Résumé

Dans les dernières années avec le boom de data due à l'explosion de l'internet au monde, et l'avancement technologique des calculateurs, l'intelligence artificielle ne cesse d'accroitre en sens exponentiel dans tous les champs y compris transport, médecine et agriculture ... etc. La reconnaissance d'image l'un des nouvelles technologies de AI en plein d'évolution, où est déjà utilisé dans les voitures autonomes pour détecter et classifier les objets dans la route avec haute précision à partir de l'apprentissage d'un Algorithme par un ensemble d'image de routes. En revanche, l'apprentissage des algorithmes nécessite un nombre massif de data, qui est un autre challenge pour les développeurs. A cet égard, nous avons proposé un système de reconnaissance de fruit pourri pour l'implanter dans une machine de tri de légumes ou fruits. L'Algorithme proposé est basé sur les réseaux de neurone convolutionel comme un outil puissant de l'apprentissage automatique. Les résultats de notre travail sont un prototype de machine de tri, il a été travaillé automatiquement.

Mots clés : Triage, Automatique, Agriculture, Intelligence artificielle, MobileNet, Jetson Nano

Abstract

In the last few years with the data boom due to the explosion of the internet in the world, and the technological advancement of computers, artificial intelligence keeps increasing in exponential direction in all fields including transportation, medicine and agriculture ... etc. Image recognition is one of the new AI technologies in full evolution, where is already used in autonomous cars to detect and classify objects in the road with high accuracy from learning an Algorithm by a set of road images. On the other hand, learning algorithms requires a massive amount of data, which is another challenge for developers. In this respect, we have proposed a rotten fruit recognition system to be implemented in a vegetable or fruit sorting machine. The proposed algorithm is based on convolutional neural networks as a powerful tool in machine learning. The results of our work are a prototype of sorting machine, it has been worked automatically.

Keywords : Sorting, Automatic, Agriculture, Artificial Intelligence, MobileNet, Jetson Nano

الملخص

.في السنوات القليلة الماضية مع ازدهار البيانات بسبب انفجار الإنترنت في العالم ، والتقدم التكنولوجي لأجهزة الكمبيوتر ، استمر الذكاء الاصطناعي في الزيادة في الاتجاه الأسي في جميع المجالات بما في ذلك النقل والطب والزراعة ... إلخ. التعرف على الصور هي إحدى تقنيات الذكاء الاصطناعي الجديدة في تطور كامل ، حيث يتم استخدامها بالفعل في السيارات ذاتية القيادة لاكتشاف وتصنيف الأشياء على الطريق بدقة عالية من تعلم خوارزمية من خلال مجموعة من صور الطريق. من ناحية أخرى ، تتطلب خوارزميات التعلم قدرًا هائلاً من البيانات ، وهو تحدٍ آخر للمطورين. في هذا الصدد ، اقترحنا نظام التعرف على الفاكهة الفاسدة ليتم تنفيذه في آلة فرز الخضار أو الفاكهة. تعتمد الخوارزمية المقترحة على الشبكات العصبية التلافيفية كأداة قوية في التعلم الآلي. نتائج عملنا عبارة عن نموذج أولي لآلة الفرز ، تم تشغيلها آليًا.

الكلمات المفتاحية: فرز ، آلي ، زراعة ، ذكاء إصطناعي ، موبايل نت ، جيتسون نانو

Remerciements

Mes remerciements vont d'abord à الله Tout-Puissant pour la volonté, la santé, et la patience, qu'il m'a accordé durant toutes ces années d'études.

J'exprime ma profonde gratitude à mes parents pour leurs encouragements, leur soutien et pour les sacrifices qu'ils ont endurés, sans oublier de remercier ma famille pour leur soutien et surtout mes tantes et leurs maris.

Je remercie également mon Institut de Maintenance et de Sécurité Industrielle qui m'a donné l'opportunité de passer ce stage pour mieux exploiter mes connaissances dans l'intérêt de ma formation et de ma qualification professionnelle.

Ainsi je voudrais remercier toutes les personnes qui m'ontaidé, de près ou de loinà accomplir ce travail surtout mon encadreur Mr. Benarbia Taha qui a su suivre et diriger mon projet de fin d'étude, ses conseils particuliers et ses critiques qui ont été pour moi un encouragement permanent.

Je tiens également à remercier mon équipe « Automafy » ilies BOUROUH, Hichem Ahmed MAHBOUBI et Dr.Mohamed BRAHIMI.

Enfin je tiens à exprimer ma gratitude à tous mes amis et collègues tous ceux qui ont une bonne impression dans mon cœur, Comme des signes d'amour et de respect.

Younes LABRI

Dédicace

I dedicatethis work

To my dear parents, for all their sacrifices, theirlove, their support and their prayers throughout my studies,

To my dear sister for their constant encouragements and their moral support,

To my dear brothers for their support and encouragement,

To my dear aunt for their support and encouragement,

To my dear family for their support throughout my university

To my support “Amel Meriem BENARBIA”

To dear brothers from D53:”Salah,Nadir,Mouad,Ayoub,Mokhtar,Youcef B, Youcef ,Moahmed lay,Tifour,Belckacem,Zedek”

To my team Automafy “ilies BOUROUH, Moahmed BRAHIMI, Tarik”

To my best friends from Oran “Chafia CHAKER,Lotfi,Ilyes Djennane”

To my dear 2 end family “MasterMinds-Club” for motivation, support and encouragement,

May this work be the fulfillment of your long-awaited wishes and the leak of your unfailing support !!

Thank you for always being there for me.

Younes LABRI

Sommaire

Chapitre II : Généralités sur intelligence artificielle deep learning and Convolutionnal Neural Networks

Chapitre III : Environnement de travail

Liste des Figures

Liste des abréviations

AI : artificial intelligence

CNN : convolutional neural network

RGB : Red, Green, Bleu

Introduction générale

Dans une interview avec l'exportateur, Abdel Razzaq Mokrani, sur El-Badil TV[1], il a déclaré : "Le problème est qu'ici en Algérie, l'agriculteur vous montre une marchandise et vous trouvez une qualité différente plus tard". En fait, ce n'est pas un problème individuel pour Abdel Razzaq seulement, en 2019, beaucoup de marchandises ont été rejetées pour ne pas suivre les normes d'exportation fixées par les pays de destination.

En effet, le processus de contrôle de la qualité est encore très manuel, ce qui empêche de nombreux exportateurs comme Mokrani de développer leur activité.

Le processus actuel de triage pour le marché des fruits est encore manuel en Algérie. La plupart des solutions disponibles sur le marché nécessitent de grandes et coûteuses machines, qui ne sont pas adaptées aux acteurs algériens. Néanmoins, les normes de qualité en Algérie ont une marge de progression pour être compétitives sur les marchés internationaux des dattes et des produits transformés à base de tomates, entre autres. Cela passe par d'énormes améliorations du processus de triage ! L'automatisation est largement utilisée sur d'autres marchés, ce qui offre un avantage concurrentiel sur les acteurs algériens ; l'automatisation est donc désormais une question de vie ou de mort.

Comme la solution ce projet aidera les agriculteurs à résoudre leurs problèmes. Ce projet utilise la vision par ordinateur, intelligence artificielle et l'apprentissage profond, formés sur un ensemble de données de biopsie proprement exécuté sur un ordinateur monocarte pour faire progresser le diagnostic assisté par ordinateur qui aiderait les agriculteurs à trier les fruits.

L’agriculteur mettrait les fruits dans le système, qui les trierait automatiquement pour le fermier, les triant en fruits frais ou fruits pourris.

Structure du mémoire

Ce mémoire se présente sous forme de cinq chapitres :

Le chapitre 1 nous passerons en revue, des notions théoriques en commençant par la vision, s'étendant aux traitements d'image et enfin aux visions par ordinateur.

Le chapitre 2 nous présente un aperçu général sur l’intelligenceartificielle et du Deep Learning.

Le chapitre 3 et 4 illustrent l’implémentation et l’expérimentation de notre système, les outils et logiciels que nous avons eu à utiliser pour le développement, et la réalisation de notre système.

Le chapitre 5 nous parlerons du résultat de notre système.

Et enfin, nous terminerons ce mémoire par une conclusion générale.

Chapitre I :

Vision par ordinateur
(Computer Vision)

I.1 INTRODUCTION

La vision par ordinateur est l'un des sous-domaines les plus en vogue de l'intelligence artificielle et de l'apprentissage automatique, étant donné sa grande variété d'applications et son énorme potentiel. Son objectif : reproduire les puissantes capacités de la vision humaine.

Mais, qu'est-ce que la vision par ordinateur exactement ? Comment est-elle actuellement appliquée dans différents secteurs ? Quels sont les cas d'utilisation commerciale les plus connus ?

Dans ce chapitre, nous allons présenter le concept de base de la vision par ordinateur et la manière dont elle est utilisée dans le monde réel.

I.2 DEFINITION DE LA VISION

Qu'il s'agisse d'un ordinateur ou d'un être humain, la vision se résume à deux composantes :

- Tout d'abord, un dispositif de détection capte le plus de détails possibles d'une image. L'œil capte la lumière qui traverse l'iris et la projette sur la rétine, où des cellules spécialisées transmettent les informations au cerveau par l'intermédiaire des neurones. Une caméra capture les images de manière similaire et transmet les pixels à l'ordinateur. Dans cette partie, les caméras sont meilleures que les humains car elles peuvent voir dans l'infrarouge, voir plus loin ou avec plus de précision.
- Ensuite, le dispositif d'interprétation doit traiter les informations et en extraire le sens. Le cerveau humain résout cette tâche en plusieurs étapes dans différentes régions du cerveau.[2]

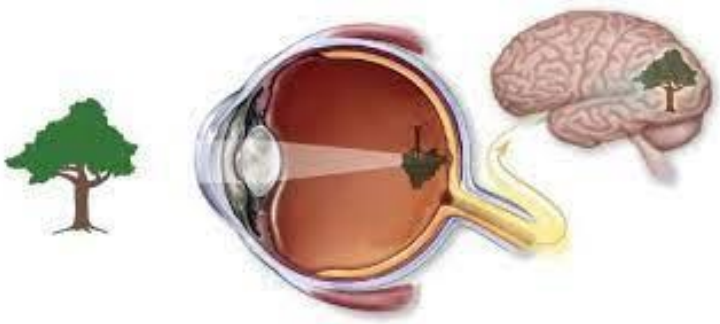


Figure I-1 Vision Human

I.3 TRAITEMENT IMAGE

Le traitement d'images numériques est l'utilisation d'un ordinateur pour traiter des images numériques à travers un algorithme. En d'autres termes, le traitement d'images est l'ensemble des méthodes permettant d'effectuer certaines opérations sur une image, afin d'obtenir une image améliorée (suppression des erreurs) ou d'en extraire des informations utiles.[3]

I.3.1 IMAGE ET PIXEL

L'image est définie comme étant une fonction f (x, y) à deux dimensions, où x et y sont les coordonnées spatiales, et l'amplitude à tous points (x, y) correspondant à l'intensité ou au niveau de gris.

Une image numérique est constituée d'un ensemble de points appelés pixels (abréviation venant de l'anglais : Picture Element). Les pixels sont approximativement rectangulaires, parfois carrés. Leur taille peut être modifiée en ajustant l'écran ou la carte graphique, le pixel est donc le plus petit composant d'une image numérique. L'ensemble de ces pixels est contenu dans un tableau bidimensionnel constituant l'image.[4]

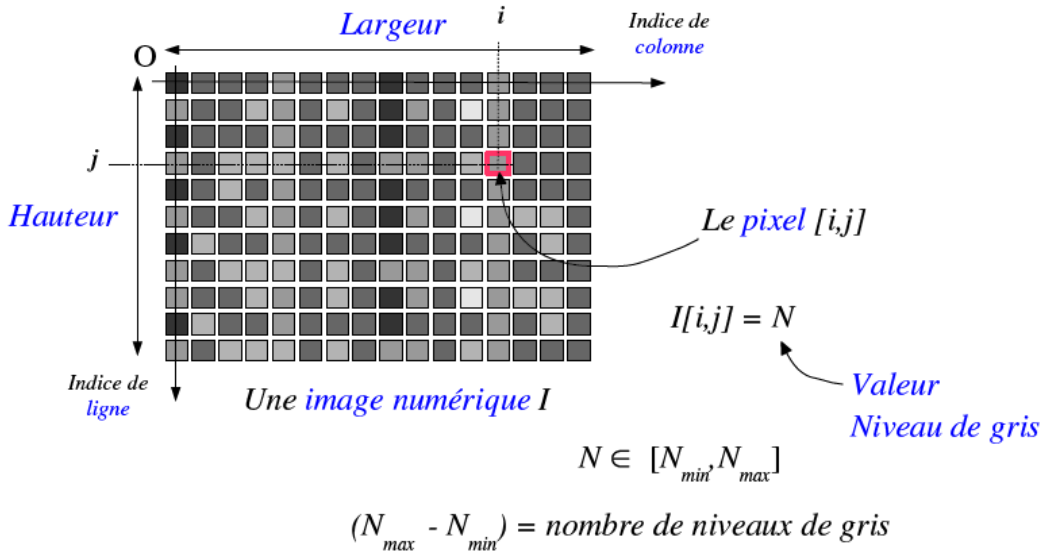


Figure I-2Explication des pixels

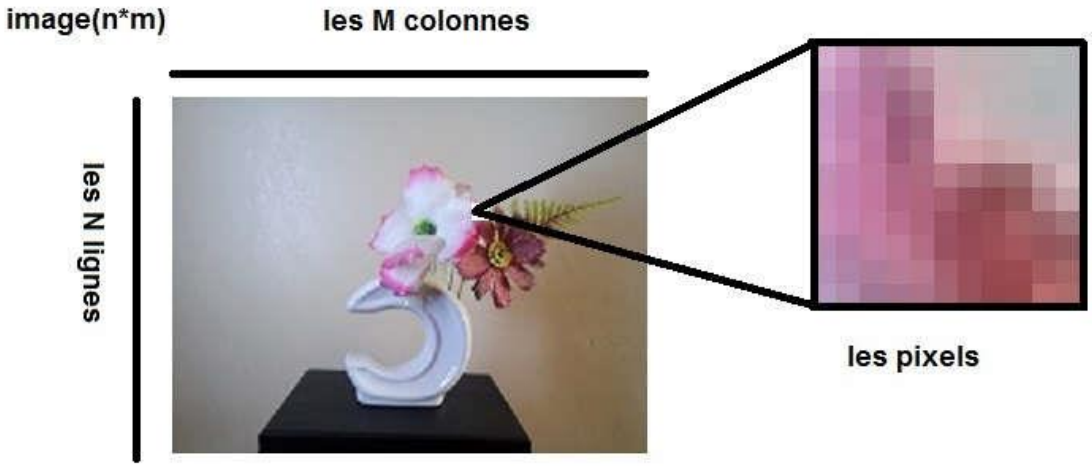


Figure I-3Représentation des notions d'image et pixel (c) Représentation matricielle (d) Image réelle

I.3.2 RESOLUTION D'UNE IMAGE

La résolution d'une image est le nombre de pixels par pouce qu'elle contient (1 pouce = 2.54 centimètres). Elle est exprimée en "PPP" (points par pouce) ou DPI (dots per inch). Plus il y a de pixels (ou points) par pouce et plus il y aura d'information dans l'image (plus précise). Par exemple, une résolution de 300dpi signifie que l'image comporte 300 pixels dans sa largeur et 300 pixels dans sa hauteur, elle est donc composée de 90 000 pixels (300x300 ppp). Grâce à cette formule, il est facile de connaître la dimension maximale d'un tirage.[5]

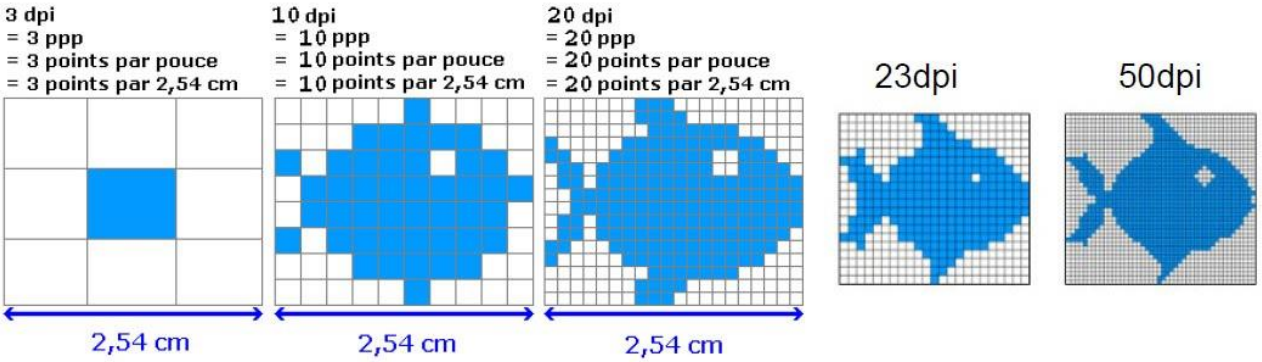


Figure I-4 Exemples de résolution d'une image

I.3.3 HISTOGRAMME D'UNE IMAGE

L'histogramme des niveaux de gris ou des couleurs d'une image est une fonction qui donne la fréquence d'apparition de chaque niveau de gris (couleur) dans l'image. Pour comparer deux images obtenues sous des éclairages différents, pour mesurer certaines propriétés sur une image, on modifie souvent l'histogramme correspondant. Il permet de donner un grand nombre d'information sur la distribution des niveaux de gris (couleur) et de voir entre quelles bornes est repartie la majorité des niveaux de gris (couleur) dans les cas d'une image trop claire ou d'une image trop foncée. Il peut être utilisé pour améliorer la qualité d'une image (Rehaussement d'image) en introduisant quelques modifications, pour pouvoir extraire les informations utiles de celle-ci.[6]

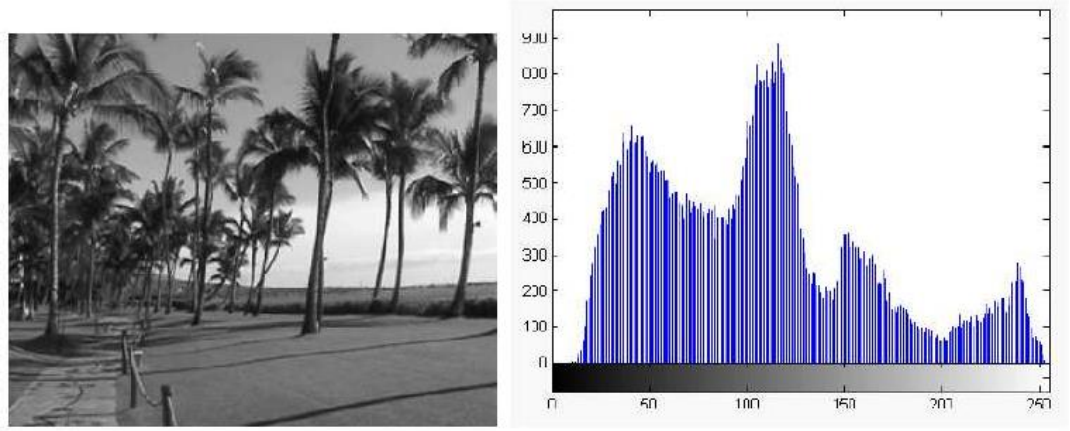


Figure I-5 L'image et son histogramme Gray

I.4 TYPES DES IMAGES

Nous distinguons trois types d'images :

I.4.1 IMAGE BINAIRE

Une image binaire est une image numérique qui n'a que deux valeurs possibles pour chaque pixel. Généralement, les deux couleurs utilisées pour une image binaire sont le noir et le blanc.[5]

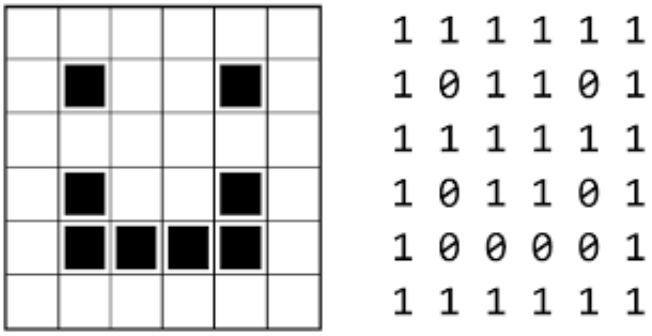


Figure I-6 Exemple d'une image binaire

I.4.2 IMAGE EN NIVEAUX DE GRIS

Chaque pixel est un niveau de gris, allant de 0 (noir) à 255 (blanc). Cet intervalle de valeurs signifie que chaque pixel est codé sur huit bits (un octet). 256 niveaux de gris sont généralement suffisants pour la reconnaissance de la plupart des objets d'une scène. Le nombre de pixels est calculé comme suit :[7]

$$N_{pixels} = N_{rows} \times N_{columns}$$

Figure I-7Exemple d'une image gray

I.4.3 IMAGE COULEUR (OU RED, GREEN, BLEU RGB)

Chaque pixel est codé sur 24 bits (3 octets) où chaque octet (8 bits) correspond à une des couleurs suivantes couleurs : Rouge(R), Vert(V), Bleu(B). L'image est considérée comme une liste de matrices. Le nombre de pixels de nombre de pixels est calculé comme suit :[7]

$$N_{pixels} = N_{rows} \times N_{columns} \times 3$$

Figure I-8 Exemple d'une image RGB

I.5 PRETRAITEMENT

L'image à traiter comporte une grande quantité de données qui est généralement bruitée par des pixels indésirables qui pourraient modifier l'information utile. Généralement, le bruit d'image est considéré comme un champ aléatoire centré additif qui peut provenir soit du dispositif d'acquisition (influences magnétiques…) soit de la scène elle-même (lumière parasite…).

Avant d'extraire les objets et d'analyser une image, il est donc souvent nécessaire d'améliorer l'image. Les techniques de prétraitements les plus courantes qu'on va présenter sont :

I.5.1 LA MODIFICATION D'HISTOGRAMME

Pour modifier les caractéristiques de l'image (accentuer les contrastes en général), une approche générale consiste à appliquer une fonction qui associe à chaque valeur d'intensité dans l'image une nouvelle valeur. Cette fonction va modifier l'histogramme de l'image[8]. L'idée est de modifier la répartition des niveaux pour obtenir un histogramme plat étendu à l'ensemble des valeurs possibles. Dans cette opération la dynamique originale [min, max] est étalée à [0, 255]. On cherche à effectuer le même nombre de pixels à chaque niveau de gris ; c'est pourquoi on appelle cette opération équi-population. L'effet obtenu permet de mieux séparer les valeurs les plus représentées dans l'image et de rapprocher les valeurs marginales. Cela a pour effet d'améliorer le contraste de l'image. Cette transformation peut aussi être appliquée pour réduire le nombre de niveaux (passer de 256 à 16 niveaux) pour

la visualisation ; elle s'apparente alors à une classification puisqu'il s'agit de représenter plusieurs valeurs initiales par la même valeur finale.[3]

I.5.2 SEGMENTATION

La segmentation est l'une des étapes critiques de traitement et de l'analyse d'images qui conditionne la qualité des mesures effectuées ultérieurement. Elle permet de cerner les formes des objets sur lesquels doit porter l'analyse, la bonne méthode de segmentation sera celle qui permettra d'arriver à une bonne interprétation. Elle devra donc avoir simplifié l'image sans pour autant en avoir trop réduit le contenu la Figure I.9.représente exemple de segmentation d'image., il y a des nombreuses méthodes de segmentation on peut citer la segmentation basée sur le seuillage.[3]

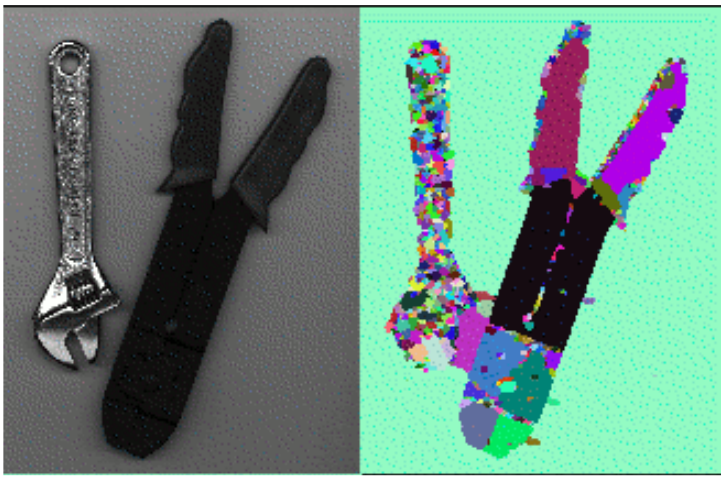


Figure I-9 Exemple de segmentation d'image

I.5.3 LA REDUCTION DU BRUIT PAR FILTRAGE

Il existe un grand nombre de filtres possibles, et à quelques exceptions près, on peut les classer en 2 grandes catégories : [9]

- Les filtres linéaires : la méthode de filtrage par moyenne est la plus utilisée à cause de la simplicité de son algorithme et la qualité de résultats qu'elle donne par rapport à d'autres filtres.
- Les filtres non linéaires, c'est le filtrage médian réalise un lissage de l'image un peu plus performant que le filtre moyenne en ce qui concerne les détails dans l'image.

I.6 VISION PAR ORDINATEUR (COMPUTER VISION)

La vision par ordinateur est le domaine de l'informatique qui se concentre sur la création de systèmes numériques capables de traiter, d'analyser et de donner un sens aux données visuelles (images ou vidéos) de la même manière que les humains. Le concept de vision par ordinateur repose sur l'apprentissage des ordinateurs à traiter une image au niveau du pixel et à la comprendre. Techniquement, les machines tentent de récupérer les

informations visuelles, de les traiter et d'interpréter les résultats grâce à des algorithmes logiciels spéciaux.[10]

I.6.1 COMMENT FONCTIONNE LA VISION PAR ORDINATEUR ?

La technologie de la vision par ordinateur tend à imiter le mode de fonctionnement du cerveau humain. Mais comment notre cerveau résout-il la reconnaissance visuelle des objets ? Selon l'une des hypothèses les plus répandues, notre cerveau s'appuie sur des motifs pour décoder les objets individuels. Ce concept est utilisé pour créer des systèmes de vision par ordinateur.

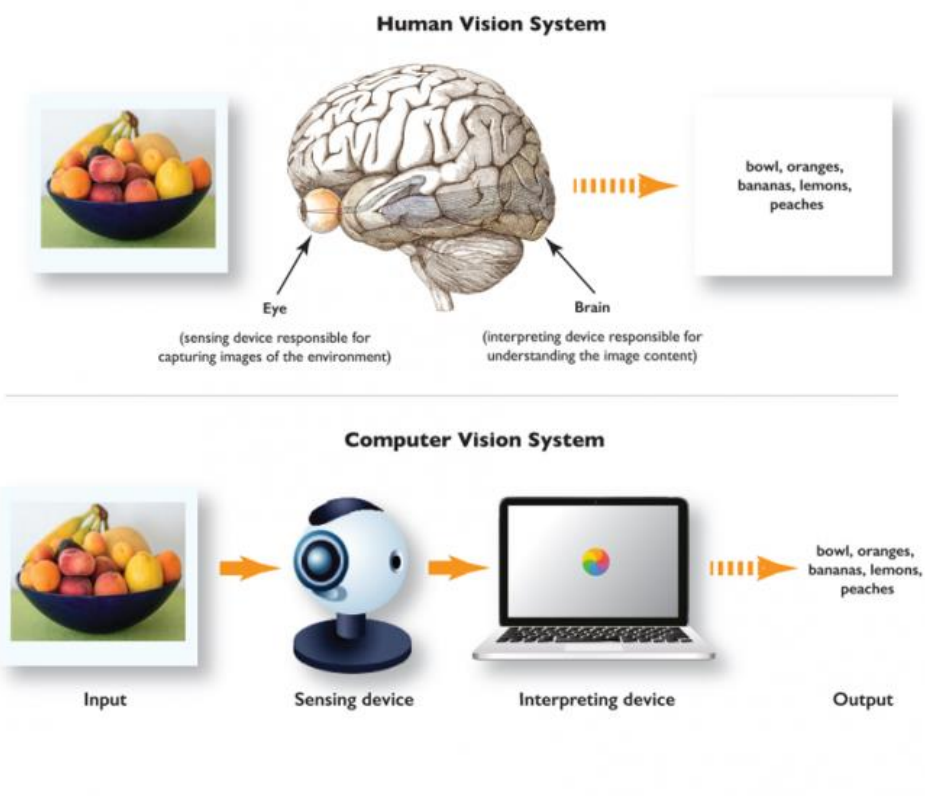


Figure I-10 Vision human et Vision par ordinateur

Les algorithmes de vision par ordinateur que nous utilisons aujourd'hui sont basés sur la reconnaissance des formes. Nous entraînons les ordinateurs sur une quantité massive de données visuelles - les ordinateurs traitent les images, étiquettent les objets qui s'y trouvent et trouvent des motifs dans ces objets. Par exemple, si nous envoyons un million d'images de fleurs, l'ordinateur les analysera, identifiera les motifs qui sont similaires à toutes les fleurs et, à la fin de ce processus, créera un modèle "fleur". Par conséquent, l'ordinateur sera capable de détecter avec précision si une image particulière est une fleur chaque fois que nous lui enverrons des photos.

Golan Levin, dans son article Traitement d'images et vision par ordinateur, fournit des détails techniques sur le processus que les machines suivent pour interpréter les images. En bref, les machines interprètent les images comme une série de pixels, chacun ayant son

propre ensemble de valeurs de couleur. Par exemple, vous trouverez ci-dessous une photo d'Abraham Lincoln. La luminosité de chaque pixel de cette image est représentée par un seul nombre de 8 bits, allant de 0 (noir) à 255 (blanc). Ces nombres sont ce que le logiciel voit lorsque vous saisissez une image. Ces données sont fournies en entrée à l'algorithme de vision par ordinateur qui sera chargé de l'analyse et de la prise de décision.[11]

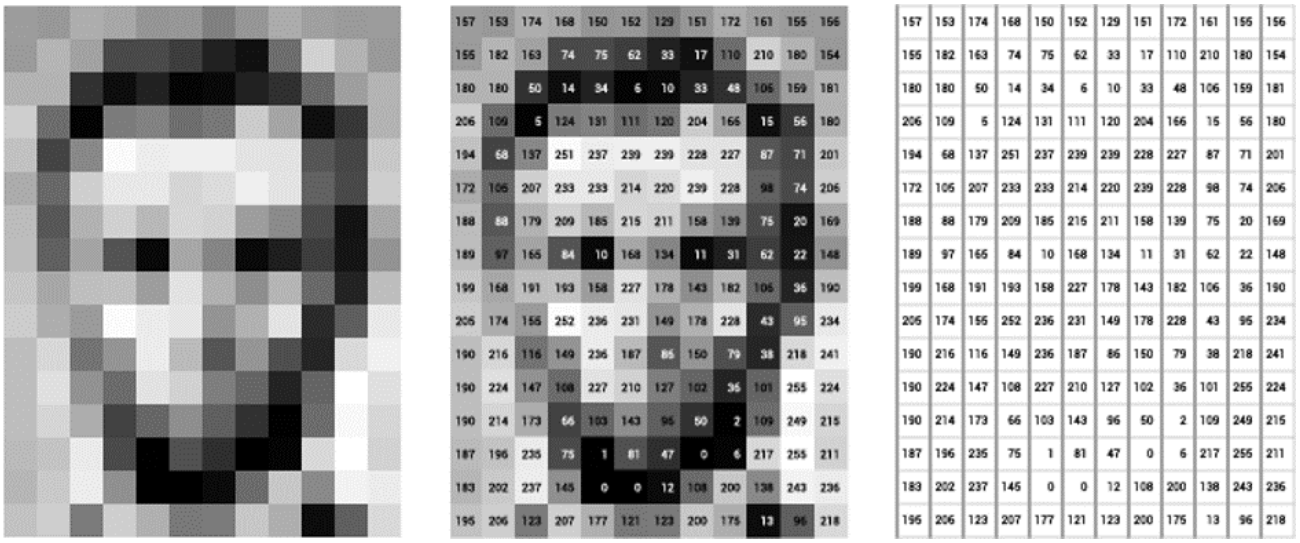


Figure I-11Abraham Lincoln

I.6.2 L'EVOLUTION DE LA VISION PAR ORDINATEUR

La vision par ordinateur n'est pas une technologie nouvelle ; les premières expériences dans ce domaine ont débuté dans les années 1950 et, à l'époque, elle était utilisée pour interpréter des textes dactylographiés et manuscrits. À l'époque, les procédures d'analyse de la vision par ordinateur étaient relativement simples mais exigeaient beaucoup de travail de la part des opérateurs humains qui devaient fournir manuellement des échantillons de données à analyser. Comme vous l'avez probablement deviné, il était difficile de fournir beaucoup de données en faisant cela manuellement. De plus, la puissance de calcul n'était pas suffisante, de sorte que la marge d'erreur pour cette analyse était assez élevée.

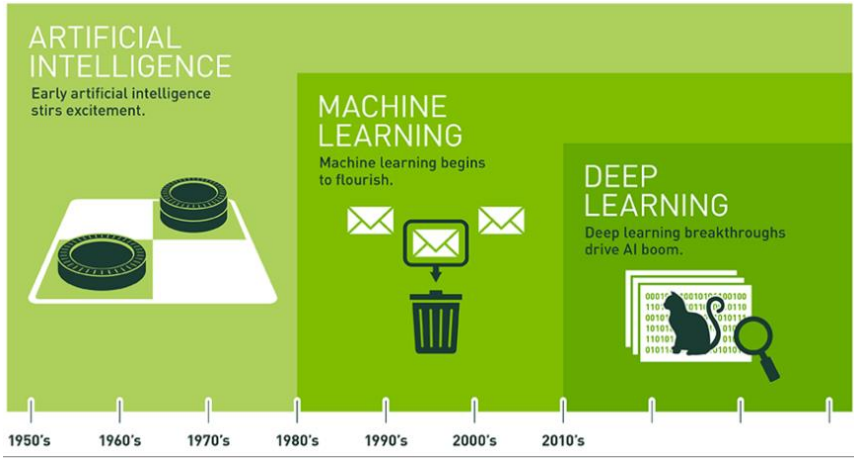


Figure I-12 I.6.2 L'évolution de la vision par ordinateur

I.7 APPLICATIONS DE LA VISION PAR ORDINATEUR :

I.7.1 SOINS DE SANTÉ

L'un des domaines d'application les plus importants est la vision informatique médicale, ou le traitement d'images médicales, qui se caractérise par l'extraction d'informations à partir de données d'images afin de diagnostiquer un patient. La détection de tumeurs, d'artériosclérose ou d'autres modifications malignes en est un exemple ; les mesures des dimensions des organes, du flux sanguin, etc. en sont un autre. Elle soutient également la recherche médicale en fournissant de nouvelles informations : par exemple, sur la structure du cerveau, ou sur la qualité des traitements médicaux. Les applications de la vision par ordinateur dans le domaine médical comprennent également l'amélioration des images interprétées par l'homme - images ultrasoniques ou radiographiques par exemple - afin de réduire l'influence du bruit.

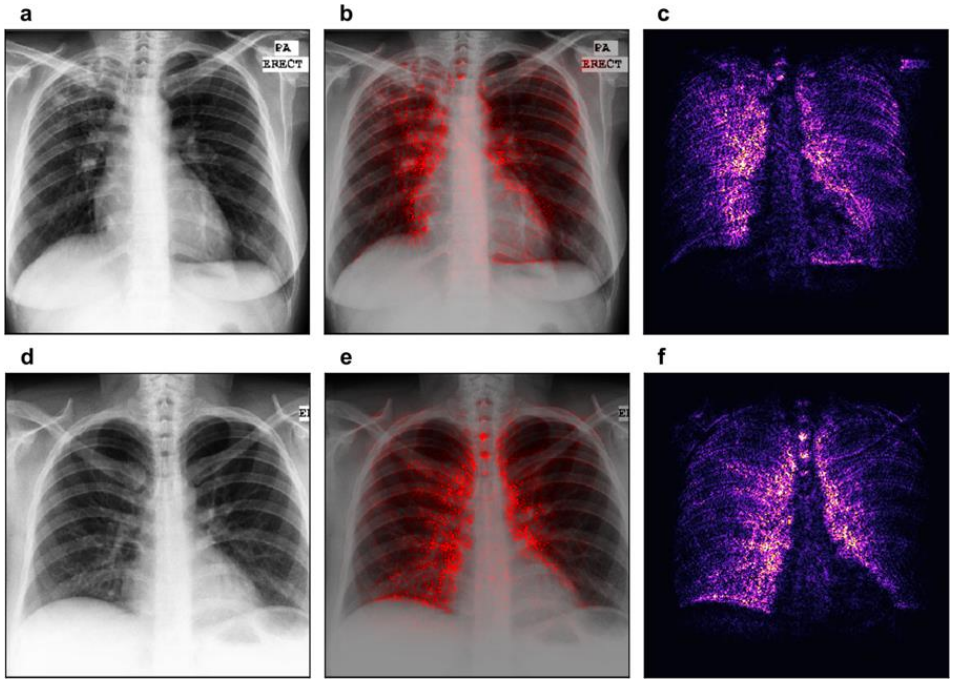


Figure I-13 Medical

I.7.2 AUTOMOBILE

Certaines des applications les plus célèbres de la vision par ordinateur ont été réalisées par Tesla avec sa fonction Autopilote

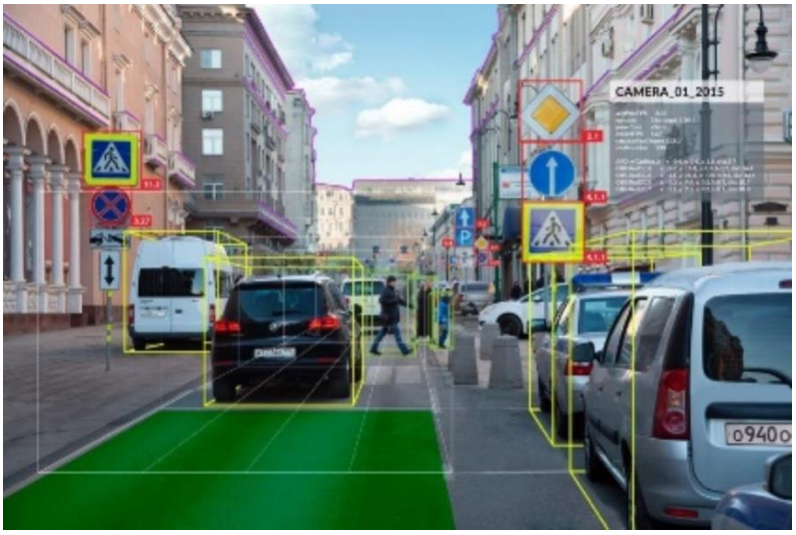


Figure I-14Exemple d'utilisation de la vision par ordinateur automobile

I.7.3 MILITAIRE

Les applications militaires constituent probablement l'un des plus grands domaines de la vision par ordinateur. Les exemples évidents sont la détection de soldats ou de véhicules ennemis et le guidage de missiles. Les systèmes plus avancés de guidage de missiles envoient le missile vers une zone plutôt que vers une cible spécifique, et la sélection de la cible est effectuée lorsque le missile atteint la zone, sur la base des données d'image acquises localement. Les concepts militaires modernes, tels que la "connaissance du champ de bataille", impliquent que divers capteurs, y compris les capteurs d'images, fournissent un riche ensemble d'informations sur une scène de combat qui peuvent être utilisées pour soutenir les décisions stratégiques. Dans ce cas, le traitement automatique des données est utilisé pour réduire la complexité et pour fusionner les informations provenant de plusieurs capteurs afin d'augmenter la fiabilité.[12]

Figure I-15 Exemple d'application militaire

I.7.4 AGRICULTURE

Utilisation potentielle de la vision par ordinateur dans l'agriculture. L'industrie agricole a bénéficié de plusieurs contributions de modèles de vision par ordinateur et d'intelligence artificielle (IA) dans des domaines tels que la plantation, la récolte, l'analyse avancée des conditions météorologiques, le désherbage et la détection et la surveillance de la santé des plantes.[13]

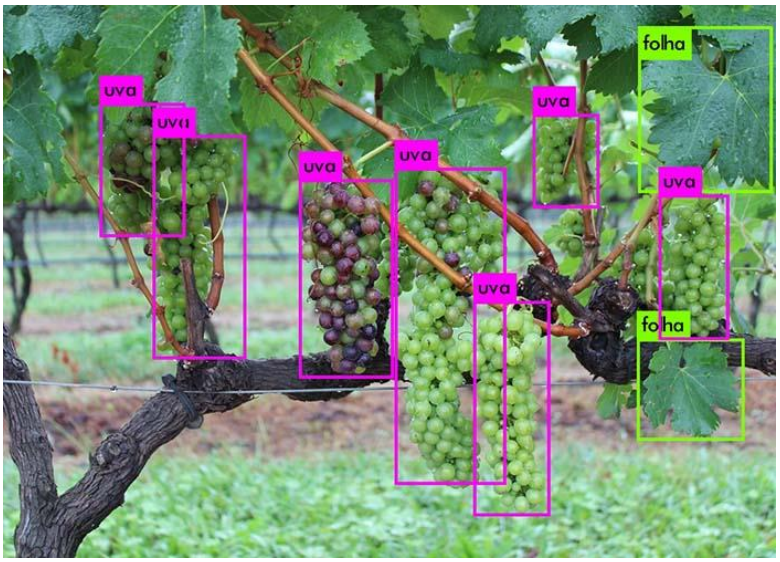


Figure I-16Exemple d'utilisation de la vision par ordinateurAgriculture

I.8 CONCLUSION

Dans ce chapitre, nous avons abordé un résumé des définitions et des concepts de base liés à l'image numérique représentée par sa forme matricielle. Il s'agissait des concepts de pixel, d'image, de caractéristiques et de formats. Nous avons parlé des bases de la vision par ordinateur et de ses différentes applications.

Chapitre II :

Généralités sur intelligence artificielle deep learning and Convolutionnal Neural Networks

II.1 INTRODUCTION

Dans ce chapitre, nous allons présenter une introduction à l'intelligence artificielle, l'apprentissage automatique, l'apprentissage profond, les réseaux de neurones convolutifs, les différents modèles qui peuvent être utilisés et les limites de ces algorithmes. Nous terminons ce chapitre en pesant les compromis de chaque modèle pour trouver le modèle qui convient à notre système embarqué.

II.2 INTELLIGENCE ARTIFICIELLE (AI)

L'intelligence artificielle est la branche de l'informatique consacrée à la création de systèmes permettant d'effectuer des tâches qui requièrent habituellement l'intelligence humaine. Il s'agit d'un terme général qui englobe une grande variété de sous-domaines et de techniques ; dans ce chapitre, nous nous concentrons sur l'apprentissage profond en tant que type d'apprentissage automatique.

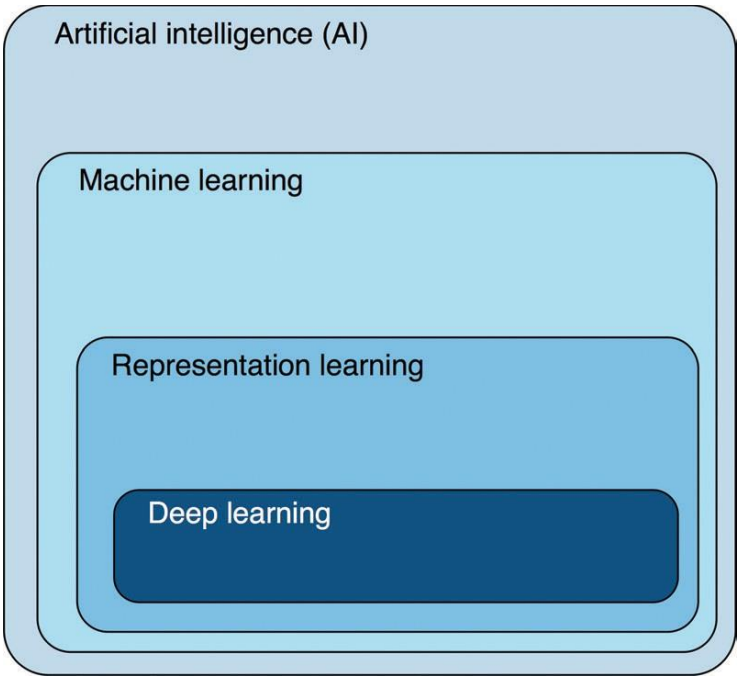


Figure II-1

II.3 MACHINE LEARNING

L'apprentissage automatique est le sous-domaine de l'intelligence artificielle dans lequel les algorithmes sont formés pour exécuter des tâches en apprenant des modèles à partir de données plutôt que par une programmation explicite [14]. Dans l'apprentissage automatique classique, des experts discernent et encodent les caractéristiques qui semblent distinctives dans les données, et des techniques statistiques sont utilisées pour organiser ou séparer les données sur la base de ces caractéristiques (Fig 2). Par exemple, pour analyser une image, un expert en traitement d'images peut programmer un algorithme pour décomposer les images d'entrée en éléments de base (bords, gradients et textures).

L'analyse statistique de la présence de ces caractéristiques dans une image donnée peut ensuite être utilisée pour classer ou interpréter l'image.

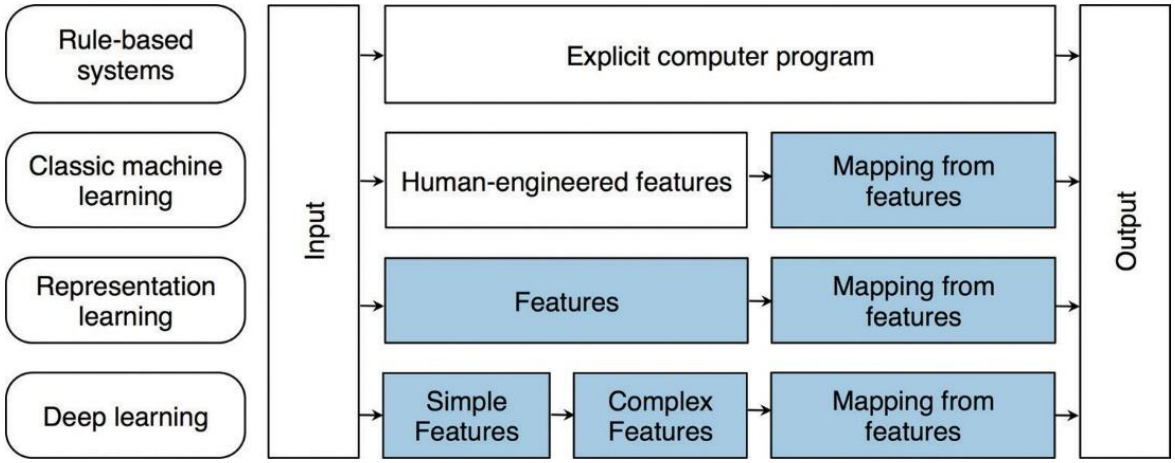


Figure II-2

Cependant, pour de nombreuses tâches complexes de vision par ordinateur, il n'est généralement pas évident, même pour un expert, de définir les caractéristiques d'image optimales à utiliser par un algorithme d'apprentissage automatique. Par exemple, il peut ne pas être évident d'apprendre à un ordinateur à reconnaître un organe sur la base de la luminosité des pixels (figure 3). Par conséquent, il peut être souhaitable qu'un système informatique ne se contente pas d'apprendre les correspondances entre les caractéristiques et les résultats souhaités, mais qu'il apprenne et optimise les caractéristiques elles-mêmes.

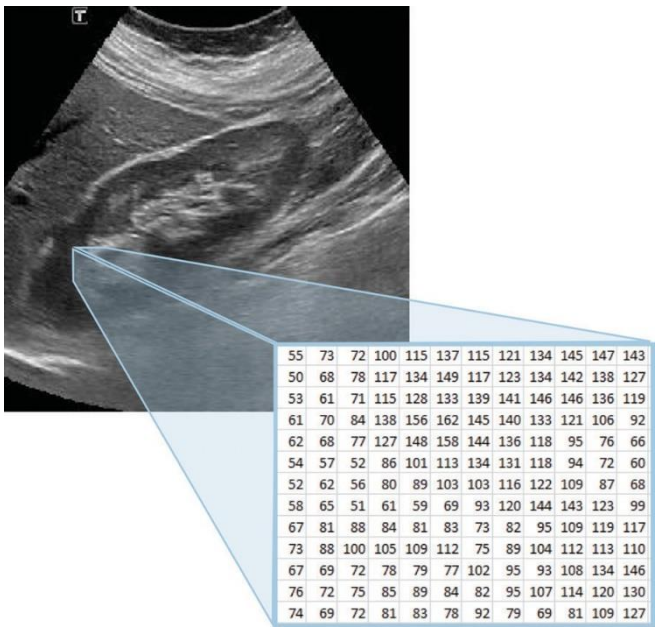


Figure II-3 Exemple des pixels

II.4 BREVE DESCRIPTION DES ALGORITHMES POPULAIRES

II.4.1 RANDOM FOREST

Random Forest est un algorithme d'apprentissage supervisé. Comme vous pouvez déjà le voir dans son nom, il crée une forêt et la rend aléatoire. La "forêt" qu'il construit est un ensemble d'arbres de décision, formés pour la plupart avec la méthode de mise en sac. L'idée générale de la méthode de mise en sac est qu'une combinaison de modèles d'apprentissage améliore le résultat global.

En termes simples : La forêt aléatoire construit plusieurs arbres de décision et les fusionne pour obtenir une prédiction plus précise et plus stable.

Un grand avantage de la forêt aléatoire est qu'elle peut être utilisée pour les problèmes de classification et de régression, qui constituent la majorité des systèmes actuels d'apprentissage automatique. Ci-dessous, vous pouvez voir à quoi ressemble une forêt aléatoire avec deux arbres :

À quelques exceptions près, un classificateur de forêt aléatoire possède tous les hyperparamètres d'un classificateur d'arbre de décision et également tous les hyperparamètres d'un classificateur de mise en sac, pour contrôler l'ensemble lui-même. Au lieu de construire un classificateur de mise en sac et de le faire passer par un classificateur d'arbre de décision, vous pouvez simplement utiliser la classe de classificateur de forêt aléatoire qui est plus pratique et optimisée pour les arbres de décision. Notez qu'il existe également un régresseur de forêt aléatoire pour les tâches de régression.

L'algorithme de forêt aléatoire apporte un caractère aléatoire supplémentaire au modèle, lorsqu'il fait croître le modèle, lorsqu'il fait croître les arbres. Au lieu de rechercher la meilleure caractéristique lors de la division d'un nœud, il recherche la meilleure caractéristique parmi un sous-ensemble aléatoire de caractéristiques. de caractéristiques. Ce processus crée beaucoup de diversité, ce qui permet généralement d'obtenir un meilleur modèle.

Par conséquent, lorsque vous créez un arbre dans une forêt aléatoire, seul un sous-ensemble aléatoire de caractéristiques est pris en compte pour la division d'un nœud. Vous pouvez même rendre les arbres plus aléatoires, en utilisant des seuils aléatoires au-dessus de chaque caractéristique plutôt que de rechercher les meilleurs seuils possibles (comme un arbre de décision normal).[15]

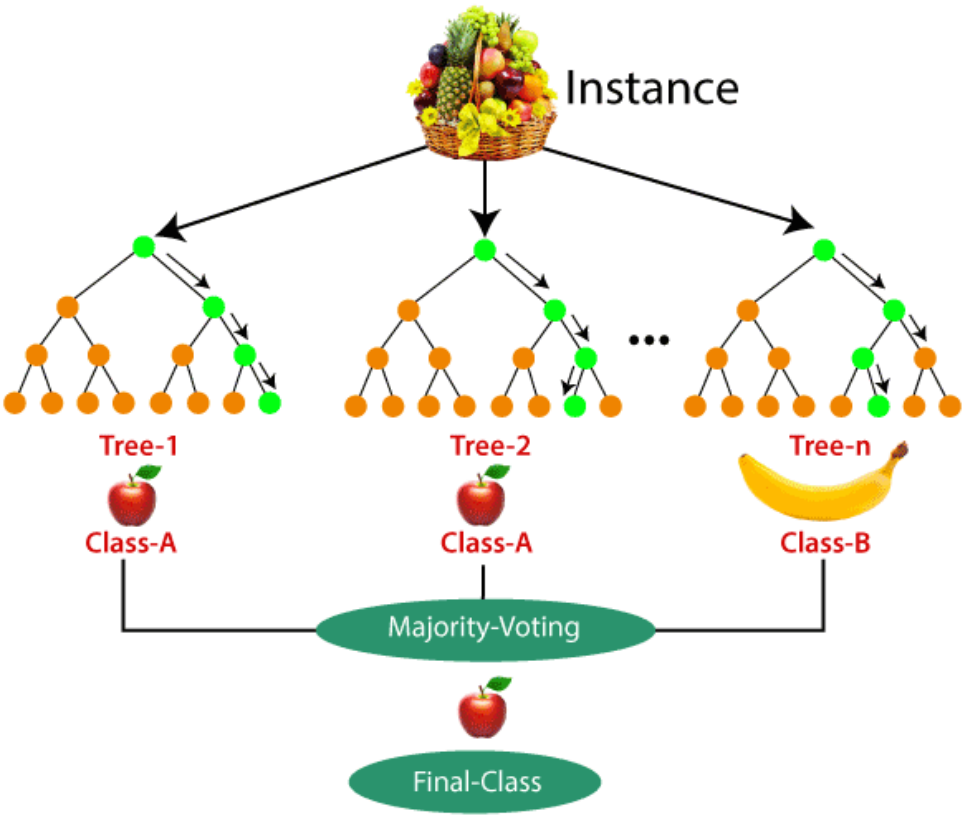


Figure II-4Random forest classifier

II.4.2 SVM (SUPPORT VECTOR MACHINE)

La machine à vecteurs de support est un algorithme d'apprentissage automatique supervisé, principalement utilisé pour la classification. L'avantage de cet algorithme par rapport aux autres algorithmes d'apprentissage automatique est qu'il permet non seulement de séparer les données en classes, mais aussi de trouver un hyperplan de séparation (l'analogue d'un plan dans un espace à plus de trois dimensions) qui maximise la marge séparant chaque point de l'hyperplan. En outre, les machines à vecteurs de support peuvent également traiter le cas des données qui ne sont pas linéairement séparables. Il existe deux façons de traiter ce type de données : l'une consiste à introduire des marges souples et l'autre à utiliser l'astuce dite du noyau.[16]

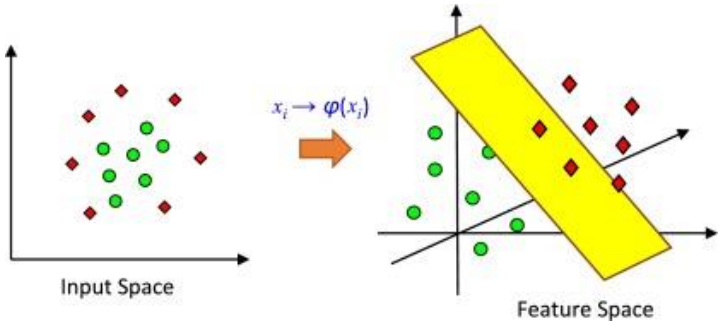


Figure II-5 SVM classifier

II.5 DEEP LEARNING

Récemment, grâce à l'optimisation des algorithmes, à l'amélioration du matériel de calcul et à l'accès à une grande quantité de données d'imagerie, l'apprentissage profond a démontré une supériorité indiscutable sur le cadre classique d'apprentissage automatique. L'apprentissage profond est une classe d'apprentissage automatique qui utilise des architectures de réseaux neuronaux artificiels qui ressemblent à la structure des fonctions cognitives humaines.[17]

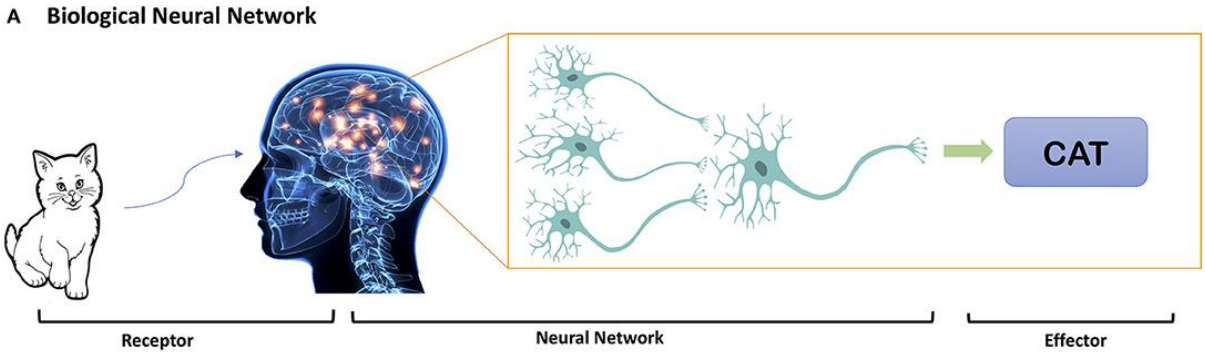


Figure II-6 réseau neuronal biologique

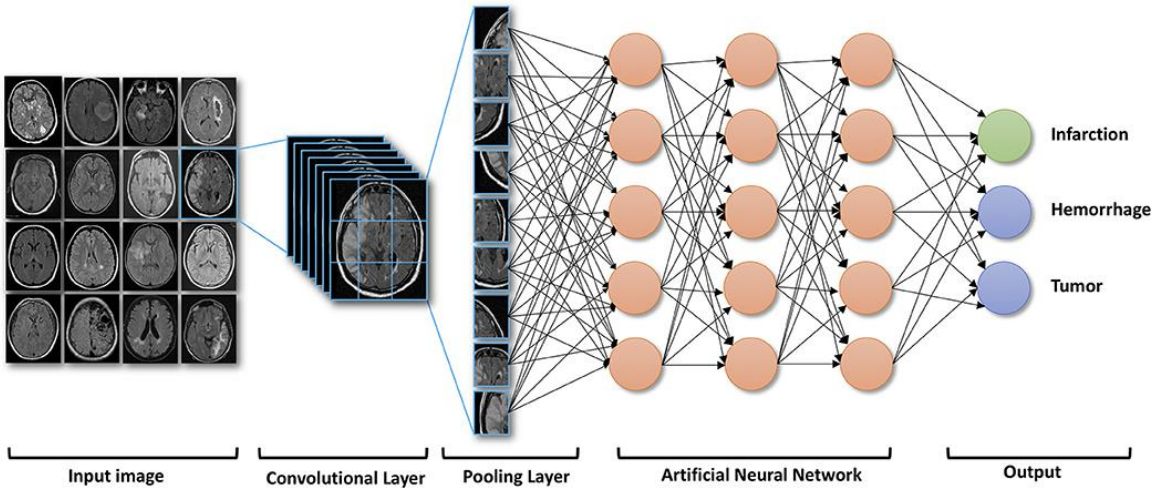


Figure II-7réseau neuronal biologique

Il s'agit d'un type d'apprentissage par représentation dans lequel l'algorithme apprend une composition de caractéristiques qui reflètent une hiérarchie de structures dans les données (18). Les réseaux de neurones convolutionnels (CNN) et les réseaux de neurones récurrents (RNN) sont différents types de méthodes d'apprentissage profond utilisant des réseaux de neurones artificiels (ANN).

II.5.1 RESEAUX NEURONAUX (NEURAL NETWORKS)

Dans le cerveau, les neurones échangent des informations via des synapses chimiques et électriques. Les signaux électrochimiques se propagent de la zone synaptique vers le

soma, le corps de la cellule, en passant par les dendrites (figure 5). Lorsqu'un certain seuil d'excitation est atteint, la cellule libère un signal d'activation à travers son axone vers les synapses avec les neurones voisins. Des signaux complexes peuvent être codés par des réseaux de neurones sur la base de ce paradigme ; par exemple, une hiérarchie de neurones dans le cortex visuel est capable de détecter des bords en combinant des signaux provenant de récepteurs visuels indépendants.[17]

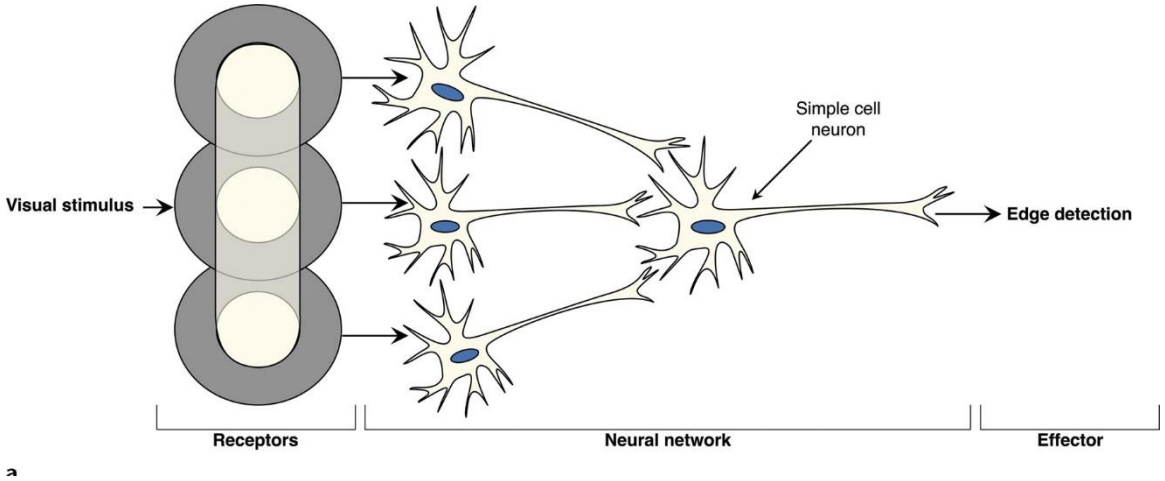


Figure II-8réseau neuronal biologique

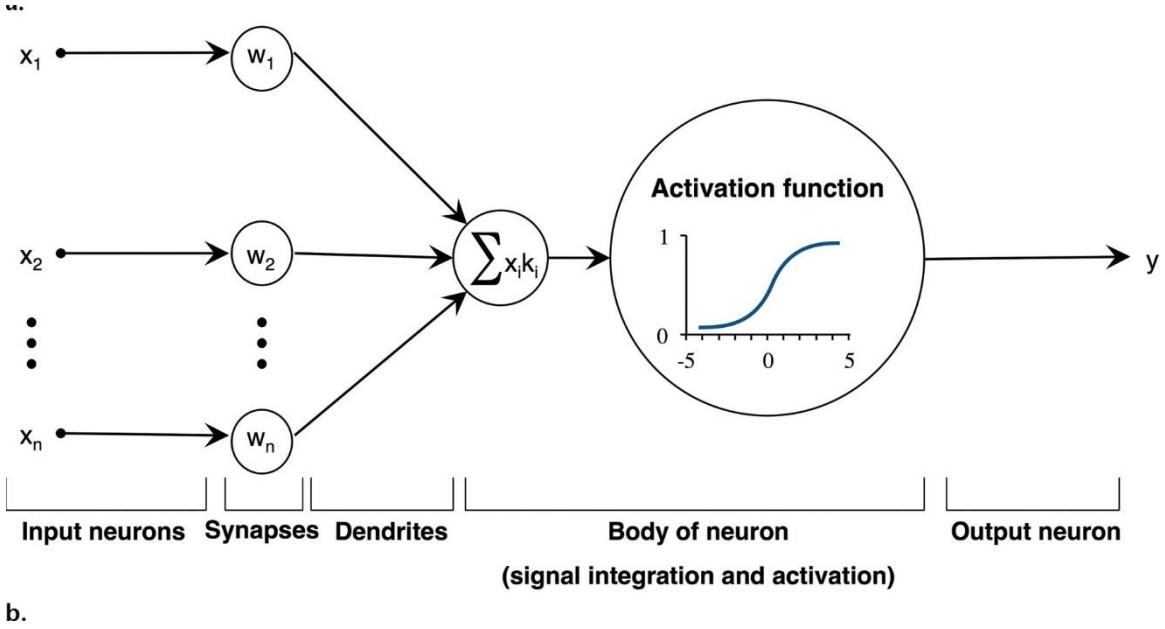


Figure II-9 réseau neuronal mathématique

II.5.2 RESEAUX NEURONAUX ARTIFICIELS (ARTIFICIAL NEURAL NETWORKS)

Les réseaux neuronaux artificiels s'inspirent de ce processus biologique. L'unité de base d'un réseau neuronal artificiel, le neurone ou nœud artificiel, est un modèle simplifié qui imite le mécanisme de base du neurone biologique. Dans le neurone biologique. Le neurone artificiel prend comme d'entrée un ensemble de valeurs représentant des

caractéristiques, chacune multipliées par un poids correspondant. Les caractéristiques pondérées caractéristiques pondérées sont additionnées et passées par une fonction d'activation non linéaire. De cette façon, un neurone artificiel peut être considéré comme produisant une décision en en pondérant un ensemble de preuves.[17]

Bien qu'un neurone artificiel individuel soit simple, les architectures de réseaux neuronaux appelées perceptrons multicouches, qui se composent de milliers de neurones, peuvent représenter des fonctions non linéaires très complexes. Ces perceptrons multicouches sont généralement construits en assemblant plusieurs neurones pour former une couche et en empilant ces couches, reliant la sortie d'une couche à l'entrée de la couche suivante. L'aspect "profond" de l'apprentissage profond fait référence à l'architecture multicouche des perceptrons multicouches (figure 6). La première couche, appelée couche d'entrée, représente les données d'entrée telles que les intensités des pixels individuels, tandis que la couche de sortie produit des valeurs cibles telles qu'un résultat de classification. Les couches intermédiaires des perceptrons multicouches sont appelées couches cachées, car elles ne produisent pas directement les sorties visibles souhaitées, mais calculent plutôt des représentations intermédiaires des caractéristiques d'entrée qui sont utiles dans le processus d'inférence.[17]

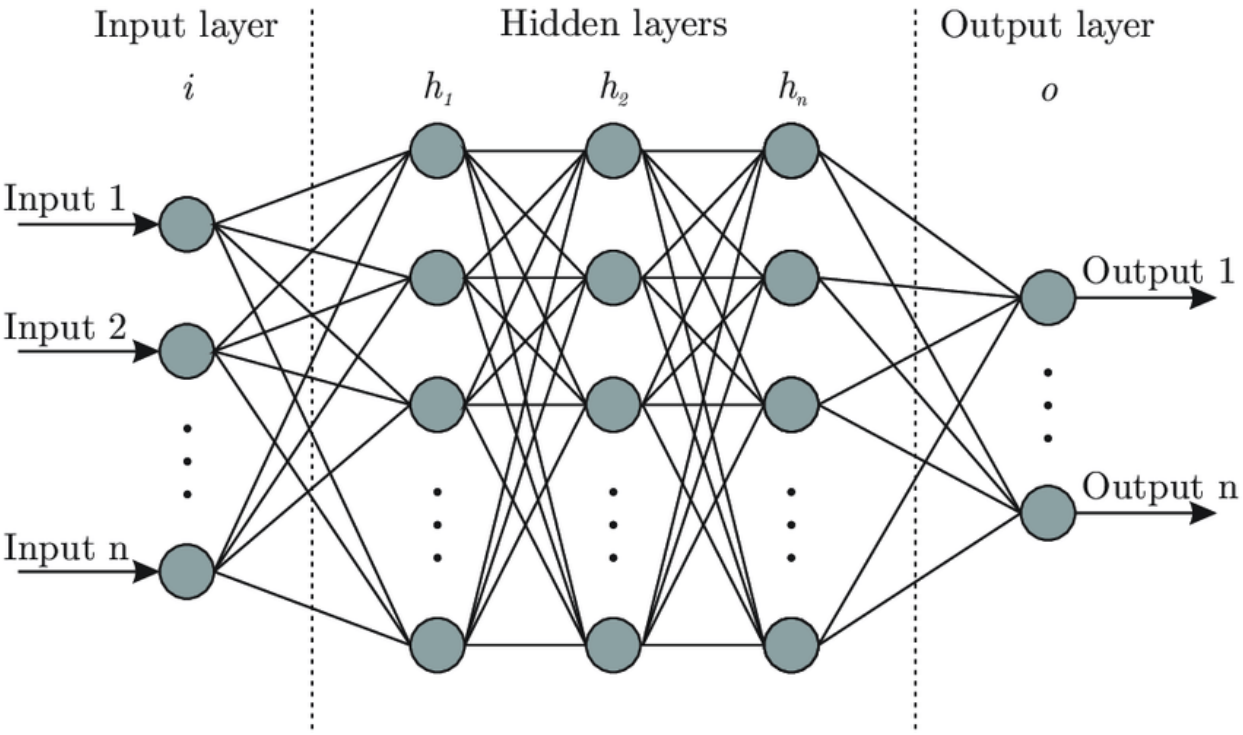


Figure II-10 CNN

II.6 RÉSEAUX NEURONES CONVOLUTIONNELS (CNN)

Les premiers CNN à utiliser la rétro propagation ont été utilisés pour la reconnaissance de chiffres manuscrits (21). Ces premiers CNN ont été inspirés par le Neocognitron, une première architecture de réseau neuronal capable de reconnaître des formes visuelles en combinant des couches de cellules simples et de cellules complexes (22). La conception du

Neocognitron s'inspire des travaux de Hubel et Wiesel (23), qui ont décrit ces deux types de cellules dans le cortex visuel primaire, une découverte qui leur a valu le prix Nobel de physiologie et de médecine en 1981.[18]

Les CNN peuvent composer des caractéristiques dont l'étendue spatiale est de plus en plus grande. Près de l'entrée, nous ne devons-nous préoccuper que des caractéristiques locales, capturées par les noyaux convolutifs ; les interactions distantes entre les pixels semblent faibles. Le même schéma se produit à chaque couche de représentation du modèle. Dans un CNN, plus la couche de représentation est profonde, plus la caractérisation de la position spatiale de l'élément est grossière (en raison de la réduction de l'échantillonnage et de la mise en commun) ; ainsi, les noyaux de ces couches plus profondes considèrent les éléments sur des échelles spatiales de plus en plus grandes.

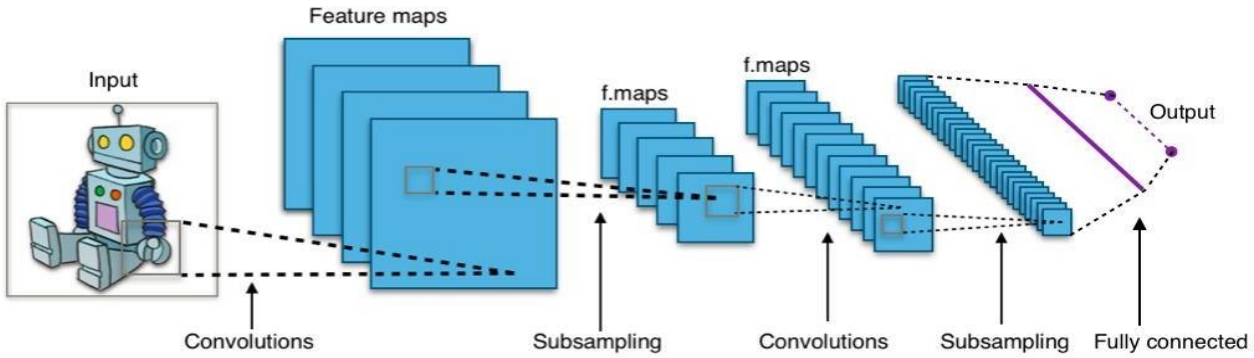


Figure II-11 Réseau neuronal convolutif

II.7 ARCHITECTURE GLOBALE DE CNN

Les CNN sont composés de trois types de couches : des couches convolutives, des couches de regroupement et des couches entièrement connectées. Lorsque ces couches sont empilées, une architecture CNN a été formée. Une architecture CNN simplifiée pour la classification MNIST est illustrée à la figure II- 12.

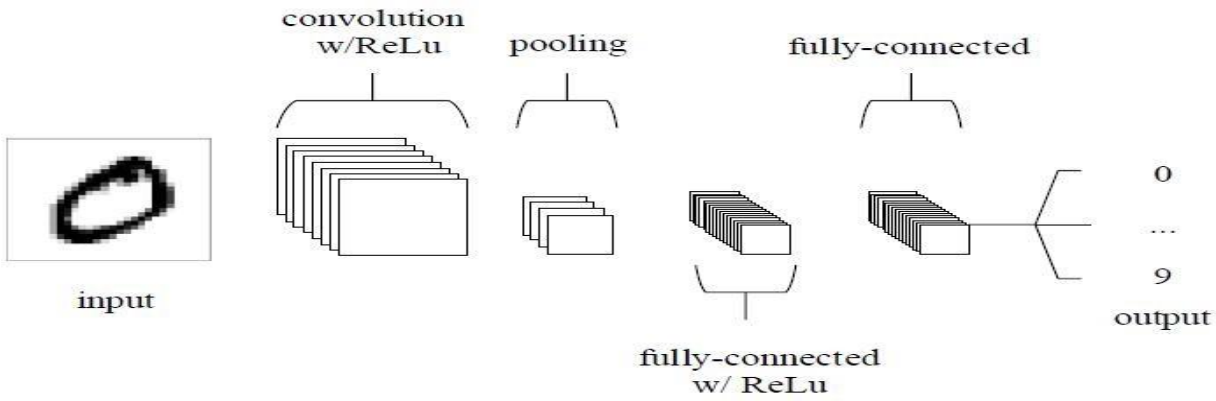


Figure II-12Réseau neuronal convolutif

II.7.1 COUCHE CONVOLUTIVE

Les couches convolutives constituent le noyau du réseau convolutif. Ces couches se composent d'une grille rectangulaire de neurones qui ont un petit champ réceptif étendu à travers toute la profondeur du volume d'entrée. Ainsi, la couche convolutive est juste une convolution d'image de la couche précédente, où les poids spécifient le filtre de convolution.

La couche convolutive déterminera la sortie des neurones qui sont connectés aux régions locales de l'entrée par le calcul du produit scalaire entre leurs poids et la région connectée au volume d'entrée. ReLu vise à appliquer une fonction d'activation «élémentaire» telle qu'une fonction sigmoïde à la sortie de l'activation produite par la couche précédente.[19]

II.7.2 COUCHE DE POOLING

Après chaque couche convolutive, il peut y avoir une couche de pooling. Cette couche sous échantillonne le long de la dimensionnalité spatiale de l'entrée donnée, ce qui réduira davantage le nombre de paramètres au sein de cette activation. Il y a plusieurs façons de faire cette mise en commun, comme prendre la moyenne ou le maximum, ou une combinaison linéaire prise par des neurones dans le bloc. Par exemple, la Fig. 3 montre le max pooling sur une fenêtre 2 × 2.[19]

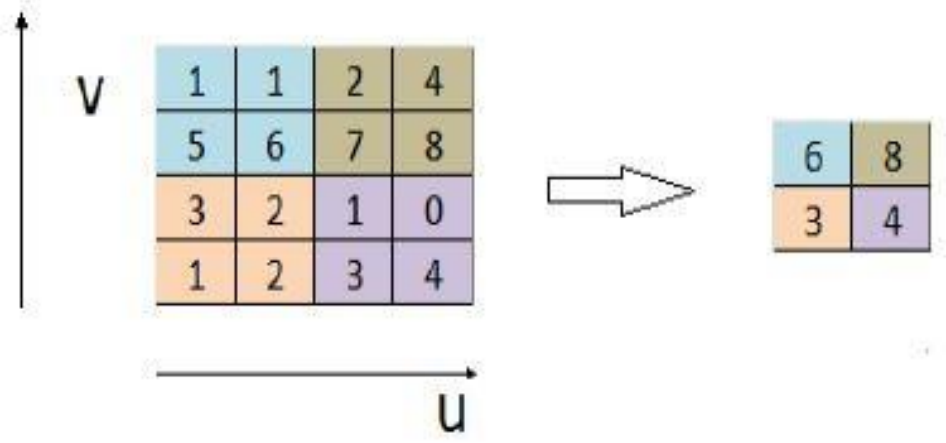


Figure II-13 Max pooling

II.7.3 COUCHE TOTALEMENT CONNECTEE

Enfin, après les couches de convolution et pooling, le raisonnement de haut niveau dans le réseau neuronal se fait via des couches totalement connectées.

Dans les réseaux de neurones convolutifs, chaque couche agit comme un filtre de détection pour la présence de caractéristiques spécifiques ou de motifs présents dans les données d'origine. Les premières couches d'un réseau convolutif détectent des caractéristiques qui peuvent être reconnues et interprétées facilement. Les couches ultérieures détectent de plus en plus des caractéristiques plus abstraites. La dernière couche du réseau convolutif est capablede faire une classification ultra-spécifique en combinant

toutes les caractéristiques spécifiques détectées par les couches précédentes dans les données d'entrée.

Les couches totalement connectées font les mêmes tâches que celles des ANN standard et tenteront de produire des notes de classe à partir des activations, pour les utiliser pour la classification. Il est également suggéré d'utiliser ReLu entre ces couches pour améliorer les performances.

II.7.4 COUCHE DE CORRECTION (RELU)

C'est une couche pour améliorer l'efficacité du traitement en intercalant entre les couches de traitement une couche qui va opérer une fonction mathématique (fonction d'activation) sur les signaux de sortie.

Cette fonction force les neurones à retourner des valeurs positives.

$$\text{ReLu} : f(x) = \max(0, x)$$

II.8 DIFFÉRENTES ARCHITECTURES CNN

Au cours des dix dernières années, le réseau de neurones convolutifs a beaucoup progressé et a conduit au développement de multiples architectures (la plupart de ces modèles ont été conçus pour le concours annuel ImageNet, qui consiste à classer 1000 objets ; l'organisation tente d'aider la recherche en IA en offrant des stages intensifs aux gagnants) avec des différences minimes mais notables, dont certaines sont célèbres :

II.8.1 ALEXNET

En 2012, Alex Krizhevsky, Ilya Sutskever et Geoff Hinton ont remporté le concours ImageNet Large Scale Visual Recognition Challenge avec une précision de 84,6 %[3]. Le modèle a largement dépassé les performances du deuxième finaliste (erreur de 16 % dans le top 5, contre 26 % pour le finaliste). Krizhevsky a utilisé les GPU pour entraîner l'AlexNet, ce qui a permis un entraînement plus rapide des modèles CNN et a déclenché une vague d'intérêt et de nouveaux travaux basés sur les CNN.[20]

Le réseau se compose de 5 couches convolutionnelles et de 3 couches entièrement connectées.

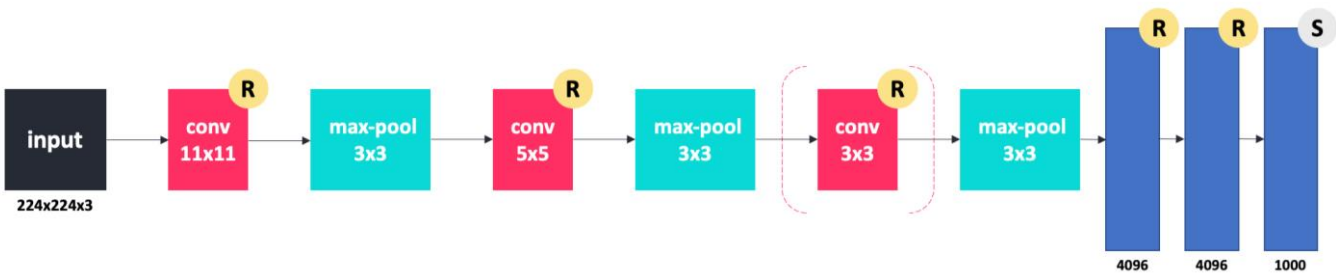


Figure II-14 AlexNet

II.8.2 VGG-19

Le modèle atteint une précision de 92,7 % au test top-5 dans ImageNet, qui est un jeu de données de plus de 14 millions d'images appartenant à 1000 classes. Il a été proposé par Karen Simonyan et Andrew Zisserman du laboratoire du groupe de géométrie visuelle de l'Université d'Oxford en 2014.[20]

Le 16 dans VGG16 fait référence au fait qu'il possède un total de 16 couches qui ont des poids.

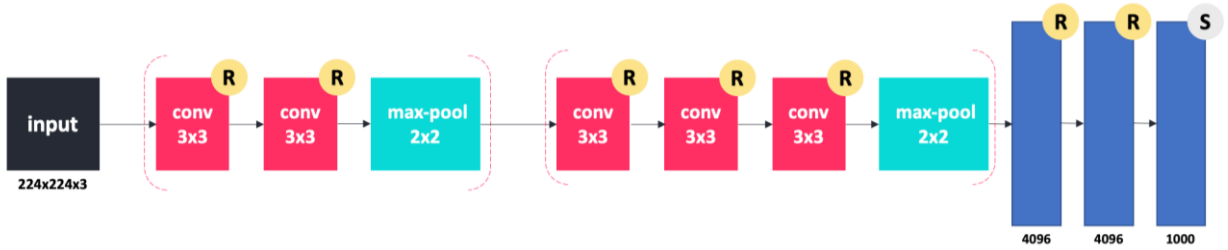


Figure II-15 VGG-19

II.8.3 RESNET

Le modèle ResNet a été créé pour résoudre le problème de la disparition du gradient (lorsque les réseaux ConvNets sont plus profonds, la valeur de la dérivée lors de la rétropropagation vers les couches initiales devient presque insignifiante).

Ces raccourcis sont ajoutés entre deux couches (appelés bloc résiduel), où nous ne prévoyons aucun changement dans la valeur des neurones pour limiter les problèmes de complexité.[20]

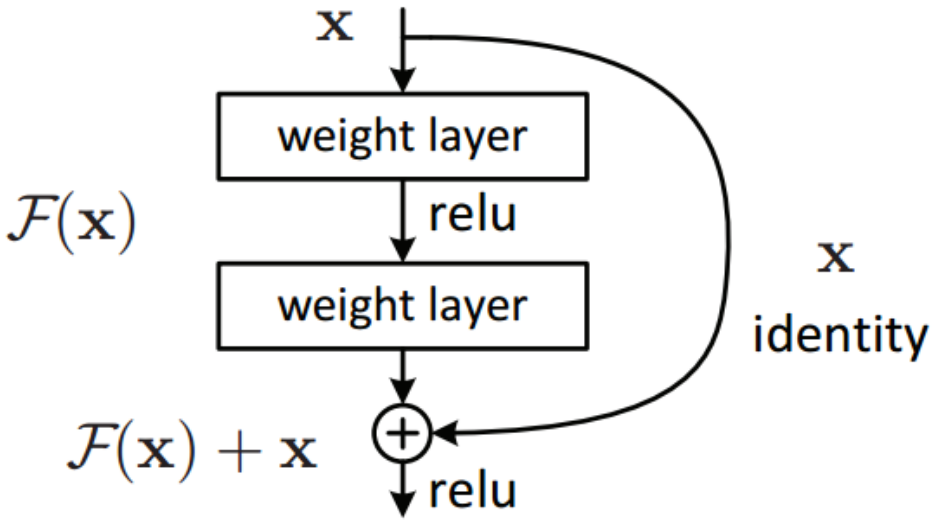


Figure II-16 ResNet

II.8.4 MOBILENET

Les MobileNets ont été l'une des premières initiatives visant à construire des architectures CNN pouvant être facilement déployées dans des applications mobiles. L'une des principales innovations est la convolution séparable en profondeur, qui est visualisée ci-dessous. Une convolution séparable sépare un noyau de convolution normal en deux noyaux. Par exemple, au lieu d'un noyau 3x3, nous obtenons un noyau 3x1 et un noyau 1x3. Cette séparation réduit le nombre des opérations nécessaires pour effectuer la convolution et donc beaucoup plus efficace. Cependant, il n'est pas toujours possible de séparer sur la dimension spatiale, il est donc plus courant de séparer sur la dimension de la profondeur (canal). Cette convolution séparable en profondeur est utilisée dans MobileNet. L'article introduit également un multiplicateur de largeur qui vous permet de mettre facilement à l'échelle le modèle CNN en fonction de votre cas d'utilisation. Le modèle est devenu une solution facile pour les applications mobiles de détection d'objets, de détection de visages et de classification d'images.[21]

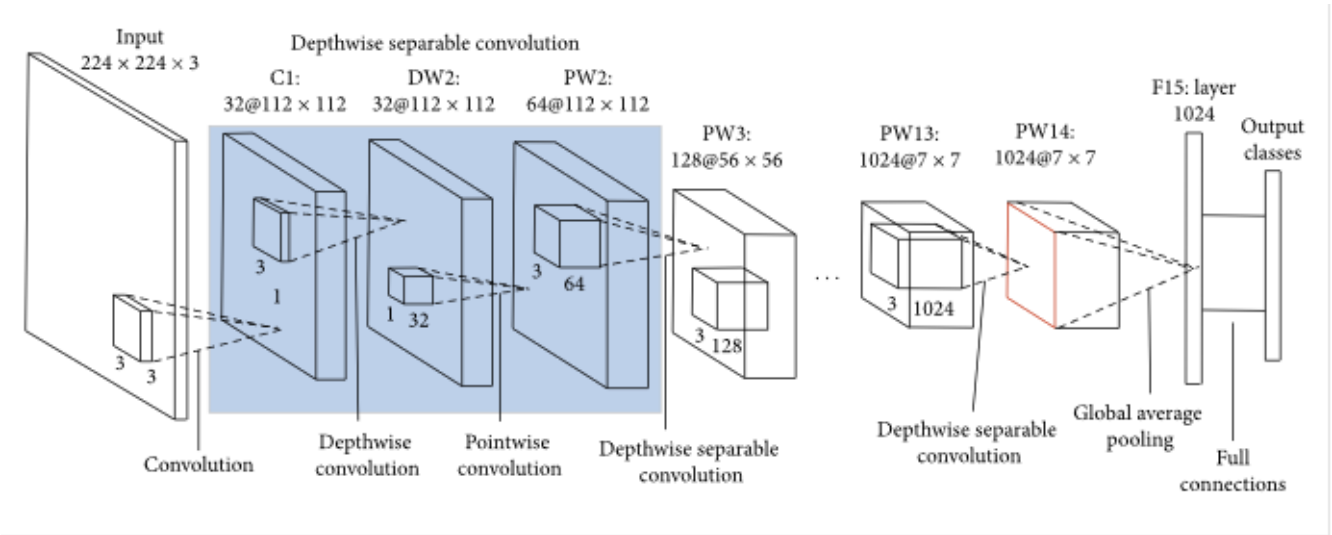


Figure II-17 MobileNet

II.9 CONCLUSION

Dans ce chapitre, nous avons parlé de la définition de l'intelligence artificielle, du machine Learing et du Deep learing.Nous avons présenté les différents types de réseaux neuronaux convolutifs pour nous a aidé à choisir l'architecture parfaite de CNN.

Nous terminons la première partie de notre thèse (partie théorique). Dans le prochain chapitre, nous parlerons de la partie implémentation et des résultats de notre système

Chapitre III :

Environnement de travail

III.1 INTRODUCTION

Avant d'entamer la mise en œuvre du projet, il est nécessaire de définir notre environnement de travail, à la fois matériel (Hardware) et logiciel (Software). Environnement de travail, tant au niveau matériel que logiciel. en outre, dans ce chapitre, une présentation des différents composants du matériel sera détaillé; y inclutJetson Nano, les composants du PCB et les logiciels ; LunixOS, la bibliothèque OpenCV, la bibliothèque Numpy,la bibliothèque Tensorflow,la bibliothèque Keras, et le langage de programmation Python.

III.2 MATÉRIEL (HARDWARE)

III.2.1 JETSON NANO

Le kit de développement NVIDIA Jetson est un petit ordinateur puissant qui permet de faire tourner plusieurs réseaux neuronaux en parallèle pour des applications telles que la classification d'images, la détection d'objets, la segmentation et le traitement de la parole. Le tout dans une plateforme facile à utiliser qui ne consomme que 5 watts. C'est la solution idéale pour prototyper un prochain produit basé sur l'IA et le mettre rapidement sur le marché.[22]

Un ordinateur monocratie est un ordinateur complet construit sur une seule carte de circuit imprimé, avec microprocesseur, mémoire, entrée/sortie et autres caractéristiques requises pour un ordinateur fonctionnel.

Un Jetson contient un processeur ARM qui est l'une des familles les plus simples de CPU, il a également une faible consommation d'énergie, c'est pourquoi ils sont les processeurs les plus utilisés dans les systèmes embarqués.

Par ailleurs, Le Jetson a été conçu pour encourager l'enseignement et l'apprentissage du codage, il permet l'exécution des différentes distributions GNU/Linux et d'autres programmes compatibles.

La Fondation NVIDIA dispose d'un catalogue décent de 4 modèles qui présentent des caractéristiques et des fonctionnalités diverses :

Figure III-1Jetson Nano

Figure III-2 JETSON XAVIER NX

Figure III-3 JETSON AGX XAVIER

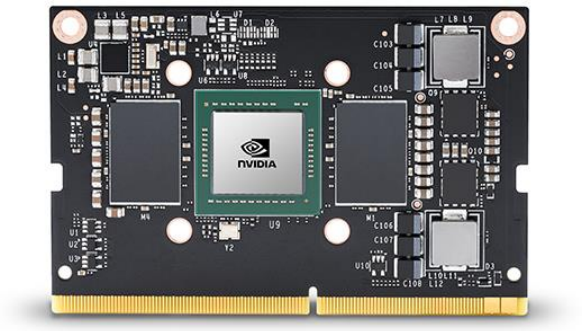


Figure III-4 JETSON TX2

Pour pouvoir utiliser le Jetson nano, il faut au moins :

- D'un support mémoire (carte SD ou Micro SD selon le modèle) qui contient un système d'exploitation compatible.
- D'une source d'alimentation ; avec des batteries, ou un transformateur électrique qui doit fournir le courant et la tension minimum.

De plus, ni la souris, ni le clavier, ni l'écran ne sont nécessaires pour utiliser le Jetson. Sur certains systèmes d'exploitation, SSH (Secure Shell) et VNC (Virtual Networking

Computing) sont actifs et configurés par défaut et faciles à activer et à configurer via le support mémoire si le système d'exploitation ne les active pas par défaut. Ces protocoles de communication permettent à l'utilisateur de contrôler le Jetson via un réseau dès le premier démarrage.

Afin de réaliser le projet, on a choisi le Jetson Nano en raison de son faible coût, de sa disponibilité et de la raison principale : ses spécifications sont suffisamment proches des exigences du projet.

Les spécifications sont énumérées ci-dessous :[23]

- GPU: NVIDIA Maxwell architecture with 128 NVIDIA CUDA® cores
- CPU: Quad-core ARM Cortex-A57 MPCore processor
- Memory: 4 GB 64-bit LPDDR4, 1600MHz 25.6 GB/s
- Camera: 12 lanes (3x4 or 4x2) MIPI CSI-2 D-PHY 1.1 (1.5 Gb/s per pair)
- Connectivity: Gigabit Ethernet, M.2 Key E
- Display: HDMI 2.0 and eDP 1.4
- USB: 4x USB 3.0, USB 2.0 Micro-B
- Others: GPIO, I2C, I2S, SPI, UART
- Mechanical 69.6 mm x 45 mm

III.2.2 ARDUINO UNO

L'Arduino Uno est une carte microcontrôleur open-source basée sur le microcontrôleur Microchip ATmega328P et développée par Arduino.cc. La carte est équipée d'un ensemble de broches d'entrée/sortie numériques et analogiques qui peuvent être interfacées avec diverses cartes d'extension et autres circuits.[24]

Figure III-5 Arduino Uno

III.2.3 RELAY SHILED

Le Relay Shield offre une solution pour contrôler les dispositifs à courant élevé qui ne peuvent pas être contrôlés par les broches d'E/S numériques de l'Arduino en raison des limites de courant et de tension.[25]

Figure III-6 Relay shiled

III.2.4 MOTEUR PAS A PAS

Le moteur pas-à-pas appartient au groupe des moteurs synchrones et dispose d'un élément moteur pivotant doté d'un arbre, le rotor, et d'un élément moteur fixe, que l'on appelle le stator. Tandis que le rotor agit comme un aimant permanent, le stator, lui, est composé de bobines d'excitation disposées en quinconce et qui produisent un champ magnétique. Celui-ci est à la base des capacités de positionnement du moteur et marque la différence de ce moteur par rapport aux servomoteurs. Alors que ces derniers utilisent différents capteurs de mesure de position et les valeurs qui en découlent, le moteur pas-à-pas, parfois aussi appelé stepper, fonctionne sans aucun capteur.

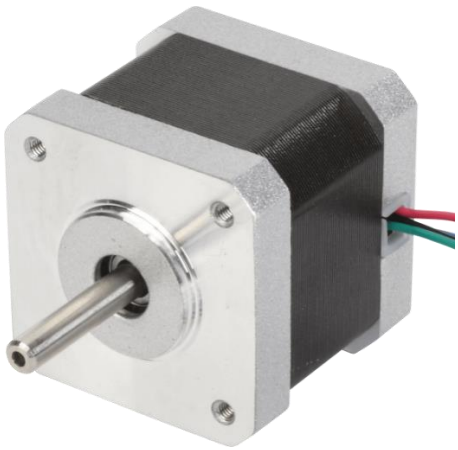


Figure III-7 Moteur pas à pas

III.2.5 DRV8825

Cette carte d'extension pour le circuit d'attaque de moteur pas à pas bipolaire à micropas DRV8825 de TI comporte une limitation de courant réglable, une protection contre les surintensités et les surchauffes, et six résolutions de micropas (jusqu'à 1/32 de pas). Il

fonctionne de 8,2 V à 45 V et peut délivrer jusqu'à environ 1,5 A par phase sans dissipateur thermique ni flux d'air forcé (jusqu'à 2,2 A par bobine avec un refroidissement supplémentaire suffisant).[26]

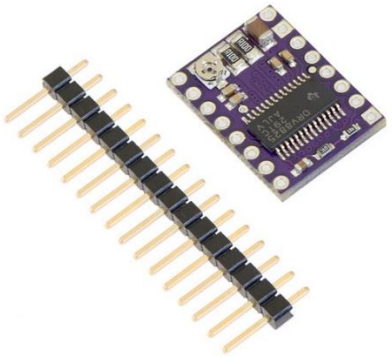


Figure III-8DRV8825

III.3 LOGICIEL (SOFTWARE)

III.3.1 UBUNTU

Ubuntu est une distribution Linux basée sur Debian et composée principalement de logiciels libres et open-source. Ubuntu est officiellement publiée en trois éditions : Desktop, Server, et Core pour les appareils et robots de l'Internet des objets. Toutes les éditions peuvent fonctionner sur l'ordinateur seul, ou dans une machine virtuelle.[27]

Figure III-9 Logo Ubuntu

III.3.2 ARDUINO IDE

L'environnement de développement intégré Arduino est une application multiplateforme qui est écrite dans des fonctions de C et C++. Il est utilisé pour écrire et télécharger des programmes sur les cartes compatibles Arduino, mais aussi, à l'aide de noyaux tiers, sur les cartes de développement d'autres fournisseurs.[24]

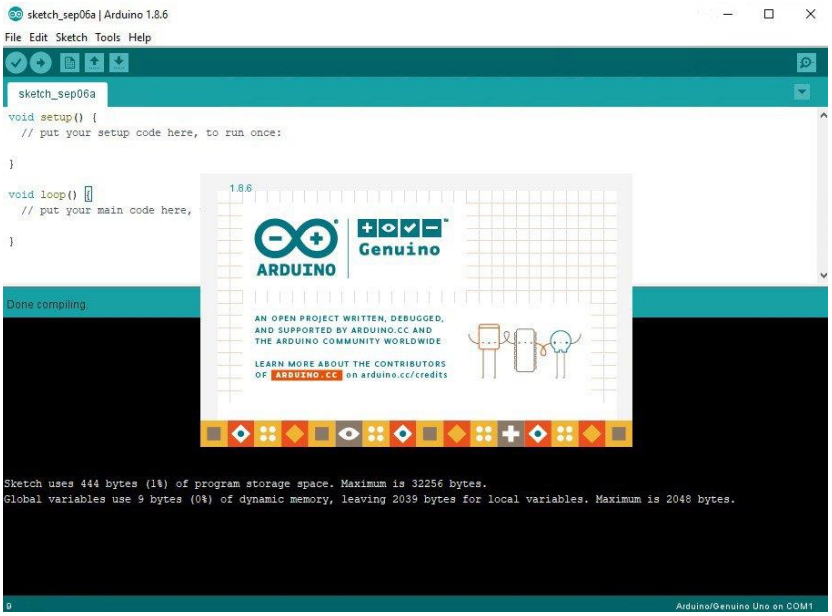


Figure III-10 Arduino IDE

III.3.3 GOOGLE COLABORATORY

Google Colaboratory (en abrégé Google Colab) est une plateforme Google destinée à aider les étudiants, les scientifiques et les chercheurs à exécuter du code Python avec un minimum de configuration minimale et un accès gratuit aux GPU (Graphic Processing Units). Google Colab fonctionne sur un navigateur, il est donc portable.

Par ailleurs, Google Colab est utilisé dans ce projet pour faire le "gros travail", en d'autres termes, il est utilisé pour entraîner notre modèle sur le Google Cloud sans avoir à s'embêter à utiliser l'ordinateur personnel ou les spécifications limitées du Jetson Nano.[28]

Figure III-11 Google Colab Logo

III.3.4 PYTHON

Python est un langage de programmation interprété, interactif et orienté objet. Il intègre des modules, des exceptions, un typage dynamique, des types de données dynamiques de très haut niveau et des classes. Il prend en charge plusieurs paradigmes de programmation au-delà de la programmation orientée objet, comme la programmation procédurale et fonctionnelle. Le langage est fourni avec une grande bibliothèque standard qui couvre des domaines tels que le traitement des chaînes de caractères, les protocoles Internet, l'ingénierie logicielle et les interfaces de système d'exploitation.

La principale raison pour laquelle nous avons choisi d'utiliser Python dans ce projet est sa portabilité, ce qui signifie en quelques mots : nous n'avons pas à utiliser le Jetson Nano pour l'entraînement du modèle ; la mémoire et la puissance de traitement sont limitées par rapport aux spécifications du système pour la session Google Colab, mais nous pouvons quand même utiliser le modèle pour que le système fonctionne.[29]

Figure III-12 Python Logo

Comme mentionné ci-dessus, Python a accès à de nombreuses bibliothèques, voici quelques-unes des bibliothèques utilisées dans ce projet :

III.3.4.1 OpenCV

OpenCV (Open Source Computer Vision Library) est une bibliothèque logicielle open source de vision par ordinateur et d'apprentissage automatique. OpenCV a été construit pour fournir une infrastructure commune pour les applications de vision par ordinateur et pour accélérer l'utilisation de la perception artificielle dans les produits commerciaux. Étant un produit sous licence BSD, OpenCV permet aux entreprises d'utiliser et de modifier facilement le code.

La bibliothèque compte plus de 2500 algorithmes optimisés, ce qui inclut un ensemble complet d'algorithmes de vision par ordinateur et d'apprentissage automatique classiques et de pointe. Ces algorithmes peuvent être utilisés pour détecter et reconnaître des visages, identifier des objets, classer des actions humaines dans des vidéos, suivre les mouvements de la caméra, suivre des objets en mouvement...[30]

Figure III-13 OpenCV Logo

III.3.4.2 NumPy

NumPy est une bibliothèque pour le langage de programmation Python, qui ajoute un support pour les tableaux et matrices multidimensionnels de grande taille, ainsi qu'une large collection de fonctions mathématiques de haut niveau pour opérer sur ces tableaux.[31]

Figure III-14 NumPy Logo

III.3.4.3 TensorFlow

TensorFlow est une bibliothèque de logiciels open-source pour la programmation différentielle et de flux de données pour diverses tâches. De même, TensorFlow est utilisé dans l'apprentissage automatique par les réseaux de neurones. Développé par Google en 2011 sous le nom de DistBelief, TensorFlow a été officiellement mis en ligne en 2017 de manière gratuite. La bibliothèque est capable de fonctionner sur plusieurs CPU et GPU et est disponible sur plusieurs plateformes, y compris sur mobile. Le nom vient des tableaux multidimensionnels connus sous le nom de tenseurs, qui sont couramment utilisés dans les réseaux neuronaux.[32]

Figure III-15 TensorFlow Logo

III.3.4.4 Keras

Keras est une bibliothèque open-source de composants de réseaux neuronaux écrits en Python. Keras est capable de fonctionner au-dessus de TensorFlow, Theano, PlaidML et autres. La bibliothèque a été développée pour être modulaire et conviviale, mais elle a initialement commencé dans le cadre d'un projet de recherche pour le système d'exploitation intelligent neuro-électronique ouvert ou ONEIROS. L'auteur principal de Keras est François Chollet, un ingénieur de Google qui a également écrit XCeption, un modèle de réseau neuronal profond. Bien que Keras ait été officiellement lancé, il n'a été intégré à la bibliothèque centrale TensorFlow de Google qu'en 2017. Un support supplémentaire a également été ajouté pour l'intégration de Keras avec Microsoft Cognitive Toolkit.[33]

Composée d'une bibliothèque de composants d'apprentissage automatique couramment utilisés, notamment des objectifs, des fonctions d'activation et des optimiseurs, la plateforme open-source de Keras offre également un support pour les réseaux neuronaux récurrents et convolutifs. En outre, Keras propose le développement de plateformes mobiles pour les utilisateurs ayant l'intention de mettre en œuvre des modèles d'apprentissage profond sur les smartphones, tant iOS qu'Android. En 2018, la bibliothèque compte 22 % d'utilisation à travers de ses plus de 200 000 utilisateurs.

Figure III-16 Keras Logo

III.4 CONCLUSION

Dans ce chapitre, l'environnement de travail, tant matériel que logiciel, nécessaire à la mise en œuvre de notre système a été présenté. Dans le chapitre suivant "Implémentation", nous allons combiner les algorithmes mentionnés dans le chapitre précédent pour créer le système décrit.

Chapitre IV :

Implémentation

IV.1 INTRODUCTION

Dans ce chapitre, nous allons utiliser les informations acquises dans le deuxième chapitre et l'environnement de travail décrit dans le troisième chapitre pour mettre en œuvre le système présenté précédemment, nous allons parler en détail à la fois du matériel, le circuit imprimé créé et du logiciel, le modèle créé pour aider à détecter les fruits frais et pourris.

IV.2 MATÉRIEL (HARDWARE)

En ce qui concerne la partie matérielle, nous pouvons la diviser en deux grandes parties, les schémas du système et le prototype mécanique de la machine.

IV.2.1 SCHEMAS

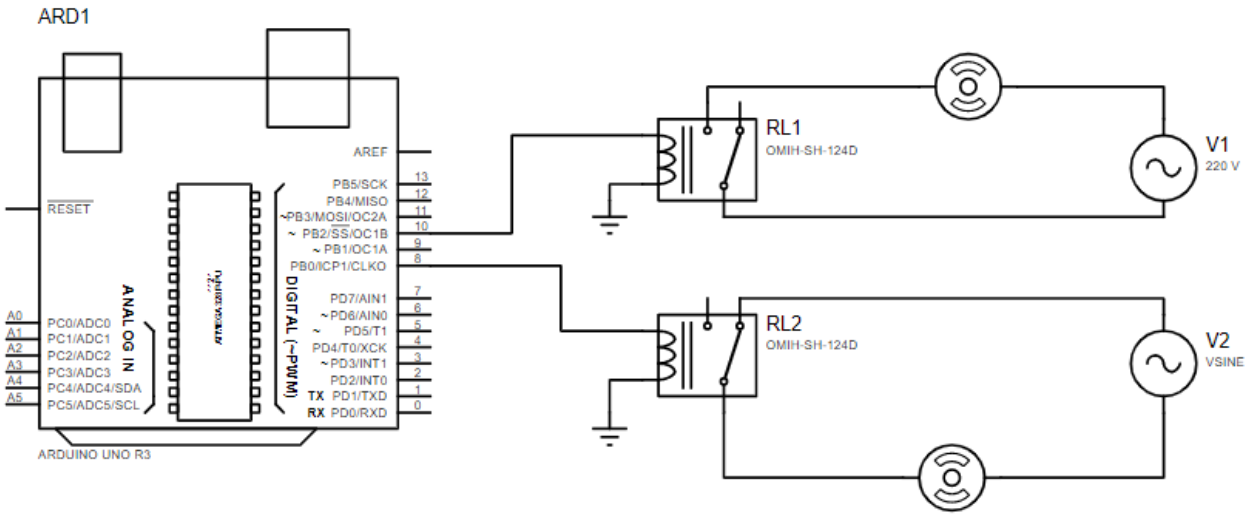


Figure IV-1 Schema Arduino

IV.2.2 PROTOTYPE MECANIQUE

En ce qui concerne le modèle mécanique, il y a eu de nombreuses suggestions et quelques formes, mais en raison de la difficulté à trouver des matériaux, nous avons réalisé un modèle réduit présenté ci-dessous :

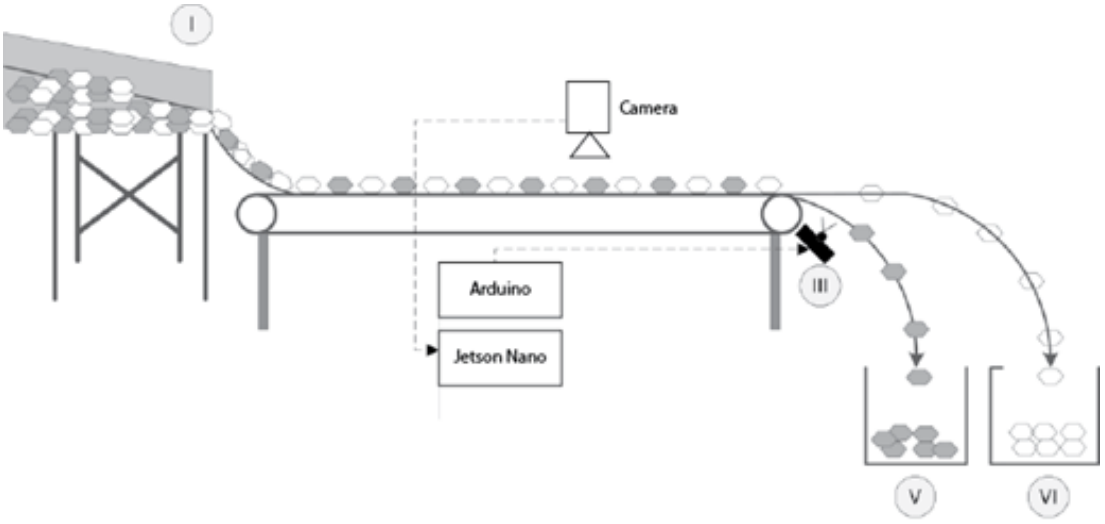


Figure IV-2 Exemple 01

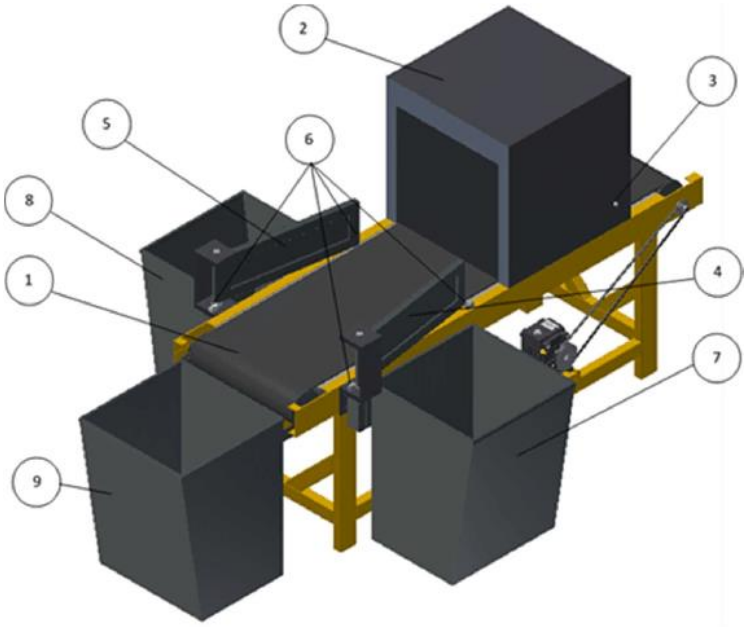


Figure IV-3 Exemple 2

IV.3 LOGICIEL (SOFTWARE)

La partie logicielle du système peut être divisée en 5 parties principales : la compréhension du modèle, la manière dont l'extraction des données a été effectuée, la formation, la validation et enfin le fonctionnement de l'ensemble du code lorsqu'il est combiné au matériel.

IV.3.1 LA COLLECTION DES DONNEES

Afin d'appliquer une méthode d'apprentissage supervisé ou non supervisé, il est d'abord nécessaire de collecter une grande quantité de données de qualité, ce qui affecte

directement la précision du modèle final. Dans notre Project, nous intéressons à la création de modèles prédictifs pour classer les fruits et détecter les anomalies.

Nous avons utilisé des datasets gratuits provenant de différentes sources, et nous avons créé nos propres datasets.

Nous utilisons un dataset de trois fruits (pommes, bananes, oranges), divisé en fruits frais et fruits pourris. Nous divisons ce jeu de données en un ensemble d'entraînement qui contient 80% des données, et les 20% restants sont utilisés comme ensemble de validation.

Le tableau suivant résume les jeux de données utilisés :

Classes	Train set (80%)	Validation set (20%)	Total
Pommes fraîches	**193**	**46**	**239**
Banane fraîche	**189**	**43**	**232**
Oranges fraîches	**181**	**41**	**22**
Pommes pourries	**263**	**65**	**328**
Banane pourrie	**253**	**61**	**314**
Oranges pourries	**178**	**48**	**226**
Total	**1257**	**304**	**1561**

Table 1Description des datasets

Pour accéder à dateset vous pouvez scanner le code QR ou aller à la ligne :

- https://drive.google.com/drive/folders/1TX5_33mDFWj5PABbo6sBFDLXNfmELker?usp=sharing

Figure IV-4 Code QR de dataset

IV.3.2 MODEL BUILDING

Aprèsla préparation et le nettoyage des données, maintenant on peut passer à la phase d’apprentissage, cette phase est très coûteuse en termes deressources, donc nous avons

choisis d'utiliser la plateforme "Google Colab" qui offre des "CPU" et "GPU" gratuits, ainsi le service de stockage "Google Drive" pour sauvegarder datasets.

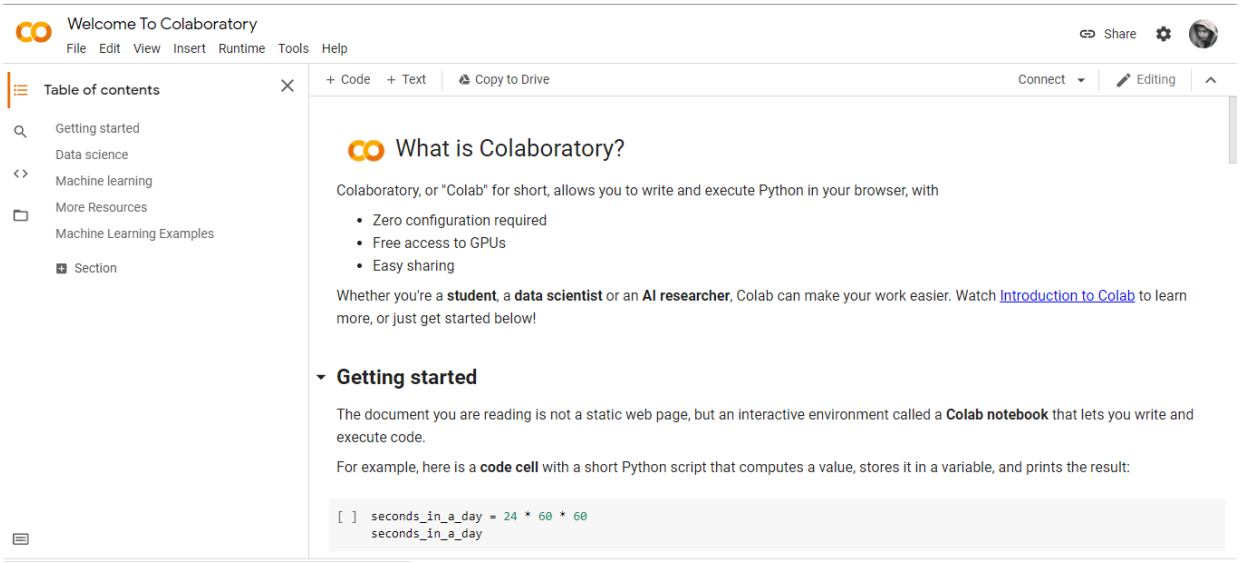


Figure IV-5 Google Colab

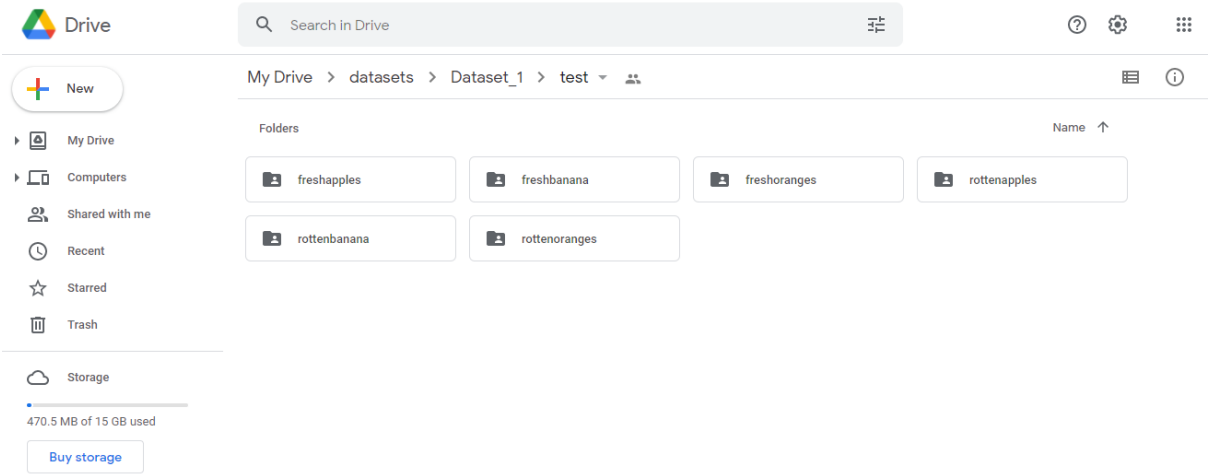


Figure IV-6 Google Drive

Le tableau représente les capacités de calcul de l'environnement d'exécution :

Paramètre	PC Portable	Colab cloud service
NVIDIA GPU	920m	Tesla K80
GPU RAM	4GB	12GB
CPU RAM total	8GB	13GB
Core	4	2
Brand	Intel(R) Core(TM) i3-4005U	Intel(R) Xeon(R)
Frequence	1.7GHZ	2.20GHz

Pour construire le modèle de classification, nous utilisons un réseau neuronal convolutif (CNN). Nous choisissons l'architecture standard du réseau dorsal MobileNet comme modèle de base pour notre classificateur. Comme couches supérieures, nous ajoutons une couche de mise en commun de la moyenne globale, une couche entièrement connectée avec 256 nœuds, et enfin, une couche entièrement connectée avec 6 nœuds représente la couche de sortie. La figure suivante donne les détails du modèle CNN :

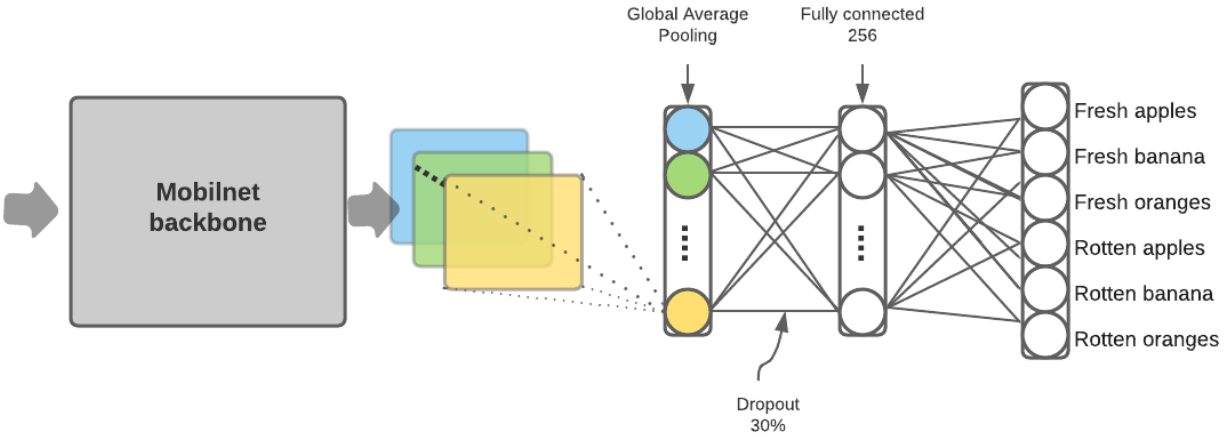


*Figure IV-7*architecture de notre model

Pour entraîner ce modèle, nous tirons parti de la stratégie d'apprentissage par transfert et en particulier de l'apprentissage par réglage fin. Cela signifie que les poids du modèle de base sont initialisés à partir d'un modèle déjà entraîné sur ImageNet. Ensuite, le modèle est réentraîné sur le nouveau jeu de données en utilisant de nouvelles couches supérieures correspondant au nombre de classes du nouveau jeu de données. Le choix de l'apprentissage par transfert est motivé par sa capacité à améliorer les résultats lorsque de petits ensembles de données sont utilisés comme dans notre cas.

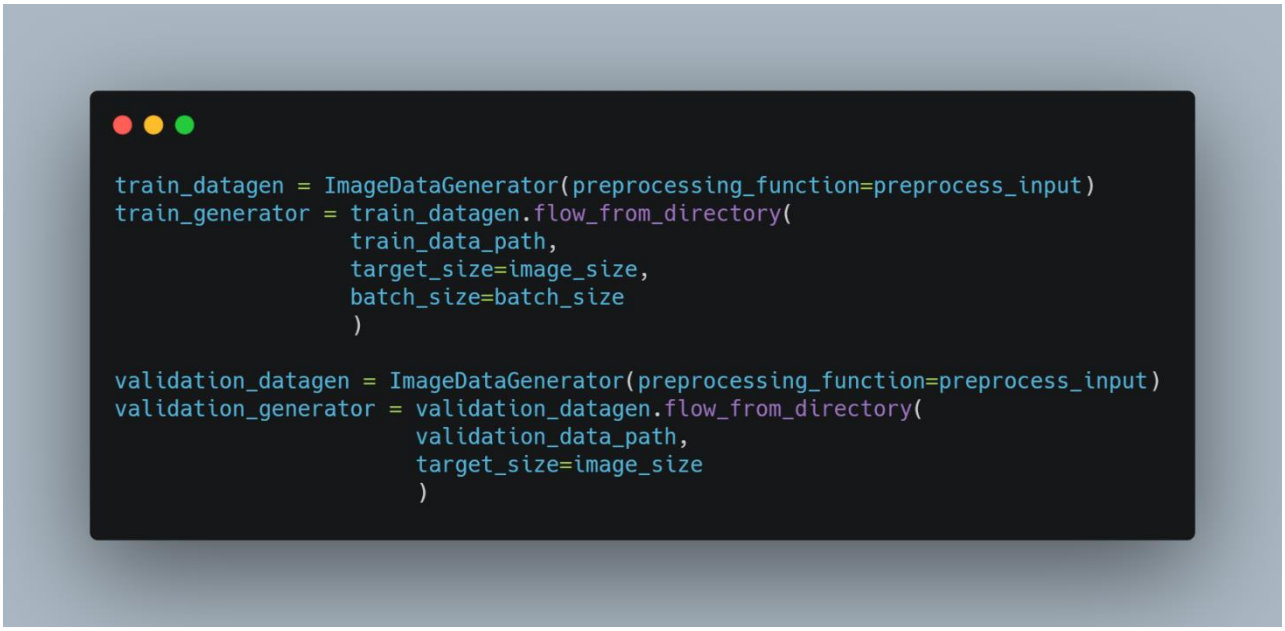
```
def build_model_graph(class_number=class_number):
  base_model = MobileNet(weights='imagenet', include_top=False,input_shape = (224,224,3))
  x = base_model.output
  x = GlobalAveragePooling2D()(x)
  #x = Dense(512, activation='relu')(x)
  x = Dropout(0.3)(x)
  x = Dense(256, activation='relu')(x)
  predictions = Dense(class_number, activation='softmax')(x)

  model = Model(base_model.input, predictions)

  return model
```

Figure IV-8 Build Model

IV.3.3 TRAIN THE MODEL

Nous parlerons du test de validation plus tard, pour l'instant nous allons expliquer la formation. Elle a été réalisée en utilisant le modèle décrit dans la section précédente et en suivant les étapes suivantes :

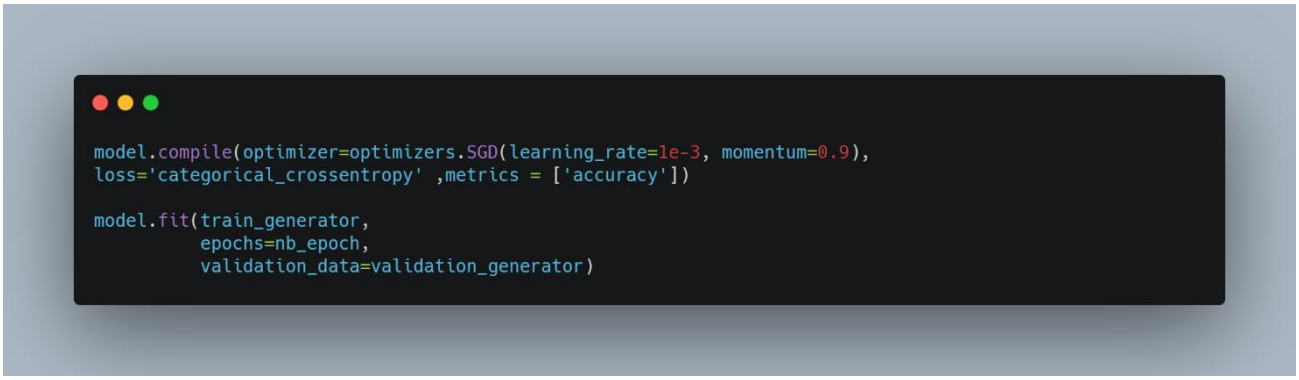
- Chargement des images, redimensionnement à une taille fixe et attribution d'une étiquette (frais ou pourri) en fonction de l'image.



```
train_datagen = ImageDataGenerator(preprocessing_function=preprocess_input)
train_generator = train_datagen.flow_from_directory(
                  train_data_path,
                  target_size=image_size,
                  batch_size=batch_size
                  )

validation_datagen = ImageDataGenerator(preprocessing_function=preprocess_input)
validation_generator = validation_datagen.flow_from_directory(
                       validation_data_path,
                       target_size=image_size
                       )
```

Figure IV-9 Chargement des images

- Une fois que le modèle est entrainé , nous allons essayer de lui attribuée certains paramètres :



```
model.compile(optimizer=optimizers.SGD(learning_rate=1e-3, momentum=0.9),
loss='categorical_crossentropy' ,metrics = ['accuracy'])

model.fit(train_generator,
          epochs=nb_epoch,
          validation_data=validation_generator)
```

Figure IV-10 Algorithme de traing

```
Epoch 1/15
40/40 [==============================] - 18s 388ms/step - loss: 0.9732 - accuracy: 0.6508 - val_loss:
0.9860 - val_accuracy: 0.6053
Epoch 2/15
40/40 [==============================] - 15s 368ms/step - loss: 0.1504 - accuracy: 0.9602 - val_loss:
0.4067 - val_accuracy: 0.8651
Epoch 3/15
40/40 [==============================] - 15s 367ms/step - loss: 0.0920 - accuracy: 0.9737 - val_loss:
0.3963 - val_accuracy: 0.8651
Epoch 4/15
40/40 [==============================] - 15s 369ms/step - loss: 0.0617 - accuracy: 0.9849 - val_loss:
0.1728 - val_accuracy: 0.9441
Epoch 5/15
40/40 [==============================] - 15s 369ms/step - loss: 0.0310 - accuracy: 0.9984 - val_loss:
0.1357 - val_accuracy: 0.9507
Epoch 6/15
40/40 [==============================] - 15s 368ms/step - loss: 0.0322 - accuracy: 0.9936 - val_loss:
0.0972 - val_accuracy: 0.9737
Epoch 7/15
40/40 [==============================] - 15s 369ms/step - loss: 0.0189 - accuracy: 1.0000 - val_loss:
0.0894 - val_accuracy: 0.9671
Epoch 8/15
40/40 [==============================] - 15s 370ms/step - loss: 0.0174 - accuracy: 0.9992 - val_loss:
0.0615 - val_accuracy: 0.9803
Epoch 9/15
40/40 [==============================] - 15s 372ms/step - loss: 0.0162 - accuracy: 0.9984 - val_loss:
0.0514 - val_accuracy: 0.9836
Epoch 10/15
40/40 [==============================] - 15s 366ms/step - loss: 0.0152 - accuracy: 0.9984 - val_loss:
0.0574 - val_accuracy: 0.9803
Epoch 11/15
40/40 [==============================] - 15s 374ms/step - loss: 0.0165 - accuracy: 0.9952 - val_loss:
0.0449 - val_accuracy: 0.9836
Epoch 12/15
40/40 [==============================] - 15s 371ms/step - loss: 0.0111 - accuracy: 1.0000 - val_loss:
0.0657 - val_accuracy: 0.9737
Epoch 13/15
40/40 [==============================] - 15s 367ms/step - loss: 0.0099 - accuracy: 0.9984 - val_loss:
0.0526 - val_accuracy: 0.9770
Epoch 14/15
40/40 [==============================] - 15s 371ms/step - loss: 0.0125 - accuracy: 0.9984 - val_loss:
0.0463 - val_accuracy: 0.9836
Epoch 15/15
40/40 [==============================] - 15s 373ms/step - loss: 0.0120 - accuracy: 0.9984 - val_loss:
0.0578 - val_accuracy: 0.9770
```

Figure IV-11 Résultat du modèle obtenu.

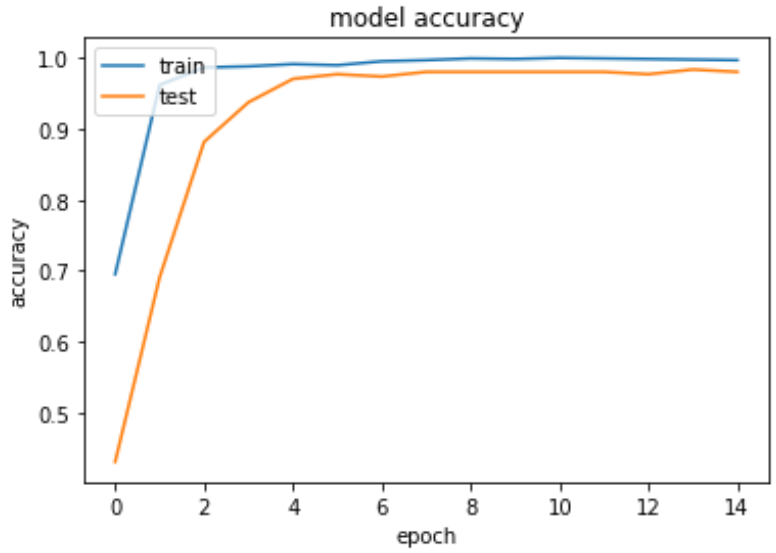


Figure IV-12 model accuracy

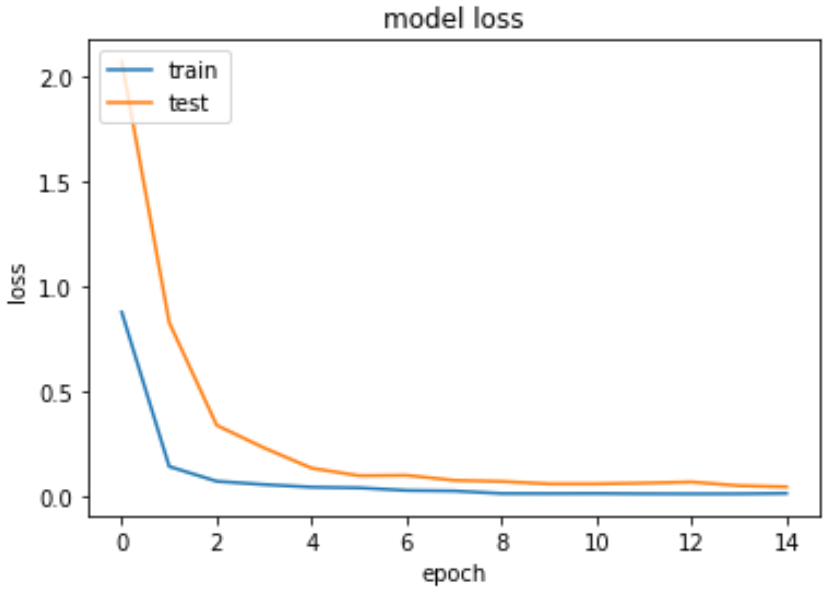


Figure IV-13 Model loss

- **Epoch** : désigne le nombre d'itération dans notre base de données. Optimiser : permet de réduire le poids des erreurs. (15epoch)
- **Batch_size** = 32
- **Loss** : désigne le taux d'erreur.
- **Accuracy** : designe le taux de précision.

Si nous faisons une comparaison de la Figure IV 12 nous constatons qu'entre ces deux itérations nous constatons que le taux d'erreur baisse tant disque notre précision augmente cela signifie que notre modèle a été bien entrainer et réponds d'ailleurs à la définition du réseau de neurones qui disait que plus le réseau de neurones est profond meilleur sont ses performances.

IV.3.4 CONFIGURATION DE JETSON NANO

Une fois l'apprentissage terminé, le modèle doit être optimisé et adapté pour être utilisé dans le Jetson Nano pour l'inférence. Nous devons configurer le NVIDIA Jetson Nano pour la vision par ordinateur et l'apprentissage profond.

1. Dans cette étape, nous allons télécharger l'image d'OS Jetpack 4.2 de NVIDIA basée sur Ubuntu et la flasher sur une microSD. Vous aurez besoin de la microSD flashée et prête à fonctionner pour suivre les étapes suivantes.
   Allez-y et commencez votre téléchargement, en vous assurant que vous téléchargez l'image "Jetson Nano Developer Kit SD Card" comme indiqué dans la capture d'écran suivante :

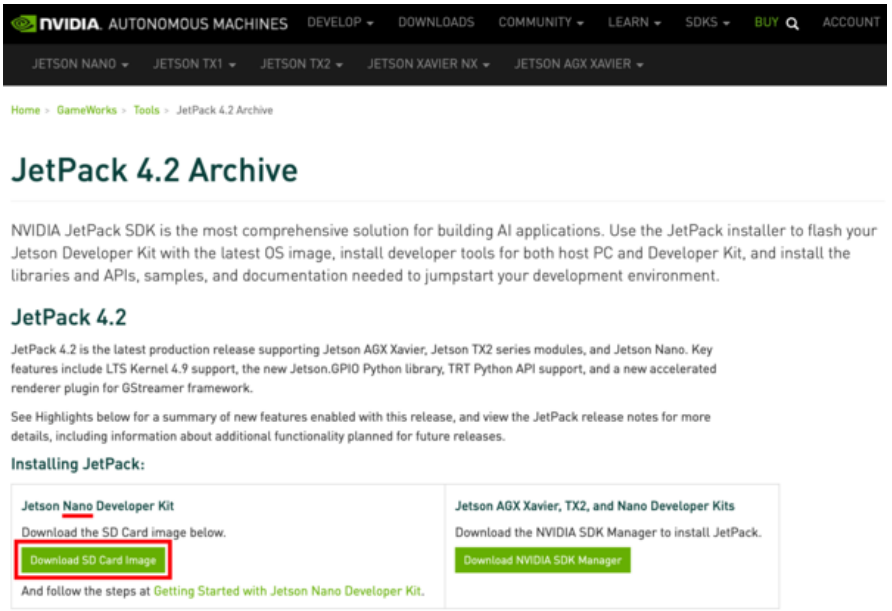


Figure IV-14 The first step to configure your NVIDIA Jetson Nano for computer vision and deep

2. Démarrez votre Jetson Nano avec la microSD et connectez-vous à un réseau.

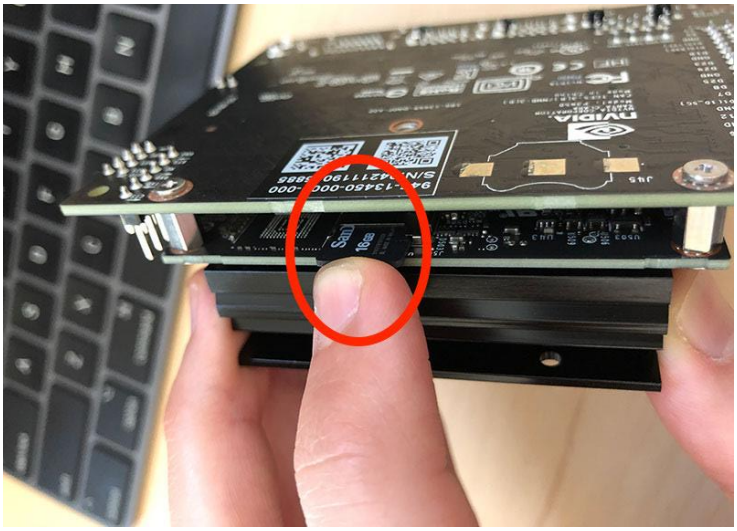


Figure IV-15Pour insérer votre microSD flashée par Jetpack après qu'elle ait été flashée.

3. De là, connectez votre écran, votre clavier, votre souris et votre interface réseau. Enfin, mettez sous tension. Insérez la fiche d'alimentation de votre adaptateur électrique dans votre Jetson Nano.

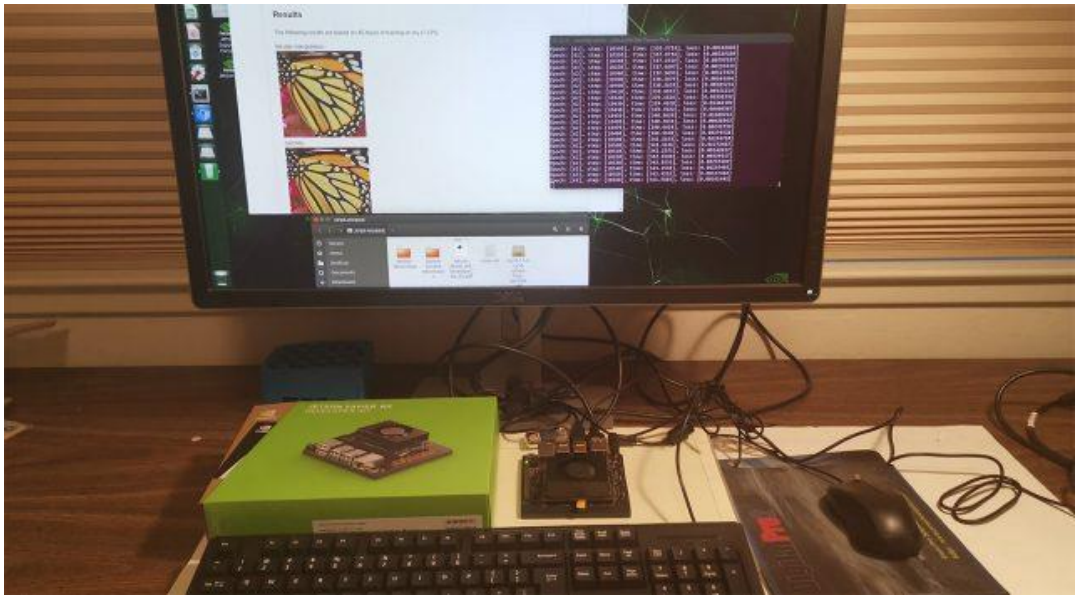


Figure IV-16 Jetson Nano mode desktop

4. Une fois que vous voyez votre bureau NVIDIA + Ubuntu 18.04, vous devez configurer les paramètres de votre réseau câblé ou sans fil selon vos besoins en utilisant l'icône dans la barre de menu comme indiqué dans la Figure IV 22.

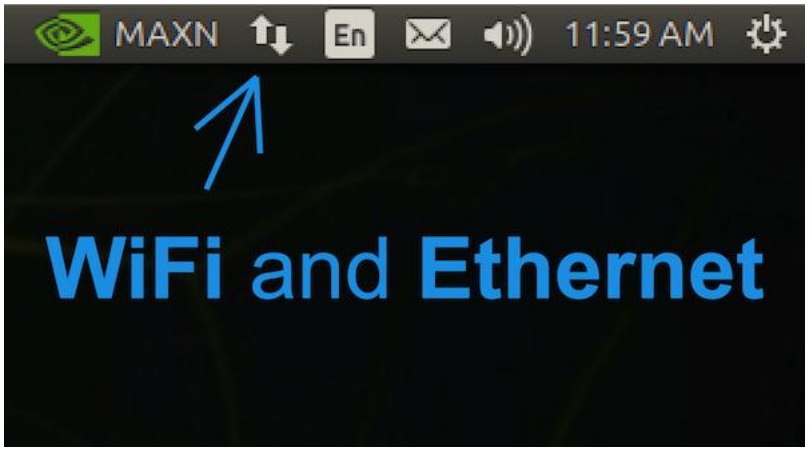


Figure IV-17 Configuration de wifi

Après avoir confirmé que vous disposez d'un accès Internet sur votre NVIDIA Jetson Nano, vous pouvez passer à l'étape suivante.

5. Dans cette étape, nous devons installer toutes les librairies dont nous avons besoin pour notre projet.

IV.3.5 DEPLOIEMENT DU MODELE

Pour optimiser le modèle, nous avons utilisé le TensorRT SDK pour adapter le modèle.

Les applications basées sur TensorRT sont jusqu'à 40 fois plus rapides que les plates-formes à processeur seul pendant l'inférence. Avec TensorRT, vous pouvez optimiser les

modèles de réseaux neuronaux formés dans tous les principaux frameworks, calibrer pour une précision moindre avec une grande exactitude, et déployer sur des centres de données hyperscale, des plateformes de produits embarqués ou automobiles.[34]

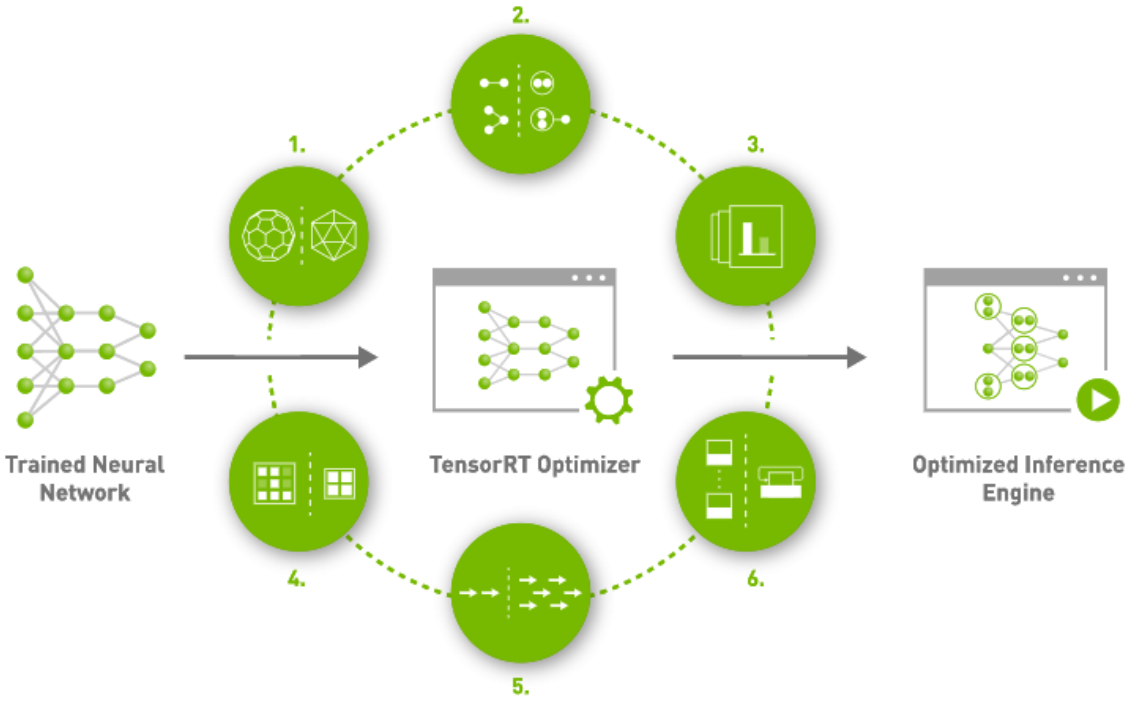


Figure IV-18architecture de optimization

IV.4 CONCLUSION

Dans ce chapitre, nous avons discuté du processus de mise en œuvre de notre système, en commençant par le matériel, puis les parties logicielles et enfin nous avons parlé de la façon d'optimiser le modèle. Dans le prochain chapitre, nous parlerons des liens entre les systèmes complets et les résultats.

Chapitre V :

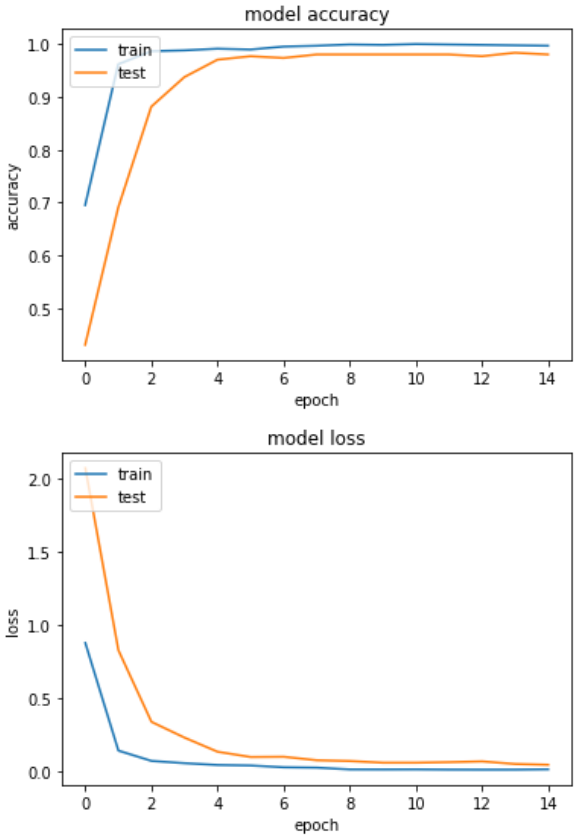
Résultats

V.1 INTRODUCTION

Dans ce chapitre, nous parlerons des différents résultats de notre système, des différents effets des paramètres de formation, nous essaierons également de trouver les meilleurs paramètres de modèle. Enfin, nous discuterons d'autres améliorations qui permettraient d'accroître l'efficacité du système.

V.2 ENTRAINEMENT DU MODÈLE

Comme nous l'avons expliqué précédemment, nous avons entraîné notre modèle en utilisant l'apprentissage par transfert car le jeu de données utilisé n'est pas grand. Comme prévu, les courbes de précision et de perte pendant l'entraînement montrent clairement que le modèle converge juste après 6 itérations à 98% de la précision de validation.



V.3 MATRICE DE CONFUSION

Pour avoir une idée claire de la façon dont le modèle se comporte entre les classes, nous avons calculé la matrice de confusion présentée dans la figure suivante. La matrice de confusion montre que les vrais positifs sont supérieurs à 97 % dans toutes les classes, à l'exception des oranges pourries (94 %). Le modèle a mal classé 6,2 % des oranges pourries en oranges fraîches et 3,1 % des pommes pourries en pommes fraîches. Ce problème peut être résolu en ajoutant davantage d'images pourries à l'ensemble d'entraînement.

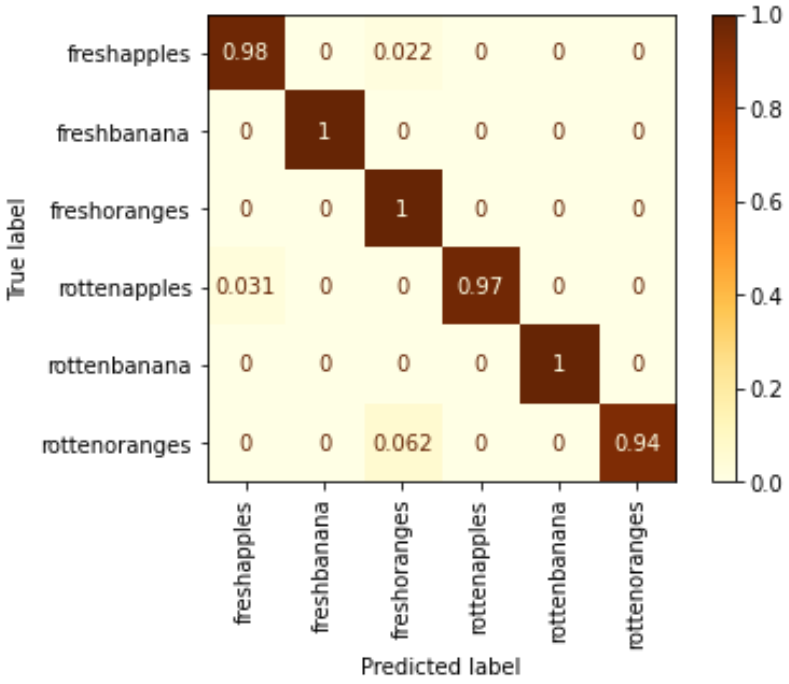


Figure V-1 Matrice de confusion

V.4 IMAGES MAL CLASSEES

Nous avons extrait les images mal classées de notre ensemble de données de validation pour analyser les erreurs. Ces erreurs sont illustrées dans la figure suivante. A partir de ces erreurs, on peut remarquer que le modèle en général n'échoue pas à classifier un fruit comme un autre fruit. Parfois, le modèle manque certains défauts dans les pommes et les oranges, ce qui peut être corrigé en incluant plus d'images avec différents types de défauts.

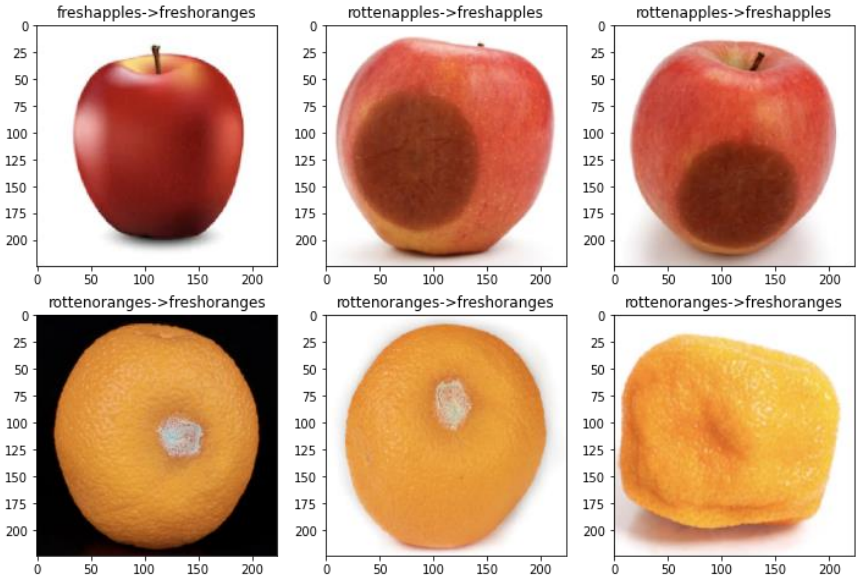


Figure V-2 Images mal classées

V.5 RESULTAT POUR LES IMAGES DE TEST TELECHARGEES SUR INTERNET

Pour valider le modèle, nous avons téléchargé 6 images sur internet réparties dans les 6 classes. Les 6 images, présentées dans la figure suivante, ont été classées correctement avec notre classifieur bien qu'elles ne fassent pas partie de notre jeu de données, ce qui nous a motivé à déployer notre modèle dans la machine de tri et à l'améliorer progressivement en le réentraînant avec les données à venir.

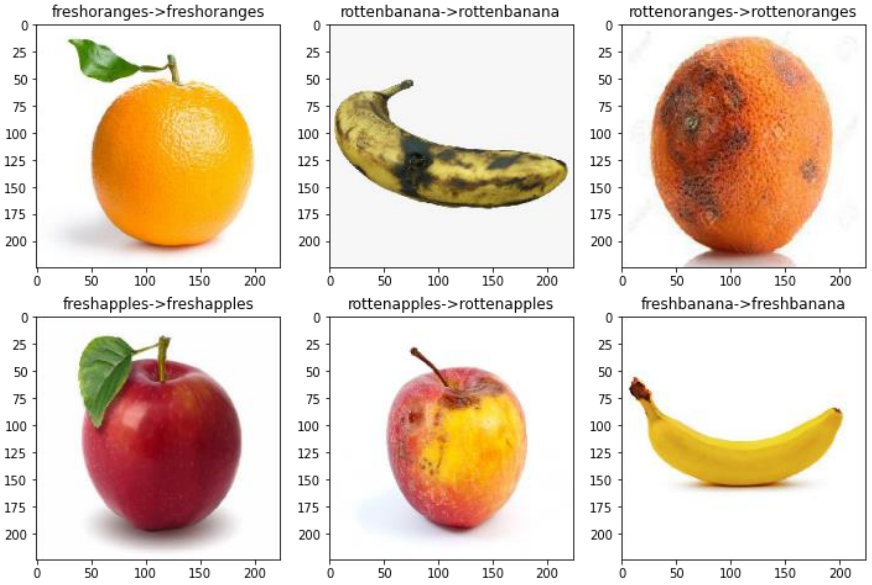


Figure V-3 Résultat pour les images de test téléchargées sur Internet

V.6 CONCLUSION

Dans ce chapitre, nous avons parlé des résultats finaux de notre modèle, en l'affinant et en créant un système qui permettrait d'augmenter le taux de détection et de diminuer les faux positifs.

Conclusion générale

Les solutions de triage sont très importantes dans l'écosystème agricole. Après avoir communiqué avec les exportateurs et les agriculteurs, nous avons détecté un grand potentiel de marché pour cette solution. Cependant, cette solution n'est pas facile à mettre en œuvre car son aspect transdisciplinaire rend difficile la recherche du profil et la communication nécessaires.

Dans ce projet, de nombreux problèmes ont été trouvés, surmonter ces problèmes permettrait d'améliorer le système :

- Le modèle pourrait être entraîné sur un ensemble de données plus important, ce qui introduirait de nouveaux modèles à détecter par le modèle, et pourrait donc améliorer la précision globale du système.
- J'aurais aimé travailler avec un ingénieur agronome lors de la recherche de ce projet, le problème aurait été plus clair à un stade plus précoce. Ce problème peut être résolu grâce à des données de source ouverte concernant la recherche agricole en ligne.
- Les données algériennes ne sont pas disponibles, la création et la sauvegarde de données pour la recherche est une bonne option et devrait être prise en considération.
- Le système pourrait être mis en œuvre de manière à faciliter le tri en temps réel lorsqu'il est complété par un équipement mécanique réel.

Reference bibliographie

[1] A. R. Mokrani, Interviewee, *Al iktissad al badil.* [Interview]. 03 2021./ www.youtube.com/watch?v=g?rhdE_6Rzk

[2] Benoît, A, «Le système visuel humain au secours de la vision par ordinateur,» Doctoral dissertation, Institut National Polytechnique de Grenoble-INPG, 2017.

[3] Ph. Bolon, J-M. Chassery, J-P. Cocquerez, D. Demigny, C. Graffigne, A. Montanvert, S. Philipp, R. Zéboudj, J. Zerubia, Analyse d'images : Filtrage et segmentation, 1995.

[4] Bergounioux, Maïtine, Quelques méthodes mathématiques pour le traitement d'image, 2009.

[5] Bergounioux.M, Introduction au traitement mathématique des images, Springer, 2015.

[6] Soissons, Alexandre Tauvy / Nicolas Carayon / Sebastien, «www.tsi.enst.fr,» 06 06 2021. [En ligne]. Available: http://www.tsi.enst.fr/pages/enseignement/ressources/mti/egal-histo/index.html.

[7] Isdant, Raphaël, traitement numérique de l'image, 2009.

[8] Ali, Benabdallah, «mise en oeuvre d'une technique automatique de segmentation de sillons corticaux,» Mémoire pour l'obtention du diplôme de Magister en Informatique 2011, 2011.

[9] LAOUAMER, LAMRI, «COMME EXIGENCE PARTIELLE DE LA MAÎTRISE EN MATHÉMATIQUES RT INFORMATIQUE APPLIQUÉES,» UNIVERSITÉ DU QUÉBEC, 2006.

[10] Babich, Nick, «www.xd.adobe.com,» 2020. [En ligne]. Available: https://xd.adobe.com/ideas/principles/emerging-technology/what-is-computer-vision-how-does-it-work/. [Accès le 09/06/2021].

[11] Levin, Golan, Computer vision for artists and designers: pedagogic tools and techniques for novice programmers, 2006.

[12] Mort, Andrew, «How Computer Vision Applications are Changing the World,» 12 03 2021. [En ligne]. Available: https://techsee.me/blog/computer-vision-applications/.

[Accès le 13 06 2021].

[13] Marr, Bernard, «7 Amazing Examples Of Computer And Machine Vision In Practice,» 2019. [En ligne]. Available: https://www.forbes.com/sites/bernardmarr/2019/04/08/7-amazing-examples-of-computer-and-machine-vision-in-practice/?sh=10c8e0721018. [Accès le 13 06 2021].

[14] Erickson, B. J., Korfiatis, P., Akkus, Z., & Kline,, Machine learning for medical imaging, Radiographics, 37(2), 505-515, 2017.

[15] Mbaabu, Onesmus, «Introduction to Random Forest in Machine Learning,» 11 12 2020. [En ligne]. Available: https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/. [Accès le 01 06 2021].

[16] Ray, Sunil, «Understanding Support Vector Machine(SVM) algorithm from examples (along with code),» 13 10 2017`. [En ligne]. Available: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/. [Accès le 03 06 2021].

[17] Gabriel Chartrand, Phillip M. Cheng, Eugene Vorontsov, Michal Drozdzal, Simon Turcotte, Christopher J. Pal, Samuel Kadoury, «Deep Learning: A Primer for Radiologists,» 13 11 2017. [En ligne]. Available: https://pubs.rsna.org/doi/full/10.1148/rg.2017170077. [Accès le 01 06 2021].

[18] TOUZET, Claude, LES RESEAUX DE NEURONES ARTIFICIELS, 1992.

[19] VANCAPPEL, Kévin, «Deep Learning : le Réseau neuronal convolutif (CNN),» 26 01 2021. [En ligne]. Available: https://fr.blog.businessdecision.com/tutoriel-deep-learning-le-reseau-neuronal-convolutif-cnn/. [Accès le 03 06 2021].

[20] Anwar, Aqeel, «“Difference between AlexNet, VGGNet, ResNet and Inception.,» 07 06 2019. [En ligne]. [Accès le 01 05 2021].

[21] Tsang, Sik-Ho, «Review: MobileNetV1 — Depthwise Separable Convolution (Light Weight Model),» 14 10 2018. [En ligne]. Available: https://towardsdatascience.com/review-mobilenetv1-depthwise-separable-convolution-light-weight-model-a382df364b69. [Accès le 13 06 2021].

[22] NVIDIA, «Jetson Nano Developer Kit,» [En ligne]. Available: https://developer.nvidia.com/embedded/jetson-nano-developer-kit. [Accès le 23 06 2021].

[23] Nvidia, Developer, «Developer Nvidia,» 08 06 2021. [En ligne]. Available: https://developer.nvidia.com/embedded/jetson-nano.

[24] Arduino, «Arduino,» [En ligne]. Available: https://www.arduino.cc/en/guide/introduction. [Accès le 25 06 2021].

[25] sparkfun, «Relay Shield v2.0,» [En ligne]. Available: https://www.sparkfun.com/products/retired/13769. [Accès le 2021].

[26] pololu, «DRV8825 Stepper Motor Driver,» [En ligne]. Available: https://www.pololu.com/product/2133. [Accès le 2021].

[27] ubuntu, «ubuntu,» [En ligne]. Available: https://ubuntu.com/. [Accès le 2021].

[28] Colaboratory., Google, «Google Colaboratory.,» 08 06 2021. [En ligne]. Available: colab.research.google.com.

[29] Python, «Python,» Python, [En ligne]. Available: https://www.python.org/. [Accès le 2021].

[30] OpenCV, «opencv,» 2021. [En ligne]. Available: https://opencv.org/about/.

[31] Numpy, «numpy,» numpy, [En ligne]. Available: https://numpy.org/. [Accès le 26 06 2021].

[32] Tensorflow, «tensorflow.org,» [En ligne]. Available: https://www.tensorflow.org/about.

[33] Keras, «deepai.org,» 2021. [En ligne]. Available: https://deepai.org/machine-learning-glossary-and-terms/keras.

[34] Nvidia, «Developer.nvidia.com,» 2021. [En ligne]. Available: https://developer.nvidia.com/tensorrt. [Accès le 2021].

[35] Uno, Arduino, «components101,» 2018. [En ligne]. Available: https://components101.com/microcontrollers/arduino-uno. [Accès le 2021].

[36] R. Isdant, traitement numérique de l’image, 2009.

Annexe 01

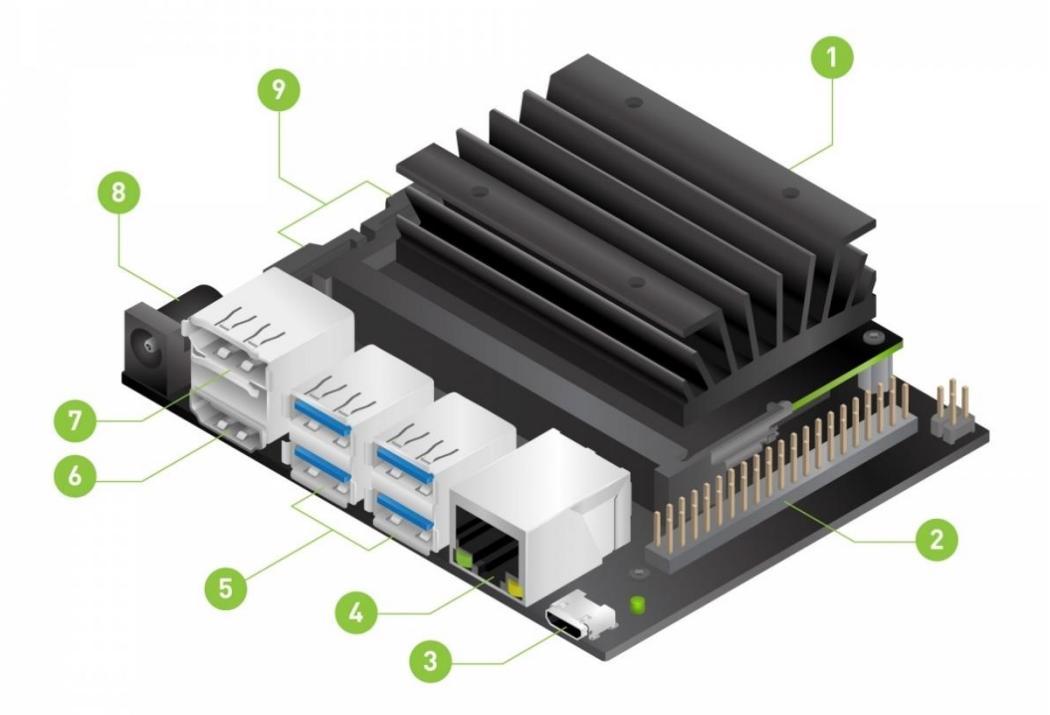


Figure 0-1Jetson Nano

1. microSD card slot for main storage
2. 40-pin expansion header
3. Micro-USB port for 5V power input, or for Device Mode
4. Gigabit Ethernet port
5. USB 3.0 ports (x4)
6. HDMI output port
7. DisplayPort connector
8. DC Barrel jack for 5V power input
9. MIPI CSI-2 camera connectors

Les spécifications sont énumérées ci-dessous :[23]

- GPU: NVIDIA Maxwell architecture with 128 NVIDIA CUDA® cores
- CPU: Quad-core ARM Cortex-A57 MPCore processor
- Memory: 4 GB 64-bit LPDDR4, 1600MHz 25.6 GB/s
- Camera: 12 lanes (3x4 or 4x2) MIPI CSI-2 D-PHY 1.1 (1.5 Gb/s per pair)
- Connectivity: Gigabit Ethernet, M.2 Key E

- Display: HDMI 2.0 and eDP 1.4
- USB: 4x USB 3.0, USB 2.0 Micro-B
- Others: GPIO, I2C, I2S, SPI, UART
- Mechanical 69.6 mm x 45 mm

Annexe 2

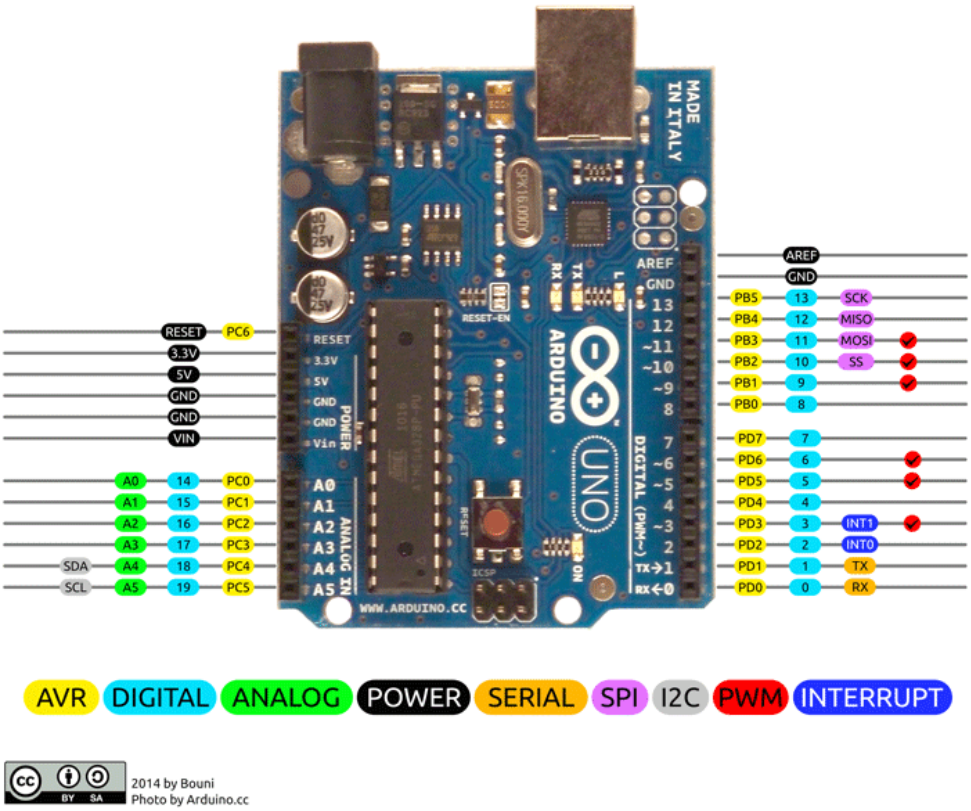


Figure 0-1 ARduino Uno

Description de la broche[35]

Pin Category	**Pin Name**	**Détails**
Power	Vin, 3.3V, 5V, GND	Vin : Tension d'entrée de l'Arduino lorsqu'il utilise une source d'alimentation externe. 5V : Alimentation régulée utilisée pour alimenter le microcontrôleur et les autres composants de la carte. 3.3V : Alimentation 3.3V générée par le régulateur de

		tension embarqué. Le courant maximum consommé est de 50mA. GND : broches de mise à la terre.
Analog Pins	A0 – A5	Utilisé pour fournir une entrée analogique dans la gamme de 0-5V
Input/Output Pins	Digital Pins 0 - 13	Peut être utilisé comme broche d'entrée ou de sortie.
Serial	0(Rx), 1(Tx)	Utilisé pour recevoir et transmettre des données série TTL.