.** *Principles of digital image processing.* s.l. : Springer, 2009. Vol. 111.

[40]. *Imagenet large scale visual recognition challenge.* **Russakovsky, Olga and Deng, Jia and Su, Hao and Krause, Jonathan and Satheesh, Sanjeev and Ma, Sean and Huang, Zhiheng and Karpathy, Andrej and Khosla, Aditya and Bernstein, Michael and others.** s.l. : Springer, 2015, International journal of computer vision, Vol. 115, pp. 211--252.

[41]. *The pascal visual object classes (voc) challenge.* **{Everingham, Mark and Van Gool, Luc and Williams, Christopher KI and Winn, John and Zisserman, Andrew.** s.l. : Springer, 2010, International journal of computer vision, Vol. 88, pp. 303--338.

[42]. **Szegedy, Christian and Vanhoucke, Vincent and Ioffe, Sergey and Shlens, Jon and Wojna, Zbigniew.** Rethinking the inception architecture for computer vision. *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2016, pp. 2818--2826.

[43]. *Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs.* **Chen, Liang-Chieh and Papandreou, George and Kokkinos, Iasonas and Murphy, Kevin and Yuille, Alan L.** 4, s.l. : IEEE, 2017, IEEE transactions on pattern analysis and machine intelligence, Vol. 40, pp. 834--848.

44. [En ligne]

[45]. **Microsoft.** *Visual Studio Code Documentation.* [En ligne] https://code.visualstudio.com/docs.

[46]. **Abadi, Mart{\'\i}n and Barham, Paul and Chen, Jianmin and Chen, Zhifeng and Davis, Andy and Dean, Jeffrey and Devin, Matthieu and Ghemawat, Sanjay and Irving, Geoffrey and Isard, Michael and others.** $TensorFlow$\}$: a system for $\{$Large-Scale$\}$ machine learning. *12th USENIX symposium on operating systems design and implementation (OSDI 16).* 2016, pp. 265--283.

[47]. **Bradski, Gary and Kaehler, Adrian.** *Learning OpenCV: Computer vision with the OpenCV librar.* s.l. : " O'Reilly Media, Inc.", 2008.

[48]. *Keras: The python deep learning library.* **Chollet, Fran{\c{c}}ois and others.** 11, s.l. : Miller Freeman Inc., 2018, Astrophysics source code library, Vol. 25, pp. ascl--1806.

[49]. *Python.* **Python, Why.** s.l. : Citeseer, 2021, Python releases for windows, Vol. 24.

[50]. **He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian.** Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2016, pp. 770--778.