الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique

Mémoire de Master

Filière : Électronique
Spécialité : Systèmes Embarqué

Présenté par

BOUMEHIRA OUSSAMA
&
FERHAT ABDELFATAH

Détection et reconnaissance de pièces de monnaies algériennes par l'apprentissage profond

Proposé par : Pr. NAMANE ABDERRAHMANE

Année Universitaire 2021-2022

Remerciements

D'abord et avant tout, Nous remercions Dieu le tout puissant qui nous a donné la volonté et le courage pour réaliser ce travail.

Nous remercions vivement notre encadreur Mr Namane Abderrahmane pour avoir dirigé ce mémoire.

Nous avons eu le plaisir de travailler sous votre direction. Nous vous remercions pour votre gentillesse et spontanéité avec lesquelles vous avez dirigé ce travail, ainsi que pour votre disponibilité et vos conseils que grâce à eux nous avons pu atteindre notre objectif.

Nous espérons que votre confiance que vous nous accordez et que ce mémoire est à la hauteur de vos espérances.

Nous remercions également, notre chef de spécialité Mme Naceur Djamila qui nous a accepté pour être parmi ses étudiants au cours de ces deux années académiques dans la spécialité électronique des systèmes embarqués.

Nous remercions enfin, tous les enseignants du cycles License et Master et toutes les personnes qui nous ont soutenus de près ou loin dans la réalisation de ce travail.

:ملخص

الهدف من هذا المشروع هو إنشاء نظام قادر على الكشف و التعرف على العملات النقدية الجزائرية باستخدام التعلم العميق و تقنيات معالجة الصور. هذا المشروع الخاص بالتخرج تم تقسيمه الى قسمين ، القسم الأول يستعمل التعلم العميق باستعمال نموذج YOLOV4 للكشف و التصنيف ، و القسم الثاني يستعمل تقنيات معالجة الصور بهدف حساب قطر كل قطعة نقدية. تمت العمل على هذا المشروع من خلال صور تم التقاطها بكاميرا من نوع LOGITECH C920.

الكلمات المفتاحية:

التعلم العميق، العملات المعدنية، معالجة الصور,YOLO V4 .

Résumé : L'objectif de ce projet est de réaliser un système capable de faire la détection et la reconnaissance des pièces monnaies algériennes à l'aide de l'apprentissage profond et les techniques de traitement d'image. Ce projet de fin d'études est divisé en deux parties ; la première partie utilise l'apprentissage profond est basée sur le YOLO V4 pour la détection et classification, quant à la deuxième partie, elle vise l'utilisation du traitement d'image qui a comme objectif le calcul du diamètre de chaque pièce. Ce travail a été réalisé en utilisant comme moyen d'acquisition d'images la caméra LOGITECH C920e.

Mot clé :

YOLO V4, apprentissage profond, pièces de monnaies, traitement d'image.

Abstract : This project aims to créate a system capable of detecting and recognizing Algerian coins using deep learning and image processing techniques. This graduation project is divided into two parts; the first part uses deep learning based on the YOLO V4 for detection and classification, and the second part aims to use image processing which aims to calculate the diameter of each coin. This work was carried out using the LOGITECH C920e camera as the means of image acquisition.

Keyword

YOLO V4, Deep Learning, Coins detection, image processing

Listes des acronymes et abréviations

YOLO : You Only Look Once

RNA : Réseaux neurone artificiel

ML : Machine Learning

CNN : Convolutional Neurone Network

OCR : Optical Character Recognition

R-CNN : Region-based Convolutional Neural Network

SSD : Single Shot MultiBox Detector

IA : Intelligence Artificiel

ConvNet : Convolutional Neural Network

RNN : Recurrent Neural Network

SVM : Support vector machines

COCO : Common Objects In Context

IOU : Intersection Over Union

NMS : Non Max Suppression

SPP : spatial pyramid pooling

PAN : Path Integral Based Convolution

CSPNet : Cross Stage Partial Network

BoF : Bag of Freebies

BoS : Bag of Specials

FCN : fully convolutional network

IPS : Image par Seconde

AP : Average Precision

GPU : Graphics processing unit

FPS : Frame Per Second

Map : mean Average Precision

OpenCv : Open Source Computer Vision Library

CVAT : Computer Vision Annotation Tool

IDE : Integrated Development Environment

CFG : Configuration

VCS : Version Control System

Sommaire

Introduction générale 01

Chapitre 1 : Aperçu général des pièces monnaies.....03

1.1 Introduction 03

1.2 Pièces de monnaies 03

1.3 Historique des pièces de monnaie 04

1.3.1 Pièces dans le monde entier.....04

1.3.2 Pièces algériennes.....06

1.4 Pièces algériennes actuellement utilisées 09

1.5 Systèmes de reconnaissance des pièces utilisés sur le marché.....13

1.5.1 Systèmes basés sur la méthode Mécanique.....14

1.5.2 Systèmes basés sur la méthode Électromagnétique.....14

1.5.3 Systèmes basés sur le traitement Image.....15

1.6 Travaux réalisés sur la reconnaissance des pièces.....16

1.6.1 Système de reconnaissance de pièces de monnaie utilisant un RNA.....16

1.6.2 Détection de pièces en temps réel avec ML approche sur appareil iOS..16

1.6.3 Reconnaissance des pièces malaisiennes basées à l'aide de l'apprentissage Profond.....16

1.6.4 Développement d'un réseau neuronal précis pour la reconnaissance des pièces.....17

1.6.5 Reconnaissance des pièces de monnaie indiennes fondées sur l'apprentissage Profond.....17

1.7 Conclusion.....17

Chapitre 2 : Traitement d'image et réseau de neurone.....18

2.1 Introduction.....18

2.2 Techniques de traitement d'image.....18

2.2.1 Conversion d'image couleur en niveau de gris.....19

2.2.2 Seuillages d'une image.....19

2.2.3 Filtrage Gaussien.....22

2.2.4 Filtres morphologiques.....23

2.2.5 Détection de contour.....25

2.3 Réseaux de neurones.....25

2.3.1 Neurone biologique.....25

2.3.2 Neurone artificiel.....26

2.3.3 Architecture des réseaux de neurones.....29

2.3.4 Apprentissage.....30

2.4 Conclusion.....32

Chapitre 3 : la détection des objets à l'aide de l'apprentissage profond 33

3.1 Introduction 33

3.2 L'intelligence artificielle (IA).....33

3.3 Apprentissage profond.....34

3.4 Architectures de l'apprentissage profond.....35

3.4.1 Réseaux neuronaux convolutifs (CNN).....35

3.4.2 Réseaux neuronaux récurrents (RNN).....36

3.5 Détection d'objets à l'aide d'apprentissage profond.....37

3.5.1 Définition.....37

3.5.2 Les Modèles de Détection 38

3.6 Détection d'Object avec YoloV4 41

3.6.1 Yolov4 Architecture 41

3.6.2 Sac de cadeaux et sac de promotions (bof & bos).....45

3.6.3 Fonctionnements de Yolov4.....47

3.6.4 Avantages et inconvénients du YOLOv4.....55

3.7 Conclusion.....56

Chapitre 4 : Implémentation et conception 57

4.1 Introduction 57

4.2 Environnement de travail.....57

4.2.1 Spécifications de l'ordinateur portable et le Webcam.....57

4.2.2 Les Logiciels utilisé.....58

4.2.3 Les bibliothèques utilisé.....59

4.3 Implémentation du système.....60

4.3.1 Détection à l'aide de YOLO v4.....60

4.3.2 Calcul du diamètre.....71

4.3.3 Combinaison des deux systèmes d'indentification 79

4.4 discussions des résultats.....83

4.5 Création de l'interface graphique.....84

4.6 Conclusion.....87

Conclusion générale.....88

Bibliographie.....89

Liste des figures

Figure 1.1: Pièces mondiales.....04

Figure 1.2: Pièce d'Euthydemos d'Agathokles.....04

Figure 1.3: Miletos Electrum Stater.....05

Figure 1.4: Pièces de monnaie algériennes anciennes.....06

Figure 1.5: 2 boudjous-Mahmoud II 1821-1829.....07

Figure 1.6: ½ boudjou - Abdel Kader 1834-1847.....07

Figure 1.7: 100 francs en 1950.....08

Figure 1.8: 1, 2, 5 santeem en aluminium 1964.....08

Figure 1.9: 10, 20, 50 santeem en bronze-aluminium 1964.....08

Figure 1.10: La face de 1 dinar contient le drapeau de l'Algérie et le dos contiennent la valeur 1964.....09

Figure 1.11: Pièce algérienne de 200 dinars 2012.....09

Figure 1.12: Trieur automatique amélioré par Bikecyclist.....14

Figure 1.13: COMPTEUR AUTOMATIQUE DE PIÈCES MODÈLE DB-350 Series.....15

Figure 1.14: Les techniques de traitement d'image.....15

Figure 2.1: Image niveau gris 19

Figure 2.2: Seuillage d'une image à niveau de gris 20

Figure 2.3: Application du filtre gaussien 23

Figure 2.4: Ensemble de formes… 23

Figure 2.5: Détection du contour pièces monnaies 25

Figure 2.6: Un neurone avec son arborisation dendritique.....26

Figure 2.7: Neurone formel de Mac Culloch et Pitts 26

Figure 2.8: Passage du neurone biologique vers le neurone formel 28

Figure 2.9: Fonction d'activation d'un neurone formel 28

Figure 2.10: Fonctions d'activation possibles d'un neurone formel 29

Figure 2.11: Réseau de neurone monocouche 30

Figure 2.12: Réseau de neurone multicouche ……..30
Figure 2.13: Apprentissage supervisé ……..31
Figure 2.14: Apprentissage non supervisé ……..32
Figure 3.1: Concept de l'intelligence artificielle ……..34
Figure 3.2: Classification du chat et chien a l'aide de l'apprentissage profond ……..34
Figure 3.3: Modèle linière et non-Linéaire ……35
Figure 3.4: Réseaux neuronaux convolutifs ……36
Figure 3.5: Réseaux neuronaux récurrents ……37
Figure 3.6: le modèle R-CNN ……38
Figure 3.7: le modèle SSD ……39
Figure 3.8: le model YOLO ……40
Figure 3.9: Architecture de YOLO v4 ……41
Figure 3.10: l'architecture de cspdaknet53……42
Figure 3.11: Spatial Pyramid Pooling ……43
Figure 3.12: modification de pan ……44
Figure 3.13: Schéma fonctionnel de YOLOv4……45
Figure 3.14: Exemples des augmentations des données ……46
Figure 3.15: les additions de BoS ……46
Figure 3.16: Les différents modules/méthodes de BoF et BoS ……47
Figure 3.17: La structure globale de YOLOv4……48
Figure 3.18: Diviser l'image en SxS grille ……48
Figure 3.19: le vecteur prédit dans le cas d'une seule boite ……49
Figure 3.20: intersection sur union ……50
Figure 3.21: Example de l'intersection sur union ……50
Figure 3.22: Exemple d'un image ou le centre de deux objet dans la même cellule…..51
Figure 3.23: vecteur prédit dans le cas de plusieurs boites dans la cellule ……51
Figure 3.24: tenseur spécifiant les emplacements de la boite englobante et probabilités de classe……52

Figure 3.25: les résultats de suppression non maximale ……..53
Figure 3.26: les déférentes échelle de détection ……..55
Figure 3.27: performances de yolov4 par fps et AP ……..55
Figure 4.1 : Quelques échantillons d'images de la base de données utilisée……..62
Figure 4.2 : Etapes d'étiquetage sur CVAT……63
Figure 4.3 : enregistrement de l'étiquetage sous format yolo1.1 dans CVAT……63
Figure 4.4 : Example et explication de format yolo1.1 généré par CVAT……64
Figure 4.5 : Types des augmentations a Roboflow……64
Figure 4.6 : cloner le référentiel darknet dans le collab……65
Figure 4.7 : importation des fichier obj.names et obj.data……66
Figure 4.8 : importation le fichier yolov4 cfg ……66
Figure 4.9 : Fichier cfg de yolo v4……67
Figure 4.10 : la division du base de donnes pour (train/ valid/ test)……68
Figure 4.11 : téléchargement de fichier de poids(.weights) pré-entraîné de yolov4……68
Figure 4.12 : graphe de la performance de notre entraînement de yolo v4……69
Figure 4.13 : Performance de notre entraînement de YOLO v4……69
Figure 4.14 : Précision obtenue pour chaque classe……70
Figure 4.15 : Graphe montrant les performances du système proposé……71
Figure 4.16 : Mesure manuel de diamètre de pièce de monnaie à l'aide de Pied à coulisse……72
Figure 4.17 : Ecarte type de diamètre de chaque pièce……73
Figure 4.18 : Conversion de l'image couleur en niveaux de gris……73
Figure 4.19 : application d'un filtre gaussien……74
Figure 4.20 : Binarisation de l'image filtrée ……74
Figure 4.21 : application de fermeture morphologique……75
Figure 4.22 : application de l'ouverture morphologique ……75
Figure 4.23 : Diagramme de L'architecture du système développé……79
Figure 4.24 : Exemple d'image d'entre……80
Figure 4.25 : Résultat obtenu par le Yolov4……80
Figure 4.26 : Recadrer de la pièce et augmentation de la surface de l'image……81

Figure 4.27 : Résultat après la binarisation et fermeture, ouverture pour calculer la surface……81
Figure 4.28 : Résultats final obtenu après l'étape de comparaison……82
Figure 4.29 : Résultat final après la reconnaissance de la pièce……82
Figure 4.30 : Page d'accueil de L'interface graphique de notre system……84
Figure 4.31 : Choix d'une image pour identification de la pièce……85
Figure 4.32 : Résultat d'identification pour des images fixes……85
Figure 4.33 : Emplacement de la pièce référence à l'endroit voulu et cliquer sur S……86
Figure 4.34 : Identification d'une pièce de 100da……86
Figure 4.35 : Rejet d'une fausse pièce ……87

Liste des tableaux

Tableau 1.1: Tableau de pièces de monnaies algériennes et leurs spécifications……10

Tableau 2.1: Passage du neurone biologique vers le neurone formel……..27

Tableau 4.1: le nombre des images nous avons prendre pour chaque pièce……61

Tableau 4.2 : Tableau de résultats obtenus pour l'ensemble de test.……..70

Tableau 4.3: Tableau contenant les résultats de test pour les fausses pièces ……..71

Tableau 4.4: Mesures de diamètres des pièces à l'aide de Pied a coulisse……72

Tableau 4.5: Résultats de test mettant en évidence le diamètre de la pièce de monnaie ……..78

Tableau 4.6: Tableau contenant les résultats de test pour les fausses pièces ……78

Tableau 4.7: Tableau contenant les résultats de test par pièces ……..83

Tableau 4.8: Tableau contenant les résultats de test par les fausses pièces ……83

Introduction générale

La détection d'objets est une technique de vision par ordinateur qui nous permet d'identifier et de localiser des objets dans une image ou une vidéo. Avec ce type d'identification et de localisation, la détection d'objets peut être utilisée pour compter les objets dans une scène et déterminer et suivre leurs emplacements précis, tout en les étiquetant avec précision. Elle permet d'utiliser des caméras comme capteurs pour détecter des défauts sur des objets manufacturés, inspecter des pièces en cours de fabrication, les compter, les trier, les classer, les mesurer, ... à partir de leur apparence visuelle.

Plus précisément, la détection d'objets dessine des cadres autour de ces objets détectés, qui nous permettent de localiser où ces objets se trouvent (ou comment ils se déplacent) dans une scène donnée, La détection d'objets est souvent confondue avec la reconnaissance d'images.

La reconnaissance d'image attribue une étiquette à une image. Une photo d'une pièce de 50 DA reçoit l'étiquette « 50 DA ». Une photo de deux pièces de 50 DA et 100 DA reçoit l'étiquette « 50 DA » et « 100 DA » respectivement. La détection d'objets, par contre, dessine une boîte autour de chaque pièce de 50 DA et étiquette la boîte « 50 DA ». Le modèle prédit où se trouve chaque objet et quelle étiquette doit être appliquée. De cette façon, la détection d'objets fournit plus d'informations sur une image que la reconnaissance.

La détection d'objets peut être décomposée en approches basées sur l'apprentissage automatique et en approches basées sur l'apprentissage profond.

Notre objectif consiste à résoudre le problème du tri des pièces en Algérie en créant un système qui nous aide à détecter et à reconnaître un volume élevé de pièces à haute vitesse et précision en utilisant Open Cv (Traitement d'image) et Yolo V4 l'apprentissage profond).

La mémoire contient quatre différents chapitres :

Le 1er chapitre présente les pièces monnaies de façon générale, donner les types utilisés en Algérie et présenter les différent systèmes et travaux à propos ce thème.

Le 2éme chapitre, Introduit les techniques du traitement d'image utilisées dans notre système pour détecter la pièce monnaie, ainsi définir le réseau neurone monocouche et multicouche.

Le 3éme chapitre présente l'intelligence artificielle et l'apprentissage profond tout en expliquant le modèle Yolo V4 d'une maniéré plus détaillée.

Le 4éme chapitre, détaille le système proposé et montre les étapes du système développé tout en discutant les résultats obtenus.

Enfin, une conclusion générale pour résumer les résultats obtenus et les perspectives.

Chapitre 1 Aperçu général des pièces monnaies

1.1 Introduction

L'argent est l'un des moyens les plus largement utilisés à travers les âges, malgré les développements dans la sphère économique concernant les transactions financières, également un moyen important dans la vie de dépenser divers besoins spéciaux, que ce soit l'achat, la vente ou le commerce de biens.

Les gens ont recours au stockage d'espèces au lieu de marchandises pendant un certain temps et les déboursent à tout moment en raison de leur valeur fixe et de leur non-dommage. Dans le même temps, des programmes et des systèmes ont été développés pour reconnaître les devises et les compter afin d'éviter diverses opérations de fraude.

En outre, pour faciliter le fonctionnement des magasins et des diverses installations qui possèdent d'énormes quantités de devises.

Dans ce premier chapitre, nous allons définir les pièces de monnaie, parler de l'histoire des pièces au niveau mondial et national et des types de pièces algériennes utilisées aujourd'hui et enfin nous discuterons les différents systèmes de reconnaissance des pièces.

1.2 Pièces de monnaies

Une pièce de monnaie est une petite pièce de métal ronde (généralement, selon le pays ou la valeur) utilisée principalement comme moyen d'échange ou de monnaie légale. Elle est standardisée en poids et produit en grande quantité afin de faciliter les échanges, et peuvent être utilisés principalement comme symbole de la monnaie légale du commerce dans une région, un pays ou un territoire donné (voir la figure 1.1).

Les pièces de monnaie sont faites de cuivre, de zinc et de nickel. Le ratio de cuivre est généralement plus élevé et dans certains pays, la pièce est faite d'étain et d'aluminium et certains utilisent de l'argent et de l'or.

Figure 1.1 Pièces mondiales [1]

1.3 Historique des pièces de monnaie

1.3.1- Pièces dans le monde entier

La durée de vie des pièces s'étend de l'Antiquité à nos jours, et les pièces sont liées à l'histoire économique, à l'histoire des techniques d'instruments de la pièce, comme le montrent les images imprimées sur les pièces et la date de la collecte de la monnaie. Les devises sont encore largement utilisées à des fins monétaires et autres [2].

Les pièces de monnaie ont été inventées vers le 6ème ou 5ème siècle avant JC. Introduit comme mode de paiement. L'invention de la pièce est encore un mystère. Selon Herdotous (I, 94), la pièce a d'abord été coulée par les Lydiens, mais Aristote affirme que la première pièce a été coulée par les Desmoïdes de Sirme, épouse du roi Midas de Phrygie. Les numismates supposent que les premières pièces de monnaie sur l'île d'Égine en Grèce ont été inventées par un souverain local ou le roi Pheidon d'Argos (voir figure 1.2).

Figure 1.2 Pièce d'Euthydemos d'Agathokles [2]

Égine, Samos et Milet ont tous frappé des pièces de monnaie pour les Égyptiens, par l'intermédiaire du poste de traite grec de Naucratis dans le delta du Nil. Il est certain que lorsque la Lydie a été conquise par les Perses en 546 avant notre ère, les pièces de monnaie ont été introduites en Perse. Les Phéniciens n'ont frappé aucune pièce de monnaie avant le milieu du Ve siècle avant notre ère, qui s'est rapidement propagée aux Carthaginois qui ont frappé des pièces de monnaie en Sicile. Les Romains n'ont commencé à frapper des pièces de monnaie qu'à partir de 326 avant notre ère.

Les pièces de monnaie ont été apportées en Inde par l'intermédiaire de l'Empire achéménide, ainsi que des royaumes successeurs d'Alexandre le Grand. Surtout les royaumes indo-grecs ont frappé des pièces de monnaie (souvent bilingues) au 2ème siècle avant notre ère. On dit les plus belles pièces de monnaie de l'âge classique.

D'avoir été frappé par Samudragupta (335-376 CE), qui s'est dépeint à la fois comme conquérant et musicien.

Figure 1.3 Miletos Electrum Stater [2]

La première pièce était faite d'électrum (voir figure 1.3), un alliage d'argent et d'or. De nombreuses pièces lydiennes précoces semblent avoir été inventées par les marchands comme jetons pour les transactions commerciales. L'État de Lydien a également coulé des pièces de monnaie, dont la plupart se réfèrent au roi Alyattes de Lydie. Certaines pièces lydiennes ont une soi-disant légende, une sorte de dédicace. Un exemple célèbre trouvé à Kalia dit : « Je suis un badge Phanes. » On ne sait toujours pas qui était Phanes.

En Chine, les pièces d'or ont été normalisées pour la première fois sous la dynastie Qin (221-207 av. J.-C.). Après l'effondrement de la dynastie Qin, l'empereur Han a ajouté deux autres offres légales. Pièces d'argent et « billets de banque en cuir de cerf », le prédécesseur des chinois a inventé les billets de banque [2].

1.3.2 Pièces algériennes

L'État algérien a une longue histoire. Au cours des siècles passés, la capitale de l'Algérie a été considérée comme le site des colonies phéniciennes et romaines.

Ce fut le début de l'instrument des pièces royales, l'or, ou le dinar algérien localement dans la capitale dans les années 1520, peu de temps après que les pirates turcs ont pris le contrôle d'Alger, Grâce à ses liens commerciaux et à l'accès de l'Algérie à l'or du désert, cela a conduit, que l'Algérie est rapidement devenue l'un des principaux sites de production royale.

L'invasion de l'Occident par les Espagnols par Oran en 1509, et l'invasion de l'Algérie par les Ottomans en 1516.

Puis, dans la seconde moitié du seizième siècle, la Monnaie a été construite en Algérie, qui a travaillé à émettre des pièces d'or au nom des sultans ottomans après le règne nominal, mais ils ne les ont pas respectées.

La conception générale des pièces et télescopes royaux standard a suivi le modèle du XIIe siècle, et est devenue plus tard la base de toutes les pièces d'or marocaines jusqu'au seizième siècle [3].

D'autre part, l'émission de pièces d'argent carrées appelées ; Batlac, Wak a été limitée en Algérie jusqu'au dix-huitième siècle, et les autorités locales de la capitale algérienne ont commencé à émettre des pièces d'argent plus grandes dans la deuxième décennie du dix-huitième siècle(voir figure 1.4). [3].

Figure 1.4 Pièces de monnaie algériennes anciennes [4]

La croissance des exportations et la disponibilité accrue de certains types sont des raisons pour lesquelles l'État algérien a connu une grande stabilité monétaire pour le reste du siècle dans les années 1830, la pièce royale a été remplacée par environ 9,5 francs français, ou 8,5 Batlak, qui était la monnaie d'argent originale de l'Algérie, l'argent pur, chacun contenant environ cinq grammes d'argent pur.

L’émission de monnaies d’argent en provenance d’Algérie, appelée Boudjous (voir figure 1.5), avait commencé, et équivalait à trois vracs d’argent, ou vingt-quatre milliards de mazuma, en 1829.[3].

Figure 1.5 2 boudjous-Mahmoud II 1821-1829 [4]

L’Algérie a également émis des pièces de cuivre, surnommées Haruba par les Européens, qui étaient initialement en fractures akça (désigne une pièce d’argent qui était l’unité monétaire principale de l’Empire ottoman. Akça moyen turc a évolué à partir du mot "argent ou argent argent), mais avec la dépréciation de akça, cela a augmenté la valeur nominale du cuivre.

Les huit vraies pièces espagnoles, le soi-disant requin, ont ensuite émergé comme un échange de premier plan en Algérie au cours de la première partie du 17ème siècle. L’occupation française de l’Algérie, qui commença en 1830, ne fut achevée qu’en 1848 à cause de la lutte de la résistance locale. Pendant cette période, la nouvelle monnaie a été ouverte à Constantine et Médéa, la maison de présentation de l’instrument, et le camp ont produit des pièces de style ottoman au nom de la Résistance Abdul Qadir [3] (voir figure 1.6).

Figure 1.6 ½ boudjou - Abdel Kader 1834-1847 [4]

En 1848, le Pedjo n'était plus considéré comme la monnaie locale de l'Algérie avec l'occupation Française, puisqu'en 1959 l'inflation économique nécessitait un nouveau franc (voir figure 1.7), égal à cent vieux francs, et divisé en cent cents [3].

Figure 1. 7 100 francs en 1950 [4]

Malgré l'indépendance de l'Algérie, en 1962, l'utilisation du nouveau franc en Algérie a continué jusqu'en 1964, où le franc a été remplacé par le dinar, où le dinar a été émis pour la première fois le 1er avril pour l'année 1964, et a été divisé en 100 santeem.

Pièces d'une catégorie, deux, cinq santeem, en aluminium (voir figure 1.8), dix catégorie, vingt, cinquante santeem, en bronze plus aluminium (voir figure 1.9), et un dinar était fait de cupro-nickel (voir figure 1.10). [3].

Figure 1.8 1, 2, 5 santeem en aluminium 1964[4]

Figure 1.9 10, 20, 50 santeem en bronze-aluminium 1964 [4]

Le visage du dinar représentait le logo de l'État algérien, tandis que le verso, les valeurs du dinar, représentaient les figures arabes orientales. Plus tard, des pièces commémoratives ont été émises en Algérie à partir de divers sujets. Néanmoins, des pièces d'une catégorie un, deux, cinq, dix et vingt santeem ont été produites pour la

dernière fois dans les années 1980, ci-dessous la pièce de un dinar qui a été fabriqué en 1964 (voir figure 1.10).

Figure 1.10 La face de 1 dinar contient le drapeau de l'Algérie et le dos contiennent la valeur 1964 [4]

La Banque algérienne a également émis une nouvelle série de devises dans la catégorie dinar, un, deux, cinq, dix, vingt, cinquante et deux cents dinars en raison de l'inflation. Depuis, les pièces de la plus haute valeur de la chaîne, à savoir deux cents dinars, ont été produites en 2012 (voir la figure 1.11) [3].

Figure 1.11 Pièce algérienne de 200 dinars 2012 [4]

1.4 Pièces algériennes actuellement utilisées

De nos jours, il existe 8 types de pièces utilisées en Algérie et ces pièces représentent 6 valeurs différentes : 5, 10, 20, 50, 100 et 200 dinars, nous allons maintenant discuter les spécifications de chaque type telles que leur poids et leur diamètre ainsi que des matériaux de fabrication (voir tableau 1.1).

 	**Type 1 :** Année de réalisation : **1972** Valeur : **5 dinars** Poids : **12 g** Diamètre : **31 mm** Composition : **Argent 75%** Forme : **Rond**
 	**Type 1 :** Année de réalisation :**1974** Valeur : **5 dinars** Poids : **12 g** Diamètre : **31 mm** Composition : **Nickel** Forme : **Rond**
 	**Type 1 :** Année de réalisation : **1984** Valeur : **5 dinars** Poids : **12 g** Diamètre : **31mm** Composition : **Nickel** Forme : **Rond**
 	**Type 2 :** Année de réalisation :**1992** Valeur : **5 dinars** Poids : **6.15 g** Diamètre : **24.65 mm** Composition : **Acier inoxydable** Forme : **Rond**
 	**Type 3 :** Année de réalisation : **1979** Valeur : **10 dinars** Poids : **11.37 g** Diamètre : **29,3 mm** Composition : **Bronze-aluminium** Forme : **Décagonal (10 côtés)**

 	**Type 4 :** Année de réalisation : **1992** Valeur : **10 dinars** Poids : **4.95 g** Diamètre : **26,5 mm** Composition : **Bimétallique (centre aluminium-magnésium et bague en acier inoxydable)** Forme : **Rond**
 	**Type 5 :** Année de réalisation :**1992** Valeur : **20 dinars** Poids : **8.62 g** Diamètre : **27,5 mm** Composition : **Bimétallique (centre en laiton et bague aluminium-bronze)** Forme : **Rond**
 	**Type 6 :** Année de réalisation :**1992** Valeur : **50 Dinars** Poids : **9.27 g** Diamètre : **28,5 mm** Composition : **Bimétallique (centre en acier inoxydable et bague en aluminium bronze)** Forme : **Rond**
 	**Type 6 :** Année de réalisation : **1994** Valeur : **50 Dinars** Poids : **9.26 g** Diamètre : **28,5 mm** Composition : **Bimétallique (centre en acier inoxydable et bague en aluminium bronze)** Forme : **Rond**

	**Type 6 :** Année de réalisation : **2004** Valeur : **50 Dinars** Poids : **9.3 g** Diamètre : **28,5 mm** Composition : **Bimétallique (centre en acier inoxydable et bague en aluminium bronze)** Forme : **Rond**
	**Type 7 :** Année de réalisation : **1992** Valeur : **100 dinars** Poids : **11 g** Diamètre : **29,5 mm** Composition : **Bimétallique (centre en aluminium bronze et bague en acier inoxydable)** Forme : **Rond**
	**Type 7 :** Année de réalisation : **2002** Valeur : **100 dinars** Poids : **11 g** Diamètre : **29,5 mm** Composition : **Bimétallique (centre en aluminium bronze et bague en acier inoxydable)** Forme : **Rond**
	**Type 7 :** Année de réalisation :**2018** Valeur : **100 dinars** Poids : **11 g** Diamètre : **29,5 mm** Composition : **Bimétallique (**centre **cupronickel et bague en acier inoxydable)** Forme : **Rond**

	**Type 7 :** Année de réalisation : **2021** Valeur : **100 dinars** Poids : **11 g** Diamètre : **29,5 mm** Composition : **Bimétallique (centre cupronickel et bague en acier inoxydable)** Forme : **Rond**
	**Type 8 :** Année de réalisation :**2012** Valeur : **200 dinars** Poids : **12 g** Diamètre : **28 mm** Composition : **Bimétallique (centre cupro-aluminium-nickel et anneau cupronickel)** Forme : **Rond**
	**Type 8 :** Année de réalisation : **2020** Valeur : **200 dinars** Poids : **12 g** Diamètre : **28 mm** Composition : **(centre cupro-aluminium-nickel & bague cupronickel)** Forme : **Rond**

Tableau 1.1 Tableau de pièces de monnaies algériennes et leurs spécifications [4]

1.5 Systèmes de reconnaissance des pièces utilisés sur le marché

Sur le marché, il existe trois types de base de systèmes de reconnaissance de pièces de monnaie qui utilisent différentes méthodes :

A-Systèmes basés sur la méthode Mécanique

B-Systèmes basés sur la méthode Électromagnétique

C-Systèmes basés sur le traitement Image

1.5.1 Systèmes basés sur la méthode Mécanique

Les systèmes basés sur des méthodes mécaniques distinguent les pièces en utilisant des paramètres tels que le diamètre ou le rayon, l'épaisseur, le poids et le magnétisme. Cependant, ces paramètres ne peuvent pas être utilisés pour différencier les différents matériaux de la pièce. Cela signifie que si nous donnons deux pièces, une originale et une fausse, avec le même diamètre, l'épaisseur, le poids et le magnétisme mais des matériaux différents à un système de reconnaissance de pièces basé sur une méthode mécanique, il traitera les deux comme des pièces originales, rendant ces systèmes facilement trompés (voir la figure 1.12) [5].

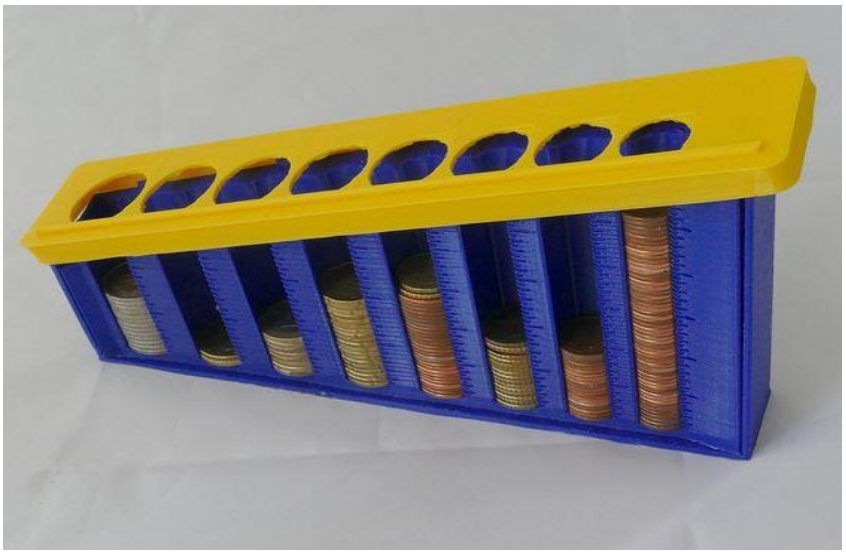


Figure 1.12 Trieur automatique amélioré par Bikecyclist

1-5-2 Systèmes basés sur la méthode Électromagnétique

Les systèmes basés sur la méthode électromagnétique peuvent distinguer différents matériaux parce que les pièces sont passées à travers une bobine oscillante à une fréquence spécifique dans ces systèmes, et différents matériaux provoquent des changements différents dans l'amplitude et la direction de la fréquence (voir la figure 1.13). En conséquence, ces variations, ainsi que d'autres paramètres tels que le diamètre, l'épaisseur, le poids et le magnétisme, peuvent être utilisés pour distinguer les pièces. Bien que les systèmes de reconnaissance de pièces basés sur la méthode électromagnétique améliorent la précision de la reconnaissance, ils peuvent toujours être trompés par certaines pièces de jeu [5].

Figure 1.13 COMPTEUR AUTOMATIQUE DE PIÈCES MODÈLE DB-350 Series

1-5-3 Systèmes basés sur le traitement Image

Les systèmes de reconnaissance de pièces basés sur des images sont également arrivés sur la scène ces dernières années (voir la figure 1.14). Dans ces systèmes, l'image de la pièce à reconnaître est capturée en premier, soit par caméra, soit par balayage. Les images sont ensuite traitées à l'aide de diverses techniques de traitement d'image telles que FFT, Ondelettes de Gabor, DCT, détection de bord segmentation, soustraction d'image, et diverses caractéristiques sont extraites des images. Différentes pièces sont ensuite reconnues en fonction de ces caractéristiques [5].

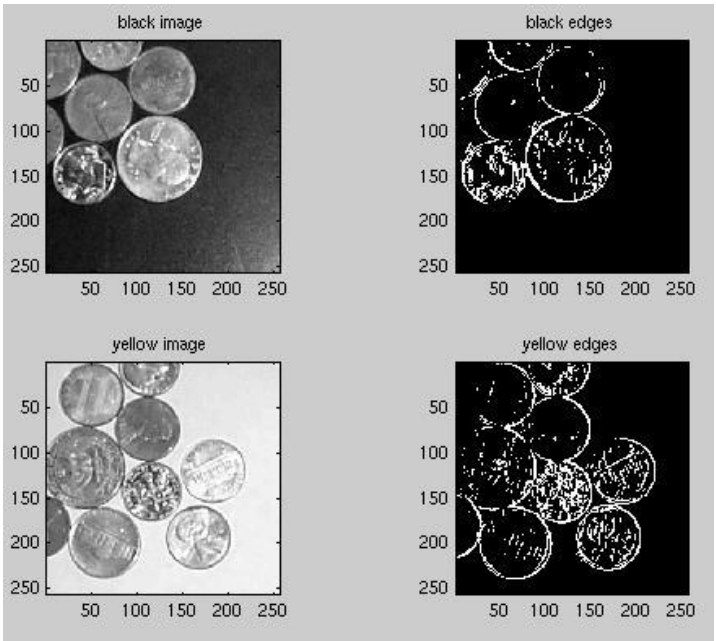


Figure 1. 14 Les techniques de traitement d'image

1.6 Travaux réalisés sur la reconnaissance des pièces

Différents travaux de recherches ont été réalisés dans l'identification de pièces de monnaies. Parmi ces travaux nous pouvons citer cinq travaux qui consistaient à la détection et reconnaissance des pièces monnaies.

1.6.1 Système de reconnaissance de pièces de monnaie utilisant un RNA

Dr. Seema Bawa, Shatrughan Modi [1] a développé un Système automatisé de reconnaissance des pièces de monnaie basé les réseau neurones artificiels (RNA). Ce travail est utilisé pour la reconnaissance de pièces indiennes avec invariance de rotation. Ce système est capable de reconnaître les pièces des deux côtés. Les caractéristiques extraites de l'image de la pièce, utilisent diverses techniques telles que la transformée de Hough Transformation, le calcul de la moyenne des tendances, …etc. Après avoir entrainé un RNA avec les caractéristiques extraites comme entrées, un taux de reconnaissance de 97,74 % a été atteint pour un ensemble test. Cela signifie que leur système commet seulement une erreur de 2,26 %, ce qui est très encourageant [6].

1.6.2 Détection de pièces en temps réel avec ML approche sur appareil iOS

Nataliya Didukh et Ihor Zhvanko [7] ont développé un modèle d'apprentissage automatique sur appareil iOS pour reconnaître les pièces en euros en temps réel. L'architecture YOLO est utilisée pour implémenter le modèle à partir de zéro dans Keras. Le classificateur en tant que réseau neuronal de base pour la détection d'objets basé sur l'architecture DenseNet avec différents hyperparamètres est formé et évalué sur un ensemble de données publiques. Les modèles pré-dessinés ont été convertis pour effectuer la détection, formés avec la fonction de perte personnalisée et évalués. À la fin, une application iOS développée utilise un modèle de détecteur pour effectuer la détection en temps réel.

1.6.3 Reconnaissance des pièces malaisiennes basées sur l'aide de l'apprentissage profond

Lina Suhaili Rosidi et al. [8] ont proposé un projet pour la reconnaissance de monnaies malaisiennes. Leur système a été développée en modélisant le réseau neuronal convolutionnel (CNN) pour reconnaître les images de pièces. L'ensemble de donnée de pièces de monnaie qui a été développé se compose de 2400 images de quatre classes de pièces de monnaie, 5 , 10 , 20 et 50 sen.

Les modèle de CNN Pré-entrainés sont AlexNet, GoogleNet, et MobileNetV2, ils ont évalué les performances de chaque modèle formé à l'aide d'une matrice de confusion

et GoogleNet a obtenu la meilleure performance avec 99,2% de précision de test, 99,2% de précision, 99,18% de rappel et 99,19% de score F1.

1.6.4 Développement d'un réseau neuronal précis pour la reconnaissance des pièces

Nikolay Fonov et Urkaeva Ksenia [9] ont présenté un système de reconnaissance des pièces de l'URSS qui est basé sur un réseau de neurones convolutionnels et une base de données qui stocke des images de pièces et des informations sur chaque pièce. En outre, deux algorithmes de traitement de l'image d'entrée ont été étudiés. Ces algorithmes sont composés de quatre étapes : rognage de l'arrière-plan de l'image, trouver le centre de la pièce en la faisant tourner, réduire la taille de l'image de la pièce pour augmenter la vitesse de reconnaissance du réseau neuronal et trouver le nombre maximum de vecteurs colinéaires.

1.6.5 Reconnaissance des pièces de monnaie indiennes fondées sur l'apprentissage profond

A.U. Tajane et al. [10] ont proposé un système Pour la reconnaissance et la détection des pièces indiennes à l'aide du modèle AlexNet (Réseau de neurones convolutionnels pré-entrainé).

Le modèle est pré-entrainé sur plus de 1600 images et peut classer les images en quatre catégories d'objets comme une, deux, cinq et dix rupées. Le modèle pré-entrainé est testé sur divers ensembles de données standard et propres enregistrées qui se composent d'images ayant subi une rotation et décalage.

1.7 Conclusion

Les pièces de monnaies font partie intégrante de notre vie quotidienne, c'est l'une des choses dont l'économie nationale ou mondiale ne peut pas se passer en ce moment, malgré les changements dans le niveau des moyens de paiement.

Dans ce chapitre, nous avons défini et introduit les pièces de monnaies et leurs types. Nous avons aussi discuté l'opération de la fabrication des pièces dans l'industrie ainsi que l'histoire des pièces mondiales et algériennes. Un tableau résumant les images de pièces monnaies algériennes avec leurs informations, notamment, l'année de fabrication, la valeur de la pièce et sa dimension (diamètre) a été présenté. Nous avons aussi présenté un état de l'art concernant différents travaux menés dans le domaine de détection et identification de pièces de monnaies. Ces travaux utilisent le traitement d'images comme moyen pour faire la classification de pièces.

Chapitre 2 Traitement d'image et réseau de neurone

2.1 Introduction

La détection d'objets est une importante tâche de vision par ordinateur utilisée pour détecter les instances d'objets visuels de certaines classes telles que les pièces de monnaie. Il y a une énorme quantité de techniques utilisées pour la détection d'objets dans notre cas, nous allons discuter des techniques de traitement d'image qui ne nécessitent généralement pas de données historiques pour la formation et sont non supervisés dans la nature, nous discuterons également une autre partie importante dans la détection des objets est les réseaux neuronaux artificiels.

Le traitement d'images à l'aide de réseaux neuronaux artificiels (RNA) a été utilisé avec succès dans divers domaines d'activité tels que la géotechnique, le génie civil, la mécanique, la surveillance industrielle, le département de la défense, l'automatisation et les transports.

2.2 Techniques de traitement d'image

Dans cette section nous allons présenter seulement les méthodes utilisées dans le développement de ce travail, notamment, la conversion d'image couleur en niveau de gris, le filtrage, le seuillage ou binarisation, la morphologie mathématique et finalement la détection de contour.

2.2.1 Conversion d'image couleur en niveau de gris

Le codage dit en niveaux de gris permet d'obtenir plus de nuances que le simple noir et blanc. Il offre des possibilités supplémentaires pour coder le niveau de l'intensité lumineuse. La couleur est codée souvent sur trois octets, soit 3x8 bits .

Une image en noir et blanc signifie que les valeurs de pixels ne peuvent être que 0 ou 255 et rien entre les deux. Pour passer d'une image couleur (RVB) vers une image à niveaux de gris ou noir et blanc, on doit exploiter une des deux équations du codage YUV ou YIQ qui sont identiques et qui est :

$$G = 0{,}299*R + 0{,}587*V + 0{,}114*B \qquad (2.1)$$

Avec G est le niveau de gris du pixel, et R, V et B sont les composantes rouge, vert et bleue du pixel [11].

Pour afficher une image à niveaux de gris sur un écran couleur, les trois composantes RVB doivent être identiques comme suit :

$$R = V = B$$

L'application de cette conversion est illustrée par les images suivantes (voir figure2.1) :

Figure 2.1 : Image niveau gris

2.2.2 Seuillages d'une image

Le seuillage est un processus consistant à répartir les pixels d'une image à niveau de gris en deux classes de pixels (ou plus), c.-à-d. « avant-plan » et « arrière-plan ». Il est

principalement utilisé dans diverses tâches de traitement d'image, telles que l'élimination du bruit dans le processus OCR qui permet une plus grande précision de reconnaissance d'image, segmentation, etc [12].

Afin d'obtenir une image en seuil, habituellement, nous convertissons l'image originale en une image en échelle de gris, puis nous appliquons la technique de seuillage. Cette méthode est aussi connue sous le nom de binarisation car nous convertissons l'image en une forme binarisée, i.e. si la valeur d'un pixel est inférieure à la valeur seuil, convertissez-la en 0 (Noir) et si la valeur d'un pixel est supérieure à la valeur seuil, convertissez-la en 1 (blanc) ou vice-versa (voir Figure 2.2).

Figure 2.2 : Seuillage d'une image à niveau de gris [12]

A- Seuillage Simple

Si la valeur de pixel est supérieure à une valeur seuil, on lui assigne une valeur (peut-être blanche), sinon on lui assigne une autre valeur (peut-être noire) [12].

Le seuillage simple permet d'obtenir de bons résultats dans certaines images mais pas pour la plupart des images, d'où la nécessité de modifier la valeur du seuil ou effectuer quelques opérations supplémentaires pour obtenir le bon résultat.

B- Seuillage Adaptive

C'est la méthode selon laquelle la valeur seuil est calculée pour les petites régions et, par conséquent, il y aura des valeurs seuils différentes pour les différentes régions [13].

C- Méthode d'Otsu

La méthode de seuillage d'Otsu correspond aux critères discriminants linéaires qui supposent que l'image ne comprend que l'objet (premier plan) et l'arrière-plan, et l'hétérogénéité et la diversité de l'arrière-plan est ignorée. Otsu a fixé le seuil afin de minimiser le chevauchement des distributions de classes. Compte tenu de cette définition, la méthode d'Otsu segmente l'image en deux régions claire et sombre T0 et T1, où la région T0 est un ensemble de niveaux d'intensité de 0 à t ou en notation définie T0 = {0, 1, ..., t} et la région T1 = {t + 1, ..., n-1, n} où t est la valeur seuil, n est le niveau de gris maximal de l'image (par exemple 255). T0 et T1 peut être affecté à l'objet et à l'arrière-plan ou vice versa (l'objet n'occupe pas nécessairement toujours la région de lumière). La méthode de seuillage d'Otsu scanne toutes les valeurs de seuillage possibles et calcule la valeur minimale pour les niveaux de pixels de chaque côté du seuil. Le but est de trouver la valeur seuil avec l'entropie minimale pour la somme de premier plan et de fond. La méthode d'Otsu détermine la valeur seuil à partir des informations statistiques de l'image où pour une valeur seuil choisie t la variance des clusters T0 et T1 peut être calculée. La valeur seuil optimale est calculée en minimisant la somme des variances de groupe pondérées, où les pondérations sont la probabilité des groupes respectifs. Donnée : p(i) comme probabilités histogrammes de la valeur de gris observée i=1, ..., n [14].

$$P(i) = \frac{number\{(r,c)|image(r,c)=i}{(R,C)} \qquad (2.2)$$

Où **r**, **c :** est l'index pour la ligne et la colonne de l'image, respectivement, **R et C** : est le nombre de lignes et de colonnes de l'image, respectivement.

$\omega_b\,(t)$, $\mu_b(t)$, et $\sigma_b^2(t)$ en tant que poids, moyenne, et variance de la classe T0 avec valeur d'intensité de 0 à t, respectivement.

$\omega_f\,(t)$, $\mu_f(t)$, et $\sigma_f^2(t)$ en tant que poids, moyenne, et variance de la classe T1 avec valeur d'intensité de t+1 à n, respectivement.

$\sigma_w^2(t)$ représente la somme pondérée des variances de l'image. La meilleure valeur de seuil t* est la valeur avec la variance minimale à l'intérieur de la classe. L'écart de classe se définit comme suit :

$$\sigma_w^2(t) = \omega_b\,(t)^*\,\sigma_b^2(t) + \omega_f\,(t)^*\,\sigma_f^2(t) \qquad (2.3)$$

$$t^* = argmin\,(\sigma_w^2(t)) \qquad (2.4)$$

Où

$\omega_b\,(t) = \sum_{i=1}^{t} P(i)$ $\qquad$ $\omega_f\,(t) = \sum_{i=t+1}^{l} P(i)$

$\mu_f(t) = \frac{\sum_{i=t+1}^{l} i*P(i)}{\omega_f\,(t)}$ $\qquad$ $\mu_b(t) = \frac{\sum_{i=1}^{t} i*P(i)}{\omega_b\,(t)}$

$\sigma_b^2(t) = \frac{\sum_{i=1}^{t}(i-\mu_b(t))^2 *P(i)}{\omega_b\,(t)}$ $\qquad$ $\sigma_f^2(t) = \frac{\sum_{i=t+1}^{t}(i-\mu_f(t))^2 *P(i)}{\omega_f\,(t)}$

2.2.3 Filtrage Gaussien

Un filtre gaussien est un filtre passe-bas utilisé pour réduire le bruit (composants haute fréquence) et les zones de flou d'une image. Le filtre est implémenté comme un noyau symétrique de taille impaire qui est passé par chaque pixel de la région d'intérêt pour obtenir l'effet désiré. Un filtre gaussien pourrait être considéré comme une approximation de la fonction gaussienne (mathématiques), la fonction gaussienne est [15] :

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (2.5)$$

Où σ est l'écart-type de la distribution, la distribution est supposée avoir une moyenne de 0 [16].

En utilisant la fonction ci-dessus un noyau gaussien de n'importe quelle taille peut être calculé, en lui fournissant des valeurs appropriées. Une approximation du noyau gaussien (bidimensionnelle) avec écart type = 1 apparaît comme suit (voir Figure 2.3) :

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Figure 2.3 : Application du filtre gaussien [16]

2.2.4 Filtres morphologiques :

La morphologie mathématique est dédiée à l'analyse des structures spatiales. Elle est dite morphologique parce qu'elle vise à analyser la forme (morphologie) des objets. Une image est vue comme un ensemble de formes (voir figure 2.4).

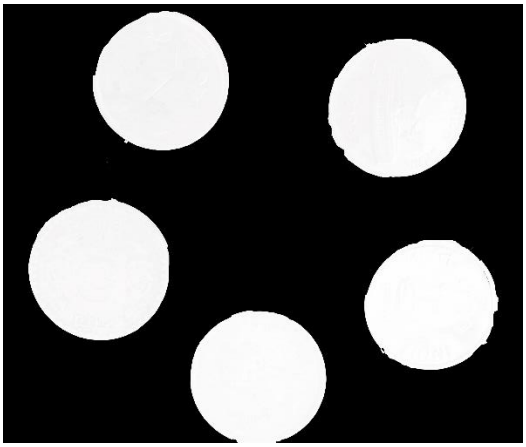


Figure 2.4 : Ensemble de formes [11]

Les filtres morphologiques sont souvent utilisés pour éliminer des pixels isolés considérés comme un bruit dans une image. Pour analyser les images, ces méthodes utilisent des formes connues appelées éléments structurants [11].

A- Dilatation :

La dilatation a pour objectif de dilater les objets ou formes dans une image. Pour une image binaire X et un élément structurant B, nous avons :

$$D^B(x) = X \oplus B = \{x \mid Bx \cap X \neq \Phi\} \quad (2.13)$$

Où

$$Bx = \{b + x \mid b \in B\} \quad (2.14)$$

Bx : l'ensemble de B translaté de x.

Dans une image binaire cette opération permet d'éliminer les pixels noirs isolés mais ajoute des pixels blancs au contour des objets présents dans l'image. Le résultat de cette opération est l'augmentation de la taille de ces objets.

B- Erosion :

L'érosion a pour objectif de d'éroder les objets ou formes dans une image. Pour une image binaire X et un élément structurant S, nous avons :

$$E^B(x) = X \oplus B = \{x \mid Bx \subseteq X\} \quad (2.15)$$

Elle permet d'éliminer les pixels blancs isolés au milieu des parties noires de l'image. Le résultat de cette opération est la diminution de la taille des objets présents dans l'image.

C- Ouverture :

L'ouverture est constituée par une opération d'érosion suivie d'une dilatation. Elle permet d'éliminer les taches blanches dans le fond de l'image.

D- Fermeture :

La fermeture est l'opération inverse de l'ouverture, qui consiste à faire subir à l'image une dilatation suivie d'une érosion. Elle permet d'éliminer les noirs qui se trouvent dans l'objet.

2.2.5 Détection du contour

La détection des contours est une technique de traitement d'image utilisée pour identifier les points d'une image numérique avec des discontinuités, simplement pour dire, des changements marqués dans la luminosité de l'image (voir la figure 2.5). Ces points où la luminosité de l'image varie fortement sont appelés les bords (ou limites) de l'image [17].

Figure 2.5 : Détection du contour pièces monnaies [10]

C'est l'une des étapes de base du traitement des images, de la reconnaissance des formes dans les images et de la vision par ordinateur. Lorsque nous traitons des images numériques à très haute résolution [17].

2.3 Réseaux de neurones

Les réseaux de neurones artificiels (RNA) sont inspirés de la méthode de travail du cerveau humain qui est totalement différente de celle d'un ordinateur. Le cerveau humain se base sur un système de traitement d'information parallèle et non linéaire, très compliqué, ce qui lui permet d'organiser ses composants pour traiter, d'une façon très performante et très rapide, des problèmes très compliqués tel que la reconnaissance des formes [19].

2.3.1 Neurone biologique

Le neurone est une cellule composée d'un corps cellulaire et d'un noyau. Le corps cellulaire ramifie pour former ce que l'on nomme les dendrites. Celles-ci sont parfois si nombreuses que l'on parle alors de chevelure dendritique ou d'arborisation dendritique. C'est par les dendrites que l'information est acheminée de l'extérieur vers le soma, corps du neurone. L'information traitée par le neurone chemine ensuite le long de l'axone

(unique) pour être transmise aux autres neurones. La transmission entre deux neurones n'est pas directe. En fait, il existe un espace intercellulaire de quelques dizaines d'Angströms

(10^{-10}m) entre l'axone du neurone afférent et les dendrites (on dit une dendrite) du neurone efférent. La jonction entre deux neurones est appelée la synapse (voir la figure 2.8) [20].

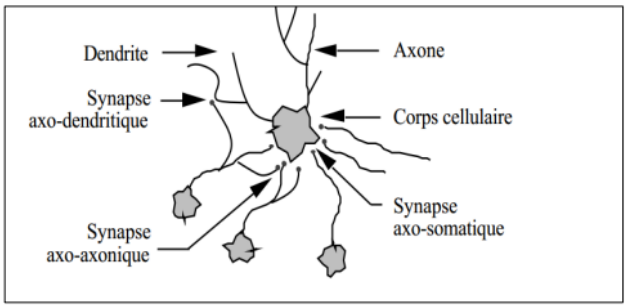


Figure 2.6 : Un neurone avec son arborisation dendritique [20]

Un neurone est constitué de trois parties ; le corps cellulaire, les dendrites et l'axone [21].

2.3.2 Neurone artificiel

Le perceptron, encore appelé neurone artificiel ou neurone formel, cherche à reproduire le fonctionnement d'un neurone biologique. Il existe différents niveaux d'abstraction, suivant la précision de la modélisation voulue (voir Figure 2.9) [22].

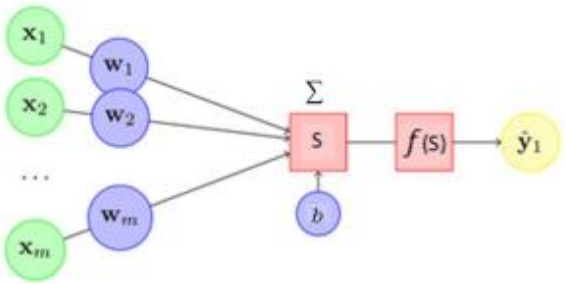


Figure 2.7 : Neurone formel de Mac Culloch et Pitts [22]

Tels que :

• (X_1,X_2 , …, X_m) : sont les entrées du neurone (signaux qui lui parviennent).

• (w_1,w_2,…, w_m) : les poids associés à chaque connexion.

• b : le seuil d'activation (C'est un paramètre supplémentaire dans le réseau de neurones qui est utilisé pour ajuster la sortie avec la somme pondérée des entrées du neurone.)

• S : la somme pondérée des entrées (potentiel d'activation).

$$S = \sum_{i=1}^{n} w_i X_i + b \qquad (2.16)$$

$\hat{Y}$ = ƒ (S) : la sortie du neurone (réponse du neurone « activé $\hat{Y}$ =1, ou non activé $\hat{Y}$ =0 »).

$$\hat{Y} = \begin{cases} 1 \; si \; S > 0 \\ 0 \; si \; S \leq 0 \end{cases}$$

On pourra résumer une modélisation de tel neurone par le tableau 2.1 et la Figure 2.10, qui nous permettra de voir clairement le passage du neurone biologique vers le neurone formel.

Neurone biologique	Neurone artificiel
Synapses	Poids de connexions
Axones	Signal de sortie
Dendrite	Signal d'entrée
Somma	Fonction d'activation

Tableau 2.1 : Passage du neurone biologique vers le neurone formel [20]

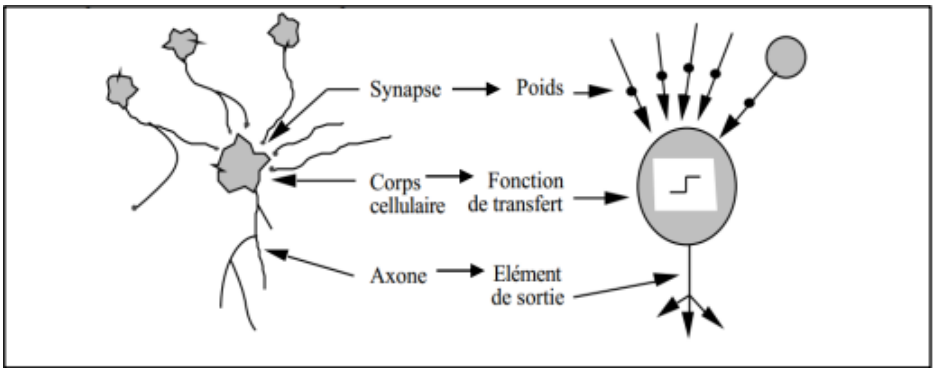


Figure 2.8 : Passage du neurone biologique vers le neurone formel [20]

A- Fonction d'activation

La fonction d'activation (fonction de transfert) joue un rôle très important dans le comportement des neurones, elle contient un paramètre sur la somme des entrées pondérées en plus du seuil d'activation. Il en existe plusieurs types dont la nature est déterminée en fonction du réseau (voir la figure 2.11) [20].

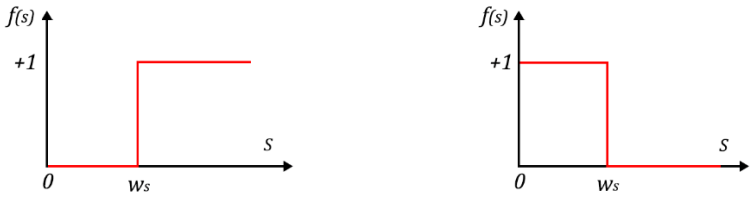


Figure 2.9 : Fonction d'activation d'un neurone formel [22]

A partir de ce modèle ont été définis divers modèles de neurones et avec d'autres fonctions d'activations. La figure 2.12 suivante montre d'autres fonctions d'activation.

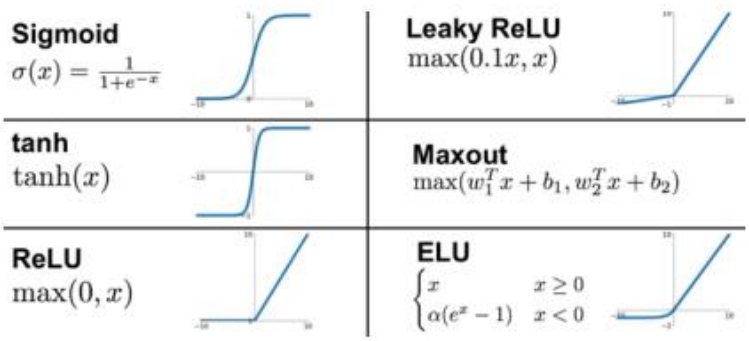


Figure 2.10 Fonctions d'activation possibles d'un neurone formel [22]

2.3.3 Architecture des réseaux de neurones

L'architecture d'un réseau de neurones artificiel est définie par la structure de ses neurones et leur connectivité. Elle est spécifiée par le nombre d'entrées, de sorties, de nœuds et la façon selon laquelle sont interconnectés et organisés les nœuds. Une fameuse architecture des réseaux de neurones est celle basée sur des couches où les nœuds de chaque couche n'ont aucune connexion entre eux. Cette architecture est utilisée dans presque 90 % des applications commerciales et industrielles [19].

A- Perceptron monocouche

Le perceptron simple est dit simple parce qu'il ne dispose que de deux couches ; la couche en entrée et la couche en sortie. Le réseau est déclenché par la réception d'une information en entrée. Le traitement de la donnée dans ce réseau se fait entre la couche d'entrée et la couche de sortie qui sont toutes reliées entre elles. Le réseau intégral ne dispose ainsi que d'une matrice de poids. Le fait de disposer d'une seule matrice de poids limite le perceptron simple à un classificateur linéaire permettant de diviser l'ensemble d'informations obtenues en deux catégories distingues. (Voir figure 2.13) [23].

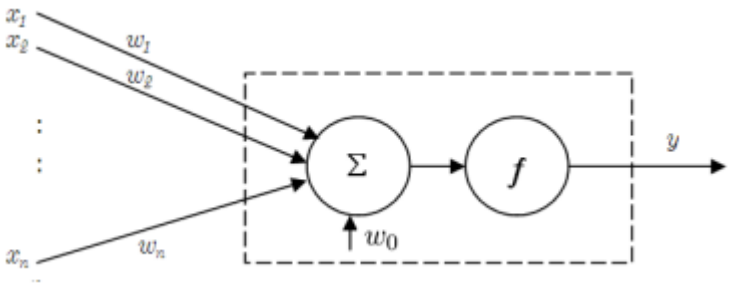


Figure 2.11 : Perceptron monocouche [24]

B- Perceptron multicouche

Dans ce réseau, les neurones de la première couche reçoivent toutes les informations entrées, ceux de la deuxième reçoivent toutes les sorties des neurones de la première couche, et ainsi de suite jusqu'au neurone de sortie qui reçoit celles de la dernière couche (Voir Figure 2.14).

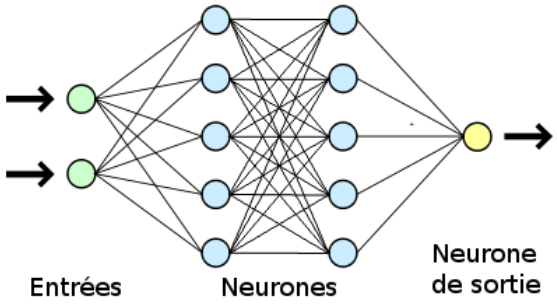


Figure 2.12 : Réseau de neurone multicouche [23]

Les couches 1 et 2 s'appellent des couches cachées tandis que la couche 3 est la couche de sortie [19].

2.3.4 Apprentissage

L'apprentissage est un processus par lequel les paramètres libres d'un réseau de neurones sont adaptés, au moyen d'un processus continu de stimulation par l'environnement dans lequel le réseau est intégré. Le type d'apprentissage est déterminé par la manière dont les changements de paramètres auront lieu. Il existe deux grandes stratégies d'apprentissage [20].

A- Apprentissage supervisé

C'est actuellement le mode d'apprentissage le plus couramment utilisé (Voir figure 2.15). Son principe est élémentaire : on soumet au réseau à un grand nombre d'exemples pour lesquels l'entrée commise par le réseau. Le plus répandu des algorithmes d'apprentissage supervisé est l'algorithme de rétro propagation du gradient d'erreur qui, appliqué aux réseaux Multi Couches [21].

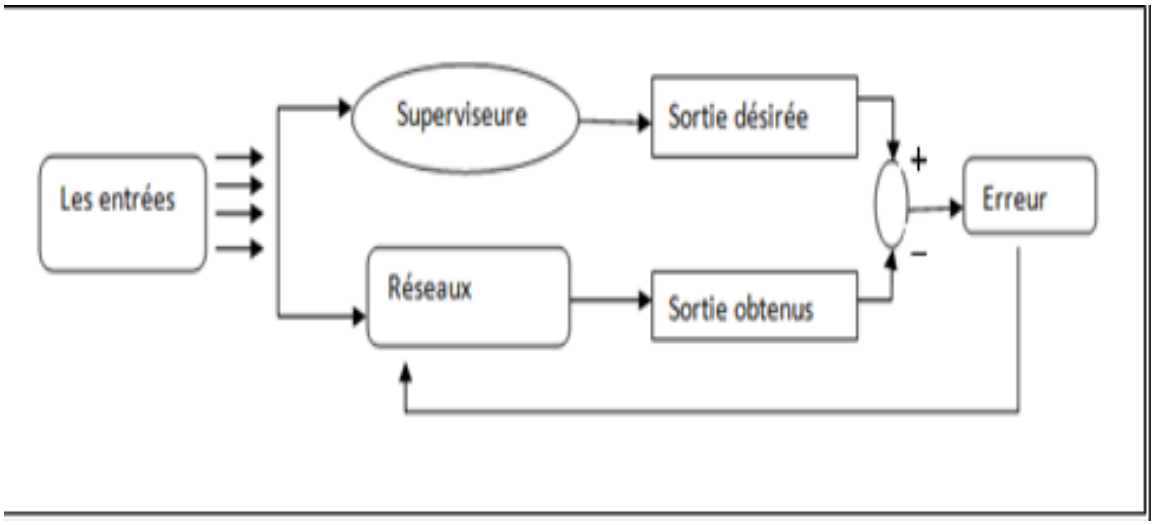


Figure 2.13 : Apprentissage supervisé [20]

B- Apprentissage non supervise

L'apprentissage non supervisé implique la fourniture d'un réseau autonome avec un nombre suffisant d'exemples contenant des répétitions (c'est-à-dire la redondance) de sorte qu'il efface automatiquement les modèles. Ces réseaux sont souvent appelés auto-organisateurs ou apprentissage compétitif. Dans l'apprentissage non supervisé, les données ne contiennent pas d'informations sur un résultat souhaité, il n'y a pas de superviseur ou d'expert humain. Il s'agit de déterminer les paramètres du réseau neuronal selon un critère défini. Dans ce cas, les exemples présentés à l'entrée entraînent une auto adaptation du réseau pour produire des valeurs de sortie proches en réponse à des valeurs d'entrée similaires. Ce type d'apprentissage a souvent moins de complexité dans le calcul par rapport à l'apprentissage supervisé bien sûr, l'architecture réseau, préalablement définie par l'utilisateur, est une forme de non supervision (Voir la figure 2.16) [20].

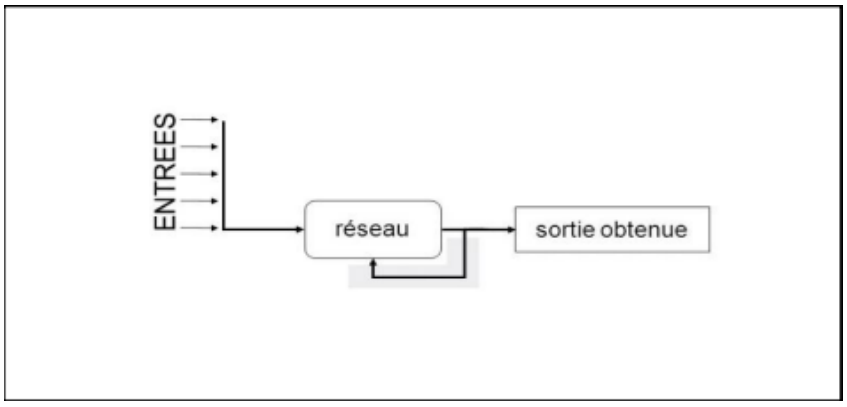


Figure 2.14 : Apprentissage non supervisé [20]

2.4 Conclusion

Dans ce chapitre nous avons présenté différentes techniques du traitement d'images. Toutes ces techniques sont utilisées dans notre projet de fin d'études, notamment, l'utilisation du contour et la notion de DPI pour estimer le diamètre de la pièce. Nous avons aussi présenté l'outil de mesure de diamètre de pièce monnaie (pied à coulisse) et finalement nous avons expliqué de manière générale les réseaux de neurones artificiels.

Chapitre 3 Détection et reconnaissance d'objet à l'aide de l'apprentissage profond

3.1 Introduction

L'intelligence artificielle a connu un développement remarquable et un saut qualitatif ces dernières années dans différents domaines, surtout après l'émergence d'apprentissage profond qui a apporté diverses techniques traditionnelles ou modernes pour qu'il soit possible de reconnaître les images ou les objets dans l'image.

Il y a divers types de modèles de détection d'objets R-CNN, SSD, YOLO avec différentes techniques. Ces techniques sont différentes les unes des autres en termes de précision, de vitesse et de ressources et même appareils requis. Parmi ces modèles nous avons choisi le YOLO V4 dont la vitesse et précision sont optimales pour la détection des objets.

3.2 Intelligence artificielle (IA)

L'intelligence artificielle (IA) est une nouvelle science technique qui étudie et développe des théories, des méthodes, des techniques et des systèmes d'application pour simuler et étendre l'intelligence humaine. En 1956, le concept d'Al a été proposé pour la première fois par John McCarthy, qui a défini le sujet comme "la science et l'ingénierie de la fabrication de machines intelligentes, en particulier le programme informatique intelligent. L'intelligence artificielle s'occupe de faire fonctionner les machines d'une manière intelligente, semblable à la façon dont l'esprit humain fonctionne. À l'heure actuelle, l'Al est devenu un cours interdisciplinaire qui implique divers domaines (Voir la figure 3.1) [25].

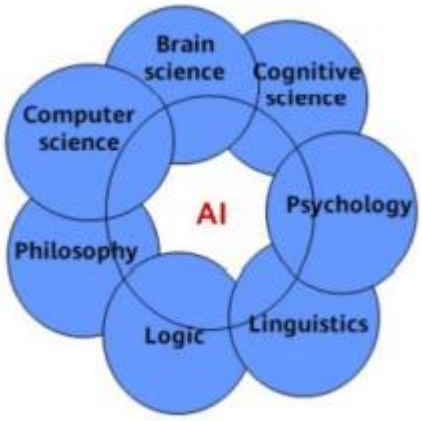


Figure 3.1: Concept de l'intelligence artificielle [25]

3.3 Apprentissage profond

L'apprentissage profond ou Deep Learning (en anglais) est l'une des technologies principales de l'apprentissage automatique. Avec l'apprentissage profond, nous parlons d'algorithmes capables de mimer les actions du cerveau humain grâce à des réseaux de neurones artificielles. Les réseaux sont composés de dizaines voire de centaines de « couches » de neurones, chacune recevant et interprétant les informations de la couche précédente (Voir figure 3.2).

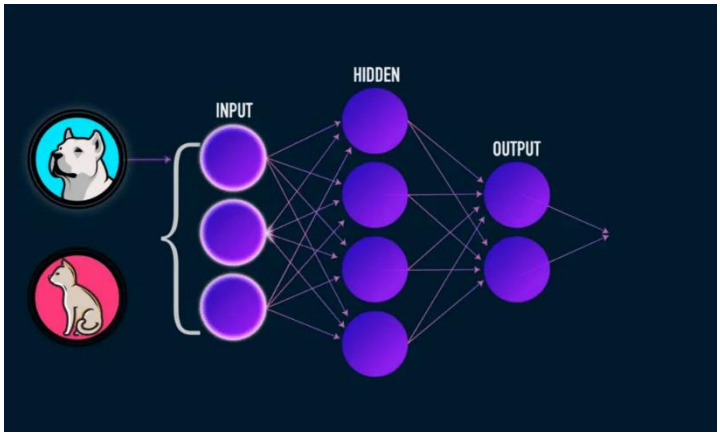


Figure 3.2: Classification du chat et chien à l'aide de l'apprentissage profond [26]

Chaque neurone artificiel est représenté dans la figure 3.3 par un rond, peut être vu comme un modèle linéaire. En interconnectant les neurones sous forme de couche, nous transformons notre réseau de neurones en un modèle non-linéaire très complexe [26].

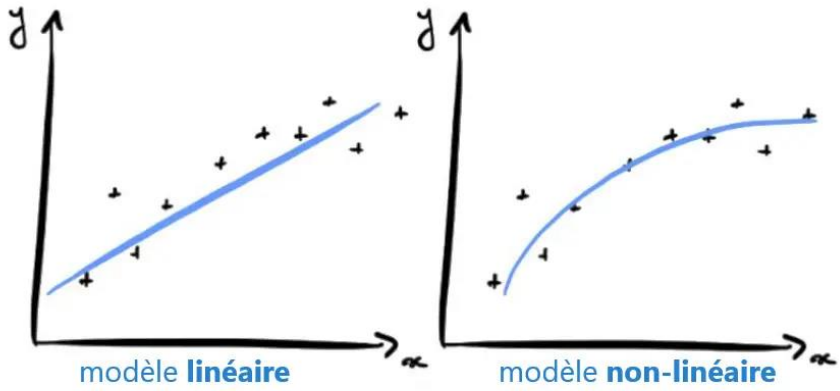


Figure 3.3: Modèle linière et non-Linéaire [26]

Pour illustrer le concept, prenons un problème de classification entre chien et chat à partir d'images. Lors de l'apprentissage, l'algorithme va ajuster les poids des neurones de façon à diminuer l'écart entre les résultats obtenus et les résultats attendus. Le modèle pourra apprendre à détecter les triangles dans une image puisque les chats ont des oreilles beaucoup plus triangulaires que les chiens.

3.4 Architectures de l'apprentissage profond

3.4.1 Réseaux neuronaux convolutifs (CNN)

Le CNN est un réseau neuronal multicouches qui a été biologiquement inspiré par le cortex visuel animal. L'architecture est particulièrement utile dans les applications de traitement d'images. Les réseaux de neurones convolutifs, également appelés ConvNets, ont été introduits pour la première fois dans les années 1980 par Yann LeCun, à l'époque, l'architecture était axée sur la reconnaissance manuscrite des caractères, comme l'interprétation des codes postaux. En tant que réseau profond, les premières couches reconnaissent les caractéristiques (comme les bords), et les couches ultérieures recombinent ces caractéristiques en attributs de niveau supérieur de l'entrée (Voir la figure 3.4).

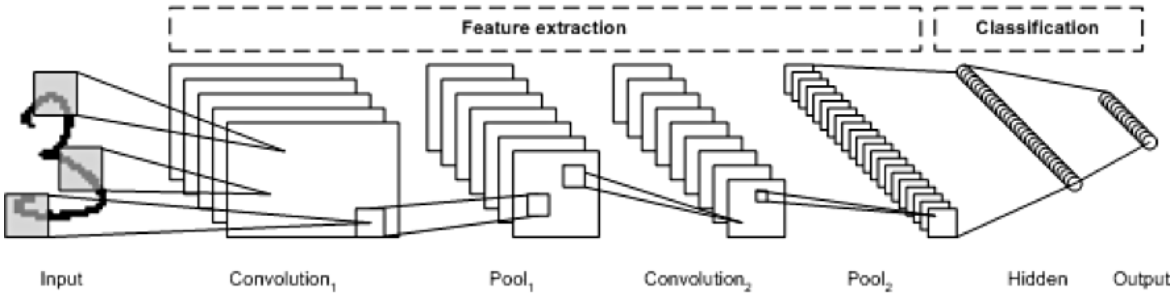


Figure 3.4: Réseaux neuronaux convolutifs [27]

L'architecture CNN est composée de plusieurs couches qui implémentent l'extraction et la classification des fonctionnalités. L'image est divisée en champs réceptifs qui alimentent une couche convolutive, qui extrait ensuite les caractéristiques de l'image d'entrée. La prochaine étape est la mise en commun, qui réduit la dimensionnalité des caractéristiques extraites (par le biais d'un échantillonnage descendant) tout en conservant les informations les plus importantes (généralement, par la mise en commun maximale). Une autre étape de convolution et de mise en commun est alors effectuée qui alimente un perceptron multicouche entièrement connecté. La couche de sortie finale de ce réseau est un ensemble de nœuds qui identifient les caractéristiques de l'image (dans ce cas, un nœud par numéro identifié). Vous entraînez le réseau en utilisant la rétropropagation [27].

L'utilisation de couches profondes de traitement, de convolutions, de mise en commun et d'une couche de classification entièrement connectée a ouvert la porte à diverses nouvelles applications de réseaux neuronaux d'apprentissage profond. En plus du traitement d'image, le CNN a été appliqué avec succès à la reconnaissance vidéo et à diverses tâches dans le traitement du langage naturel.

3.4.2 Réseaux neuronaux récurrents (RNN)

Le RNN est l'une des architectures de réseau de base à partir de laquelle d'autres architectures d'apprentissage profond sont construites. La principale différence entre un réseau multicouches typique et un réseau récurrent est qu'au lieu d'avoir des connexions en flux continu, un réseau récurrent peut avoir des connexions qui remontent dans les couches précédentes (ou dans la même couche). Cette rétroaction

permet aux ANN de conserver la mémoire des entrées passées et de modéliser les problèmes dans le temps [27].

Les RNNs se composent d'un riche ensemble d'architectures. Le principal facteur de différenciation est la rétroaction au sein du réseau, qui pourrait se manifester à partir d'une couche cachée, de la couche de sortie ou d'une combinaison de celles-ci (Voir la figure 3.5).

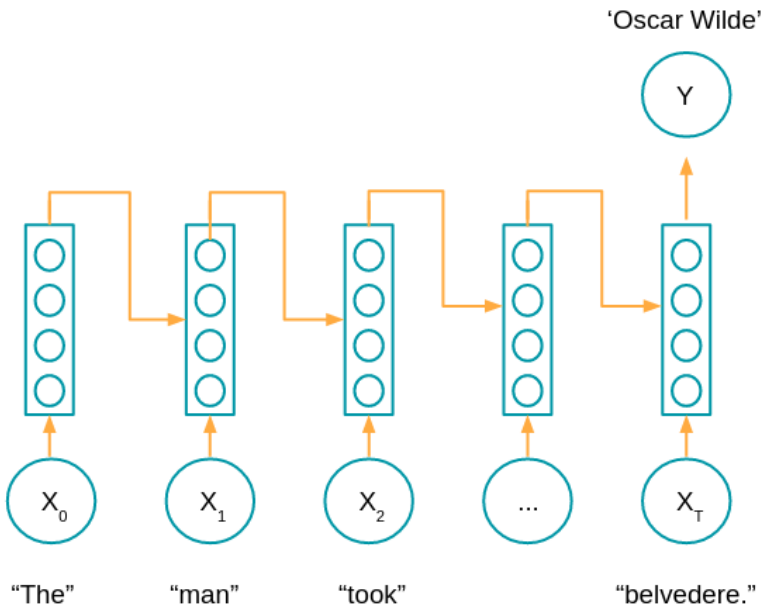


Figure 3.5: Réseaux neuronaux récurrents [28]

Les RNNs peuvent être dépliés dans le temps et entraînés avec une contre-propagation standard ou en utilisant une variante de contre-propagation qui est appelée contre-propagation dans le temps.

3.5 Détection d'objets à l'aide d'apprentissage profond

3.5.1 Définition

La détection d'objets est une technique de vision par ordinateur qui a connu un changement révolutionnaire rapide, cette technique fait la combinaison de la classification et de la localisation d'objets. En fait l'un des sujets les plus difficiles dans le domaine de la vision par ordinateur.

3.5.2 Les Modèles de Détection :

A- R-CNN

R-CNN est le grand-père de Faster R-CNN. En d'autres termes, R-CNN a vraiment donné le coup d'envoi, R-CNN, ou réseau de neurones convolutionnels basé sur la région, consistait en 3 étapes simples :

1-Analyser l'image d'entrée pour trouver des objets possibles en utilisant un algorithme appelé recherche sélective, générant ~2000 propositions de régions.

2-Exécuter un réseau neuronal convolutif (CNN) en plus de chacune de ces propositions de région.

3-Prenez la sortie de chaque CNN et introduisez-la dans un SVM pour classer la région et un régresseur linéaire pour resserrer la boîte de délimitation de l'objet, si un tel objet existe [29].

Ces trois étapes sont illustrées dans la figure 3.6 ci-dessous :

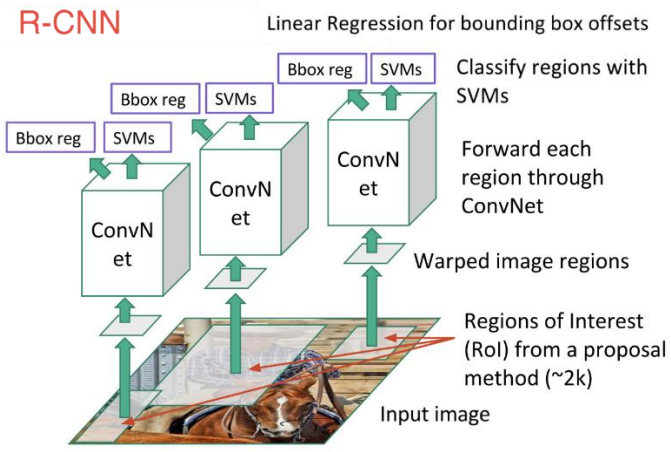


Figure 3.6: le modèle R-CNN [29]

B- SSD : Single Shot MultiBox Detector

L'approche de la SSD est basée sur un réseau convolutif feed-forward qui produit une collection de boîtes englobantes de taille fixe et des scores pour la présence d'instances de classes d'objets dans ces boîtes, suivi d'une étape de suppression non maximale pour

produire les détections finales. Les premières couches du réseau sont basées sur une architecture standard utilisée pour la classification d'images de haute qualité (VGG-16) (tronquée avant toute couche de classification), qui s'appelle réseau de base (Voir figure 3.7). Ensuite une structure auxiliaire au réseau pour produire des détections avec les caractéristiques clés suivantes :

✓ Cartes de caractéristiques multi-échelles pour la détection

✓ Prédicteurs convolutifs pour la détection

✓ Boîtes et aspect par défaut

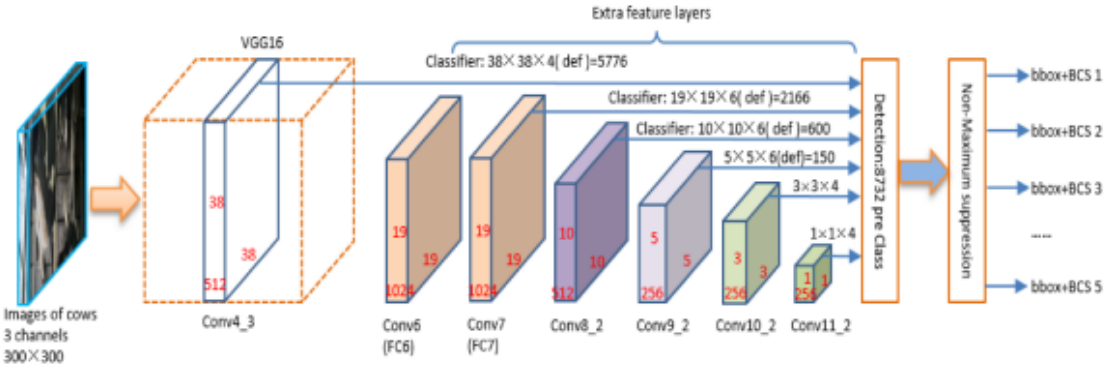


Figure 3.7: le modèle SSD [30]

Le résultat obtenu du modèle SSD512 entrainé sur COCO trainval35k puis affiné en Pascal voc 2007+2012 c'est le meilleur résultat : 81,6% mAP (mean Average Precision), et 68.0% mAP avec le modèle SSD300 apprend sur voc 2007, et le résultat obtenu des modèles SSD300 et SSD512 entrainé sur la base de données COCO trainval35k est 41.2% mAP et 46.5% mAP [30].

C- YOLO : You Only Look Once

You Only Look Once ou YOLO est l'un des algorithmes populaires de détection d'objets utilisé par les chercheurs du monde entier. Il a été décrit pour la première fois en 2015 dans l'article de Joseph Redmon [30].

Le réseau utilise les caractéristiques de l'image entière pour prédire chaque boîte englobante. Il prédit également toutes les boîtes englobantes de toutes les classes d'une image simultanément. Cela signifie que ce réseau raisonne globalement sur l'ensemble

de l'image et sur tous les objets qu'elle contient. La conception YOLO permet un apprentissage de bout en bout et des vitesses en temps réel tout en maintenant une précision moyenne élevée.

L'architecture de ce réseau est inspirée du modèle Google Net pour la classification d'images. Ce réseau comporte 24 couches convolutives suivies de 2 couches entièrement connectées. Au lieu des modules d'initialisation utilisés par Google Net, elle a utilisé simplement des couches de réduction 1 × 1 suivies de 3 × 3 couches convolutives (Voir figure 3.8).

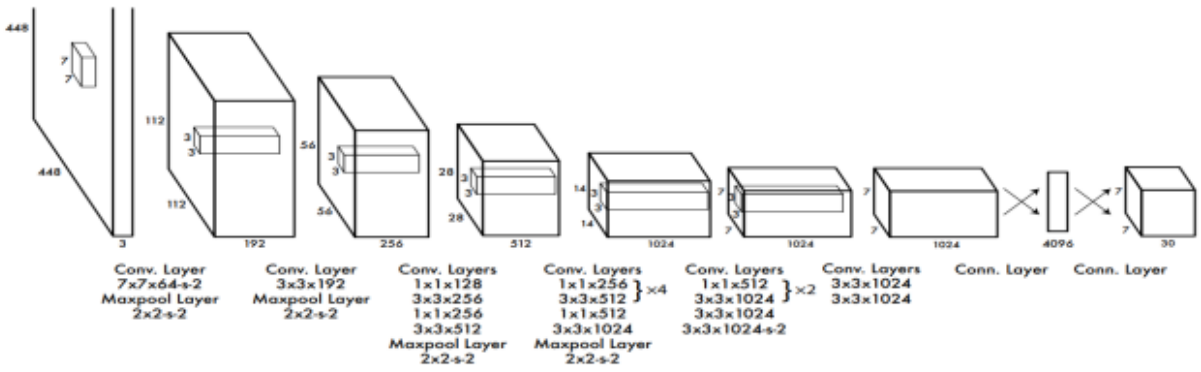


Figure 3.8: le model YOLO [30]

YOLO divise les images d'entrée en cellules S x S. chaque cellule responsable de la détection de la cible du point central dans la cellule. Chaque cellule prédit des boîtes de délimitation B (B-boite) et la probabilité de catégorie d'objet. Chaque boîte limitative représente 5 chiffres (x, y, w, h, c)

Les coordonnées (x, y) représentent le centre de la boîte par rapport aux limites de la cellule de la grille de la boîte par rapport aux limites de la cellule de la grille.

(w, h) représentent la largeur et la hauteur et sont prédites par rapport à l'image entière et finalement la prédiction de confiance représente le IoU [30].

3.6 Détection des Objectes avec YoloV4 :

Comme nous l'avons mentionné précédemment, YOLO est un algorithme de détection d'objets en temps réel, et il existe de nombreuses versions de YOLO mais nous avons utilisé la version YoloV4 en raison de la vitesse et de la précision qu'offre cette version. Dans la section suivante, nous allons expliquer comment fonctionne le Yolov4, et quels sont les avantages de son utilisation.

3.6.1 Yolov4 Architecture :

L'architecture du Yolov4 est découpée en trois parties, fonctionnellement différentes, appelées backbone (la colonne vertébrale), neck (le cou) et head (tête) (voir Figure 3.9). Il s'agit d'un découpage retrouvé dans l'architecture de nombreux modèles récents de détection d'image :

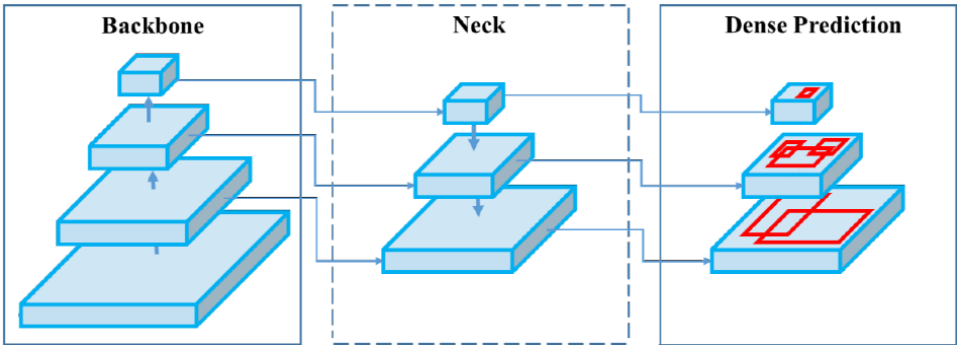


Figure 3.9: Architecture de YOLO v4 [31]

Backbone: CSPDarknet53, **Neck**: SPP, PANet, **Head:** YOLOv3

A- Backbone :

La backbone est le "corps" du réseau, ce qui va permettre l'ensemble des décisions prises par celui-ci.

Yolov4 utilise le **CSPDarknet53** comme modèle d'extracteur de fonctionnalités.

- **CSPDarknet53 :** est un réseau neuronal convolutif et une colonne vertébrale pour la détection d'objets qui utilise DarkNet-53. Il utilise une stratégie CSPNet pour partitionner la carte de caractéristiques de la couche de base en deux

parties, puis les fusionne par le biais d'une hiérarchie transversale (voir Figure 3.10). L'utilisation d'une stratégie de partition et de fusion permet un flux de gradient plus important à travers le réseau [31].

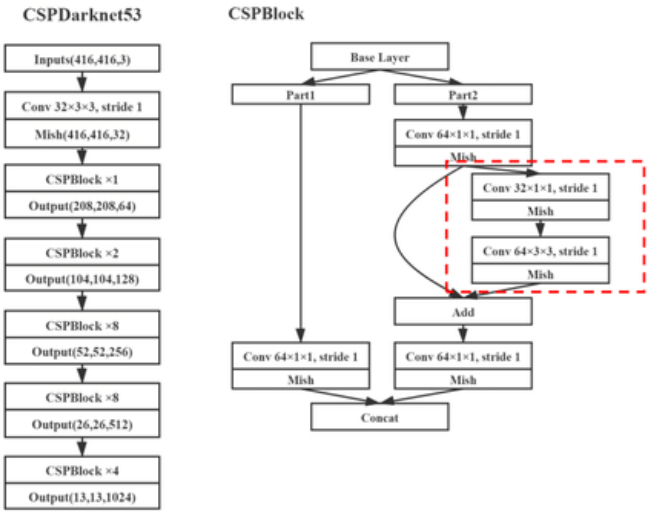


Figure 3.10: l'architecture de cspdaknet53 [31]

B- Neck :

La seconde partie de Yolov4 est le "neck" qui a pour rôle d'extraire les caractéristiques pertinentes de l'ensemble des couches de la backbone, et de les combiner en caractéristiques utiles à notre tâche de détection.

Yolov4 utilise le pooling pyramidal spatial (SPP) et le réseau d'agrégation de chemins (PAN). Ce dernier n'est pas identique au PAN d'origine, mais une version modifiée qui remplace l'ajout par un concaténation [32].

❖ **Pyramide spatiale et couche de mise en commun (SPP) :**

Un bloc supplémentaire appelé SPP (**Spatial Pyramid Pooling**) est ajouté entre le backbone CSPDarkNet53 et le réseau d'agrégation de fonctionnalités (PANet), ceci est fait pour augmenter le champ réceptif et séparer les fonctionnalités de contexte les plus importantes et n'a presque aucun effet sur la rapidité de fonctionnement du reseau.

Il est connecté aux couches finales des couches convolutives densément connectées de CSPDarkNet (voir Figure 3.11).

Le champ réceptif fait référence à la zone de l'image qui est exposée à un noyau ou à un filtre sur une instance. Au fur et à mesure que davantage de couches convolutives sont empilées, elle augmente de manière linéaire alors qu'elle augmente de manière exponentielle lorsque nous empilons des convolutions dilatées et apporte de la non-linéarité [33].

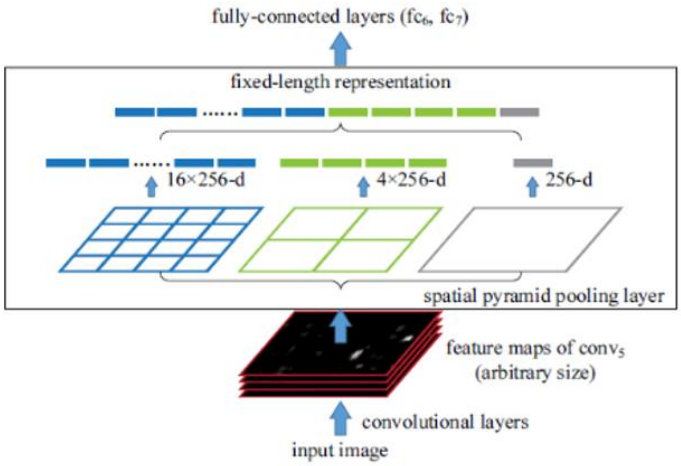


Figure 3.11: Spatial Pyramid Pooling [33]

❖ **Réseau d'agrégation de chemins (Path Aggregation Network PANet)** :

YOLOv4 utilise un réseau d'agrégation de chemins modifié, principalement en tant qu'amélioration de la conception afin de le rendre plus adapté à la formation sur un seul GPU.

Le rôle principal de PANet est d'améliorer le processus de segmentation des instances en conservant les informations spatiales qui, à leur tour, contribuent à la bonne localisation des pixels pour la prédiction du masque.

L'augmentation de chemin ascendante, la mise en commun adaptative des fonctionnalités et la fusion entièrement connectée sont les propriétés importantes qui les rendent si précis pour la prédiction de masque.

Ici, le PANet modifié concatène les couches voisines au lieu de les ajouter lors de l'utilisation du regroupement de fonctionnalités adaptatives (voir Figure 3.12)[33]

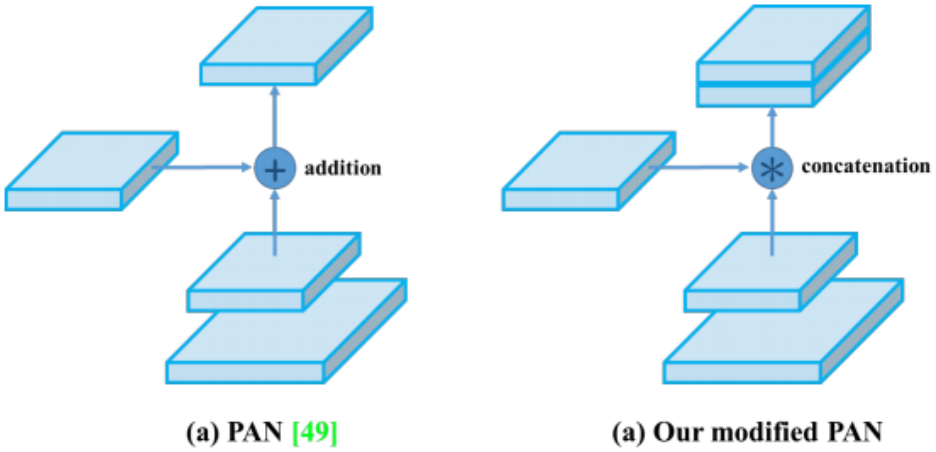


Figure 3.12: modification de pan [31]

C- Head :

Enfin, la tête est responsable de la décision finale du réseau.

La fonction principale ici est de localiser les boîtes englobantes et d'effectuer la classification.

Les coordonnées de la boîte englobante (x, y, hauteur et largeur) ainsi que les scores sont détectées. Ici, les coordonnées x et y sont le centre de la boîte b exprimé par rapport à la limite de la cellule de la grille. La largeur et la hauteur sont prédites par rapport à l'image entière [33].

La figure 3.13 suivante contient l'architecteur complet en détails.

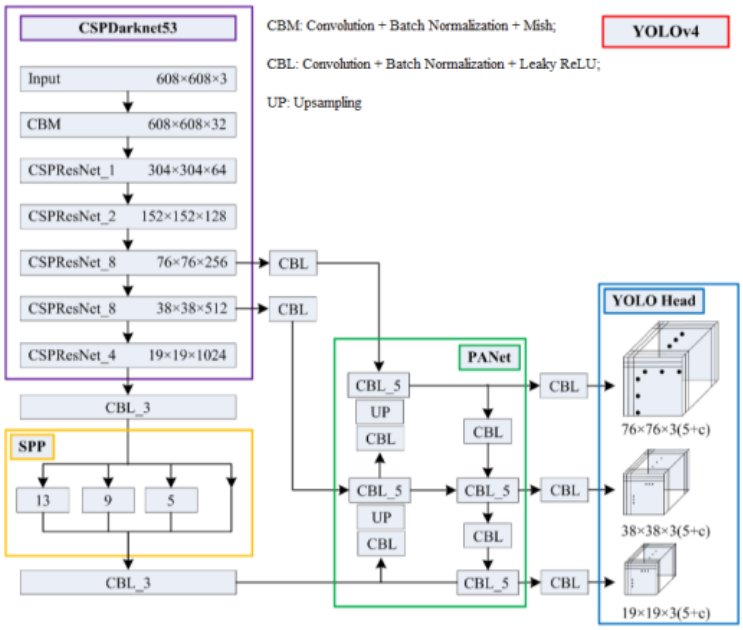


Figure 3.13: Schéma fonctionnel de YOLOv4 [34]

3.6.2 Sac de cadeaux et sac de promotions (bof & bos) :

Ce sont des méthodes utilisées pour améliorer la précision du détecteur d'objet. Ils analysent différentes méthodes dans les deux catégories, pour obtenir un réseau de neurones à vitesse de fonctionnement rapide avec une bonne précision. Ces deux catégories sont :

A- SAC DE CADEAUX (BOF) :

Méthodes qui peuvent faire en sorte que le détecteur d'objet reçoive une meilleure précision sans augmenter le coût d'inférence. Ces méthodes ne font que modifier la stratégie de formation ou ne font qu'augmenter le coût de la formation .

Un exemple de BoF est l'augmentation des données (voir Figure 3.14), qui augmente la capacité de généralisation du modèle. Pour ce faire, nous pouvons faire des :

Distorsions photométriques, par exemple : luminosité, contraste, teinte, saturation, bruit Distorsions géométriques, par exemple : mise à l'échelle aléatoire, recadrage, retournement et rotation [32].

Figure 3.14: Exemples des augmentations des données

B- SAC DE SPECIAUX (BOS) :

Le sac de spéciaux (Bag of Specials) contient différents plugins et les modules de post-traitement qui n'augmentent que légèrement le coût d'inférence mais peuvent considérablement améliorer la précision du détecteur d'objet [33].

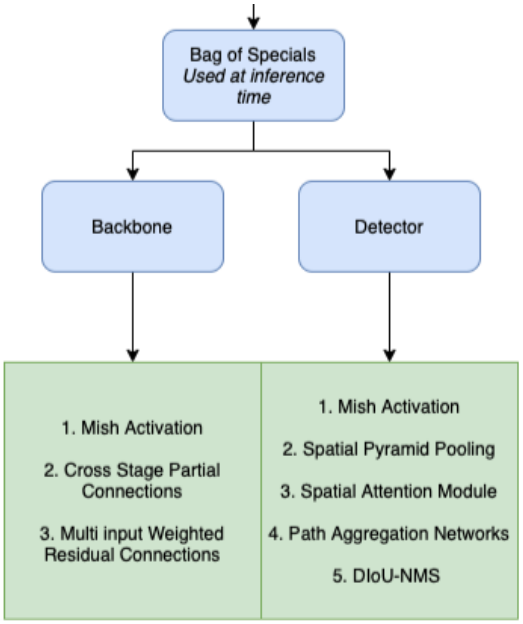


Figure 3.15: les additions de BoS [35]

- En général, ces méthodes ou *Bag of Specials* sont mentionnées dans la figure 3.15 peuvent être considérés comme un complément pour tous les détecteurs

d'objets présents actuellement afin de les rendre plus précis sur les ensembles de données de référence.

- **Résumé des BoF et BoS utilisés**

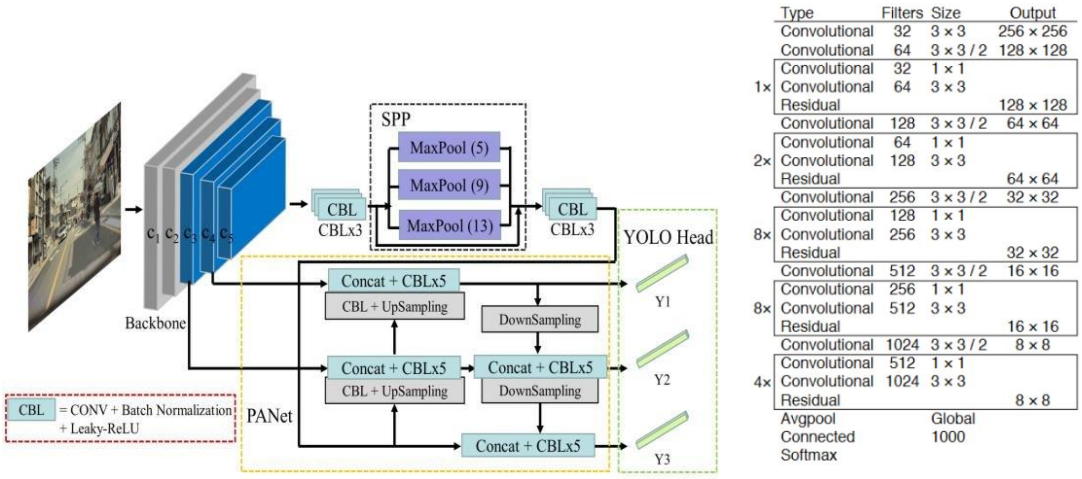
Les différents modules/méthodes de BoF et BoS utilisés dans le backbone et dans le détecteur de YOLO v4 peuvent être résumés dans la figure 3.16 comme suit [32] :

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation • DropBlock • Class label smoothing	• CIoU-loss • Cross mini-Batch Normalization • DropBlock • Mosaic data augmentation • Self-Adversarial Training • Multiple anchors for a single ground truth • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

Figure 3.16: Les différents modules/méthodes de BoF et BoS [32]

3.6.3 Fonctionnements de Yolov4 :

Le réseau utilise les caractéristiques de l'image entière pour prédire chaque boîte englobante. Il prédit également toutes les boîtes englobantes de toutes les classes d'une image simultanément. Cela signifie que ce réseau raisonne globalement sur l'ensemble de l'image et sur tous les objets qu'elle contient. La conception YOLO permet un apprentissage de bout en bout et des vitesses en temps réel tout en maintenant une précision moyenne élevée [33].



	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 3.17: La structure globale de YOLOv4 [36]

YOLO n'utilise que des couches convolutives, ce qui en fait un réseau entièrement convolutif (FCN). Il dispose de 75 couches convolutives, avec des connexions de saut et des couches de réduction 1 × 1 suivies de 3 × 3 couches convolutives [37].

L'algorithme de modèle YOLO est divisé en 3 étapes :

- **Diviser l'image en cellules avec une taille S x S :**

On divise l'image en une grille de taille S x S (par exemple 3 x3 voir Figure 3.18), ce qui donne N cellules au total. Cette cellule de la grille est responsable de la détection de cet objet[38].

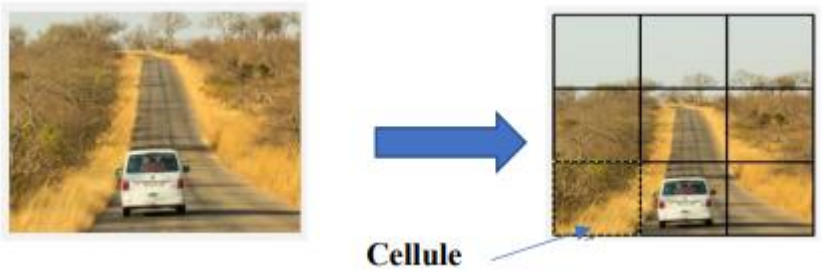


Figure 3.18: Diviser l'image en SxS grille [39]

- **Chaque cellule prédit B boîtes englobante :**

Après la divisons de l'image en N cellules, chaque cellule de la grille prédit des boîtes englobante B et des scores de confiance pour ces boîtes, Chaque boîte englobante est

constituée de 5 prédictions : x, y, w, h, et confiance. Les coordonnées (x, y) représentent le centre de la boîte par rapport aux limites de la cellule de la grille de la boîte et par rapport aux limites de la cellule de la grille (voir Figure 3.19). La largeur et la hauteur sont prédites par rapport à l'image entière. Enfin, la prédiction de confiance représente le IoU entre la boîte prédite et toute boîte de vérité terrain [38].

Exemple : où il y a 3x3 cellules (S=3), chaque cellule prédit 1 boîte limitante (B=1), et les objets sont soit chien = 1 (C=1), soit humain = 2, (C=2). Pour chaque cellule, le CNN prédit un vecteur Y :

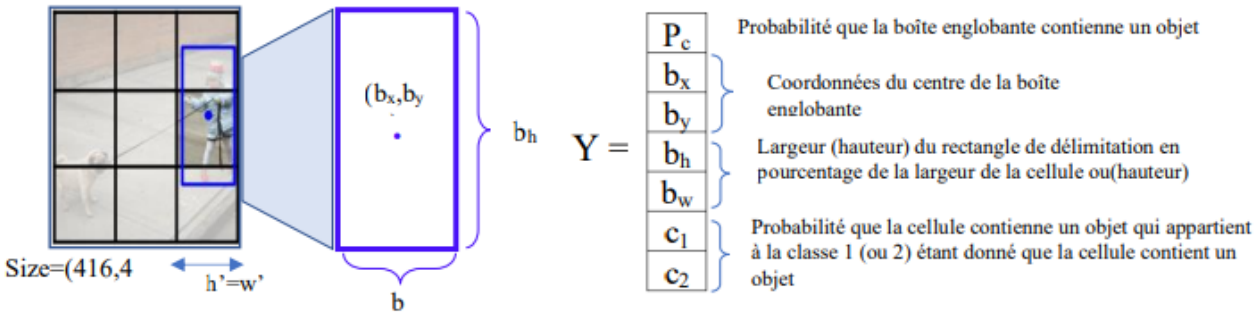


Figure 3.19: le vecteur prédit dans le cas d'une seule boite [39]

Les valeurs du vecteur Y sont calculées au format YOLO :

Pc= la prédiction de confiance représente le IoU entre la boîte prédite et la boîte de vérité terrain.

bx = (x - h')/h', **by** = (y – w')/w', **bh** = h/416, **bw** = w/416 (dans le cas ou la taille de l'image est 416x416)

a) **Intersection sur Union (IoU)** : L'intersection sur l'union (IoU) est la métrique d'évaluation de facto utilisée dans la détection d'objets. Elle est utilisée pour déterminer les vrais positifs et les faux positifs dans un ensemble de prédictions (voir figure 3.20). Lorsqu'on utilise l'IoU comme mesure d'évaluation, il faut choisir un seuil de précision [37]. Il compare la boîte prédite avec la boîte détectable et peut calculer la surface comme suit :

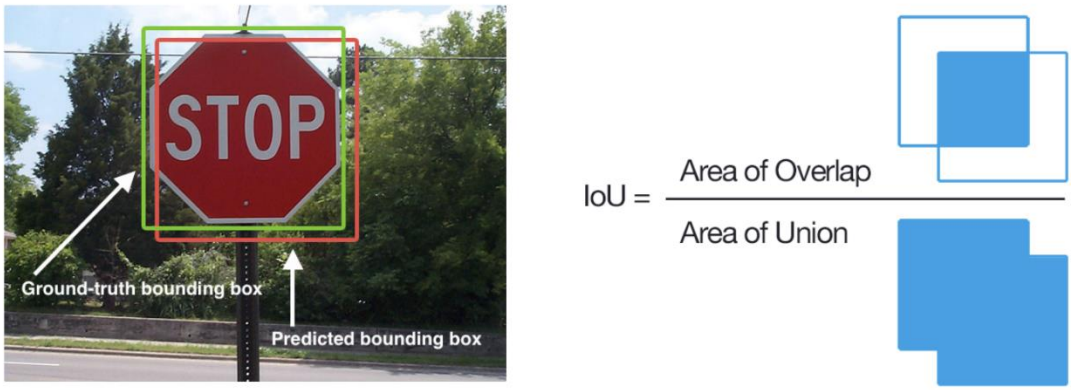


Figure 3.20: intersection sur union [40]

Pendant l'apprentissage, l'indice de confiance IoU est calculé entre la boîte prédite et la boîte de base. Dans la figure 3.21, il y a des exemples de bons et de mauvais scores d'intersection sur Union.

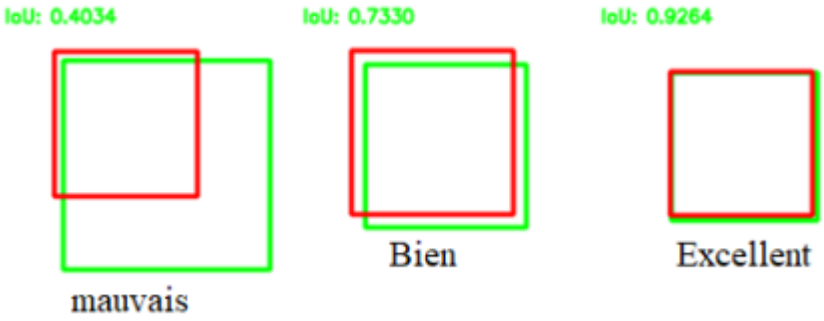


Figure 3.21: Example de l'intersection sur union [40]

D'après la figure 3.21, les boîtes englobantes prédites qui se chevauchent fortement avec les boîtes englobantes de base ont des scores plus élevés que celles qui se chevauchent moins.

b) Boîte d'ancrage (Anchor Box) :

YOLO peut bien fonctionner pour plusieurs objets où chaque objet est associé à une cellule de grille. Mais en cas de chevauchement, dans lequel une cellule de la grille contient en fait les points centraux de deux objets différents, nous pouvons utiliser

quelque chose appelé boîtes d'ancrage pour permettre à une cellule de la grille de détecter plusieurs objets [41].

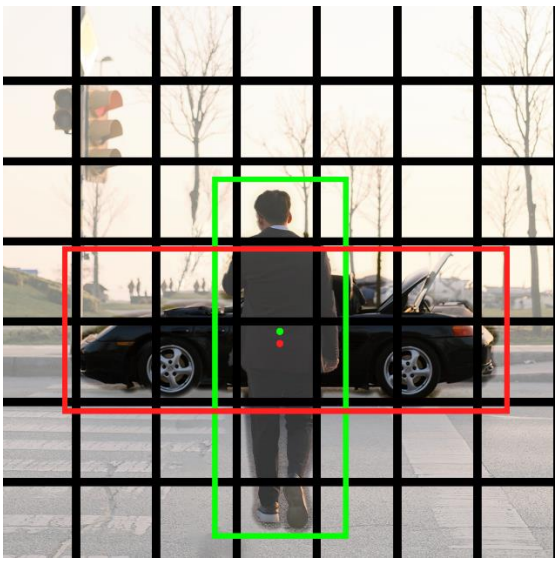


Figure 3.22 : Exemple d'image ou le centre des deux Object dans la même cellule

Dans la figure 3.22, nous voyons que nous avons une personne et une voiture qui se chevauchent dans l'image. Ainsi, une partie de la voiture est masquée. Nous pouvons également voir que les centres des deux boîtes englobantes, la voiture et la personne dans la même cellule de grille. Étant donné que le vecteur de sortie de chaque cellule de la grille ne peut avoir qu'une seule classe, il sera obligé de choisir soit la voiture, soit la personne. Mais en définissant des boîtes d'ancrage, nous pouvons créer un vecteur de cellule de grille plus long et associer plusieurs classes à chaque cellule de grille. Les boîtes d'ancrage ont un rapport d'aspect défini, et ils ont essayé de détecter les objets qui s'intègrent bien dans une boîte avec ce rapport [41].

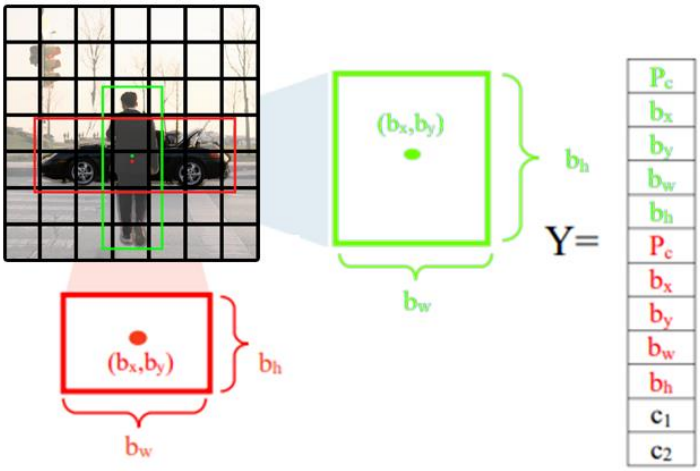


Figure 3.23: vecteur prédit dans le cas de plusieurs boites dans la cellule

Exemple, puisque nous détectons une voiture large et une personne debout, nous définirons une boîte d'ancrage qui a à peu près la forme d'une voiture, cette boîte sera plus large que haute. Et nous définirons une autre boîte d'ancrage pouvant contenir une personne debout à l'intérieur, qui sera plus haute que large. L'image de test est d'abord décomposée en une grille et le réseau produit ensuite des vecteurs de sortie, un pour chaque cellule de la grille. Ces vecteurs nous indiquent si une cellule contient un objet, la classe de l'objet et les boîtes englobantes de l'objet. Puisque nous utilisons deux boîtes d'ancrage, nous obtiendrons deux boîtes d'ancrage prédites pour chaque cellule de la grille (voir Figure 3.23) [41].

Donc en générale si on divise l'image en une grille S × S et pour chaque cellule de la grille, il prédit B boîtes englobantes, la confiance pour ces boîtes et les probabilités de classe C. Ces prédictions sont encodées sous la forme d'un tenseur S × S × (B ∗ 5 + C) (Voir figure 3.24) [33].

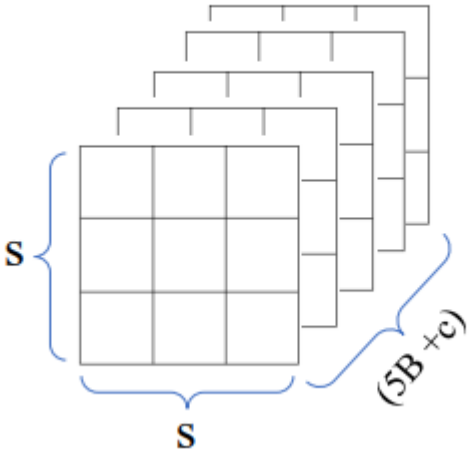


Figure 3.24 : tenseur spécifiant les emplacements de la boite englobante et probabilités de classe [39]

c) Suppression non maximale :

Cette étape est la dernière dans l'algorithme de détection. Elle est utilisée si le même objet dans l'image détecter par plusieurs boites englobantes. Cette technique est utilisée pour "supprimer" les boîtes englobantes les moins probables et ne garder que les meilleures.

Le processus de cette technique passe par 5 étapes [42] :

✓ **Étape 1 :** Sélectionner la boîte avec le score d'objectivité le plus élevé

✓ **Étape 2 :** Ensuite, comparer le chevauchement (intersection sur union) de cette boîte avec d'autres boîtes.

✓ **Étape 3 :** Supprimez les boîtes englobantes de minimal score dont le chevauchement (intersection sur union) est > 50% (on peut changer cette seuille) .

✓ **Étape 4 :** Passer ensuite au score d'objectivité le plus élevé suivant

✓ **Étape 5 :** Enfin, répéter les étapes 2 à 4 jusqu'à fini tous les objets dans l'image

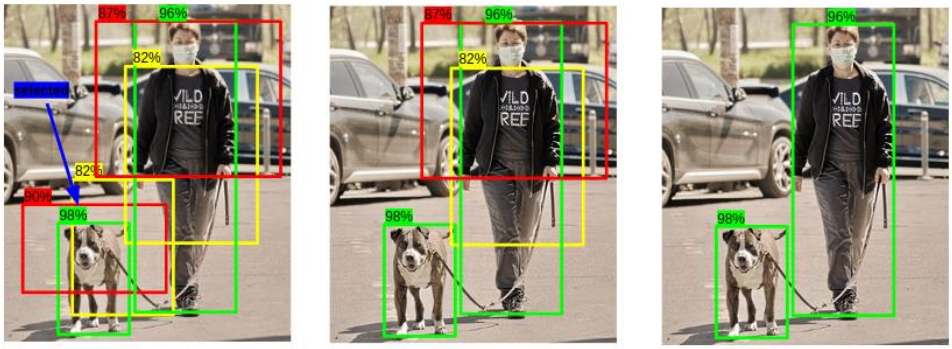


Figure 3.25: les résultats de suppression non maximale [42]

d) Le score de confiance de classe :

Le score de confiance de classe pour chaque boîte de prédiction est calculé comme suit :

$$\text{class confidence score} = \text{box confidence score} \times \text{conditional class probability} \quad \textbf{(3.1)}$$

Il mesure la confiance à la fois sur la classification et la localisation (où se trouve un objet). Nous pouvons facilement confondre ces termes de notation et de probabilité. Voici les définitions mathématiques pour votre future référence.

$$\begin{aligned} \text{box confidence score} &\equiv P_r(object) \cdot IoU \\ \text{conditional class probability} &\equiv P_r(class_i|object) \\ \text{class confidence score} &\equiv P_r(class_i) \cdot IoU \\ &= \text{box confidence score} \times \text{conditional class probability} \end{aligned} \quad \textbf{(3.2)}$$

Avec

Pr(object) est la probabilité que la boite contienne un Objet.

IoU est le résultat de l'intersection sur union entre la boite prédit et la vérité de terrain

Pr(class i) est la probabilité de la classe dans la boite

Pr(classi|object) est la probabilité que l'Objet dans la boîte englobante appartient à une classe

Les scores passent également par un sigmoïde, car il doit être interprété comme une probabilité entre 0 et 1.[37]

e) Prédiction à différentes échelles :

YOLO v4 fait des prédictions sur 3 échelles différentes. La couche de détection est utilisée pour effectuer la détection sur des cartes de caractéristiques de trois tailles différentes, ayant des foulées (stride) 32, 16, 8 respectivement. Cela signifie qu'avec une entrée de 416 x 416, nous effectuons des détections sur les échelles 13 x 13, 26 x 26 et 52 x 52 (voir figure 3.26) [37].

Le réseau sous-échantillonne l'image d'entrée jusqu'à la première couche de détection, où une détection est effectuée à l'aide de cartes d'entités d'une couche avec un pas de 32. En outre, les couches sont suréchantillonnées d'un facteur de 2 et concaténées avec des cartes d'entités d'une couche précédente ayant une carte d'entités identique de tailles. Une autre détection est maintenant effectuée au niveau de la foulée 16. La même procédure de suréchantillonnage est répétée et une dernière détection est effectuée au niveau de la foulée 8.

À chaque échelle, chaque cellule prédit 3 boîtes englobantes à l'aide de 3 ancres, ce qui porte le nombre total d'ancres utilisées à 9 (Les ancres sont différentes pour différentes échelles)

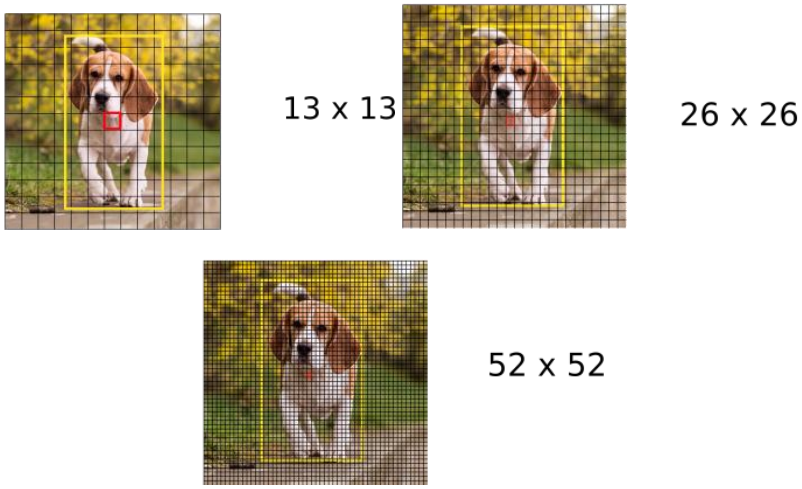


Figure 3.26: les déférentes échelle de détection [37]

Les auteurs rapportent que cela aide YOLOv4 à mieux détecter les petits objets, une plainte fréquente avec les versions antérieures de YOLO. Le suréchantillonnage peut aider le réseau à apprendre des fonctionnalités à grain fin qui sont essentielles pour détecter de petits objets.[37]

3.6.4 Avantages et inconvénients du YOLOv4

A- Les Avantages de YOLO v4:

- Traiter les images à la vitesse de 62 ips (réseau plus grand) à 120 ips (réseau plus petit), ce qui est mieux que le temps réel.
- Le réseau est capable de mieux généraliser l'image [38].

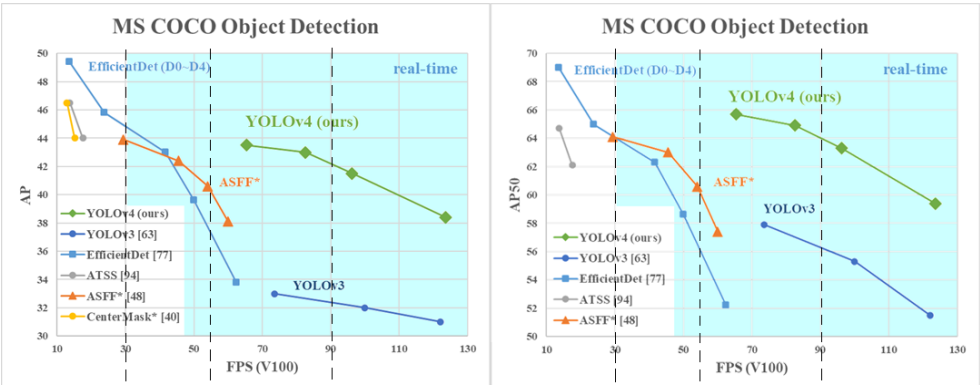


Figure 3.27 : performances de yolov4 par FPS et AP [33]

D’apres la figure 3.27 On peut voir que le YOLOv4 se débrouille très bien en détection en temps réel, atteignant une précision Moyenne (AP : average precision) entre 38 AP et 44 AP, et des images par seconde entre 60 et 120 [33].

B- Les Inconvénients de YOLOv4:

- Rappel comparativement faible et plus d’erreur de localisation par rapport à Faster R_CNN [38].
- Difficulté à détecter les objets proches car chaque grille ne peut proposer que 3 boîtes englobantes.

3.6.5 Conclusion

Dans ce chapitre nous avons présenté l’intelligence artificielle et ses domaines (apprentissage profond et automatique), les différentes architectures de l’apprentissage et la détection d’objet et ses modèles. Finalement nous avons expliqué le fonctionnement du model YOLO V4.

Chapitre 4 Implémentation et conception du système

4.1 Introduction :

Dans ce chapitre, nous discuterons de tout ce qui concerne la mise en œuvre de notre système, de l'environnement de travail à la formation du système sur yolov4 et à la création de la partie traitement d'image pour calculer le diamètre, puis à l'analyse des résultats que nous avons obtenus et à la mise en œuvre de notre système dans une interface graphique d'application pour ordinateur portable.

4.2 Environnement de travail

4.2.1 Spécifications de l'ordinateur portable et la Webcam :

• l'ordinateur portable :

Nous avons travaillé sur notre projet avec un ordinateur portable avec ces capacités :

- Dell Latitude 5320
- Processeur : Intel i7 11ème génération
- Mémoire vive : 16 Go ddr4
- Stockage : 500 Gb SSD
- GPU : Intel iris Xe – intégré

•Webcam Logitech 920 e :

Nous avons utilisé cette caméra pour tester notre système sur lequel nous travaillons, et voici ses spécifications :

- Qualité d'image : HD 1080
- FPS : 30fps
- Résolution optique : 3MP
- Champ de vision : 78°

4.2.2 Logiciels utilisés :

A- CVAT :

Computer Vision Annotation Tool (CVAT) est un outil d'annotation d'images et de vidéos gratuit, open source et basé sur le Web qui est utilisé pour étiqueter les données des algorithmes de vision par ordinateur.[43]

B- Roboflow :

Roboflow est une plate-forme de vision par ordinateur qui permet aux utilisateurs de créer des modèles de vision par ordinateur plus rapidement et avec plus de précision grâce à la fourniture de meilleures techniques de collecte de données, de prétraitement et de formation de modèles. Roboflow permet aux utilisateurs de télécharger des jeux de données personnalisés, de dessiner des annotations, de modifier les orientations des images, de redimensionner les images, de modifier le contraste des images et d'augmenter les données. Il peut également être utilisé pour former des modèles.[44]

C- Google Collab :

Collaboratory, ou « Colab » en abrégé, est un produit de Google Research. Colab permet à quiconque d'écrire et d'exécuter du code python arbitraire via le navigateur et est particulièrement bien adapté à l'apprentissage automatique, à l'analyse de données et à l'éducation.[45]

La grande chose à propos de Google Collab est qu'il nous offre un GPU gratuit de 8 Go que nous pouvons utiliser pour rendre la formation plus rapide et plus facile, mais avec la version gratuite, vous ne pouvez pas travailler plus de 12 heures d'affilée, nous avons donc acheté la version pro et nous avons obtenu un GPU Nvidia P100 avec 25 Go et avec un temps de travail illimité.

D- PyCharm:

PyCharm est un environnement de développement intégré (IDE) utilisé en programmation informatique, spécifiquement pour le langage de programmation Python. Il est développé par la société tchèque JetBrains. Il fournit une analyse de code, un débogueur graphique, un testeur d'unité intégré, une intégration avec des systèmes

de contrôle de version (VCS), et prend en charge le développement Web avec Django ainsi que la science des données avec Anaconda. [46]

4.2.3 Bibliothèques utilisées :

• **OPENCV :**

OpenCV (Open Source Computer Vision Library) est une bibliothèque logicielle de vision par ordinateur et d'apprentissage automatique open source. OpenCV a été conçu pour fournir une infrastructure commune pour les applications de vision par ordinateur et pour accélérer l'utilisation de la perception des machines dans les produits commerciaux.[47]

• **TENSORFLOW :**

TensorFlow est une bibliothèque Python pour l'informatique numérique rapide créée et publiée par Google. Il s'agit d'une bibliothèque de base qui peut être utilisée pour créer des modèles Deep Learning directement ou en utilisant des bibliothèques wrapper qui simplifient le processus construit sur TensorFlow.[48]

• **NUMPY :**

NumPy est une bibliothèque Python utilisée pour travailler avec des tableaux. Il a également des fonctions pour travailler dans le domaine de l'algèbre linéaire, de la transformée de Fourier et des matrices. NumPy a été créé en 2005 par Travis Oliphant. C'est un projet open-source et vous pouvez l'utiliser librement. NumPy signifie Numerical Python.[49]

• **Tkinter :**

Le tkinterpackage ("Tk interface") est l'interface Python standard de la boîte à outils Tcl/Tk GUI. Tk et tkintersont disponibles sur la plupart des plates-formes Unix, y compris macOS, ainsi que sur les systèmes Windows.[50]

4.3 Implémentation du système

Dans cette partie, nous allons combiner deux méthodes de détection, la première est la version complète yolov4 pour détecter l'objet (la pièce) et la seconde avec traitement d'image pour identifier l'objet en fonction de son diamètre, nous allons donc diviser cette partie en 3 sous-parties :

1. Détection d'objets à l'aide de yolov4
2. Calcul du diamètre à l'aide d'une méthode de traitement d'image
3. Combinaison des deux sous-parties 1 et 2 de manière à obtenir un produit final.

4.3.1 Détection à l'aide de yolov4 :

Dans cette partie, nous allons créer une base de données et entraîner notre réseau yolov4 et enfin faire quelques tests pour obtenir des statistiques sur notre système

A- Création de base de données :

- Collecte d'images pour l'ensemble de données :

La première étape de la création de base de données a été de rassembler des images de pièces de monnaies algériennes. L'acquisition a été réalisé par des téléphones portables à différents endroits et différents arrière-plans et pour différentes pièces (voir Figure 4.1).

Nous avons réussi à prendre 900 photos et c'était divisé comme ceci dans le tableau 4.1 :

Pièce	Images de tête	Images de queue	Total Images
Petit 5 da	50	50	100
Grand 5 da	50	50	100
Petit 10 da	50	50	100
Grand 10 da	50	50	100
20 da	50	50	100
50 da	50	50	100
Ancien 100 da	50	50	100

Nouveau 100 da	50	50	100
200 da	50	50	100
Total	450	450	**900**

Table 4.1 : les nombre des images nous avons prendre pour chaque pièce.

COIN	Images
Petit 5da	
Grand 5da	
Petit 10da	
Grand 10da	
20da	

50da	
Ancien 100da	
Nouveau 100da	
200da	

Figure 4.1 : Quelques échantillons d'images de la base de données utilisée.

- **Etiquetage d'images :**

Après avoir rassemblé 900 images pour la base de données, l'étape suivante consistait à les étiqueter pour obtenir les données dont nous avons besoin pour chaque image ; comme la classe de la pièce et les coordonnées de la boîte englobante, car c'est exactement ce dont yolov4 a besoin pour entraîner le jeu de données.

Pour étiqueter les images, nous avons utilisé le logiciel CVAT. La figure 4.2 montre quelques images du travail réalisé dans ce contexte.

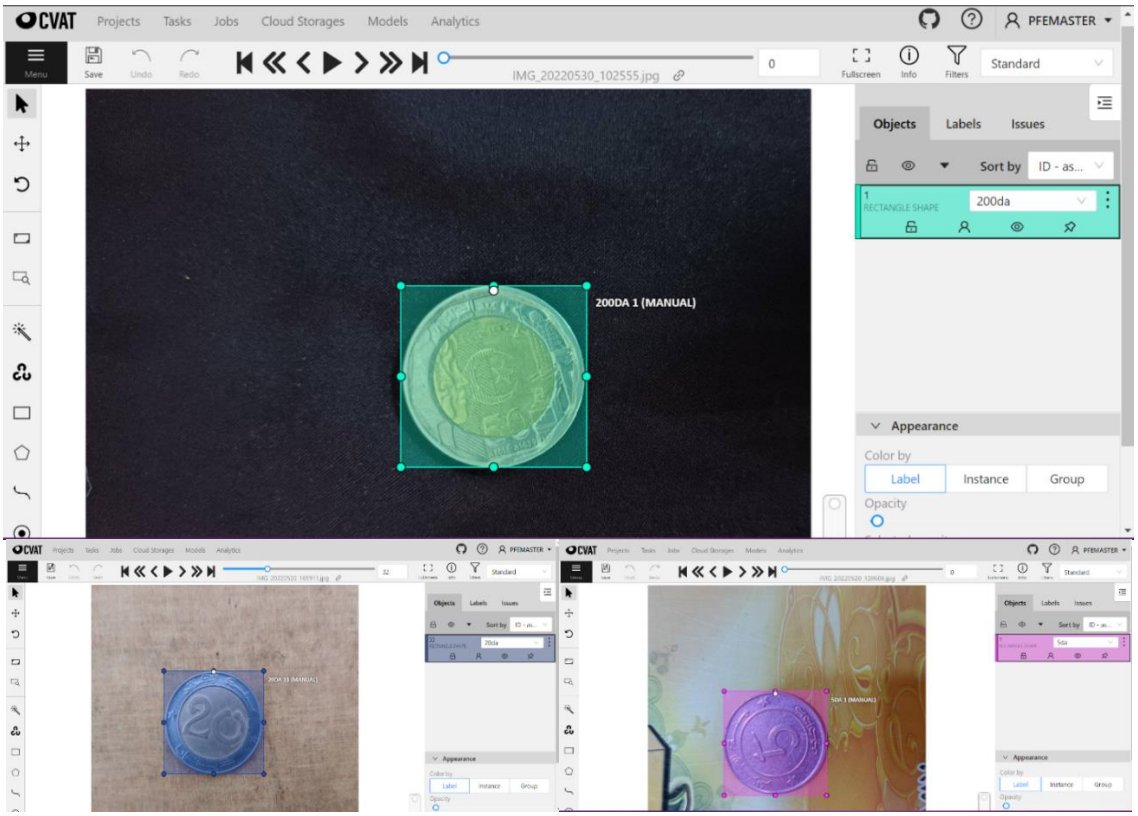


Figure 4.2 : Etapes d'étiquetage sur CVAT

Lorsque nous avons terminé l'étiquetage de toutes les images, nous enregistrons l'étiquetage au format Yolo1.1 (voir figure 4.3) et nous le téléchargeons sous forme de fichiers txt qui a l'ID de classe pour chaque image avec les coordonnées de la boîte englobante (coordonnées contiennent ; bx , by , bh et bw), avec bx, by les coordonnées du centre de l'objet et bw, bh la largeur et la hauteur de l'objet (voir figure 4.4).

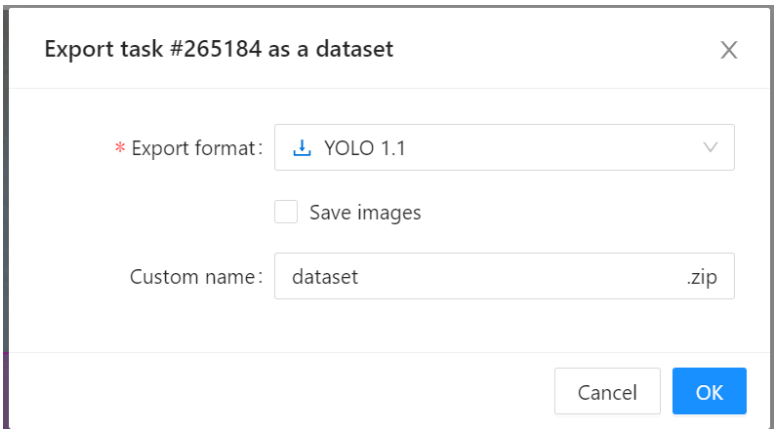


Figure 4.3 : Enregistrement de l'étiquetage sous format yolo1.1 dans CVAT.

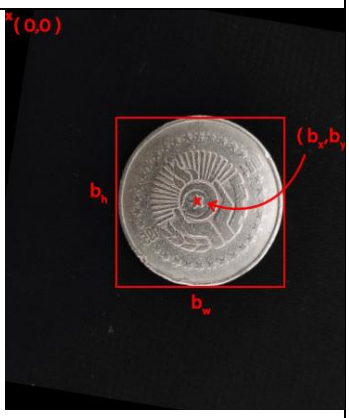
Image original	Fichier txt généré par CVAT	Explication
		

Figure 4.4 : Exemple et explication du format yolo1.1 généré par CVAT

- **Augmentation de la base de données :**

Une fois la base de données avec les annotations de chaque image est prête, de nouveaux exemples d'apprentissage sont créés en générant des versions augmentées pour chaque image de la base de données d'entraînement (voir figure 4.5).

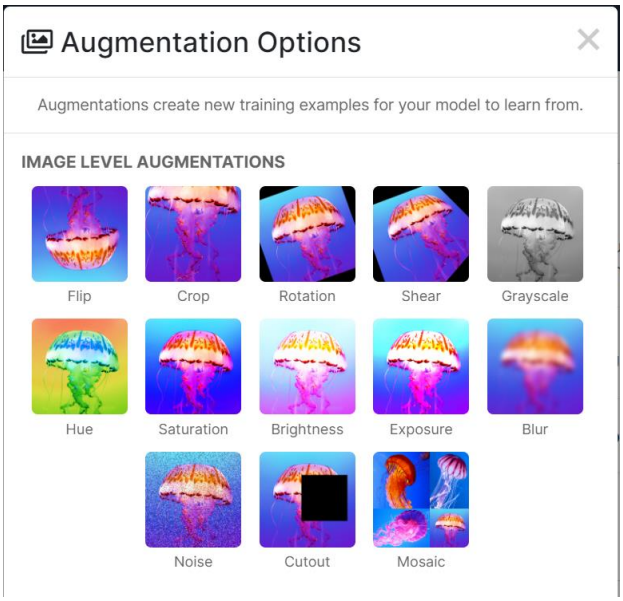


Figure 4.5 : Types des augmentations à Roboflow

- **Types des augmentations utilisés :**

i) flip vertical & horizontal

ii) 45° and -45° rotation

iii) 90° & -90° rotation

iiii) Shear vertical & horizontal avec 15°

Nous évitons de faire une augmentation avec flou, bruit, teinte, luminosité, exposition, contraste, découpe et niveaux de gris car si nous les utilisons, nous perdrons les détails de la pièce et ce sera mauvais pour l'entraînement

Une fois l'augmentation terminée, nous avons réussi à augmenter le nombre d'images dans l'ensemble de données de 900 images à 6300 images.

B- Entrainement du réseau Yolov4 :

Dans la partie de l'entrainement du réseaux Yolov4, nous avons utilisé google collab pour rendre la formation plus facile et plus rapide en utilisant le 25 Go de GPU qu'il nous offre.

La première chose que nous devons faire est de cloner le référentiel darknet dans le collab (voir figure 4.6) car le YOLO v4 en a besoin pour fonctionner, puis d'apporter quelques modifications pour activer le GPU et OPENCV dans le darknet.

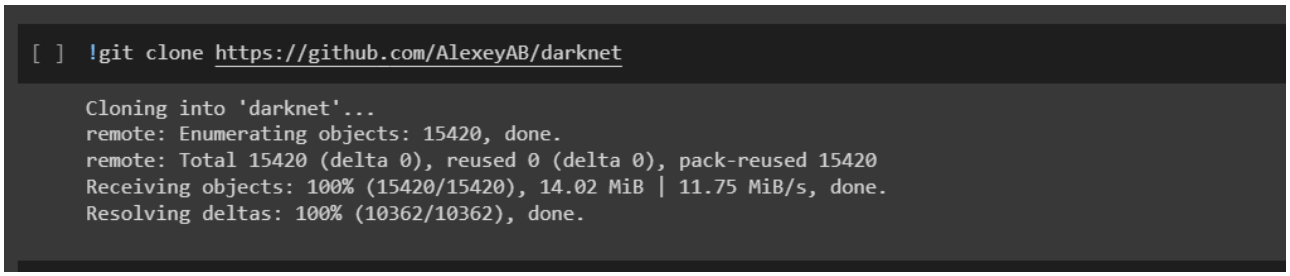


Figure 4.6 : Cloner le référentiel darknet dans le collab

Ensuite, nous téléchargeons notre base de données avec quelques fichiers supplémentaires (fichier names: contenir les nom des classes, fichier data: contenir le nombre de classes et le chemin où nous devons enregistrer le modèle de formation final (voir figure 4.7)) .

```
[ ]  # Copy the obj.names and obj.data files from your drive so that they are now in /darknet/data/ folder

     !cp /content/yolov4/obj.names data
     !cp /content/yolov4/obj.data  data

     # verify if the above files are in data folder
     !ls data/

     labels  obj  obj.data  obj.names
```

Figure 4.7 : Importation des fichier obj.names et obj.data

L'étape suivante consiste à importer le fichier yolov4 cfg (voir figure 4.8)

```
[ ]  # Copy the yolov4-custom.cfg file so that it is now in /darknet/cfg/ folder

     !cp /content/yolov4/yolov4-custom.cfg cfg

     # verify if your custom file is in cfg folder
     !ls cfg/

     yolov4-custom.cfg
```

Figure 4.8 : Importation du fichier yolov4 cfg

Le fichier cfg contient toutes les informations nécessaires pour l'entrainement comme : nombre de classes, nombre d'itérations, nombre de filtres, et beaucoup d'informations supplémentaires (voir figure 4.9).

La plupart des informations restent les mêmes car il forme l'architecture du cspdarknet53, nous ne changeons que ces variables :

Max_batches= 2000*num_classes = **16000**

Steps = 80% max_batches , 90% max_batches = 12800,14400

Filtres [1,1]= B*5+C= 3*5+8=23

Comme nous avons expliqué dans le chapitre 3, le Yolov4 contient un sac de cadeaux qui offre une augmentation automatisée de la base de données en générant de nouvelles images avec une teinte, une saturation, une exposition ...etc. Ces images sont retirées du fichier CFG (voir figure 4.9) parce qu'elles ont une tendance à perdre leurs détails, par contre nous avons déjà fait des augmentations dont nous avons besoin avec Roboflow.

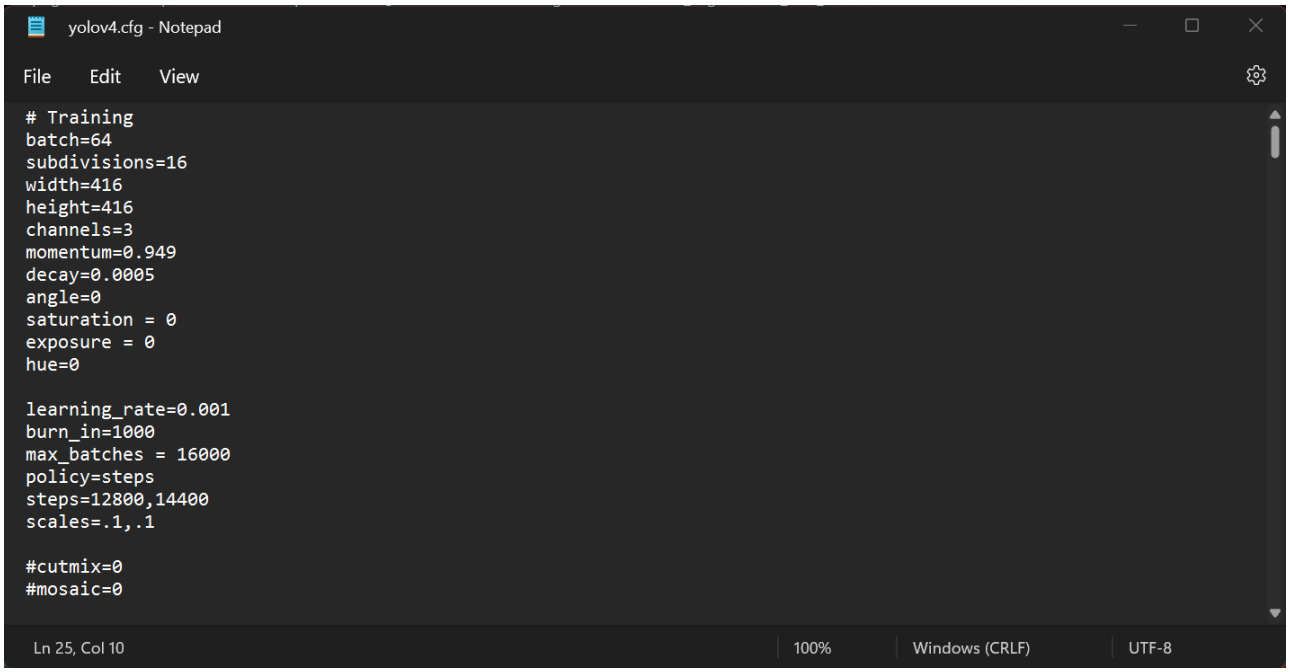


Figure 4.9 : Fichier cfg de yolo v4

Maintenant, nous générons 3 fichiers txt pour diviser la base de données en 3 parties (voir figure 4.10) :

• 70% pour l'entrainement

• 20% pour la validation

• 10% pour les tests

Ces fichiers txt contiennent les noms des images que nous avons dans la base de données et nous avons fait un script python pour diviser ces images en 3 parties avec (70% train, 20% validation, 10% test).

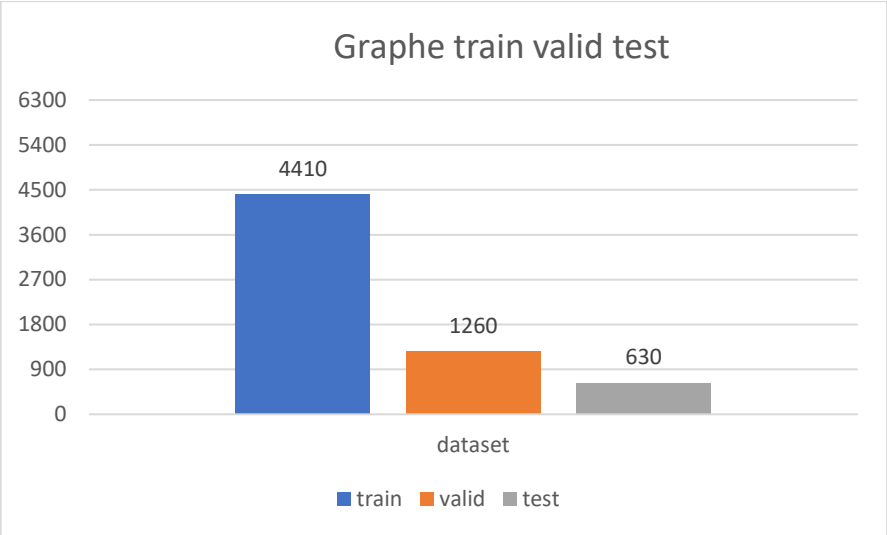


Figure 4.10 : la division du base de donnes pour (train/ valid/ test)

Dans la prochaine étape nous téléchargons un fichier de poids (weights) pré-entraîné du Yolov4 (voir figure 4.11). Cette version pré-entraînée dans la base de données COCO qui contient 80 classes, et qui provient par l'éditeur original de yolov4 Alexey AB.

```
# Download the yolov4 pre-trained weights file
!wget https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.conv.137

https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.conv.137
Resolving github.com (github.com)... 52.192.72.89
Connecting to github.com (github.com)|52.192.72.89|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/75388965/48bfe500
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.111.133, 185.199.108.
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.111.133|:443... conn
HTTP request sent, awaiting response... 200 OK
Length: 170038676 (162M) [application/octet-stream]
Saving to: 'yolov4.conv.137'

yolov4.conv.137     100%[===================>] 162.16M  13.5MB/s    in 14s
```

Figure 4.11 : Téléchargement du fichier de poids (weights) pré-entraîné de yolov4.

La dernière étape consiste à commencer l'entraînement et à attendre que le train soit terminé.

L'entrainement du réseau dure plus de 15 heures.

Les figures 4.12, 4.13 et 4.14 montrent le graphe de performance de l' entraînement.

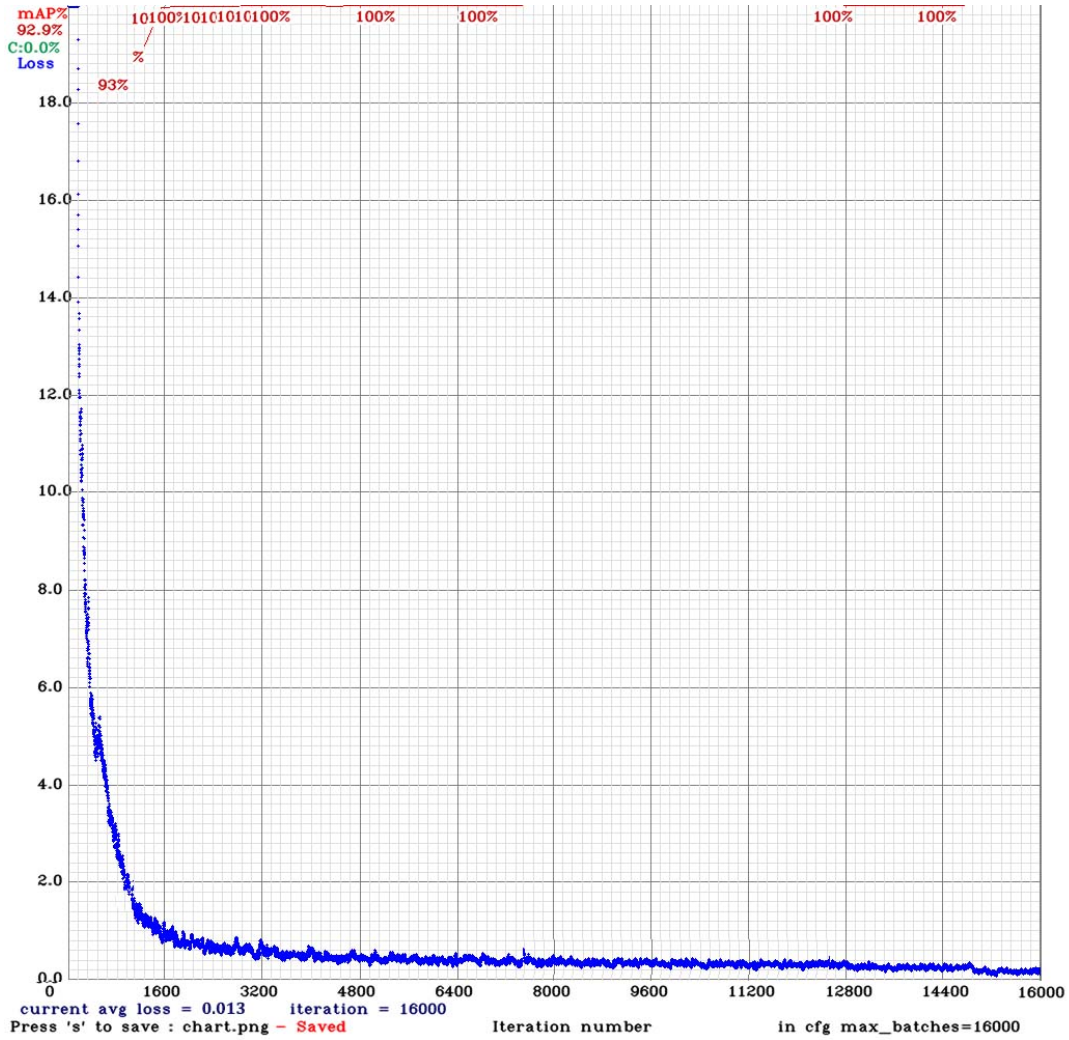


Figure 4.12 : graphe de la performance de notre entraînement de yolo v4

Le graphe de la figure 4.12 montre la variance du mAP (Précision moyenne) en fonction du nombre d'itération , le graphe est actualisé automatiquement lors l'entraînement à Google Collab.

Précision de **: 99.83%** avec perte moyenne de : **0.013%**

```
 Tensor Cores are used.
 Last accuracy mAP@0.50 = 99.82 %, best = 99.83 %
 16000: 0.009445, 0.013198 avg loss, 0.000010 rate, 1.943898 seconds, 1024000 images, 0.040615 hours left
Resizing to initial size: 416 x 416  try to allocate additional workspace_size = 72.43 MB
 CUDA allocate done!

 calculation mAP (mean average precision)...
 Detection layer: 139 - type = 28
 Detection layer: 150 - type = 28
 Detection layer: 161 - type = 28
```

Figure 4.13 : Performance de l'entraînement de YOLO v4.

```
 detections_count = 1546, unique_truth_count = 1084
class_id = 0, name = 5da, ap = 99.99%           (TP = 99, FP = 5)
class_id = 1, name = 5da, ap = 100.00%          (TP = 133, FP = 0)
class_id = 2, name = 10da, ap = 100.00%         (TP = 98, FP = 0)
class_id = 3, name = 10da, ap = 100.00%         (TP = 110, FP = 0)
class_id = 4, name = 20da, ap = 98.65%          (TP = 146, FP = 2)
class_id = 5, name = 50da, ap = 100.00%         (TP = 185, FP = 5)
class_id = 6, name = 100da, ap = 99.98%         (TP = 152, FP = 2)
class_id = 7, name = 200da, ap = 99.99%         (TP = 158, FP = 1)

 for conf_thresh = 0.25, precision = 0.99, recall = 1.00, F1-score = 0.99
 for conf_thresh = 0.25, TP = 1081, FP = 15, FN = 3, average IoU = 91.71 %
```

Figure 4.14 : Précision obtenue pour chaque classe.

Vrais positifs (TP) = le nombre de pièces correctement identifiés.

Faux positifs (FP) = le nombre de pièces incorrectement identifiés.

Après l'obtention de ces résultats, nous avons pris 420 nouvelles images pour tester le système proposé manuellement. Pour cela, nous avons créé un script pour tester chaque image et l'enregistrer avec l'annotation. Les résultats de la classification sont récapitulés dans la table 4.2.

Type de pièce	Nombre d'images	Vrai résultat	Précision %	Erreur %
Petite 5da	50	48	96%	4%
Grande 5da	50	50	100%	0%
Petite 10da	50	49	98%	2%
Grande 10da	50	50	100%	0%
20da	50	49	98%	2%
50da	50	50	100%	0%
100da	70	69	98.57%	1.43%
200da	50	49	98%	2%
TOTAL	**420**	**415**	**98.8 %**	**1.2%**

Table 4.2 : Tableau de résultats obtenus pour l'ensemble de test.

$$Précision = \frac{Vrai\ résultatl}{Nmbr\ des\ tests}\ 100 \qquad (4.1)$$

$$Erreur = \frac{faux\ résultat}{nmbr\ des\ tests}\ 100 \qquad (4.2)$$

Pendant la phase de test, nous avons aussi essayer de tester les performances du système en lui présentant de fausses pièces. Les résultats obtenus par ce test sont présentés dans la table 4.3.

Pièces	Nombre des tests	Vrai	Précision %
Fausse pièce	50	24	48%

Table 4.3 : Tableau contenant les résultats de test pour les fausses pièces

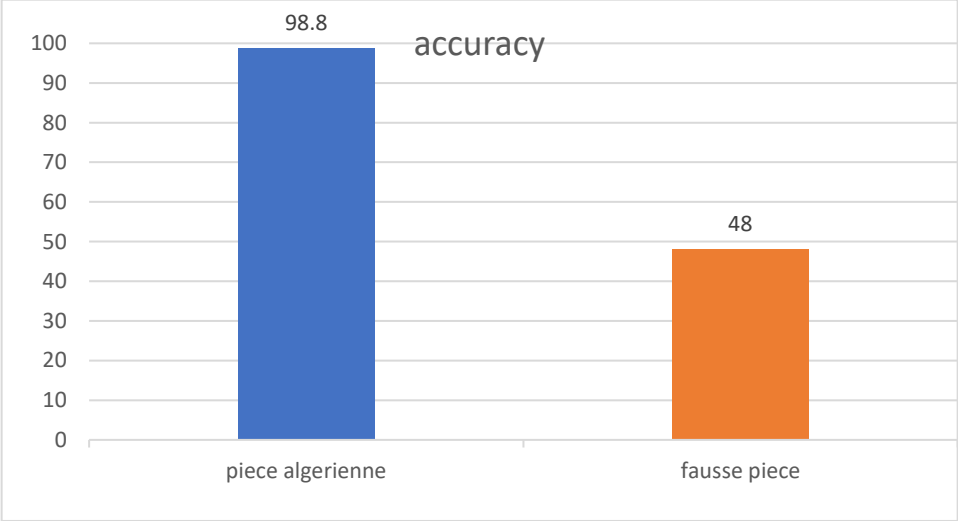


Figure 4.15 : Graphe montrant les performances du système proposé.

4.3.2 Calcul du diamètre :

Pour augmenter la précision, nous avons décidé d'introduire une partie de traitement d'image pour le calcul le diamètre de la pièce de monnaie. Chaque pièce a un diamètre bien précis et unique. Pour ce faire, cette partie doit passer par les étapes suivantes (l'arrière-plan ou fond doit être de couleur noir):

1. •Conversion de l'image couleur en image à niveaux de gris.
2. •Binarisation avec la méthode d'Otsu.

3. •Application d'une ouverture et fermeture morphologique à l'image binaire.
4. •Calcul de la surface de la pièce.
5. •Calcul du diamètre en nombre de pixels et le convertir en diamètre réel (mm).
6. •Identification la pièce en fonction de son diamètre.

Les diamètres de pièces de monnaie algériennes ont été mesurés à l'aide d'un pied à coulisse (voir figure 4.16). Nous avons fait 50 mesures pour chaque pièce comme le montre la figure la table 4.4.

Figure 4.16 : Mesure manuel de diamètre de pièce de monnaie à l'aide de Pied à coulisse

Coin	1er	2eme	3eme	4eme	5eme	6eme	7eme	8eme	9eme	.....	50eme	moyenne	l'ecart type
5da_p	24.54	24.56	24.55	24.58	24.56	24.57	24.54	24.54	24.5	.....	24.54	24.552	0.014
5da_G	30.9	31.01	30.92	30.9	30.9	31.13	30.9	30.89	31	.....	30.9	30.94	0.0125411
10da_p	26.56	26.53	26.6	26.56	26.56	26.55	26.54	26.53	26.5	.....	26.56	26.552	0.02039608
10da_G	27.96	27.9	27.93	27.96	27.98	27.98	27.96	27.96	28	.....	28.01	27.962	0.02856571
20da	27.52	27.54	27.55	27.54	27.54	27.54	27.55	27.54	27.5	.....	27.56	27.54	0.01183216
50da	28.56	28.52	28.53	28.57	28.54	28.53	28.56	28.54	28.5	.....	28.56	28.544	0.01624808
100da	29.6	29.55	29.57	29.57	29.56	29.59	29.55	29.55	29.6	.....	29.54	29.564	0.018
200da	28.04	28.05	28.04	28	28	28.04	28.06	28	28	.....	28.04	28.028	0.02181742

Table 4.4 : Mesures de diamètres des pièces à l'aide de pied à coulisse.

$$ecart_{type} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} \qquad (4.3)$$

$$avec\ n = 50, \qquad x: diamètre, \qquad \bar{x}: diamètre_{moy}$$

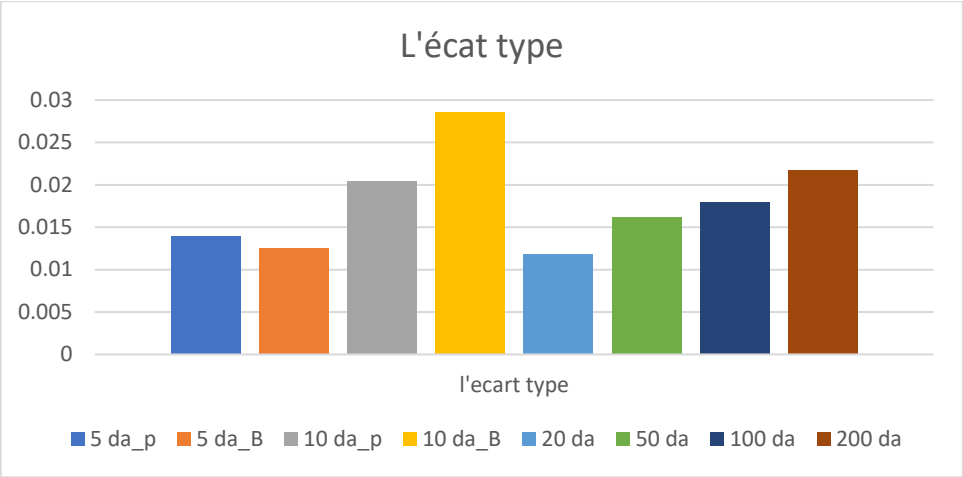


Figure 4.17 : Ecart type de diamètre de chaque pièce.

Dans le section suivante nous allons expliquer différentes étapes de la partie concernant la mesure de diamètre en utilisant le traitement d'image.

A- Conversion de l'image en niveaux de gris :

Pour ce faire, nous utilisons un code python de la bibliothèque **OpenCV** avec l'instruction **Cv2.cvtColor(img,cv2. COLOR_BGR2GRAY)** (voir figure 4.18).

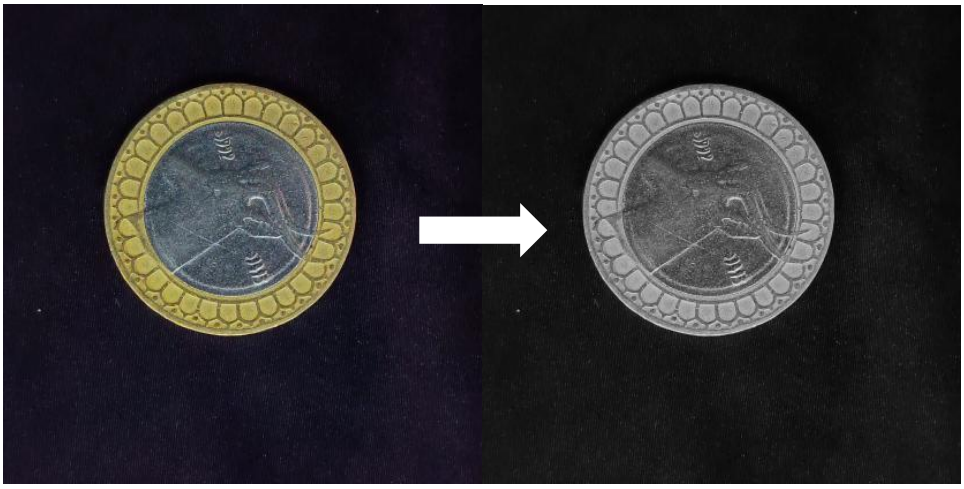


Figure 4.18 : Conversion de l'image couleur en niveaux de gris

Ensuite, nous appliquons un filtre gaussien pour éliminer tout bruit dans l'image en utilisant l'instruction **Cv2.GaussianBlur** (voir figure 4.19).

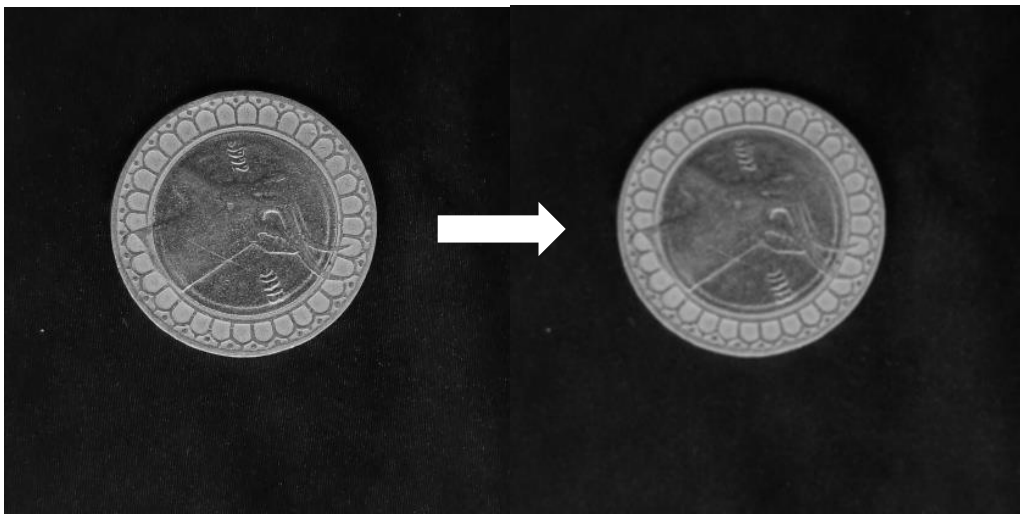


Figure 4.19 : Application d'un filtre gaussien

B- Binarisation de l'image à niveau de gris :

Pour ce faire, nous avons utilisé l'instruction :

cv2.threshold(img,0,255,cv2. THRESH_BINARY+ cv2. THRESH_OTSU) (voir figure 4.20).

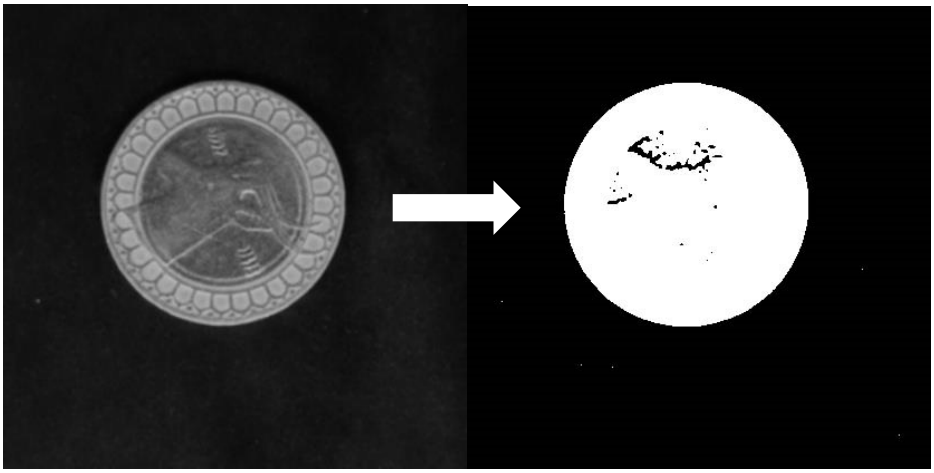


Figure 4.20 : Binarisation de l'image filtrée.

C- Fermeture morphologique :

Pour éliminer les taches noires qui se trouvent à l'intérieur de la pièce (cercle blanc), nous avons utilisé une fermeture morphologique (voir figure 4.21). Cette opération est une dilatation suivie d'une érosion appliquée à l'image de la figure 4.20. Pour ce faire, nous avons utilisé l'instruction suivante :

Cv2.morphologyEx(img , cv2. MORPH_CLOSE, Noyau)

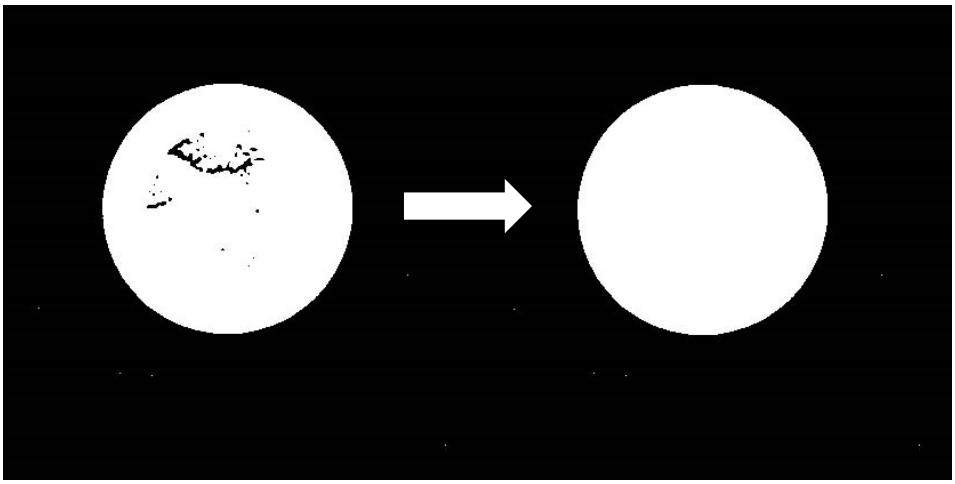


Figure 4.21 : Application de la fermeture morphologique.

D- Ouverture :

Maintenant, nous devons éliminer les taches blanches se trouvant sur l'arrière-plan qui est noir, et pour ce faire, nous devons utiliser une ouverture morphologique (voir figure 4.22) qui signifie appliquer une érosion suivie d'une dilatation à l'image de la figure 4.21 en utilisant l'instruction :

Cv2.morphologyEx(img, cv2. MORPH_OPEN, Noyau)

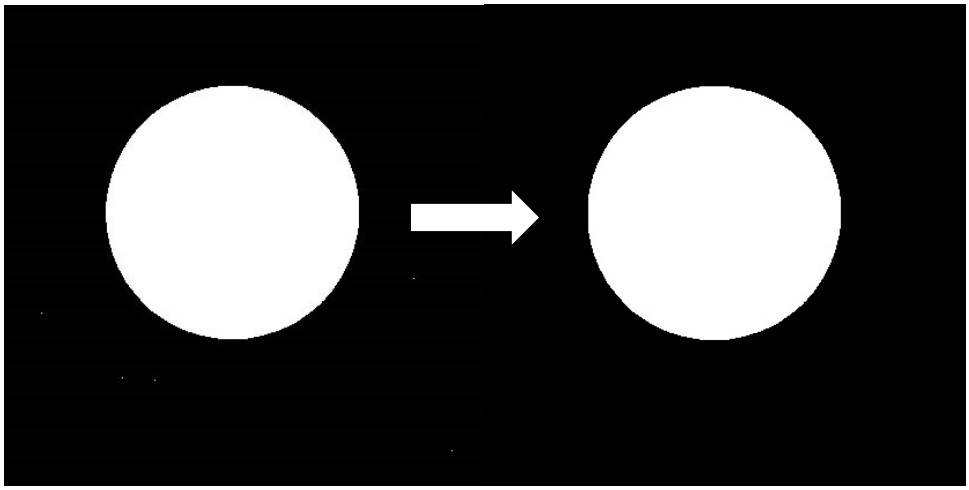


Figure 4.22 : Application de l'ouverture morphologique

E- Calcul de la surface :

Pour calculer la surface, nous devons d'abord trouver les contours, et pour ce faire, nous devons utiliser cette instruction : **cv2.findcontours**

Après avoir trouvé les contours, il suffit d'utiliser l'instruction ci-dessous pour trouver la surface.

Area=cv2.contourArea(contours)

F- Calcul du diamètre :

Pour trouver le diamètre, nous utilisons d'abord une pièce de référence. Avant que le système ne commence à fonctionner, il vous demandera de lui présenter une pièce de référence (dans notre cas, il s'agit de la pièce de 20da parce qu'après les mesures manuel du diamètre avec le pied a coulisse, nous avons pris la pièce qui correspond au minimum d'écart type) afin d'introduire une formule de mise à l'échelle. Cette formule sera développée et expliquée dans la section suivante.

Quand la pièce de référence qui est de 20da est présenté au système, celui-ci calculera la surface de cette pièce en utilisant les étapes que nous avons expliquées précédemment, puis il calculera le diamètre en fonction de la surface car nous connaissons tous la formule de la zone d'un disque :

$$Surface = \pi\ diamètre^2$$

$$diamètre = \sqrt{\frac{Surface}{\pi}} \qquad (4.4)$$

⇨ Une fois le diamètre est calculé, nous ferons une mise à l'échelle car nous savons déjà que le diamètre réel de 20da est de **27,5** mm donc pour trouver le diamètre réel de n'importe quelle pièce on va utiliser la formule suivante :

⇨ $$diamètre_{réel} = \frac{diamètre_{pièce} * 27.5}{diamètre_{ref}} \qquad (4.5)$$

$diamèter_{réel}$: est le diamètre que l'on veut trouver pour identifier la pièce.

$diamèter_{pièce}$: est le diamètre que nous trouvons après calcul de la surface

$diamèter_{ref}$: est le diamètre de 20da que nous trouvons après calcul de la surface

27.5 mm : est le diamètre réel de piece reference 20da.

E- Identification de la pièce :

La dernière étape consiste à comparer le diamètre de la pièce obtenu par la formule 4.2 avec les diamètres présentés à la table 4.4 pour identifier la pièce.

$diamèter_{réel}$ ⟵

$24.3 < diam_{5da} < 24.9$

$26.3 < diam_{10da} < 26.9$

$27.3 < diam_{20da} < 27.7$

$27.8 < diam_{200da} < 28.3$

$28.4 < diam_{50da} < 28.7$

$28.7 < diam_{b10da} < 29.3$

$29.3 < diam_{100da} < 29.7$

$30.7 < diam_{b5da} < 31.3$

⟶ $piece\ identifée$

Exemple : si on calcule le diamètre et qu'on obtient : $diamètre_{réel} = 27.6mm$

$diamèter_{réel} = 27.6mm$ ⟵

$24.3 < diam_{5da} < 24.9$

$26.3 < diam_{10da} < 26.9$

$27.3 < diam_{20da} < 27.7$

$27.8 < diam_{200da} < 28.3$

$28.4 < diam_{50da} < 28.7$

$28.7 < diam_{b10da} < 29.3$

$29.3 < diam_{100da} < 29.7$

$30.7 < diam_{b5da} < 31.3$

⟶ $la\ piece\ est : 20da$

Enfin, nous avons fait quelques tests à l'aide du programme de calcul du diamètre et nous avons trouvé de nouveaux résultats qui sont présentés dans la table 4.5.

Pièce	Nombre des testes	Vrai résultat	Précision %	Erreur %
Petit 5da	50	50	100%	0%
Grand 5da	50	50	100	0%
Petit 10da	50	50	100	0%
Grand 10da	50	47	94%	6%
20da	50	50	100	0%
50da	50	49	98%	2%
100da	70	70	100%	0%
200da	50	48	96%	4%
Total	**420**	**414**	**98.57%**	**1.43%**

Table 4.5 : Résultats de test mettant en évidence le diamètre de la pièce de monnaie.

La raison pour laquelle nous avons eu des erreurs dans Grande 10da, 50da et 200da est qu'ils ont presque le même diamètre, et quand la lumière ambiante change, la surface aussi et cela fait que le système commet des erreurs de confusion.

Pendant la phase de test, nous avons aussi essayer de tester les performances du système en lui présentant de fausses pièces. Les résultats obtenus par ce test sont présentés dans la table 4.6. Les résultats obtenus sont prometteurs et montrent bien la robustesse du système pour l'identification de fausses pièces.

Pièce	Nombre de tests	Rejeté	Précession
Fausse pièce – Type 1	50	50	100%
Fausse pièce – Type 2	50	0	0%

Table 4.6 : Tableau contenant les résultats de test pour les fausses pièces.

Fausse $pièce_{type1}$: c'est une fausse pièce avec un diamètre différent de toutes les pièces.

Fausse $pièce_{type2}$: c'est une fausse pièce avec le même diamètre qu'une des pièces.

Donc, à partir de ce résultat, nous pouvons conclure que ce système peut identifier la pièce en utilisant uniquement son diamètre et sans aucune idée des détails de la pièce.

4.3.3 Combinaison des deux systèmes d'identification

Pour augmenter la précision d'identification de la pièce de monnaies, nous allons combiner les deux systèmes présentés afin que le système puisse identifier la pièce en fonction de son diamètre et ses détails (voir figure 4.23).

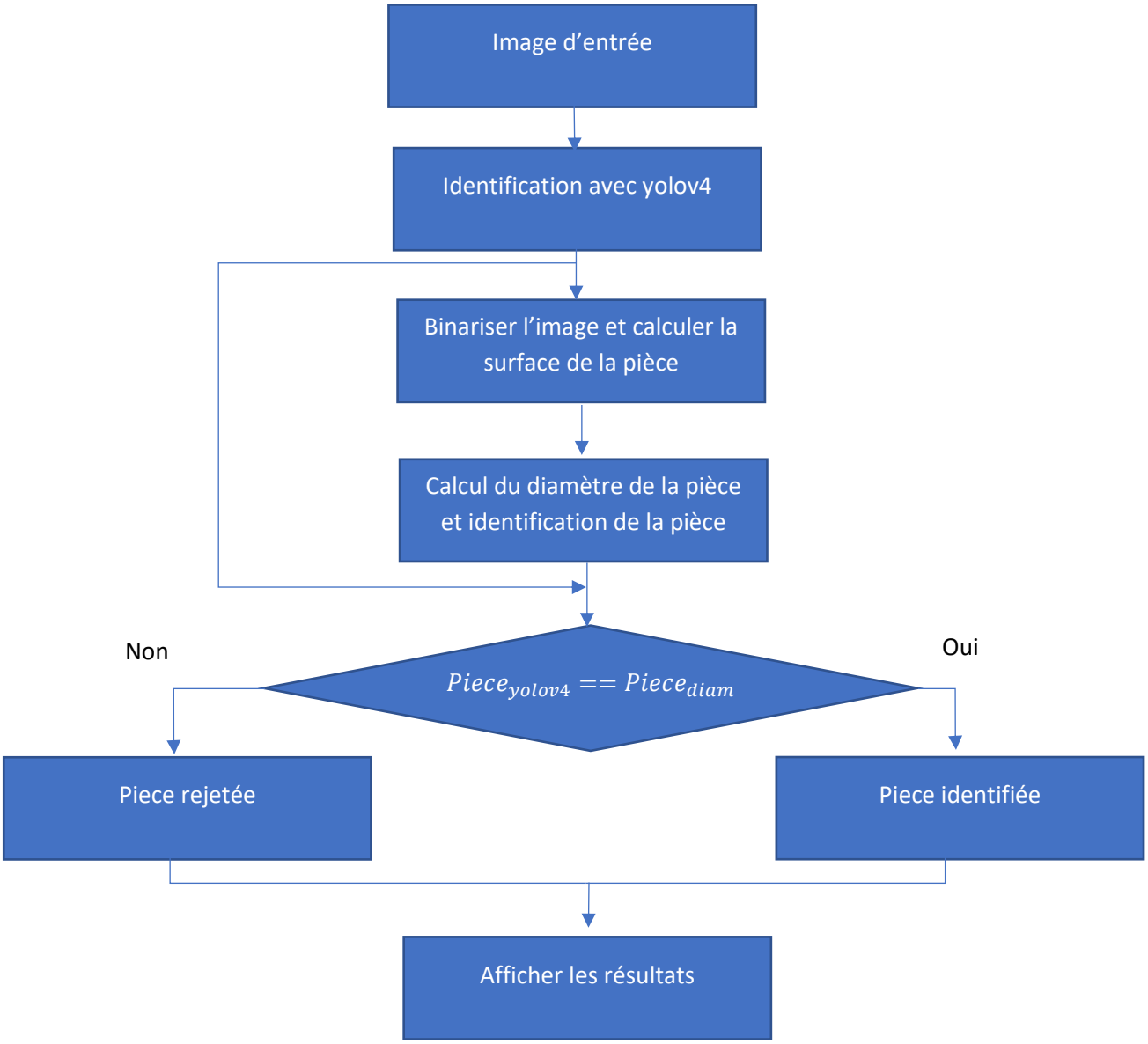


Figure 4.23 : Diagramme de l'architecture du système développé.

Dans tout ce qui suit, nous allons présenter les différentes étapes de l'architecture du système développé (voir figure 4.23).

- **Étape 1 :** Détecter la pièce à l'aide de l'algorithme yolov4 :

Donc, la première chose à faire est de présenter une image à la méthode d'identification Yolov4 (voir figure 4.24).

Figure 4.24 : Exemple d'image d'entrée

Dans cette étape, l'algorithme de yoloV4 détectera la pièce et produit 3 informations : classId, coordonnée de la boîte englobante (bb cord), et le score (voir figure 4.25).

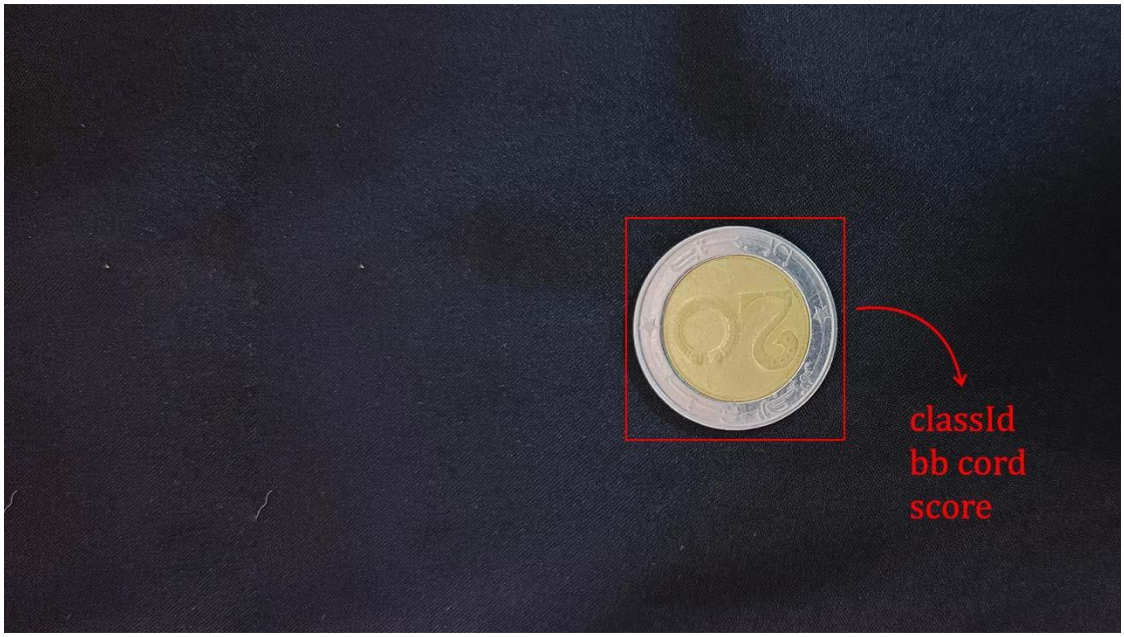


Figure 4.25 : Résultat obtenu par le Yolov4

Ensuite, lorsque nous aurons la coordonnée, nous l'utiliserons pour recadrer la pièce à partir de l'image, mais avant de le faire, nous augmentons la surface de la boîte englobante de 15% (voir figure 4.26) pour sassurer que la pièce sera complètement à l'intérieur.

Figure 4.26 : Recadrage de la pièce et augmentation de la surface de l'image.

- **Étape 2 :** calcul de la surface :

Afin de mieux calculer la zone, nous devons binariser l'image (voir figure 4.27), puis calculer la zone à l'aide de l'instruction de contour.

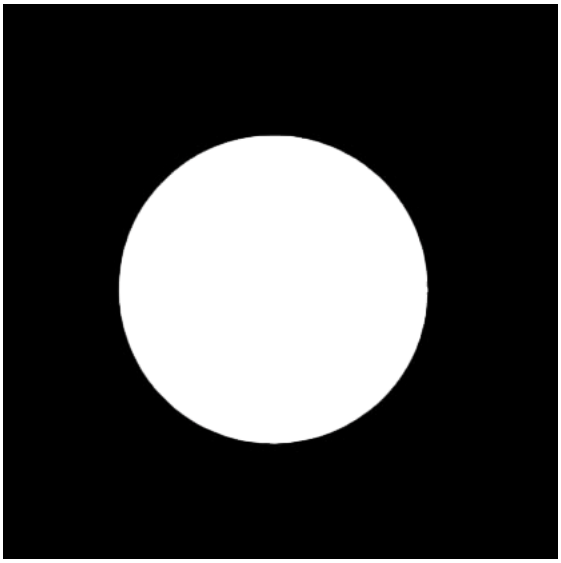


Figure 4.27 : Résultat après la binarisation et fermeture, ouverture pour calculer la surface.

- **Étape 3 :** calcul du diamètre :

Lorsque la surface est calculée, nous pouvons estimer le diamètre en utilisant l'équation (4.2) que nous avons expliqué auparavant.

- **Étape 4 :** comparaison de résultat

Dans cette étape, nous comparons le résultat du yoloV4 dans classId et le résultat de la méthode de diamètre, si les deux résultats correspondent au même type de pièce, cela signifie que la pièce est bien identifiée, sinon la pièce est rejetée (voir figure 4.28).

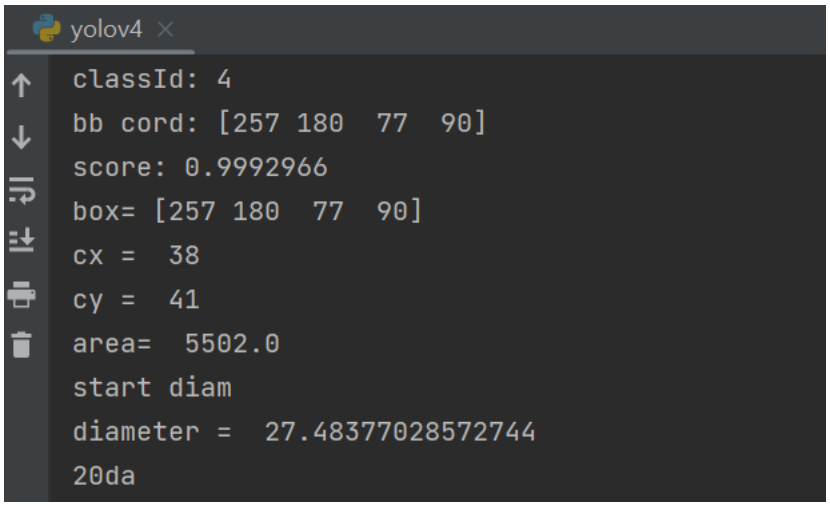


Figure 4.28 : Résultats final obtenu après l'étape de comparaison.

- **Étape 5 :** affichage des résultats

Lorsque la pièce est bien identifiée avec succès, la pièce et recadrée par un rectangle rouge et le résultat de l'identification ainsi que le score sont affiche au-dessus du rectangle (voir figure 4.29).

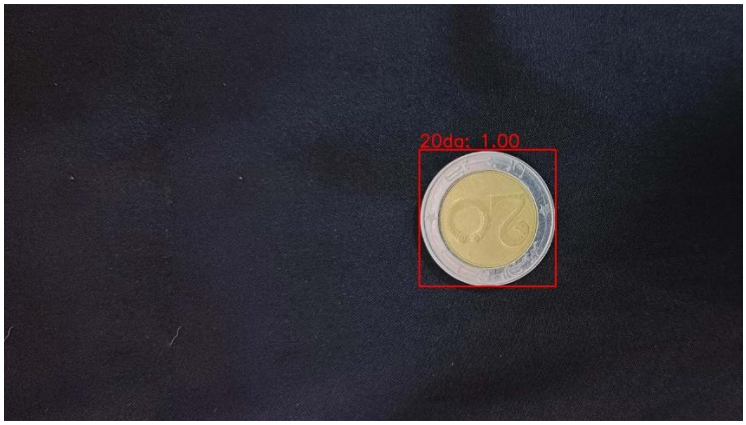


Figure 4.29 : Résultat final après la reconnaissance de la pièce.

4.4 Discussions des résultats :

Dans cette partie, nous discuterons les résultats que nous avons obtenus par le système final qui combine la détection yolov4 et le calculateur de diamètre, et pour ce faire, nous exécutons le système en détection vidéo en direct et nous testons le système 756 fois avec des pièces différentes, et aussi nous ajoutons de fausse pièce (test de 100 pièces) pour tester la robustesse du système.

Pièce	Nombre de tests	Vraie identifié	Piece rejeté	Précision %	Rejet %	Erreur %	Fiabilité %
Petit 5da	100	100	0	100%	0%	0%	100%
Grand 5da	65	65	0	100%	0%	0%	100%
Petit 10da	105	104	1	99.05%	0.95%	0%	100%
Grand 10da	51	51	0	100%	0%	0%	100%
20da	114	114	0	100%	0%	0%	100%
50da	110	110	0	100%	0%	0%	100%
100da	110	110	0	100%	0%	0%	100%
200da	101	100	1	99.01%	0.99%	0%	100%
TOTAL	**756**	754	2	99.74%	0.26%	0%	100%

Table 4.7 : Tableau contenant les résultats de test par pièces.

Pièce	Nombre de tests	Piece rejeté	Erreur %
Fausse Pièce	100	83	17%

Table 4.8 : Tableau contenant les résultats de test pour les fausses pièces.

$$Rejet = \frac{Pièce\ rejeté}{Nmbr\ des\ tests}\ 100 \qquad (4.6)$$

$$Fiabilité = \frac{vraie\ identifé}{vraie\ identifé + faux\ identidé}\ 100 \qquad (4.7)$$

Après ces résultats, nous pouvons conclure que notre système possède une grande précision dans le classement des monnaies algériennes avec 99,74% de précision et 0,26% de rejet et 0% d'erreurs. Pourtant, lorsque nous donnons au système une fausse pièce, il aura du mal à l'identifier quand elle a un diamètre similaire aux pièces de monnaie algériennes.

4.5 Création de l'interface graphique :

A la fin et pour présenter notre système nous avons créé une interface graphique (voir figure 4.30) en utilisant la bibliothèque de **tkinter.**

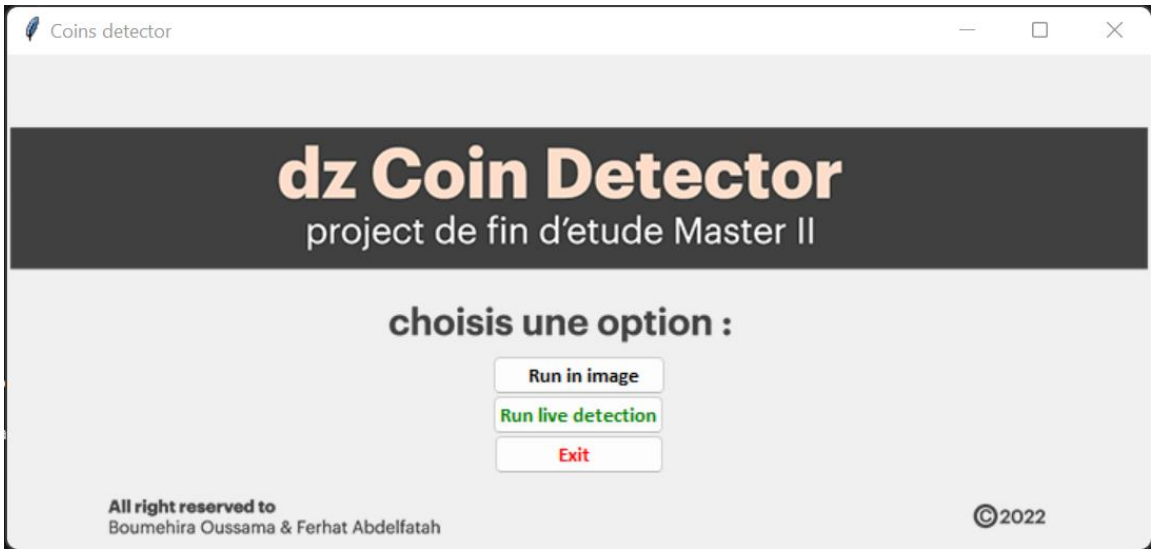


Figure 4.30 : Page d'accueil de L'interface graphique de notre système

L'interface présente deux options, la 1ère est pour faire l'identification de pièce sur une image.

Donc c'est en cliquant sur **« Run in Image »** on choisit une image à partir du PC pour faire l'identification (voir figure 4.31)

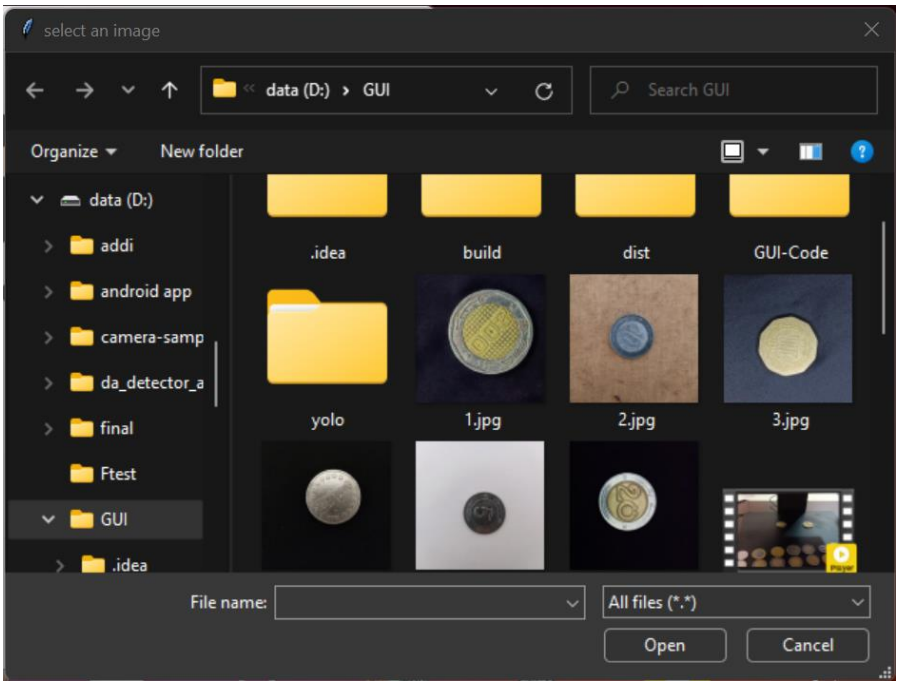


Figure 4.31 : Choix d'une image pour identification de la pièce.

Une fois l'image est choisie notre système fera automatiquement l'identification et affiche les résultats (voir figure 4.32).

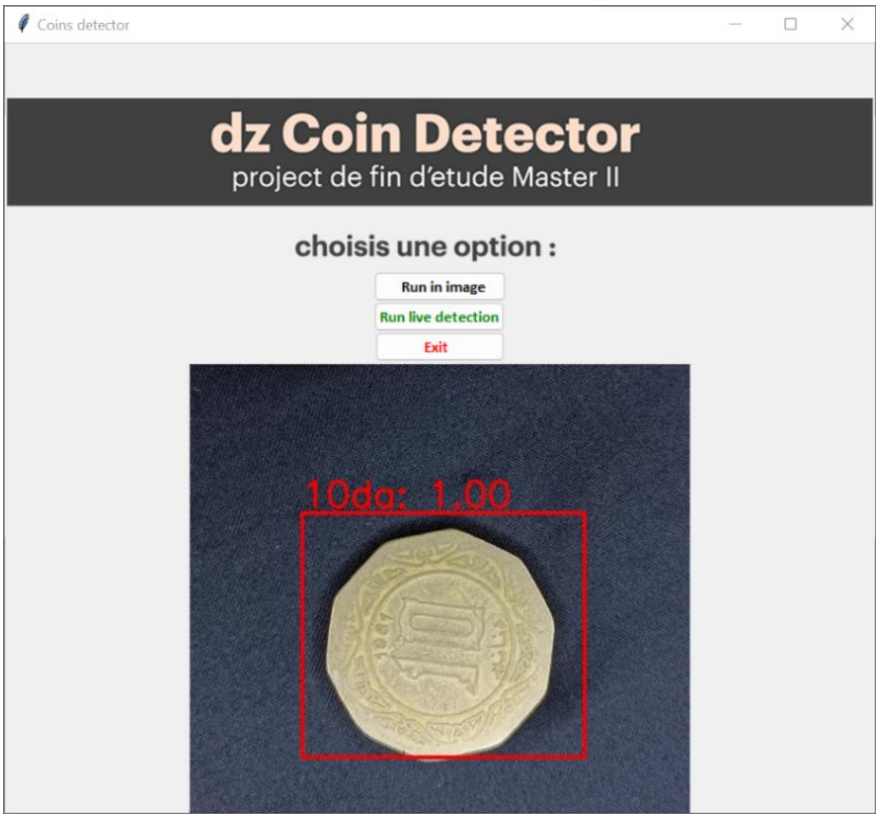


Figure 4.32 : Résultat d'identification pour des images fixes.

Pour la 2eme option, il s'agit de l'identification en temps réel moyennant une camera et notre système affichera les résultats.

Si on clique sur **« Run in live Video »** notre système demande l'emplacement de la pièce référence, on place cette pièce puis on clique sur « S » pour démarrer la détection (voir figure 4.33).

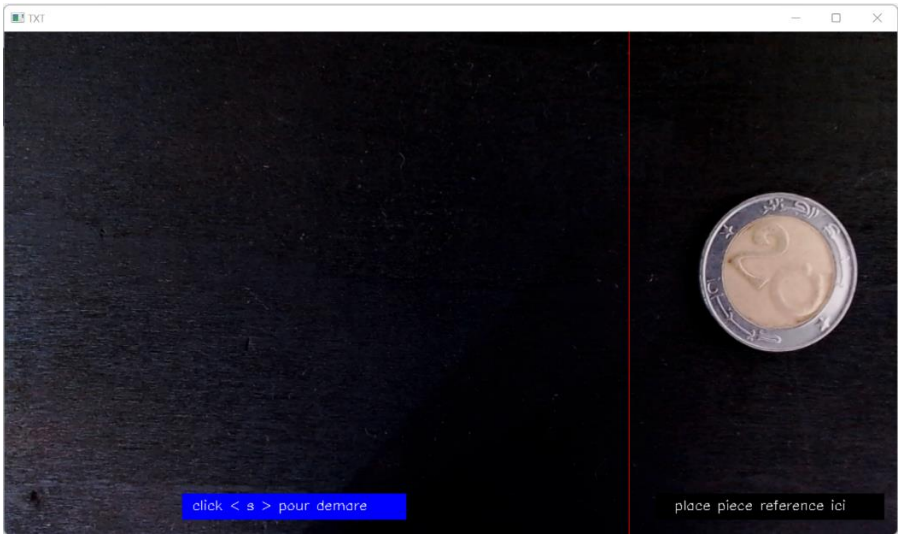


Figure 4.33 : Emplacement de la pièce référence à l'endroit voulu et cliquer sur **S**

Si on clique sur « s » la détection en vidéo démarrera automatiquement en temps réel (voir figure 4.34 et 4.35).

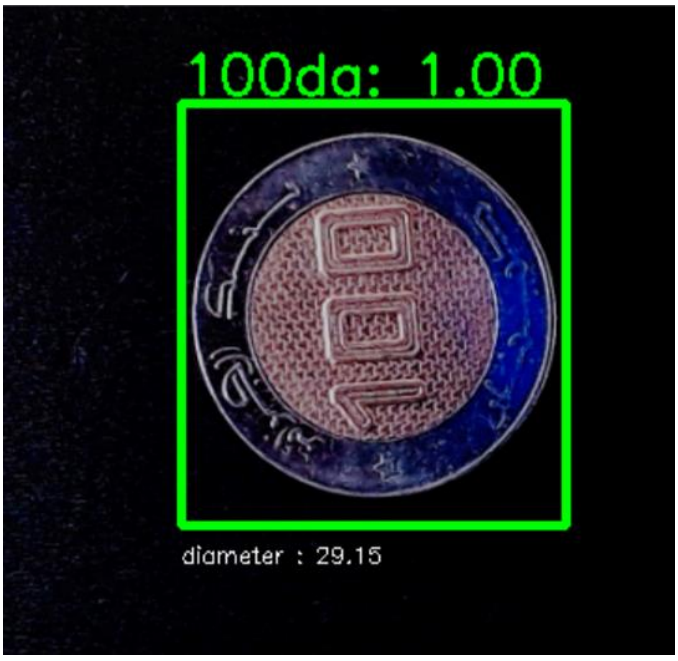


Figure 4.34 : Identification d'une pièce de 100da.

Figure 4.35 : Rejet d'une fausse pièce.

4.6 Conclusion

D'après les résultats obtenus par notre système, on peut conclure que ce dernier a une haute précision d'identification de pièces. Cela veut dire que la combinaison du modèle YOLO V4 et la partie traitement d'image ou calcul du diamètre à l'aide du pied coulisse nous a donné d'excellents résultats d'identification de pièces de monnaies algériennes.

Conclusion générale

Dans ce projet nous avons présenté un système de détection et reconnaissance des pièces monnaies algériennes à l'aide d'apprentissage profond et le traitement d'image. Nous avons pu implémenter ce système dans un programme python à l'aide de la bibliothèque Open CV et la caméra Logitech C920e, nous avons aussi créé une interface graphique pour ce système.

Au cours de ce projet, nous avons appris beaucoup de choses à propos ce projet et nous avons amélioré nos expériences dans ce domaine en appliquant les outils que nous avons appris au cours de cette formation tel que le traitement d'image, la programmation avec le langage python et l'intelligence artificielle.

Dans l'apprentissage profond nous avons utilisé le modèle YOLO V4 pour la détection et la classification des pièces monnaies, la précision de ce dernier et de 98.8% pour 420 pièces tester et 48% pour 50 fausses pièces tester.

Pour minimiser les erreurs lors la détection des pièces monnaies nous avons calculer le diamètre de chaque pièce à l'aide de la partie traitement d'image, on a aussi fait les mesures manuellement à l'aide de pied à coulisse pour confirmer le diamètre de toutes les pièces, la précision de cette partie et de 98.57% pour 420 pièces testées et un rejet de 100% pour les pièces ayant différent diamètre par rapport à nos pièces et 0% pour les pièces ayant même diamètre de nos pièces.

A la fin, la combination des deux parties nous a donné d'excellents résultats ; 99.74% de précision pour 756 pièces testées et un rejet de 83% pour 100 fausse pièces, ça indique que notre système présente une bonne robustesse.

Nous espérons que notre travail se développera à l'avenir, par exemple ajouter une partie mécanique pour ce système qui consiste à créer une machine capable à résoudre le problème de tri et comptage des pièces moyennant un convoyeur à bande.

Bibliographie

[1] **Amr Hussain** ،تعريف العملات النقدي,2021, www.almaerifaa.com , Date de consultation Juin 2022.

[2] **Jan van der Crabben**, Coinage.2011. www.worldhistory.org, Date de consultation Juin 2022.

[3] **ضحى حماده**, العملات القديمة الجزائرية.2022, www.almrsal.com , Date de consultation Juin 2022.

[4] **Numista**, Algerian coins. 2007. https://en.numista.com/catalogue/algerie-1.html . Date de consultation Juin 2022.

[5] **Modi, S.; Bawa**, S. Automated Coin Recognition System using ANN, 2013 https://arxiv.org/abs/1312.66 . Date de consultation Juin 2022.

[6] **Aby Sasi; Sreekumar K.**, International Journal of Computer Applications (0975 – 8887) Volume 131 – No.11, 2015, https://citeseerx.ist.psu.edu/ , Date de consultation Juin 2022.

[7] **Nataliya Didukh, Ihor Zhvanko,** Real-time coins detection with ML based approach on iOS device, 2021, https://autosys.informatik.haw-hamburg.de/, Date de consultation Juin 2022

[8] **Lina Suhaili Rosidi, Nur Anis Jasmin Sufri, Muhammad Amir As'ari**. Reconnaissance des pièces malaisiennes basées sur l'aide de l'apprentissage profond, Vol 12, pages 119-126, 2022.

[9] **Nikolay Fonov, Urkaeva Ksenia**, Development an Accurate Neural Network for Coin Recognition,2021, https://ieeexplore.ieee.org/abstract/document/9396592, Date de consultation Juin 2022

[10] **A.U. Tajane, J. M. Patil,A.S. Shahane,P.A. Dhulekar**, Deep Learning Based Indian Currency Coin Recognition,2018, https://ieeexplore.ieee.org/document/8529467, Date de consultation Juin 2022

[11] **Mr. Namane Abderrahmane,** 2022, Cour Vision Artificiel, Université Saad Dahleb Blida.

[12] **Sagar Kumar**, A straightforward introduction to Image Thresholding using python, 2019, https://medium.com/ , Date de consultation Juin 2022

[13] OpenCV - Adaptive Threshold, https://www.tutorialspoint.com/ , Date de consultation Juin 2022

[14] **Jamileh Yousefi** , Image Binarization using Otsu Thresholding Algorithm, mémoire de Master, University of Guelph, 2011

[15] **VasuDev4,** Apply a Gauss filter to an image with Python, 2020,geeksforgeeks.org/, Date de consultation Juin 2022

[16] Gaussian Filtering, 2010, https://www.cs.auckland.ac.nz/ , Date de consultation Juin 2022

[17] **Great Learning Team**, what is Edge Detection, 2021, mygreatlearning.com/ , Date de consultation Juin 2022

[19] **Abdelhamid Djeffal,** Réseaux de neurones, http://www.abdelhamid-djeffal.net/ , Date de consultation Juin 2022

[20] **BAA Mohammed, Khledj Omar,** Reconnaissance de caractères manuscrits ou de formes par réseau de neurones, mémoire de Master, Systèmes des Télécommunications, Université Djilali Bounaama ,2018

[21] **Habiboullah Beha Dine,Ladjedel Billal ,** Utilisation des réseaux de neurones artificiels pour la prédiction de la vitesse de vent ,mémoire de Master , SYSTEME EMBARQUE , UNIVERSITE MOHAMED BOUDIAF, 2018

[22] **Dr. Merzougui ,**2021, Cour Deep Learning, Université Chahid Mostefa Ben Boulaid Batna 2.

[23] **Djamel Belhaouci,** les Réseaux de Neurones artificiels, https://www.juripredis.com/ , Date de consultation Juin 2022

[24] **M. Avellaneda, F. Stella, A. Carelli**, A Bayesian Approach for Constructing Implied Volatility Surfaces through Neural Networks, 2000, researchgate.net/ , Date de consultation Juin 2022

[25] **Huawei Technologies Co**, AI overview, 2021 , https://www.huawei.com/ , Date de consultation Juin 2022

[26] **Hossam M**, Deep Learning ou Apprentissage Profond : qu'est-ce que c'est ? 2020, https://datascientest.com/ , Date de consultation Juin 2022

[27] **Samaya Madhavan,** Deep learning architectures, 2021, developer.ibm.com/ , Date de consultation Juin 2022

[28] **Gaël** ,3 Deep Learning Architectures explained in Human Language,2018, datakeen.co/ , Date de consultation Juin 2022

[29] **Joyce Xu,** Deep Learning for Object Detection, 2017, towardsdatascience.com/ , Date de consultation Juin 2022

[30] **Mesbah Fethia** , Détection d'objets par Deep Neural Network à l'aide du modèle YOLO en temps réel, mémoire de Master , Informatique , Université 8 Mai 1945, 2021

[31] **Bochkovskiy** , CSPDarknet53 , https://paperswithcode.com/ , Date de consultation Juin 2022

[32] **Andrej Anka,** YOLO v4: Optimal Speed & Accuracy for object detection, towardsdatascience.com/ , Date de consultation Juin 2022

[33] **Vivek Praharsha,** YOLOv4 model architecture, iq.opengenus.org/ , Date de consultation Juin 2022

[34] **Shengying Wang, Jing Zhao,** A real-time deep learning forest fire monitoring algorithm based on an improved Pruned + KD model, 2021, researchgate.net/ Date de consultation Juin 2022

[35] **Shraddha Pattanshetti** , Working of YOLOv4 Algorithm , 2021 , medium.com/ , Date de consultation Juin 2022

[36] **Hooman Samani, Chan-Yun Yang**, Anomaly Detection with Vision-Based Deep Learning for Epidemic Prevention and Control, 2021, researchgate.net/ , Date de consultation Juin 2022

[37] **Ayoosh Kathuria** , How to Implement a YOLO (v3) Object Detector from Scratch in PyTorch: Part 1 , 2018 , kdnuggets.com/ , Date de consultation Juin 2022

[38] **Acervo Lima** , YOLO: Vous ne regardez qu'une seule fois – Détection d'objets en temps réel , fr.acervolima.com/ , Date de consultation Juin 2022

[39] **Meradi Yasmine Nesrine, Khemis Sarah,** Analyse automatique d'images polmonaires pour la détection de pathologies oncologiques et des pneumopathies, mémoire de Master, Systèmes Embarqués, Université Saad Dahleb , 2020

[40] **IoU (Intersection over Union)** , wiki.hasty.ai/ , Date de consultation Juin 2022

[41] **Garima Nishad,** You Only Look Once (YOLO): Implementing YOLO in less than 30 lines of Python Code, 2019, medium.com/ , Date de consultation Juin 2022

[42] **Aishwarya Singh,** Selecting the Right Bounding Box Using Non-Max Suppression (with implementation) , 2020 , www.analyticsvidhya.com/ , Date de consultation Juin 2022

[43] Computer Vision Annotation Tool,2022, https://en.wikipedia.org/wiki/Computer_Vision_Annotation_Tool , Date de consultation Juin 2022

[44] **matthewbrems**, Roboflow,2021, https://github.com/opencv-ai/roboflow, Date de consultation Juin 2022

[45] Colaboratory, https://research.google.com/colaboratory/faq.html, Date de consultation Juin 2022

[46] PyCharm,2022, https://en.wikipedia.org/wiki/PyCharm, Date de consultation Juin 2022

[47] About OpenCv, https://opencv.org/about/, Date de consultation Juin 2022

[48] **Jason Brownlee**,Introduction to the Python Deep Learning Library TensorFlow,2022, https://machinelearningmastery.com/introduction-python-deep-learning-library-tensorflow/, Date de consultation Juin 2022

[49] NumPy Introduction, https://www.w3schools.com/python/numpy/numpy_intro.asp, Date de consultation Juin 2022

[50] tkinter — Interface Python pour Tcl/Tk, https://docs.python.org/fr/3/library/tkinter.html, Date de consultation Juin 2022