République Algérienne Démocratique et Populaire
Université Abou Bekr Belkaid– Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

pour l'obtention du diplôme de Master en Informatique

Option: *Modèle Intelligent et Décision (M.I.D)*

Thème

L'Apprentissage Profond pour le suivi d'objet par régression

Deep learning regression for object tracking

Réalisé par :

- **Madani Hicham**

Présenté le 05/10/2022 devant le jury composé de MM.

- **Mr. BENAZZOUZ Mourtada** *(Président)*
- **Mr. BERRABAH Sid-Ahmed** *(Encadreur)*
- **Mr. SMAHI Ismail** *(Examinatrice)*

Année universitaire: 2021-2022

REMERCIEMENT

Je voudrais d'abord remercier ﷲ le tout puissant qui nous a donné la force, le courage et la patience pour accomplir cette humble œuvre.

Deuxièmement, je tiens à remercier mon encadreur, M. Berrabah Sid Ahmed pour ses précieux conseils et son aide tout au long de l'entreprise et pour tout le soutien et les conseils.

J'exprime mon plus grand respect aux honorables membres du jury, Mr. BENAZZOUZ Mourtada, maître de conférences à l'Université d'Abou Bakr Belkaid à Tlemcen, et Mr. SMAHI Ismail, maître de conférences à l'Université d'Abou Bakr Belkaid à Tlemcen, qui ont accepté l'évaluation de ce travail.

Je remercie également mes parents et mes proches pour l'amour et le soutien constants qu'ils m'ont apportés tout au long de mon parcours. Merci à tous mes amis pour leurs encouragements.

TABLE DES MATIÈRES

REMERCIEMENT 1

TABLE DES MATIÈRES 2

TABLE DES FIGURES 4

INTRODUCTION GENERALE 6

CHAPITRE 1 : SUIVI D'OBJETS MOBILES

1. Introduction 9
2. Suivi d'objet 9
3. Différence entre la détection d'objet et le suivi d'objet 9
4. Certaines des défis du suivi des objects 9
5. Suivi d'objet unique (Single Object Tracking) 10
6. Suivi d'objets multiples (Multiple Object Tracking) 10
7. Classification de la méthode de suivi des objets 11
   - 7.1. Méthodes basées sur les fonctionnalités 11
   - 7.2. Méthodes basées sur la segmentation 12
   - 7.3. Méthodes basées sur l'estimation 13
   - 7.4. Méthodes basées sur l'apprentissage 13
8. Conclusion 15

CHAPITRE 2 : L'APPRENTISSAGE PROFOND

1. Introduction 17
2. L'apprentissage en profondeur 17
3. Extraction de caractéristiques 18
4. Réseau neuronal convolutif 18
5. Blocs de construction de l'architecture CNN 19
   - 5.1. Couche de convolution 20
   - 5.2. Couche de pooling 22
   - 5.3. Couche Fully-Connected (FC) 23
6. Applications de CNN pour la vision par ordinateur 24
   - 6.1. Classification des images 24
   - 6.2. Classification des images avec localisation 24
   - 6.3. Détection d'objet 24

6.3.1. Proposition de région 25
6.3.2. Regression/Classification 26
7. Conclusion 27
CHAPITRE 3 :IMPLÉMENTATION ET RÉSULTATS
1. Introduction 29
2. Outils 29
3. Faster R-CNN 30
3.1. Définition 30
3.2. Jeux de données 31
3.3. Création du fichier de configuration de formation 32
3.4. Préparation des données 33
3.5. Création du modèle 34
3.6. Apprentissage et résultats 34
4. DeepSORT 38
4.1. Définition 38
4.2. Architecture 40
4.3. Implémentation 41
4.4. Résultat 43
5. Conclusion 45
Conclusion générale 47
Référence 49
Résumé 54

TABLE DES FIGURES

Figure 1.1 : Quelques difficultés de suivi d'objets 10

Figure 1.2: Cadre ascendant (Bottom-Up based framework) 12

Figure 2.1: réseau neuronal artificiel simple 17

Figure 2.2 : Un ordinateur voit une image comme un tableau de nombres 19

Figure 2.3 : Réseau neuronal convolutif 19

Figure 2.4 : Exemple de convolution 20

Figure 2.5 : opération de convolution avec zéro-padding 21

Figure 2.6 : Fonctions d'activation 22

Figure 2.7 : Une illustration de couches Fully-Connected 23

Figure 2.8 : exemple de proposition de région 26

Figure 2.9 : Schéma du détecteur d'objets.(Regression/Classification) 26

Figure 3.1 : Faster R-CNN est un réseau unique et unifié pour la détection d'objets 30

Figure 3.2: la structure du répertoire des données 31

Figure 3.3:exemple d'un fichier XML Figure 3 31

Figure 3.4:Graphique de perte d'entraînement après l'entraînement 36

Figure 3.5: Graphique de perte de validation après entraînement. 36

Figure 3.6: Exemple 1 37

Figure 3.7: Exemple 2 37

Figure 3.8: Exemple 3 38

Figure 3.9 : Schéma de DeepSORT Extracteur de caractéristiques basé sur CNN. 39

Figure 3.10: deep_dort package 41

Figure 3.11: object tracking(faster_rcnn+deepsort) résultat 1 44

Figure 3.12: object tracking(faster_rcnn+deepsort) résultat 2 45

INTRODUCTION GENERALE

Le suivi d'objets en mouvements est important et sensible dans le domaine de la vision par ordinateur. Il a de nombreuses utilisations telles que la gestion de trafic, la robotique, la conduite automobile, la sécurité, etc. De nombreuses méthodes de suivi d'objet existent, mais il est devenu difficile de s'appuyer principalement sur elles, en raison de divers défis tels que le chevauchement des objets, la vitesse de déplacement, l'éclairage non homogène et de nombreux problèmes.

Avec l'avènement de l'intelligence artificielle, en particulier des réseaux de neurones convolutifs, certains de ces défis ont pu être surmontés. En utilisant des algorithmes de détection d'objets, la qualité de la classification est améliorée et par suite la qualité du suivi. Les implémentations classiques des réseaux de neurones convolutifs régionaux R-CNN, ont été entraînées pour classer les objets mais elles n'ont pas été entraînés pour détecter les coordonnées des rectangles délimitant ses objets.

Dans ce travail, nous proposons d'utiliser un modèle d'apprentissage appelé Régression qui à l'inverse de la classification qui produit un label, il produit des valeurs continues (valeurs réelles dans un intervalle donné). Ce type de modèle est utilisé généralement dans la prédiction des prix ou l'évolution des marchés.

Pour obtenir une régression des rectangle délimitant les objets, nous proposons de modifier l'architectures du réseau R-CNN en plaçant à la tête du réseau une couche fully-connected avec 4 neurones correspondant aux coordonnées (x,y) des coins supérieur gauche et inférieur droit. Le modèle est ensuite entraîné en utilisant une fonction de perte comme l'erreur quadratique.

Le premier chapitre est consacré au problème de suivi d'objets et les principales méthodes existantes.

Le second chapitre est dédié à la présentation de l'apprentissage en profondeur (Deep Learning) et l'un de ses modèles : Le Réseau neuronal convolutif (CNN), en précisant ces différents blocs de construction et ses domaines d'application.

Dans le troisième chapitre nous utiliserons le modèle faster_rcnn Qui est considéré comme meilleur détecteur d'objets Cependant, Grâce à sa capacité à détecter de petits objets dans les images. En termes de suivi, nous utiliserons le deepsort ,vu que c'est le plus populaire et le plus utilisé.

La conclusion et les perspectives de ce travail seront présentées à la fin du mémoire.

CHAPITRE 1 : SUIVI D'OBJETS MOBILES

1. Introduction

L'objectif principal du suivi d'objets est de connaître l'état de la cible (objet) dans une séquence d'images (trames). Cet état peut être connu par différentes caractéristiques telles que la forme (structure), l'apparence, l'emplacement ou la vitesse. C'est un domaine d'étude compliqué car de nombreuses difficultés doivent être résolues par différents algorithmes. Parmi elles, la gestion des variations d'éclairage et de la perspective de l'objet (rotation de la caméra) qui peuvent entraîner des changements dans l'apparence d'un objet. De même, les occlusions qui se produisent lorsque des objets sont mélangés avec d'autres objets ou composants de la scène ou la qualité de l'image elle-même peuvent être un problème.

2. Suivi d'objet

Le suivi d'objet consiste à suivre un objet en mouvement dans une vidéo. L'idée du suivi d'objet est d'associer ou d'établir une relation entre des objets cibles tels qu'ils apparaissent dans chaque image vidéo. En d'autres termes, le suivi d'objet analyse les images vidéo de manière séquentielle et assemble l'emplacement passé de l'objet avec l'emplacement actuel en prédisant et en créant une rectangle qui délimite l'objet.

3. Différence entre la détection d'objet et le suivi d'objet

Le suivi d'objet fait référence à la capacité d'estimer ou de prédire la position d'un objet cible dans chaque image consécutive d'une vidéo une fois que la position initiale de l'objet cible est définie. D'autre part, la détection d'objet est le processus de détection d'un objet cible dans une image ou une seule image de la vidéo

4. Certaines des défis du suivi des objects [2]

- Variation d'éclairage : l'éclairage dans la région cible peut considérablement changer.
- Encombrements d'arrière-plan (camouflage) : l'arrière-plan près de la cible a une couleur ou une texture similaire à celle de la cible.

- Variation d'échelle :La taille de l'objet à suivre dans l'image peut changer d'une image à une autre (effet zoom).
- Occlusion : la cible est partiellement ou totalement occluse (cachée) par un autre objet de la scène.
- Déformation de la cible.
- Mouvement rapide : La scène peut contenir des grands mouvements, ce qui revient à des grands changements entre les trames consécutives.

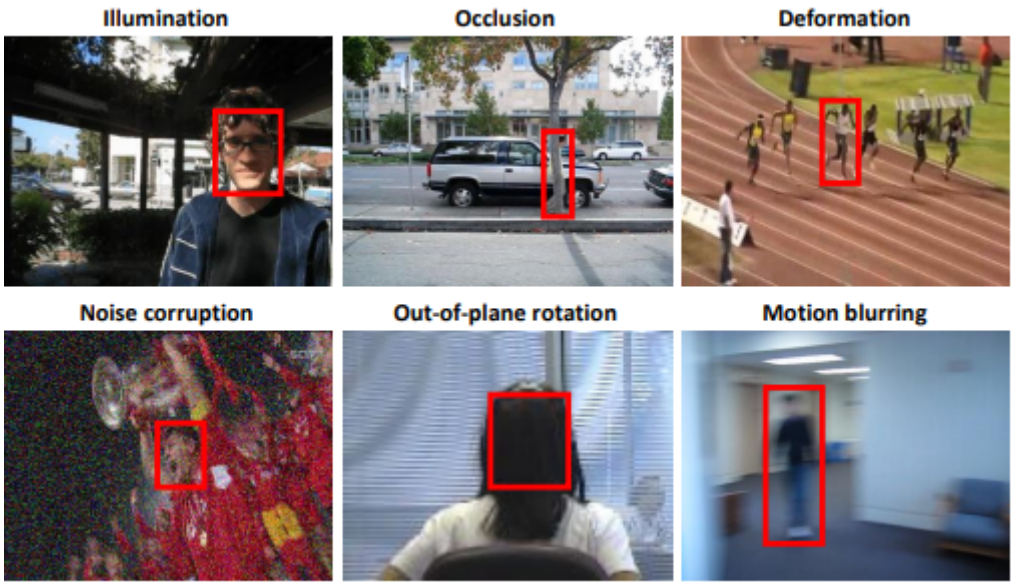


Figure 1.1 : Quelques difficultés de suivi d'objets.[1]

5. Suivi d'objet unique (Single Object Tracking)

Dans le suivi d'un seul objet, le rectangle délimitant la cible dans la première image est donnée au système de suivi (tracker). Le but du tracker est alors de localiser la même cible dans toutes les autres trames. Nous disons que les trackers d'objet unique appartiennent à la catégorie des trackers sans détection. Ils devraient être capables de suivre n'importe quel objet, sans aucune formation sur l'objet .[42]

6. Suivi d'objets multiples (Multiple Object Tracking)

Ces types de trackers peuvent suivre plusieurs objets présents dans une image. Plusieurs trackers d'objets ou MOT sont entraînés sur une grande quantité de données, contrairement aux trackers traditionnels. Par conséquent,

ils se sont avérés plus précis car ils peuvent suivre plusieurs objets et même de différentes classes en même temps tout en maintenant une vitesse élevée. Certains des algorithmes incluent DeepSORT, JDE et CenterTrack qui sont des algorithmes très puissants et gèrent la plupart des défis auxquels sont confrontés les trackers.[50]

7. Classification des méthodes de suivi des objets

7.1. Méthodes basées sur les fonctionnalités

Ces méthodes de suivi d'objets sont basées sur des caractéristiques simples telles que la couleur, la texture, le flux optique, etc. Ces caractéristiques extraites doivent être uniques afin que les objets puissent être facilement distingués dans l'espace des caractéristiques.

L'un des problèmes de ces méthodes réside dans l'étape d'extraction car les caractéristiques uniques, précises et fiables de l'objet doivent être extraites afin qu'il puisse distinguer la cible des autres objets. Voici quelques-unes des fonctionnalités utilisées pour suivre les objets.

7.1.1 Couleur

La fonction de couleur peut montrer l'apparence d'un objet. Cette fonction peut être utilisée de différentes manières. L'une des façons les plus courantes d'utiliser cette fonctionnalité consiste à utiliser des histogrammes de couleurs.

L'histogramme des couleurs montre la répartition des couleurs dans l'image. En réalité, les différentes couleurs de l'image et le nombre de pixels dans chaque couleur sont affichés. L'inconvénient des histogrammes de couleur est que deux objets différents peuvent avoir le même histogramme car la représentation dépend uniquement de la couleur de l'objet et ignore la forme et la texture de l'objet.

7.1.2 Texture

Une texture est un motif répétitif d'informations ou un arrangement de structures à intervalles réguliers. Les fonctions de texture ne sont pas

enregistrées directement. Ces caractéristiques sont générées par des techniques de prétraitement d'images.

La fonction de texture est une caractéristique importante d'une image et peut être utilisée conjointement avec la fonction de couleur pour décrire le contenu ou la zone d'une image pour aider à identifier les objets de couleurs similaires.

7.1.3 Flux optique

Le flux optique est le modèle de mouvement apparent d'un objet image entre deux images consécutives provoqué par le mouvement de l'objet ou de la caméra. Il s'agit d'un champ vectoriel 2D, où chaque vecteur est un vecteur de déplacement qui montre le mouvement du pixel d'une trame à la trame suivante.[41]

Le flux optique peut également être défini comme la distribution des vitesses apparentes de mouvement du motif de luminosité dans une image.[3]

7.2 Méthodes basées sur la segmentation

Ce type de suivi effectue deux tâches distinctes, en segmentant d'abord le premier plan, puis en suivant l'objet. La segmentation de premier plan utilise une segmentation de bas niveau pour extraire des zones de toutes les images. Certaines caractéristiques sont ensuite extraites de la zone de premier plan et suivies sur la base de ces caractéristiques comme le montre la Figure 1.2 .[5]

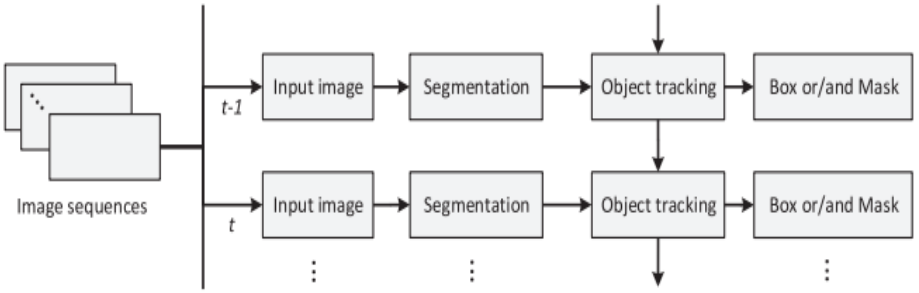


Figure 1.2: Cadre ascendant (Bottom-Up based framework)[5]

7.3 Méthodes basées sur l'estimation

Les méthodes d'estimation formulent le problème de suivi en un problème d'estimation dans lequel un objet est représenté par un vecteur d'état. Le vecteur d'état décrit le comportement dynamique d'un système, tel que la position et la vitesse d'un objet. Le cadre général du problème d'estimation de mode dynamique est tiré des méthodes bayésiennes.[4]

Les filtres bayésiens permettent à la cible de mettre à jour en permanence sa position sur le système de coordonnées, sur la base des dernières données du capteur. Cet algorithme est récursif et se compose de deux étapes : prédiction et mise à jour.[4]

L'étape prédictive utilise le modèle d'état pour estimer la nouvelle position de la cible à l'étape suivante, et l'étape de mise à jour utilise les observations actuelles pour mettre à jour la position cible à l'aide du modèle d'observation. Les étapes de prédiction et de mise à jour sont effectuées à chaque image de la vidéo.
Parmi ces méthode, nous pouvons citer les méthodes utilisant le filtre de Kalman [6] ou le filtre de Kalman étendu [7] et les méthodes utilisant les filtres à particules. [8]

7.4 Méthodes basées sur l'apprentissage

Les méthodes basées sur l'apprentissage utilisent une phase d'apprentissage pour enregistrer les caractéristiques et l'apparence de différentes cibles, et à la phase d'utilisation elles utilisent ces caractéristiques pour détecter les cibles dans les image suivantes. Les méthodes basées sur l'apprentissage sont souvent divisées en trois types : l'apprentissage génératif, l'apprentissage discriminatif et l'apprentissage par renforcement.

7.4.1 Méthodes discriminatoires

Les traqueurs d'identification considèrent généralement le pistage comme un problème de classification qui fait la distinction entre une cible et un arrière-plan. L'apprentissage discriminatoire peut être divisé en deux catégories, à savoir l'apprentissage superficiel et l'apprentissage en profondeur.

- **Apprentissage superficiel**

  Le suivi d'objets peut être considéré comme un processus de prise de décision dans lequel la classification est formée pour apprendre la différence entre une cible et un arrière-plan. Après la formation, lors des tests, déterminez si l'objet est la cible. Les caractéristiques sont extraites de différents objets et peuvent être catégorisées à l'aide de différentes méthodes telles que les machines à vecteurs de support .[9]-[12].

- **L'apprentissage en profondeur**

  Un apprentissage superficiel avec moins de couches prédit le modèle, mais l'apprentissage en profondeur a trop de couches. Une autre différence est que l'apprentissage en profondeur lui-même extrait ces caractéristiques importantes, tandis que l'apprentissage superficiel nécessite que l'expert extraie des caractéristiques importantes et discriminantes.

7.4.2 Generative Methods

Ces méthodes se concentrent sur la recherche dans des zones qui ressemblent davantage à des objets. Une méthode basée sur des filtres de corrélation est l'un de ces exemples de suivi.

L'idée principale du filtre de corrélation est d'estimer le meilleur filtre d'image afin que ce filtre produise la meilleure sortie sur l'image d'entrée.[13]

7.4.3 Apprentissage par renforcement

Dans un problème d'apprentissage par renforcement, nous rencontrons un agent qui interagit avec l'environnement par essais et erreurs et apprend à sélectionner l'action optimale pour atteindre l'objectif. [9]

8. Conclusion

Le suivi d'objets est un domaine important de la vision par ordinateur car il contient de nombreux défis, bien qu'il existe des moyens de résoudre ces problèmes, mais il n'y a aucun moyen de les résoudre tous.

Dans le chapitre suivant, nous parlerons de l'apprentissage en profondeur, qui est une partie importante de ce domaine

CHAPITRE 2 :

L'APPRENTISSAGE PROFOND

1. Introduction

L’apprentissage profond a fait des progrès remarquables ces dernières années dans divers domaines, dont la vision par ordinateur. Un domaine où l'apprentissage en profondeur a eu un impact et amélioré sa précision est le suivi d'objets. En raison de cette amélioration de la précision, nous nous concentrons sur les techniques d'apprentissage en profondeur. Les méthodes d'apprentissage en profondeur sont divisées en méthodes basées sur l'extraction de caractéristiques et en méthodes de bout en bout.

2. L'apprentissage en profondeur

deep learning ou apprentissage en profondeur est l'une des principales technologies d'apprentissage automatique. Avec le Deep Learning, on parle d'algorithmes capables de mimer les actions du cerveau humain à travers des réseaux de neurones artificiels. Le réseau est constitué de dizaines, voire de centaines de « couches » de neurones, dont chacune reçoit et interprète les informations de la couche précédente.

Chaque neurone artificiel est représenté sur la figure 2.1 par un cercle, qui peut être vu comme un modèle linéaire. En connectant des neurones dans une couche, nous transformons notre réseau de neurones en un modèle non linéaire très complexe.

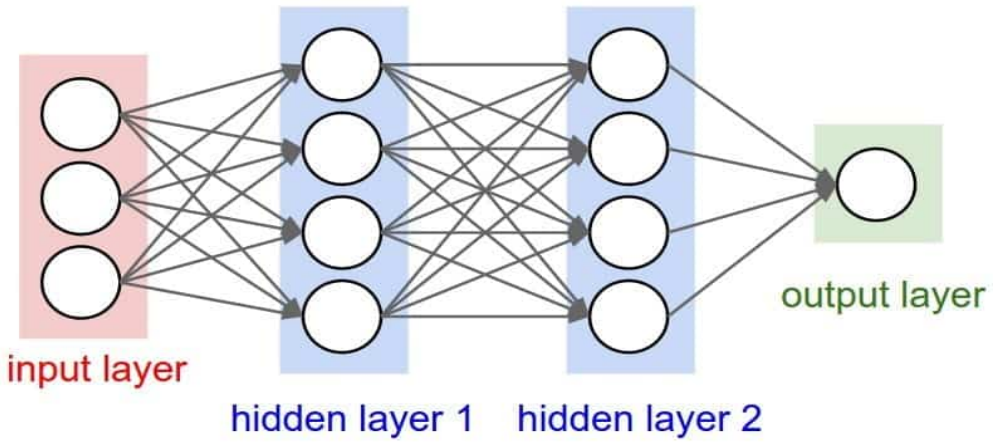


Figure 2.1: réseau neuronal artificiel simple

3. Extraction de caractéristiques

L'extraction de caractéristiques est un processus de réduction de la dimensionnalité par lequel un ensemble initial de données brutes est réduit à des groupes plus gérables pour le traitement. Une caractéristique de ces grands ensembles de données est un grand nombre de variables qui nécessitent beaucoup de ressources informatiques pour être traitées. L'extraction de caractéristiques est le nom des méthodes qui sélectionnent et/ou combinent des variables dans des caractéristiques, réduisant efficacement la quantité de données qui doivent être traitées, tout en décrivant de manière précise et complète l'ensemble de données d'origine.

4. Réseau neuronal convolutif (CNN)

CNN est un type de modèle d'apprentissage en profondeur pour le traitement de données qui ont un motif de grille, telles que des images, qui s'inspire de l'organisation du cortex visuel animal et conçu pour apprendre automatiquement et de manière adaptative les hiérarchies spatiales des caractéristiques, de bas à haut niveau. motifs. CNN est une construction mathématique généralement composée de trois types de couches (ou blocs de construction) : convolution, mise en commun et couches entièrement connectées. Les deux premières, les couches de convolution et de regroupement, effectuent l'extraction des caractéristiques, tandis que la troisième, une couche entièrement connectée, mappe les caractéristiques extraites dans la sortie finale, telle que la classification. Une couche de convolution joue un rôle clé dans CNN, qui est composé d'une pile d'opérations mathématiques, telles que la convolution, un type spécialisé d'opération linéaire. Dans les images numériques, les valeurs des pixels sont stockées dans une grille bidimensionnelle (2D), c'est-à-dire un tableau de nombres (Fig. 2.2), et une petite grille de paramètres appelée noyau, un extracteur de caractéristiques optimisables, est appliquée à chaque position de l'image. , ce qui rend les CNN très efficaces pour le traitement d'image, car une fonctionnalité peut apparaître n'importe où dans l'image. Au fur et à mesure qu'une couche transmet sa sortie à la couche suivante, les entités extraites peuvent devenir hiérarchiquement et

progressivement plus complexes. Le processus d'optimisation des paramètres tels que les noyaux est appelé formation, qui est effectuée de manière à minimiser la différence entre les sorties et les étiquettes de vérité terrain grâce à un algorithme d'optimisation appelé rétropropagation et descente de gradient, entre autres. [14]

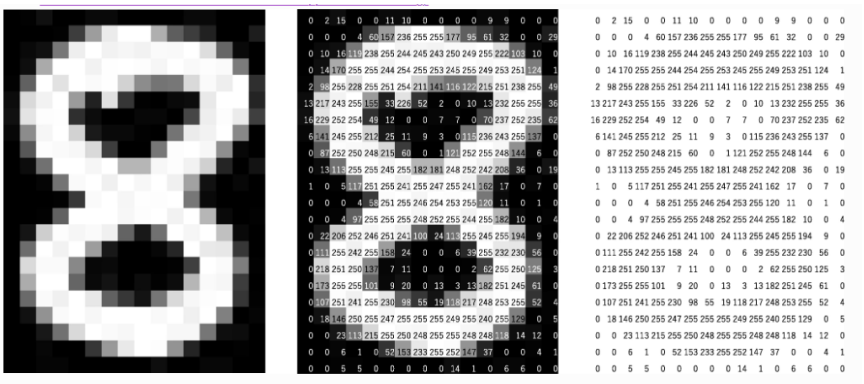


Figure 2.2 : Un ordinateur voit une image comme un tableau de nombres.[14]

5. Blocs de construction de l'architecture CNN

Une architecture CNN contient plusieurs blocs de construction tels que des couches convolutives, des couches de pooling et des couches fully-connected. Une architecture typique consiste en des itérations d'empilements de plusieurs couches de convolution et de mise en commun, suivies d'une ou plusieurs couches entièrement connectées. Les étapes au cours desquelles ces couches transforment les données d'entrée en sorties sont appelées propagation vers l'avant (Figure 2.3).

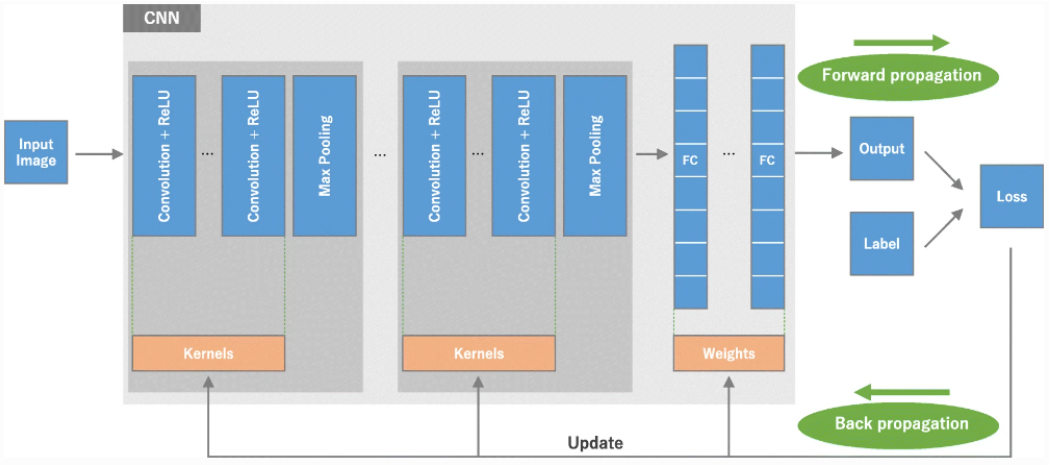


Figure 2.3 : Réseau neuronal convolutif .[14]

4.1 Couche de convolution

Une couche convolutive est le composant de base d'une architecture CNN qui effectue l'extraction de caractéristiques et consiste généralement en une combinaison d'opérations linéaires et non linéaires. c'est-à-dire une opération de convolution et une fonction d'activation.

- **Convolution** Une couche convolutive calcule une opération de convolution sur l'image d'entrée à l'aide de filtres de noyau pour extraire les caractéristiques de base. Les filtres du noyau sont de la même dimension mais avec des paramètres constants plus petits par rapport aux images d'entrée. Par exemple, pour calculer une image de scalogramme 2D 35 × 35 × 2, la taille de filtre autorisée est f × f × 2. où f=3, 5, 7, etc. Cependant, la taille du filtre doit être inférieure à la taille de l'image d'entrée. Le masque de filtre est glissé par étapes sur l'image d'entrée et le produit scalaire entre les poids du filtre du noyau et les valeurs de l'image d'entrée est estimé pour générer une carte d'activation 2D. Par conséquent, CNN apprendra une fonctionnalité visuelle. La figure 2.4 illustre un exemple simple de carte d'activation de calcul. L'équation de la couche convolutive généralisée est représentée par .[15]

$$Activation\ map = Input * Filter$$

$$= \sum_{y=0}^{columns} \left(\sum_{r=0}^{rows} Input(x - p, y - q) Filter(x, y) \right)$$

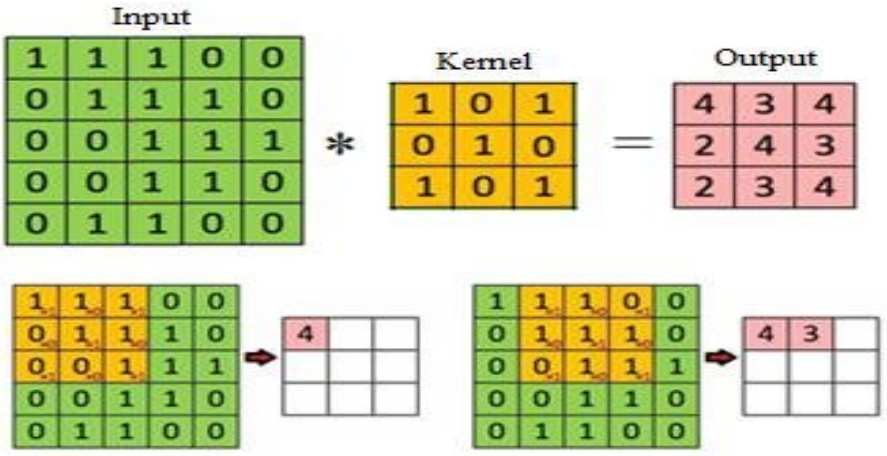


Figure 2.4 : Exemple de convolution [16]

- **Valeur de foulée(Stride Value) :** La Step Value indique le nombre de cases que le filtre décale à chaque pas. Le noyau/filtre se déplace du haut à droite vers le haut à gauche jusqu'à ce que la largeur soit complète avec la valeur de foulée spécifiée. Ensuite, il saute d'une case au début (à gauche) de la matrice 2D et va à droite avec la même valeur de pas. Ce processus se poursuit jusqu'à ce que toute la matrice 2D ait été parcourue.

- **Rembourrage zéro(Zero-padding) :** Le zéro-padding est un simple acte de rembourrage des limites de l'entrée et constitue un moyen efficace de contrôler davantage la dimensionnalité du volume de sortie.Figure 2.5

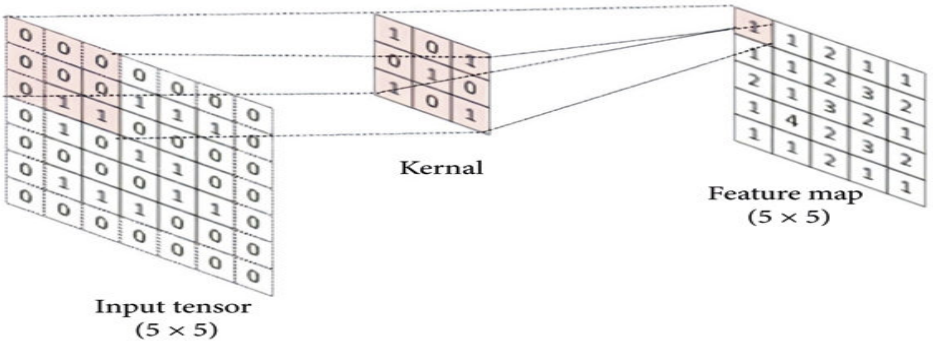


Figure 2.5 : opération de convolution avec zéro-padding [17]

- **Profondeur(Depth) :** La profondeur du volume de sortie produit par une couche convolutive peut être ajustée manuellement à la même plage d'entrées par le nombre de neurones dans la couche. Cela peut être vu dans d'autres formes d'ANN où tous les neurones de la couche cachée sont directement connectés aux neurones individuels au préalable. La réduction de cet hyperparamètre peut réduire considérablement Bien que le nombre total de neurones dans le réseau soit réduit, cela peut également réduire considérablement la capacité du modèle à reconnaître des modèles.[18]

- **Fonction d'activation non linéaire :** Les sorties d'une opération linéaire telle que la convolution sont ensuite passées à travers une fonction d'activation non linéaire. Bien que les fonctions non linéaires lisses, telles que la fonction sigmoïde ou tangente hyperbolique (tanh), aient été utilisées auparavant parce qu'elles sont des représentations mathématiques du comportement d'un neurone biologique, la fonction d'activation non linéaire la plus couramment utilisée actuellement est l'unité linéaire rectifiée (ReLU), qui calcule simplement la fonction :
  f(x) = max(0, x) Figure 2.6 .[43]

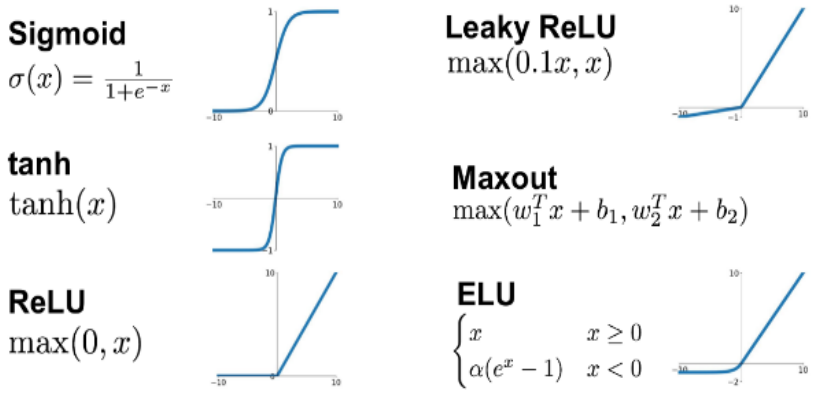


Figure 2.6 ; Fonctions d'activation.[44]

4.2 Couche de pooling

Une couche de regroupement sert à réduire la taille spatiale de la fonction convolutive. Cela permet de réduire la puissance de calcul requise pour traiter les données grâce à la réduction de la dimensionnalité. De plus, il est utile d'extraire les caractéristiques clés qui sont invariantes en rotation et en position, ce qui permet de maintenir l'efficacité du processus de formation des modèles.[19] Il existe deux grands types de pooling.

- **Max pooling :** est l'une des couches de mise en commun les plus couramment utilisées. Il est utilisé pour empêcher le surajustement en fournissant une version à complexité réduite de la représentation originale.

  Max pooling utilise la valeur la plus élevée de tous les neurones de la couche précédente.[45]

- .**Average pooling :** consiste à trouver la valeur moyenne de chaque bloc 2 × 2 de la carte des caractéristiques. Cela signifie que chaque bloc 2 × 2 est sous-échantillonné à la moyenne.[45]

4.3 Couche Fully-Connected (FC)

Les couches entièrement connectées sont généralement utilisées pour aplatir notre matrice en vecteurs afin de l'alimenter en un réseau entièrement connecté de type réseau de neurones. Comme le montre la figure 2.7, la matrice de la carte des caractéristiques est transformée en vecteur (x1, x2 …). En utilisant des couches entièrement connectées en combinant des fonctionnalités ensemble, nous pouvons créer un modèle et à l'aide de fonctions d'activation telles que Relu ou sigmoïde, nous pouvons le faire correspondre à une classe de sortie donnée pour effectuer une classification ou générer le problème pour le problème de type régression.[26]

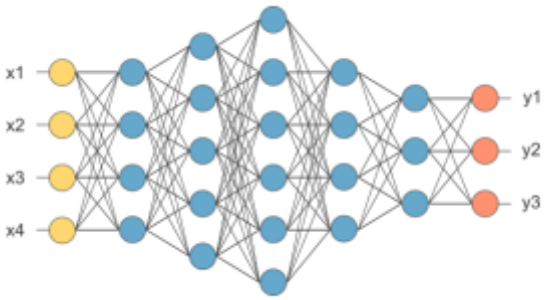


Figure 2.7 : Une illustration de couches Fully-Connected.[26]

Il existe plusieurs architectures dans le domaine des réseaux convolutifs. Les plus courants sont :

- ResNet [20]
- VGGNet (Visual Geometry Group) [21]
- GoogLeNet [22]
- ZFNet [23]
- AlexNet [24]
- LeNet-5 . [25]

6. Applications de CNN pour la vision par ordinateur

CNN est utilisé pour les tâches de vision par ordinateur, qui résolvent les problèmes de traitement d'image et de ML, tels que l'identification d'objets, la reconnaissance d'images, la classification d'images, la segmentation d'images, etc.

5.1 Classification des images

La classification des images attribue des étiquettes à des objets dans l'image ou aux images. Ce problème est également appelé "classification d'objets" et peut-être plus communément appelé "reconnaissance d'images"**.**

5.2 Classification des images avec localisation

La classification d'image par localisation attribue une étiquette de classe à l'image et indique la position de l'objet dans l'image au moyen d'un rectangle encadrant l'objet (un rectangle dessinée autour de l'objet). Il s'agit d'une version plus avancée de la classification d'images.

Nous pouvons citer l'exemple de la détection des zones cancéreuses dans une image de radiographie ou la détection des personnes dans une images.

5.3 Détection d'objet

La détection d'objets génériques vise à localiser et à classer des objets existants dans n'importe quelle image, et à les étiqueter avec des cadres de délimitation rectangulaire pour montrer les confiances d'existence. Les cadres des méthodes génériques de détection d'objets peuvent principalement être classés en deux types. On suit le pipeline traditionnel de détection d'objets, en générant d'abord des propositions de région, puis en classant chaque proposition dans différentes catégories d'objets. L'autre considère la détection d'objets comme un problème de régression ou de classification, adoptant un cadre unifié pour obtenir directement les résultats finaux (catégories et emplacements). Les méthodes basées sur la proposition de région incluent principalement R-CNN [31], Fast R-CNN [32], Faster R-CNN [34] et Mask R-CNN [35]. Les méthodes basées sur la régression/classification incluent principalement YOLO [33], SSD [36], YOLOv2 [37]. Les corrélations entre ces deux pipelines sont pointées par les ancres introduites dans Faster RCNN.

5.3.1 Proposition de région

Des algorithmes de proposition de région identifient des objets potentiels dans une image à l'aide de la segmentation. Dans la segmentation, nous regroupons les régions adjacentes qui sont similaires les unes aux autres en fonction de certains critères tels que la couleur, la texture, etc. Contrairement à l'approche de la fenêtre glissante où nous recherchons l'objet à tous les emplacements de pixels et à toutes les échelles, l'algorithme de proposition de région fonctionne par grouper les pixels en un plus petit nombre de segments. Ainsi, le nombre final de propositions générées est plusieurs fois inférieur à l'approche de la fenêtre glissante. Cela réduit le nombre de patchs d'image que nous devons classer. Ces propositions de régions générées sont d'échelles et de rapports d'aspect différents.[46]

Une propriété importante d'une méthode de proposition de région est d'avoir un rappel très élevé. C'est juste une façon élégante de dire que les régions qui contiennent les objets que nous recherchons doivent figurer dans notre liste de propositions de région. Pour ce faire, notre liste de propositions de régions peut finir par avoir un grand nombre de régions qui ne contiennent aucun objet. En d'autres termes, il est normal que l'algorithme de proposition de région produise beaucoup de faux positifs tant qu'il détecte tous les vrais positifs. La plupart de ces faux positifs seront rejetés par l'algorithme de reconnaissance d'objet. Le temps nécessaire pour effectuer la détection augmente lorsque nous avons plus de faux positifs et que la précision est légèrement affectée. Mais avoir un rappel élevé est toujours une bonne idée car l'alternative de manquer les régions contenant les objets réels a un impact important sur le taux de détection [46]. regarder la Figure 2.8

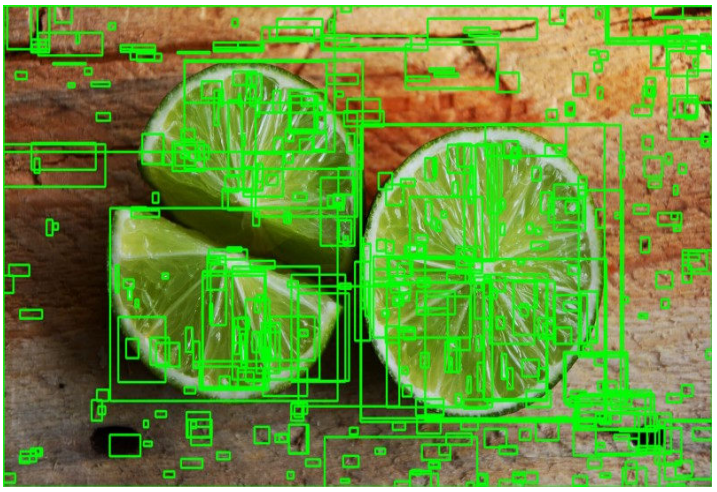


Figure 2.8 : exemple de proposition de région[48]

Plusieurs méthodes de proposition de région ont été proposées telles que

- Objectness [47]
- Constrained Parametric Min-Cuts for Automatic Object Segmentation[27]
- Category Independent Object Proposals [28]
- Randomized Prim [29]
- Selective Search [30]

5.3.2 Regression/Classification

Les principaux modèles sont divisés en deux sous-groupes : régression et classificateur. Le premier produit les coordonnées de début et de fin de la boîte englobante, le second produit l'étiquette de l'objet.Remarquez l'architecture de la figure 2.9.

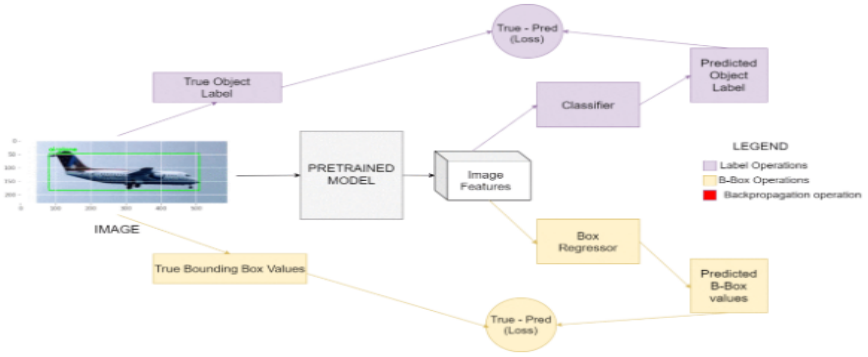


Figure 2.9 : Schéma du détecteur d'objets.(Regression/Classification)[49]

7. Conclusion

L'apprentissage en profondeur excelle dans sa capacité à s'auto-apprendre en déduisant des liens significatifs à partir d'un grand nombre de données et en comprenant des données non structurées. Il contient de nombreux modèles et nous en avons expliqué un.

Bien que le concept d'apprentissage en profondeur existe depuis plusieurs années, il n'a pas eu la capacité d'émerger fortement en raison de plusieurs facteurs, notamment la faiblesse des appareils et le manque de données de haute qualité. Nous avons parlé dans ce chapitre de l'apprentissage en profondeur et de son utilité et diverses applications qui aident à la vision par ordinateur.

Dans le chapitre suivant, nous aborderons l'un de ces algorithmes de détection d'objets.

CHAPITRE 3 :IMPLÉMENTATION ET RÉSULTATS

1. Introduction

Dans ce chapitre, nous allons construire un programme qui suit les objets, Nous commençons par présenter les outils utilisés. Ensuite nous présenterons le classificateur « Faster R-CNN » et dans la troisième section nous introduirons le « deepsort » qui effectue les tâches de suivi et dans la dernière section, nous aurons le suivi d'objet en utilisant l'apprentissage en profondeur.

2. Outils

- **Jupyter Notebook :** Il s'agit d'une application Web client/serveur avec un environnement informatique interactif qui peut combiner l'exécution de code, du texte enrichi, des mathématiques, des graphiques et d'autres contenus multimédias. Cette application, composée principalement d'un noyau et d'un tableau de bord, est l'environnement dans lequel s'effectue le développement.
- **PyTorch :**PyTorch est une bibliothèque d'IA développée par Meta et écrite en Python pour aborder l'apprentissage en profondeur (ou apprentissage en profondeur) et le développement de réseaux de neurones artificiels. A partir de plusieurs variables, vous pouvez effectuer des calculs de gradient ou utiliser des tableaux multidimensionnels obtenus à l'aide de tenseurs.
- **OpenCV :**(Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.
- **Imutils :** est un package basé sur OpenCV, qui peut appeler l'interface opencv plus simplement. Il peut facilement réaliser une série d'opérations telles que la translation d'image, la rotation, la mise à l'échelle, la squelettisation, etc.
- **NumPy :** est le package fondamental pour le calcul scientifique en Python. Il s'agit d'une bibliothèque Python qui fournit un objet tableau multidimensionnel, divers objets dérivés (tels que des tableaux masqués

et des matrices) et un assortiment de routines pour des opérations rapides sur des tableaux, y compris mathématiques, logiques, manipulation de forme, tri, sélection, E/S , transformées de Fourier discrètes, algèbre linéaire de base, opérations statistiques de base, simulation aléatoire et bien plus encore.

3. Faster R-CNN

3.1. Définition

le système de détection d'objets, appelé Faster R-CNN, est composé de deux modules. Le premier module est un réseau profond entièrement convolutif qui propose des régions, et le second module est le détecteur Fast R-CNN qui utilise les régions proposées. L'ensemble du système est un réseau unique et unifié pour la détection d'objets (Figure 3.1).

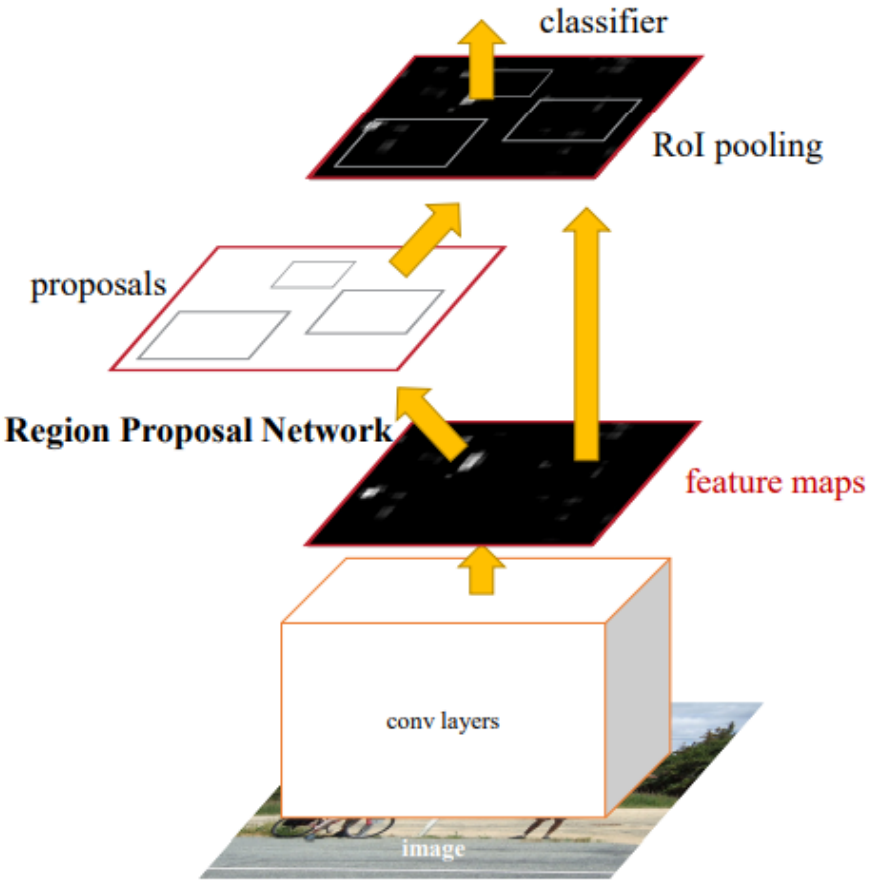


Figure 3.1 : Faster R-CNN est un réseau unique et unifié pour la détection d'objets[38]

3.2. Jeux de données

La base de données utilisée est Détection de véhicules pris à partir du site Roboflow, appartenant à 8 classes différentes. Elles sont : Auto, Vélo, Autobus, BusBike, Voiture, Cycle, Camion, Van,
où il consiste en 2346 images .
Voici la structure du répertoire des données

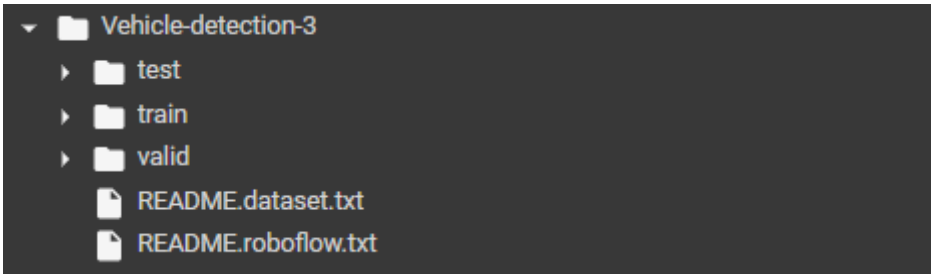


Figure 3.2: la structure du répertoire des données

nous utilisons la base de données a forme pascal voc, qui divise en 3 fichiers (train, valid , test), et pour chaque image il y a un fichier xml de même nom de l'image qui donne :

La hauteur et la largeur de l'image.

Le nom de la classe.

Les coordonnées des rectangles délimitant les objets.

Voici un exemple d'un tel fichier.

```
<?xml version="1.0"?>
- <annotation>
    <folder/>
    <filename>1_jpg.rf.90f9046ee515332233353dd2e8b70f87.jpg</filename>
    <path>1_jpg.rf.90f9046ee515332233353dd2e8b70f87.jpg</path>
  - <source>
      <database>roboflow.ai</database>
    </source>
  - <size>
      <width>640</width>
      <height>480</height>
      <depth>3</depth>
    </size>
    <segmented>0</segmented>
  - <object>
      <name>Bike</name>
      <pose>Unspecified</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
      <occluded>0</occluded>
    - <bndbox>
        <xmin>109</xmin>
        <xmax>196</xmax>
        <ymin>268</ymin>
        <ymax>480</ymax>
      </bndbox>
    </object>
</annotation>
```

Figure 3.3:exemple d'un fichier XML

3.3. Création du fichier de configuration de formation

Nous faisons toujours le fichier de configuration de formation séparé, cela garantit que nous n'avons pas besoin de manipuler d'autres parties du pipeline lorsque nous voulons modifier, par exemple, la taille de l'image d'entrée ou le nombre d'époques.

Le code suivant ira dans le fichier config.py.

```
%%writefile config.py
import torch

BATCH_SIZE = 8 # increase / decrease according to GPU memory
RESIZE_TO = 416 # resize the image for training and transforms
NUM_EPOCHS = 10 # number of epochs to train for
NUM_WORKERS = 4

DEVICE = torch.device('cuda') if torch.cuda.is_available() else
torch.device('cpu')

# training images and XML files directory
TRAIN_DIR = 'Vehicle-detection-3/train'
# validation images and XML files directory
VALID_DIR = 'Vehicle-detection-3/valid'

# classes: 0 index is reserved for background
CLASSES = [
    '__background__','Auto', 'Bike', 'Bus', 'BusBike', 'Car', 'Cycle',
'Truck', 'Van']
NUM_CLASSES = len(CLASSES)

# whether to visualize images after creating the data loaders
VISUALIZE_TRANSFORMED_IMAGES = False

# location to save model and plots
OUT_DIR = 'outputs'
```

Le fichier de configuration de la formation contient les informations suivantes:

La taille du lot à utiliser pour la formation.

Les dimensions auxquelles nous voulons que les images soient redimensionnées, c'est-à-dire RESIZE_TO.

Nombre d'époques pour s'entraîner.

Nombre de nœuds de calcul ou de sous-processus à utiliser pour le chargement des données. Cela aide beaucoup lorsque nous avons un grand ensemble de données.

Le dispositif de calcul à utiliser pour la formation. Pour la formation, nous aurons besoin d'un GPU. Un processeur est tout simplement trop

lent pour une formation RCNN plus rapide et une formation à la détection d'objets en général également.

Le TRAIN_DIR est une chaîne contenant le chemin d'accès aux images de formation et aux fichiers XML. Similaire pour VALID_DIR pour les images de validation et les fichiers XML.,ensuite, nous avons les classes et le nombre de classes.
OUT_DIR contient le chemin d'accès au répertoire pour stocker les modèles formés et les graphiques de perte.

C'est tout ce que nous avons pour le fichier de configuration de la formation.

3.4. Préparation des données

La préparation correcte de l'ensemble de données est vraiment importante pour la formation à la détection d'objets. Toute petite erreur lors du chargement des coordonnées de la boîte englobante peut complètement entraîner l'échec de l'entraînement.

Après avoir obtenu le chemin d'image actuel, nous lisons l'image et convertissons le format de couleur de BGR en RVB, le redimensionnons à la hauteur et à la largeur souhaitées et mesurons les pixels de l'image.

Nous lisons le fichier XML correspondant à notre image, puis analysons et extrayons le nom de la classe et les coordonnées de la boîte englobante, puis redimensionner ces boîtes englobantes à la hauteur et à la largeur souhaitées. Ensuite, nous ajoutons les boîtes englobantes redimensionnées à la liste des boîtes. Nous ajoutons le nom de la catégorie aux étiquettes
Nous avons un dictionnaire cible où nous stockons toutes les informations collectées
remarque : peut l'utiliser pour toute préparation d'ensemble de données de détection d'objets au format Pascal VOC.

3.5. Création du modèle

Ici, nous créons un modèle FASTER RCNN. PyTorch fournit déjà des modèles pré-formés, c'est donc très simple. Changez simplement la tête du réseau en fonction du nombre de classes dans votre jeu de données.

Le code suivant ira dans le fichier model.py.

```
1 %%writefile model.py
2 import torchvision
3
4 from torchvision.models.detection.faster_rcnn import FastRCNNPredictor
5
7 def create_model(num_classes):
8
9    # load Faster RCNN pre-trained model
10    model =
torchvision.models.detection.fasterrcnn_resnet50_fpn(pretrained=True)
11
12    # get the number of input features
13    in_features = model.roi_heads.box_predictor.cls_score.in_features
14    # define a new head for the detector with required number of classes
15    model.roi_heads.box_predictor = FastRCNNPredictor(in_features,
num_classes)
16
17    return model
```

Sur la ligne 10, nous chargeons le modèle Faster RCNN pré-entraîné avec le backbone ResNet50 FPN. Ensuite, à la ligne 13, nous obtenons le nombre d'entités en entrée. Pour ce modèle particulier, c'est 1024.

Enfin, nous changeons la tête du détecteur Faster RCNN en fonction des in_features et du nombre de classes.

3.6. Apprentissage et résultats

Pour l'apprentissage, nous avons utilisé Faster R-CNN avec la dorsale ResNet50, nombre de périodes à 10. Pour la taille du lot, il est défini sur 8, et nous utilisons l'optimiseur SGD avec un taux d'apprentissage de 0,001 et une dynamique de 0,9.

On commence par télécharger la base de données qui se compose de 1941 images pour l'apprentissage et de 299 images pour la validation.

Après avoir préparé la base de données, nous créons le modèle avec le nombre de classes spécifiées, et lançons l'apprentissage.

Nous exécutons la commande suivante.

```
!python train.py
```

Voici des exemples de sorties.

```
Number of training samples: 1941
Number of validation samples: 299

EPOCH 1 of 10
Training
Loss: 0.3054: 100% 243/243 [06:20<00:00,  1.57s/it]
Validating
Loss: 0.2147: 100% 38/38 [00:27<00:00,  1.38it/s]
Epoch #1 train loss: 0.441
Epoch #1 validation loss: 0.332
Took 6.809 minutes for epoch 0

Best validation loss: 0.33239566377903285

Saving best model for epoch: 1

SAVING PLOTS COMPLETE...

EPOCH 2 of 10
Training
Loss: 0.2978: 100% 243/243 [06:20<00:00,  1.56s/it]
Validating
Loss: 0.1450: 100% 38/38 [00:28<00:00,  1.35it/s]
Epoch #2 train loss: 0.315
Epoch #2 validation loss: 0.244
Took 6.807 minutes for epoch 1

Best validation loss: 0.24420299890794253

Saving best model for epoch: 2

SAVING PLOTS COMPLETE…
…

EPOCH 9 of 10
Training
Loss: 0.1957: 100% 243/243 [06:20<00:00,  1.57s/it]
Validating
Loss: 0.0802: 100% 38/38 [00:27<00:00,  1.36it/s]
Epoch #9 train loss: 0.181
Epoch #9 validation loss: 0.160
Took 6.815 minutes for epoch 8

Best validation loss: 0.15998098322827564

Saving best model for epoch: 9

SAVING PLOTS COMPLETE...

EPOCH 10 of 10
Training
Loss: 0.2676: 100% 243/243 [06:20<00:00,  1.56s/it]
Validating
Loss: 0.0975: 100% 38/38 [00:27<00:00,  1.36it/s]
Epoch #10 train loss: 0.172
Epoch #10 validation loss: 0.164
Took 6.806 minutes for epoch 9
SAVING PLOTS COMPLETE...
```

Au bout de 10 époques, nous avons la meilleure perte de validation de 0,1599.

Et voici les graphiques de perte d'entraînement et de validation après l'époque finale.

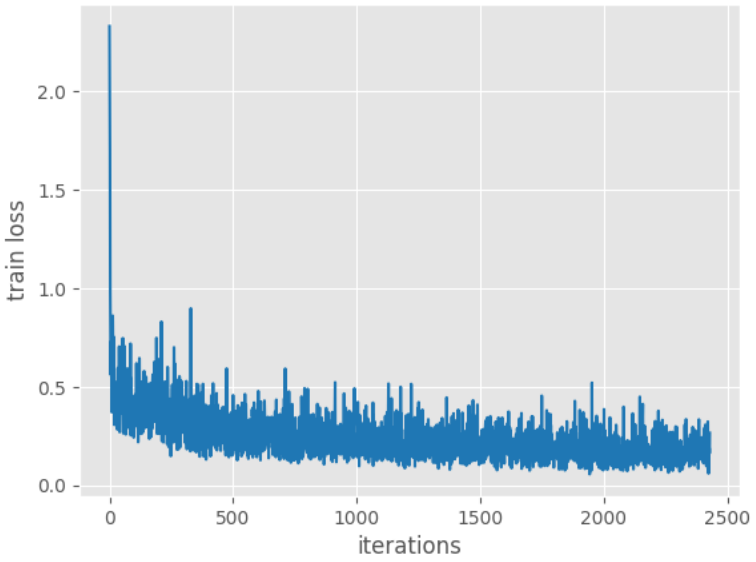


Figure 3.4:Graphique de perte d'entraînement après l'entraînement

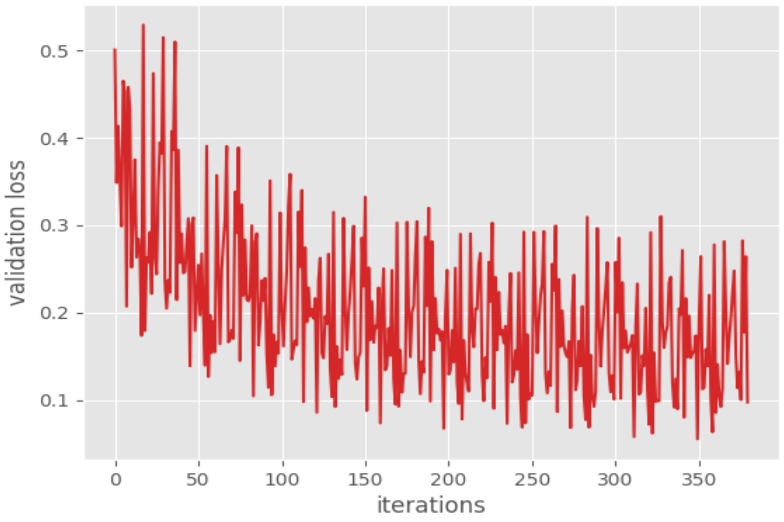


Figure 3.5: Graphique de perte de validation après entraînement.

D'après les graphiques, il semble que la perte d'entraînement ait atteint un plateau après l'itération 1500. et que la perte de validation ait diminué jusqu'à la fin de l'entraînement.

quelque résultat de test :

Figure 3.6: Exemple 1

Figure 3.7: Exemple 2

Figure 3.8: Exemple 3

4. DeepSORT [39]

4.1. Définition

DeepSORT est l'un des algorithmes de suivi d'objets les plus populaires. Il s'agit d'une extension de Simple Online Real-time Tracker ou SORT, qui est un algorithme de suivi en ligne.

SORT est un algorithme qui utilise le filtre de Kalman pour estimer l'emplacement de l'objet compte tenu de l'emplacement précédent de celui-ci. Le filtre de Kalman est très efficace contre les occlusions.

SORT comprend trois composants :

Détection : détection de l'objet d'intérêt dans la phase initiale i.

Estimation : Prédiction de l'emplacement futur i+1 de l'objet à partir de l'étape initiale à l'aide du **filtre de Kalman**. Il convient de noter que le filtre de Kalman se rapproche juste du nouvel emplacement de l'objet, qui doit être optimisé.

Association : Comme le **filtre de Kalman** estime la position future de l'objet i+1, il doit être optimisé en utilisant la position correcte. Cela se

fait généralement en détectant la position de l'objet dans cette position i+1. Le problème est résolu de manière optimale en utilisant l'algorithme **hongrois**.

Avec les bases de SORT à l'écart, nous pouvons incorporer des techniques d'apprentissage en profondeur pour améliorer l'algorithme SORT. Les réseaux de neurones profonds permettent à SORT d'estimer l'emplacement de l'objet avec une précision beaucoup plus élevée, car ces réseaux peuvent désormais décrire les caractéristiques de l'image c.

Essentiellement, le classificateur CNN est formé sur un ensemble de données spécifiques à une tâche jusqu'à ce qu'il atteigne une bonne précision. Une fois qu'il est atteint, le classificateur est supprimé et il ne nous reste que les fonctionnalités extraites de cet ensemble de données. Cette caractéristique extraite est ensuite incorporée à l'algorithme SORT pour suivre les objets.

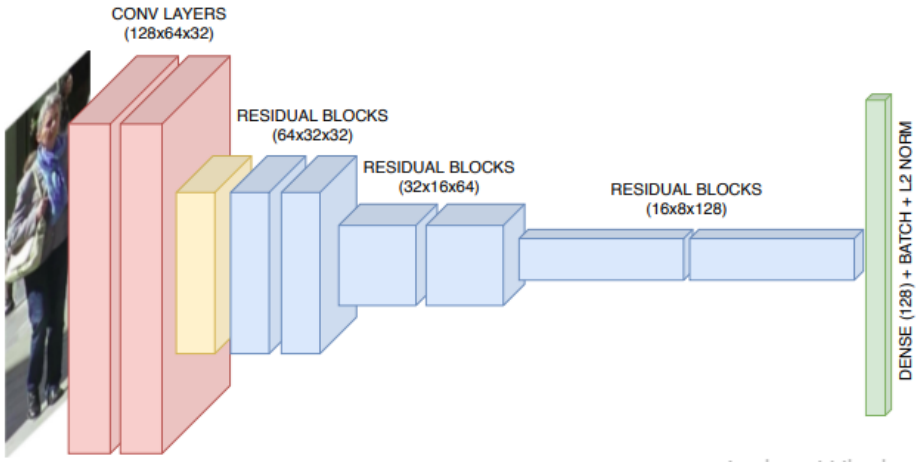


Figure 3.9 : Schéma de DeepSORT Extracteur de caractéristiques basé sur CNN [40]

4.2. Architecture

Dans DeepSort, le processus est le suivant.
Calculer les boîtes englobantes à l'aide de Faster R-CNN (détections)
Utilisez Trier (filtre de Kalman) et ReID (modèle d'identification) pour lier les boîtes englobantes et les pistes Si aucun lien ne peut être établi, un nouvel ID est attribué et il est nouvellement ajouté aux pistes.

Ce que l'on appelle les « détections » est la liste d'objets dans une image, et les « pistes » sont la liste d'objets actuellement suivies. Chaque élément de piste se voit attribuer un ID, et en attribuant un cadre de délimitation à chacun de ces éléments, vous pouvez attribuer un ID à l'objet.

ReID est principalement utilisé lors de la liaison des boîtes englobantes et des pistes. La distance entre les vecteurs de caractéristiques calculés par ReID à partir de l'image de l'objet de la cible de suivi actuelle (pistes) et les vecteurs de caractéristiques également calculés par ReID à partir de l'image de l'objet découpée par la boîte englobante (détections) dans Faster R-CNN , est utilisée pour lier la délimitation boîtes et rails. En termes simples, l'objet avec la plus petite distance est considéré comme la même objet et se voit attribuer un ID de piste. Pour calculer la distance vectorielle, les vecteurs de caractéristiques des 100 dernières images de chaque piste sont utilisés. A ce moment, les informations de coordonnées de la piste ne sont pas prises en compte.

La fonction de coût est définie comme Distance de tri * λ + distance ReID, mais dans l'article, $\lambda = 0$ s'est avéré donner empiriquement de bons résultats, de sorte que les informations de coordonnées ne sont pas prises en compte.
Si la position dans l'image actuelle, qui est supposée sur la base des informations de suivi du tri antérieur, est trop éloignée, l'ID ne sera pas

attribué. Lorsqu'une boîte englobante est laissée sans ID, Sort est utilisé pour en attribuer un.

Si la boîte englobante est « perdue » pendant 70 images, elle sera supprimée du suivi.

4.3. Implémentation

nous utilise le model faster-rcnn qui déjà prépare dans partie 1, on commence avec la présentation des package deep_sort

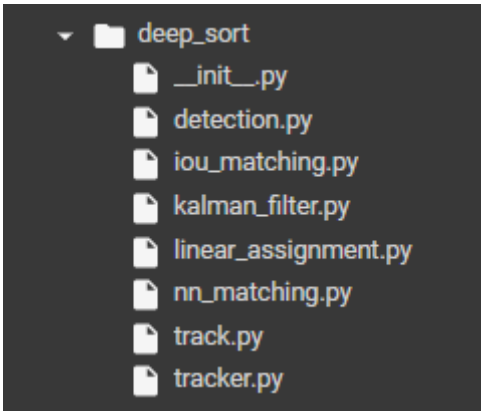


Figure 3.10: deep_dort package

detection.py : classe de base de détection.

kalman_filter.py : implémentation d'un filtre de Kalman et paramétrage concret pour le filtrage de l'espace image.

linear_assignment.py : ce module contient du code pour la correspondance des coûts minimaux et la cascade de correspondance.

iou_matching.py : ce module contient la métrique de correspondance IOU(Intersection over Union).

nn_matching.py : un module pour une métrique de correspondance du voisin le plus proche.

track.py : la classe de piste contient des données de piste à cible unique telles que l'état de Kalman, le nombre de coups, de ratés, de séquences de coups, les vecteurs de caractéristiques associés, etc.

tracker.py : il s'agit de la classe de suivi multi-cibles.

```
# DeepSORT -> Intializing tracker.
max_cosine_distance = 0.4
nn_budget = None
model_filename = './model_data/mars-small128.pb'
encoder = gdet.create_box_encoder(model_filename, batch_size=1)
metric = nn_matching.NearestNeighborDistanceMetric("cosine",
max_cosine_distance, nn_budget)
tracker = Tracker(metric,max_iou_distance=0.7,max_age=70,n_init=3)
```

On commence le téléchargement de modèle **mars-small128.pb** qui utilisé pour l'extraction de caractéristiques.

MAX_COSINE_DISTANCE est un seuil pour déterminer la similarité de l'objet par ReID. Plus la valeur est élevée, plus il est facile de supposer qu'il s'agit du même objet.

NN_BUDGET est une valeur qui indique combien de trames précédentes de vecteurs de caractéristiques doivent être conservées pour le calcul de distance pour chaque piste.

Le paramètre **max_age** du tracker spécifie après combien d'images les pistes non allouées seront supprimées. **n_init** spécifie après combien de trames les pistes nouvellement allouées seront activées. **max_iou_distance** est une valeur de seuil qui détermine dans quelle mesure les cadres de délimitation doivent se chevaucher pour déterminer l'identité de la piste non attribuée.

```
# DeepSORT -> Predicting Tracks.
  tracker.predict()
  tracker.update(detections)
```

Dans le code source, en particulier tracker.py , lorsque la fonction de mise à jour est exécutée, **_match** est appelé, et **linear_assignment.matching_cascade**, qui utilise Sort et ReID, et **linear_assignment.min_cost_matching** sont également appelés dans cet ordre.

Dans **linear_assignment.matching_cascade**, **min_cost_matching** est appelé, le calcul de la distance est effectué par ReID et la matrice de coût des détections N et des pistes M est calculée. Ici, le coût est calculé à partir des entités uniquement, sans tenir compte des informations de coordonnées.

Après cela, dans **distance_metric**, le calcul de distance de Sort est effectué en appelant **gating_distance** pour la matrice de coût qui a été

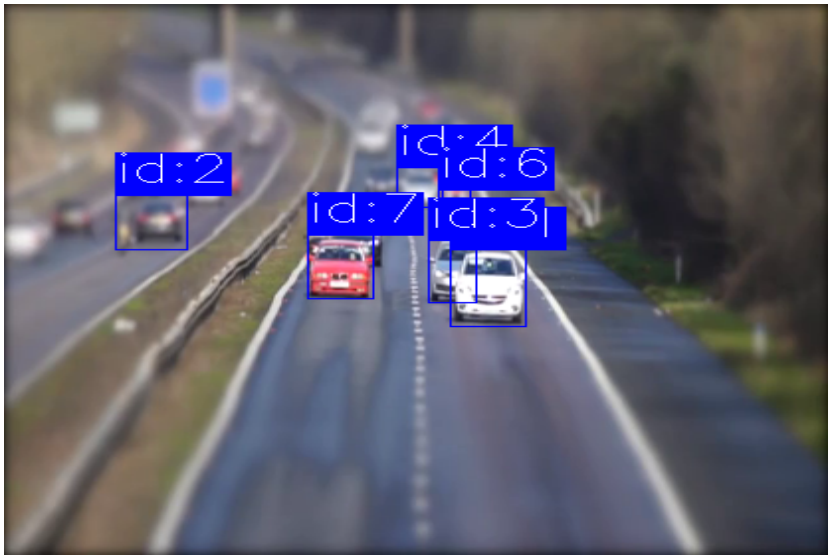
calculée par ReID, et si la distance de Sort est supérieure à une certaine valeur, la valeur de coût Infinity est attribuée.

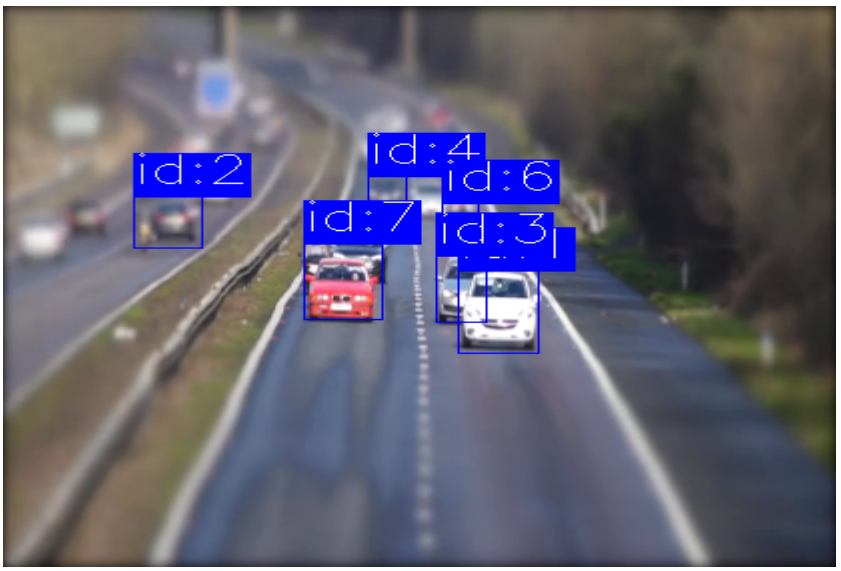
Enfin, en associant l'ID au coût le plus bas parmi les matrices de coût, on peut associer la piste à la détection qui a la distance ReID la plus proche avec une distance de Sort inférieure à un certain niveau.

4.4. Résultat

exemple 1 :

```
args={"input":"../../../car.mp4","output": "output.avi"}
detection_threshold = 0.8
RESIZE_TO = (416, 416)
```





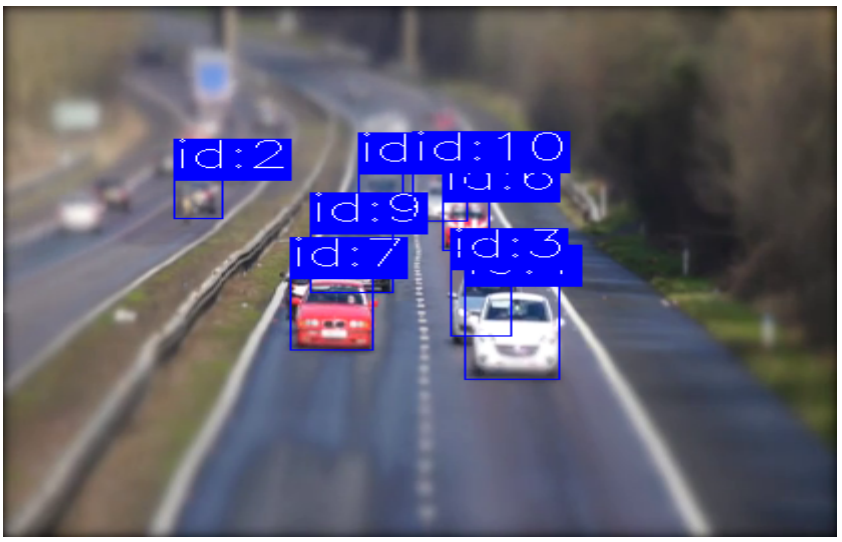
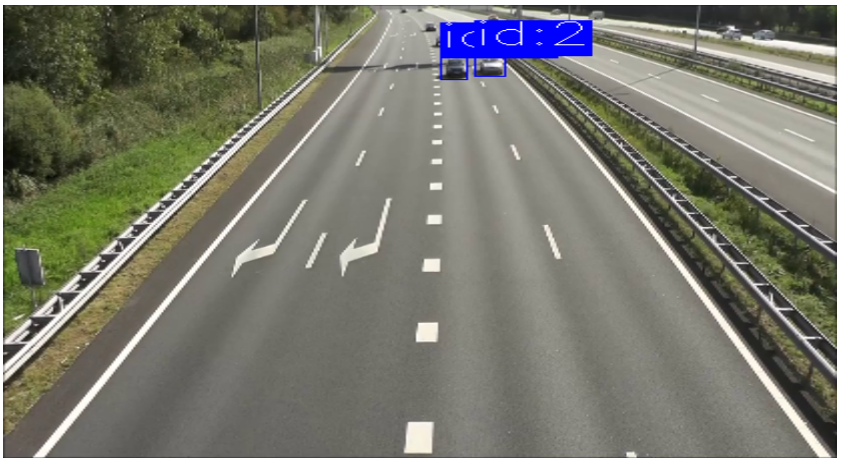
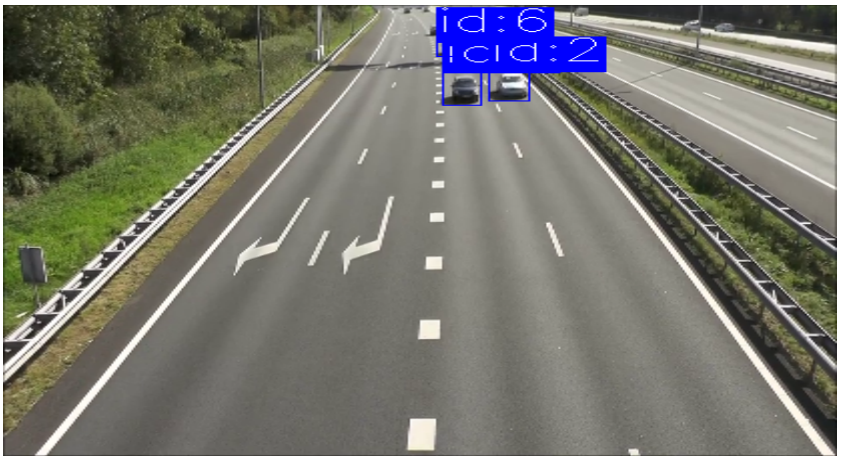


Figure 3.11: object tracking(faster_rcnn+deepsort) résultat 1

exemple 2 :

```
args={"input":"../../../Traffic.mp4","output": "output2.avi"}
detection_threshold = 0.8
RESIZE_TO = (416, 416)
```





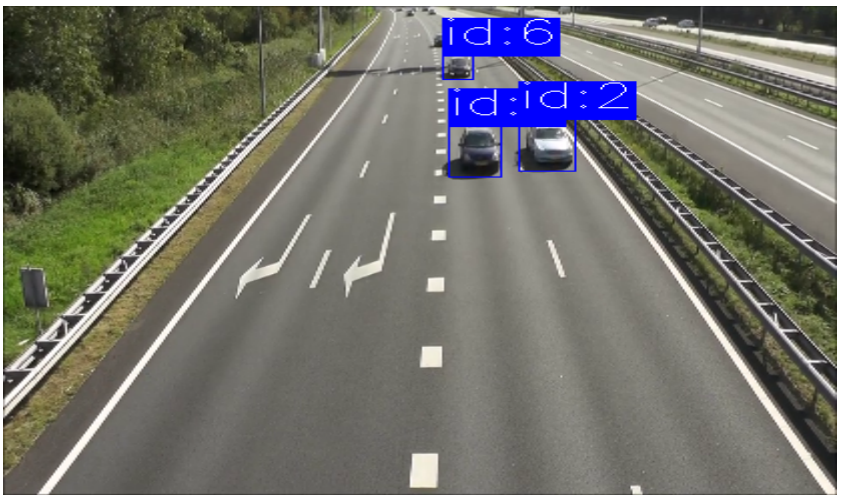


Figure 3.12: object tracking(faster_rcnn+deepsort) résultat 2

code source :https://colab.research.google.com/drive/1zjqHqN148dQznhQamZGXcx2qZOiYdiek?usp=sharing

5. Conclusion

Aucun tracker n'est meilleur que les autres. Il s'agit de savoir si nous voulons des preuves en temps réel plus rapides, des découvertes plus précises, ou peut-être une division supplémentaire. En outre, il a été démontré que les performances du tracker dépendent fortement de l'unité de détection d'objets réelle, il est donc possible de voir un tracker inférieur obtenir de meilleurs résultats uniquement en raison de son détecteur supérieur. Avec cela, **deepsort** est l'ensemble de suivi le plus rapide grâce à sa simplicité.

CONCLUSION GENERALE

Conclusion Générale

Bien que le suivi d'objets existe depuis longtemps, il présentait de nombreux défauts jusqu'à ce que le deep learning apparaisse, ou plutôt jusqu'à ce que le deep learning devienne célèbre dans l'analyse d'images, et pour cela nous avons ciblé l'une des méthodes de suivi utilisant le deep learning et avons fait ce qui suit :

Dans un premier temps nous avons introduit le traçage et ses principales méthodes , puis nous avons présenté l'apprentissage en profondeur (Deep Learning) et nous sommes allés un peu plus vers l'un de se modèles : Le Réseau neuronal convolutif (CNN), avec ces différents blocs de construction et ses domaines d'application .

Dans le troisième chapitre nous avons testé notre programme basé sur « faster rcnn » et « Deepsort » sur deux vidéos , les résultats étaient satisfaisants pour «faster rcnn » et nous avons confirmé que « Deepsort » est l'algorithme de suivi le plus rapide.

Nos résultats qui permettent de détecter des objets mobiles pourront nous servir dans plusieurs domaines . En termes de perspectives, il est intéressant d'implémenter des applications de suivi dans différents champ telles que :

la surveillance du trafic : les trackers peuvent être utilisés pour surveiller le trafic et suivre les véhicules sur la route. Ils peuvent être utilisés pour évaluer le trafic, détecter les infractions et bien d'autres.
Sports : les trackers peuvent également être utilisés dans les sports, comme le suivi de balle ou le suivi de joueurs. Cela peut à son tour être utilisé pour détecter les fautes, les buteurs dans un jeu, et bien d'autres.
Surveillance multi-caméras : Dans Tracking, la surveillance multi-caméras peut être appliquée. En cela, l'idée centrale est la ré-identification. Si une personne est suivie dans une caméra avec un identifiant, et que la personne sort du cadre et revient dans une autre caméra. Ensuite, la personne conservera le même identifiant qu'elle avait auparavant. Cette application peut aider à ré-identifier les objets qui réapparaissent dans une caméra différente et peut être utilisée dans la détection d'intrusion**.**

RÉFÉRENCES

Références

1. MEDOUAKH Saadia, ”Détection et suivi d’objets”,2019

2. Z. Soleimanitaleb, M. A. Keyvanrad and A. Jafari, "Object Tracking Methods:A Review," *2019 9th International Conference on Computer and Knowledge Engineering (ICCKE)*, 2019, pp. 282-288.

3. article about optical flow , From Wikipedia, the free encyclopedia the last edited on 27 July 2022, .

4. R. Verma, “A Review of Object Detection and Tracking Methods,” International Journal of Advance Engineering and Research Development, vol. 4, pp. 569–578, Oct. 2017.

5. Rui Yao;Guosheng Lin;Shixiong Xia;Jiaqi Zhao;Yong Zhou; (2020). *Video Object Segmentation and Tracking . ACM Transactions on Intelligent Systems and Technology,*

6. Najafzadeh, Nima; Fotouhi, Mehran; Kasaei, Shohreh (2015). *Iranian Conference on Electrical Engineering - Object tracking using Kalman filter with adaptive sampled histogram.*

7. Nordsjo, A.E. (2004). *Proceedings of the 2004 IEEE Radar Conference - A constrained extended Kalman filter for target tracking.*

8. Object Tracking: Particle Filter with Ease. Article by Darko Jurić 27 april 2015.

9. S. Avidan, “Support vector tracking,” IEEE transactions on pattern analysis and machine intelligence, vol. 26, no. 8, pp. 1064–1072, 2004.

10. M. Tian, W. Zhang, and F. Liu, “On-line ensemble SVM for robust object tracking,” in Asian conference on computer vision, 2007, pp. 355–364.

11. T. Malisiewicz, A. Gupta, and A. A. Efros, “Ensemble of exemplarSVMs for object detection and beyond.,” in Iccv, 2011.

12. S. Zhang, X. Yu, Y. Sui, S. Zhao, and L. Zhang, “Object tracking with multi-view support vector machines,” IEEE Transactions on Multimedia, vol. 17, no. 3, pp. 265–278, 2015.

13. Z. Chen, Z. Hong, and D. Tao, “An Experimental Survey on Correlation Filter-based Tracking,” Sep. 2015.

14. Convolutional neural networks: an overview and application in radiology.
Yamashita, R., Nishio, M., Do, R.K.G. *et al.* Convolutional neural networks: an overview and application in radiology. *Insights Imaging* 9, 611–629 (2018).

15. Singh, Sinam Ajitkumar (2020). *Deep Learning Techniques for Biomedical and Health Informatics || Short PCG classification based on deep learning.*

16. R. Abinayaa , D.N.V.S.L.S. Indirab , Dhanalakshmi Lanka, Acoustic based Scene Event Identification Using Deep Learning CNN, 5 April 2021 .

17. R. Nandhini Abirami , P. M. Durai Raj Vincent , Kathiravan Srinivasan , Usman Tariq , and Chuan-Yu Chang ,Deep CNN and Deep GAN in Computational Visual Perception-Driven Image Analysis,School of Information Technology and Engineering, , Taiwan,20 April 2021.

18. Keiron O’Shea , Ryan Nash,”An Introduction to Convolutional Neural Networks”,Department of Computer Science, Aberystwyth University, Ceredigion, SY23 3DB,School of Computing and Communications, Lancaster University, Lancashire, LA1 4YW,https://arxiv.org/abs/1511.08458,2 décembre 2015.

19. A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way Published inTowards Data Science by Sumit Saha .15 decembre 2018.

20. Kaiming He. Et al. Deep Residual Learning for Image Recognition. 10 Décembre 2015.

21. K. Simonyan. A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition International Conference on Learning Representations, 2015.

22. Christian Szegedy. Et al. Going Deeper with Convolutions. 17 Septembre 2014.

23. Matthew D Zeiler. Rob Fergus. Visualizing and Understanding Convolutional Networks. 12 Novembre 2013.

24. Alex Krizhevsky. Et al. ImageNet Classification with Deep Convolutional Neural Networks. NIPS. 2012.

25. Yann LeCun. Et al. Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE. Novembre 1998.

26. Rachana Patel, Sanskruti Patel ,Faculty of Computer Science and Applications, Charotar University of Science and Technology, Changa, India :"A Comprehensive Study of Applying Convolutional Neural Network for Computer Vision",· January 2020.

27. Carreira, Joao; Sminchisescu, Cristian . Computer Society Conference on Computer Vision and Pattern Recognition - "Constrained parametric min-cuts for automatic object segmentation". 2010.

28. Ian Endres , Derek Hoiem," Category Independent Object Proposals" , Department of Computer Science University of Illinois at Urbana-Champaign ,2010.

29. Santiago Manen, Matthieu Guillaumin, Luc Van Gool,"Prime Object Proposals with Randomized Prim's Algorithm", Computer Vision Laboratory ETH Zurich,ESAT - PSI / IBBT K.U. Leuven, 2013.

30. J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers , and A.W.M. Smeulders, " Selective Search for Object Recognition", Smeulders University of Trento, Italy,University of Amsterdam, the Netherlands, 2012.

31. R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in CVPR, 2014.

32. R. Girshick, "Fast r-cnn," in ICCV, 2015.

33. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in CVPR, 2016.

34. S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards realtime object detection with region proposal networks," in NIPS, 2015, pp. 91–99.

35. K. He, G. Gkioxari, P. Dollar, and R. B. Girshick, "Mask r-cnn," in ´ ICCV, 2017.

36. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in ECCV, 2016.

37. J. Redmon and A. Farhadi, “Yolo9000: better, faster, stronger,” arXiv:1612.08242, 2016.

38. Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun,”Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, **https://arxiv.org/pdf/1506.01497.pdf, 6 Jan 2016.**

39. Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In 2017 IEEE International Conference on Image Processing (ICIP), pages 3645–3649. IEEE, 2017.

40. Gioele Ciaparrone, Francisco Luque Sánchez , Siham Tabik , Luigi Troiano , Roberto Tagliaferri , Francisco Herrera,Department of Management and Innovation Systems, University of Salerno, 84084 Fisciano (SA), Italy ,Andalusian Research Institute in Data Science and Computational Intelligence, University of Granada, 18071 Granada, Spain,Department of Engineering, University of Sannio, 82100 Benevento, Italy,”DEEP LEARNING IN VIDEO MULTI-OBJECT TRACKING: A SURVEY ”,November 20, 2019.

Bibliographie

41. OpenCV(open Source Computer Vision),**OpticaFlow,** https://docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.htm **consulter le 26 août 2022**

42. Cindy Trinh Sridykhan, A tour of Video Object Tracking — Part III: Multiple Object Tracking article by Cindy Trinh Sridykhan, https://medium.com/@cindy.trinh.sridykhan/a-tour-of-video-object-tracking-part-iii-multiple-object-tracking-5e3a15ae0a7c, 22 septembre 2019.

43. Ajay krishnan,”Activation functions”,https://ajaykrish-krishnanrb.medium.com/non-linear-activation-functions-4b5e3ada8959,11 apr 2021.

44. Data Science Interview Preparation,"What is an activation function? What are commonly used activation functions", https://www.datasciencepreparation.com/blog/articles/what-is-an-activation-function-what-are-commonly-used-activation-functions/,7 May 2022.

45. Yaniv Noema, All About Pooling Layers for Convolutional Neural Networks (CNN) Published in imagescv,https://medium.com/imagescv/all-about-pooling-layers-for-convolutional-neural-networks-cnn-c4bca1c35e31 , 26 nov 2021.

46. Selective Search for Object Detection by Vaibhaw Singh Chandel .18 septembre 2017,https://learnopencv.com/selective-search-for-object-detection-cpp-python/.

47. Nathan Zhao, Understanding Objectness in Object Detection Models ,14 august 2020,https://medium.com/@zhao.nathan/understanding-objectness-in-object-detection-models-5d8c9d032488.

48. DataCamp,Lars Hulstaert, A Beginner's Guide to Object Detection,https://www.datacamp.com/tutorial/object-detection-guide **consulter le 26 août 2022**

49. Devjyoti Chakraborty *,Training an object detector from scratch in PyTorch,*https://pyimagesearch.com/2021/11/01/training-an-object-detector-from-scratch-in-pytorch/, November 1, 2021.

50. LearnOpenCV,https://learnopencv.com/understanding-multiple-object-tracking-using-deepsort/, **consulter le 26 août 2022.**

Résumé

L’apprentissage en profondeur est l’une des branches de l’intelligence artificielle qui dépend principalement des réseaux de neurones artificiels et permet l’accomplissement de tâches complexes telles que le suivi d’objets dont nous allons porter notre étude sur.

Dans cette thèse, nous avons augmenté un réseau neuronal convolutif basé sur une région avec une couche de régression qui peut estimer les quatre coordonnées de la boîte englobante de l'objet détecté.

Mots clés : L'apprentissage en profondeur, Suivi d'objet, réseaux de neurones convolutifs,détection d'objets.

Summary

Deep learning is one of the branches of artificial intelligence that mainly depends on artificial neural networks and allows the performance of complex tasks such as tracking objects which we are going to focus our study on.

In this thesis, we augmented a region based convolutional neural network with a regression layer that can estimate the four coordinates of the detected object bounding box.

Keywords: Deep learning, object tracking, convolutional neural networks, object detection.

ملخص:

التعلم العميق هو أحد فروع الذكاء الاصطناعي الذي يعتمد بشكل أساسي على الشبكات العصبية الاصطناعية ويسمح بأداء المهام المعقدة مثل تتبع الأشياء التي سنركز دراستنا عليها.

في هذه الأطروحة ، قمنا بزيادة شبكة عصبية تلافيفية قائمة على المنطقة بطبقة انحدار يمكنها تقدير الإحداثيات الأربعة لصندوق إحاطة الكائن المكتشف.

الكلمات الرئيسية : التعلم العميق ، تعقب الأشياء ، الشبكات العصبية التلافيفية، كشفا الاشياء.