

Object tracking pour le system ADAS

Rapport du mini-projet

Réalisé par :

Echcherqaoui Oussama

Feddoul Yassine

Encadre par:

Mr.HAJJI TARIK

Remerciement

Nous tenons à exprimer notre profonde gratitude envers notre Professeur MASROUR TAWIK et Mr. HAJJI TARIK, nos superviseurs du projet, pour leurs conseils patients, leurs encouragements enthousiastes et critiques utiles à ce travail de recherche et de technique. Nous tenons également à remercier Mme HASSANI IBTISSAME, pour ses conseils et son aide dans le respect de notre calendrier.

Nous tenons à remercier vivement le corps professoral responsable de notre filière, qui à sacrifier pour rendre le projet de notre filière faisable.

Enfin, nous tenons à remercier nos parents pour leur soutien et leurs encouragements tout au long de nos études.

Résumé

Notre projet a comme objectif de détecter et suivre les objets mobiles afin de prédire leurs comportements futurs.

En résumé ce rapport se compose de deux chapitres, le premier résume la bibliographie nécessaire pour la compréhension préliminaire du sujet et pouvoir entamer notre sujet, le deuxième concerne la partie pratique de notre projet où on a implémenté les différentes solutions résolvant notre problématique, faire une étude comparative ainsi conclure la solution la plus adéquate pour notre projet.

Comme nous sommes dans la première phase d'un projet de grande envergure, ce rapport fournit également une vaste bibliographie pour le sujet discuté.

Table des Illustrations

Figure 1 : Situation de notre projet.....	8
Figure 2: applications du machine learning	12
Figure 3 : machine learning vs Deep learning.....	14
Figure 4 : ARCHITECTURE D'UN RÉSEAU NEURONAL	15
Figure 5:opencv based code	19
Figure 6:resultat de detection	19
Figure 7: résultat de la detection 2	20
Figure 8: les fichiers a telecharger	20
Figure 9: yolo based code.....	22
Figure 10: résultat de detection yolo	23
Figure 11:resultat de detection 2	23
Figure 12:bounding box	24
Figure 13: la grille de detection.....	24
Figure 14: boites d'ancrages	25
Figure 15:les coordonnées des boites	25
Figure 16: résultat de la classification M-label	25
Figure 17: les couches convolutives.....	26
Figure 18:notion de IoU	28
Figure 19:Recall / Precision	28
Figure 20:comparaison des modeles	29

Table des Tableaux

Table 1 : adas longitudinal	9
Table 2 :adas latéral support.....	10
Table 3 :adas miscellaneous	10
Table 4 :RNN vs CNN	17

Table of Contents

Chapitre I : Présentation du projet et Revue de littérature	8
1. Advanced driving systems	9
a. Vue d'ensembles des technologies d'ADAS.....	9
2. Machine learning.....	11
a. Definition	11
b. Approches du Machine Learning	11
c. Applications du machine Learning	12
3. Deep Learning	13
a. Introduction to Deep Learning	13
b. Difference entre Machine Learning et Deep Learning.....	13
4. Réseau de Neurones	14
a. Définition Réseau de Neurones.....	14
b. Convolutional Neural Network	15
c. Recurrent neural network	16
d. La différence entre CNN & RNN	17
Chapitre II : notre travail	18
1. Algorithme à base de Opencv	18
2. Detection des object en utilisant Yolo v3.....	20
a. Yolo en pratique	20
b. Yolo en théorie	24
3. Comparaison entre les différents algorithmes	26
a. Précision et rappel	27
b. IoU (Intersection over union)	27
c. AP (Average Precision).....	28
d. Comparatif yolo et les alternatifs	29

Introduction générale

La plupart des accidents de la route sont dus à une erreur humaine, les systèmes avancés d'aide à la conduite (ADAS) sont des systèmes développés pour automatiser, adapter et améliorer le véhicule pour plus de sécurité et une meilleure conduite.

Le système automatisé qui est fourni par l'ADAS au véhicule a prouvé qu'il permettait de réduire le nombre de décès et des dégâts d'une manière remarquable sur les routes, en minimisant les erreurs humaines. Les dispositifs de sécurité sont conçus pour éviter les collisions et les accidents en proposant des technologies qui alertent le conducteur en cas de problème potentiel ou pour éviter les collisions en mettant en place des mesures de protection et en prenant le contrôle du véhicule.

Nous souhaitons déployer un système d'assistance à la conduite ADAS (Advanced Driving Assistance System) qui s'appuie sur des technologies de l'intelligence artificielle. Pour subvenir à ces besoins, il est primordial de passer par la prédiction des trajectoires des objets mobiles.

Pour cadrer notre travail nous avons élaborer un cahier de charge qui se compose de 4 axes principales

Axe 1 : contexte et présentation du projet, dont on a défini de façon brève notre projet.

Axe 2 : analyse du besoin, où on a exprimé l'utilité de notre projet.

Axe 3 : Expression fonctionnelle du besoin où les contraintes et les fonctionnalités de notre système sont décrites.

Axe 4 : Planning de réalisation.

Pour aboutir à cet objectif, nous devons :

- Assurer la détection des objets mobiles.
- Suivre les objets mobiles.
- Prédire les trajectoires des objets mobiles.

Chapitre I : Présentation du projet et Revue de littérature

I. Présentation du projet

La conduite dans un environnement urbain nécessite de l'interaction avec d'autres véhicules, que ce soit à la suite d'un lent véhicule, se coordonnant pour se relayer avec les véhicules aux intersections, ou manœuvrer autour d'autres véhicules pour atteindre le stationnement spots, il est presque impossible de faire un voyage en voiture sans être affecté par un autre véhicule d'une manière ou d'une autre.

En tant que conducteur, les systèmes d'assistance et les véhicules autonomes deviennent plus sophistiqué, le raisonnement sur ces interactions avec les véhicules deviennent de plus en plus importants. Dont notre rôle commence, qui se manifeste à prédire les trajectoires des objets mobiles, en se basant sur l'ensemble d'information collectées a l'intermédiaire des capteurs, cameras ..., qui vont être traiter à la suite pour aider à déterminer l'ensembles de décisions futur que l'automobile va prendre.

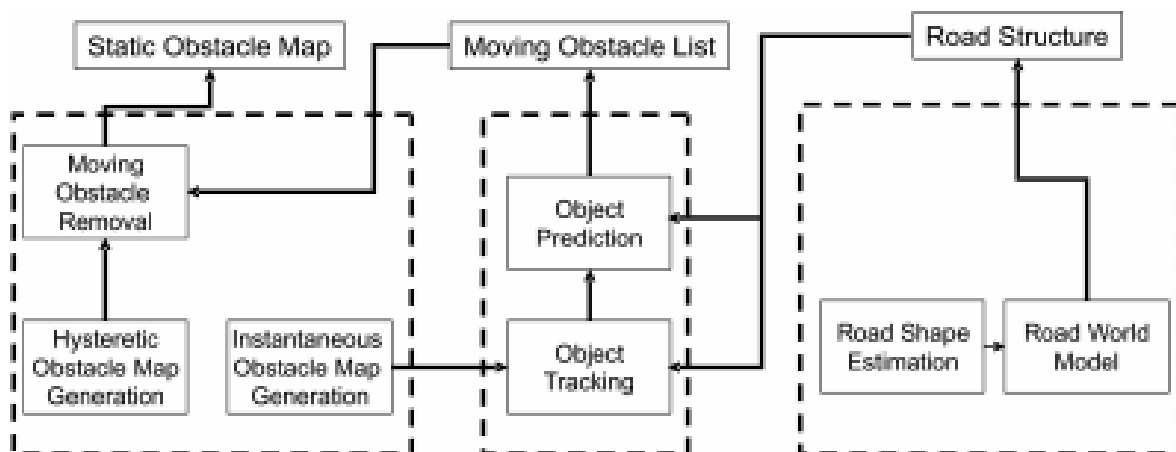


Figure 1 : Situation de notre projet

II. Revue de littérature

1. Advanced driving systems

Les systèmes avancés d'aide à la conduite sont des systèmes intelligents qui résident à l'intérieur du véhicule et assistent le conducteur principal de diverses manières. Ces systèmes peuvent être utilisés pour fournir des informations vitales sur le trafic, la fermeture et le blocage des routes, les niveaux d'embouteillage, les itinéraires suggérés pour éviter les embouteillages, etc.

a. Vue d'ensembles des technologies d'ADAS

Pour donner une idée de ce que les systèmes d'aide à la conduite représentent pour les utilisateurs, nous présentons un aperçu des technologies existantes. Pour des raisons de commodité, elles ont été divisées en sous-catégories. Ce bref aperçu de la technologie ADAS existante ne met en évidence que les types d'ADAS les plus "courants".

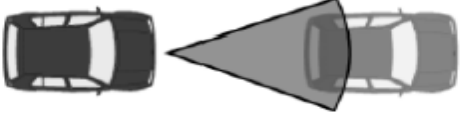
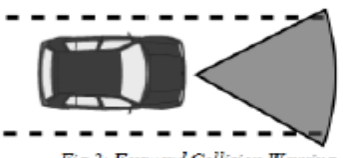
	ADAS		Description
Longitudinal	ACC	Adaptive Cruise Control	<p>ACC is becoming a more and more common accessory in modern cars. Basically, this technology keeps a safe distance between the driver's car and vehicles ahead. The driver can adjust the distance, and the system makes sure it's maintained, using throttle and brake control. Most ACC systems have influence on the driving task (they control brake and throttle), but still allow user take-overs.</p>  <p><i>Fig 1: Adaptive Cruise Control</i></p>
	FCW	Forward Collision Warning	<p>Like the ACC, this system detects vehicles in front of the driver's car. Obviously, it can be integrated with ACC. However, current systems still have problems distinguishing cars from trees, bridges from road signs, etc.</p>  <p><i>Fig 2: Forward Collision Warning</i></p>
	ISA	Intelligent Speed Assistance	<p>ISA influences the speed at which a car is driving. The maximum speed can be pre-set, or acquired from GPS data. Interfacing with the driver is done via the acceleration pedal, or by using visual or audio warnings.</p>

Table 1 : adas longitudinal

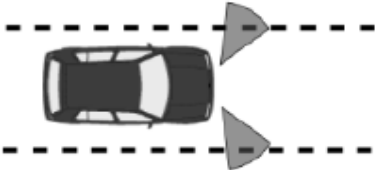

Lateral Support	LDW	Lane Departure Warning	The main task of Lane Departure Warning is to make sure a car is driving safely between road marks (i.e. in a lane). LDW uses cameras and computer systems to detect and process roadsides and lane markings, and warn the driver if necessary. Acceptance of LDW is expected to be a problem because control of the car is given to the computer, and chances of false alarms are still present.
	LKS	Lane Keeping System	<p>An extended version of the LDW system is the Lane Keeping System. Instead of warning the driver about the unintended lane departure, LKS intervenes with the driving task by using steering wheel actuators. LKS can completely take over the steering task of the driver.</p>  <p><i>Fig 3: Lane Keeping System</i></p>
	LCA	Lane Change Assistance	<p>LCA is a collection of technologies taking care of blind spots and rear-view problems. It uses sensors to detect objects and vehicles which normally can't be seen by the driver because of obstructed view. Also, approaching vehicles from behind can be detected in time, and the driver can be informed of this.</p>  <p><i>Fig 4: Lane Change Assistance</i></p>

Table 2 :adas latéral support

Miscellaneous		Night Vision Systems	These systems provide the driver with an enhanced view of the outside world. It's meant to be used during bad weather or night time. Though already implemented in several car models, the system still has a problem with its interface: how to present the enhanced image to the user. Current solutions consist of displaying the image on a monitor on the dashboard.
		Parking Assistance	The Parking Assistance system looks like Lane Change Assistance, but is meant for low speed and short distance, for example when parking a car. Using sensors a car can measure available space, and show this information to the driver. Current systems have limited use because of the low range these sensors operate with. Future developments will let the system take over control of the car during parking, letting the car park itself.
		Fuel Economy Devices	With Fuel Economy Devices the fuel flow and usage can be monitored and analysed per car. A system can intervene by informing the driver about the fuel usage, or by actively intervening, using an active gas pedal or other active systems.

Table 3 :adas miscellaneous

2. Machine learning

a. Definition

Machine Learning (ML) est l'étude des algorithmes informatiques qui s'améliorent automatiquement grâce à l'expérience et à l'utilisation des données. Il est considéré comme une partie de l'intelligence artificielle. Les algorithmes du Machine Learning construisent un modèle à partir d'un échantillon de données, appelé "données d'apprentissage", afin de faire des prédictions ou de prendre des décisions sans être explicitement programmés pour le faire.

Les algorithmes du machine Learning sont utilisés dans une grande variété d'applications, telles que le filtrage du courrier électronique et en computer vision, où il est difficile ou impossible de développer des algorithmes conventionnels pour effectuer les tâches nécessaires.

Machine Learning implique que les ordinateurs découvrent comment ils peuvent effectuer des tâches sans être explicitement programmés pour le faire. Il inclut l'apprentissage des machines à partir de données fournies afin d'exécuter certaines tâches. Pour les tâches simples confiées à des ordinateurs, il est possible de programmer des algorithmes indiquant à la machine comment exécuter toutes les étapes nécessaires pour résoudre le problème en question ; de la part de l'ordinateur, aucun apprentissage n'est nécessaire. Pour les tâches plus avancées, il peut être difficile pour un humain de créer manuellement les algorithmes nécessaires. En pratique, il peut s'avérer plus efficace d'aider la machine à développer son propre algorithme, plutôt que de demander aux programmeurs humains de spécifier chaque étape nécessaire.

b. Approches du Machine Learning

Les approches du Machine Learning sont traditionnellement divisées en trois grandes catégories, en fonction de la nature du "signal" ou du "feedback" dont dispose le système d'apprentissage :

- **L'apprentissage supervisé** : On présente à l'ordinateur des exemples d'entrées et leurs sorties souhaitées, données par un "enseignant", et l'objectif est d'apprendre une règle générale qui relie les entrées aux sorties.
- **Apprentissage non supervisé** : Aucune étiquette n'est donnée à l'algorithme d'apprentissage, le laissant seul pour trouver une structure dans son entrée.

L'apprentissage non supervisé peut être un objectif en soi (découverte de modèles cachés dans les données) ou un moyen d'atteindre un objectif (apprentissage de caractéristiques).

- **Apprentissage par renforcement** : Un programme informatique interagit avec un environnement dynamique dans lequel il doit réaliser un certain objectif (comme conduire un véhicule ou jouer à un jeu contre un adversaire). Au fur et à mesure qu'il navigue dans son espace de problème, le programme reçoit un retour d'information analogue à des récompenses, qu'il essaie de maximiser.

D'autres approches ont été développées qui ne correspondent pas exactement à cette triple catégorisation, et parfois plus d'une est utilisée par le même système d'apprentissage automatique. Par exemple, la modélisation de sujets, la réduction de la dimensionnalité ou le méta-apprentissage.

En 2020, le Deep Learning est devenu l'approche dominante pour la plupart des travaux en cours dans le domaine du Machine Learning.

c. Applications du machine Learning

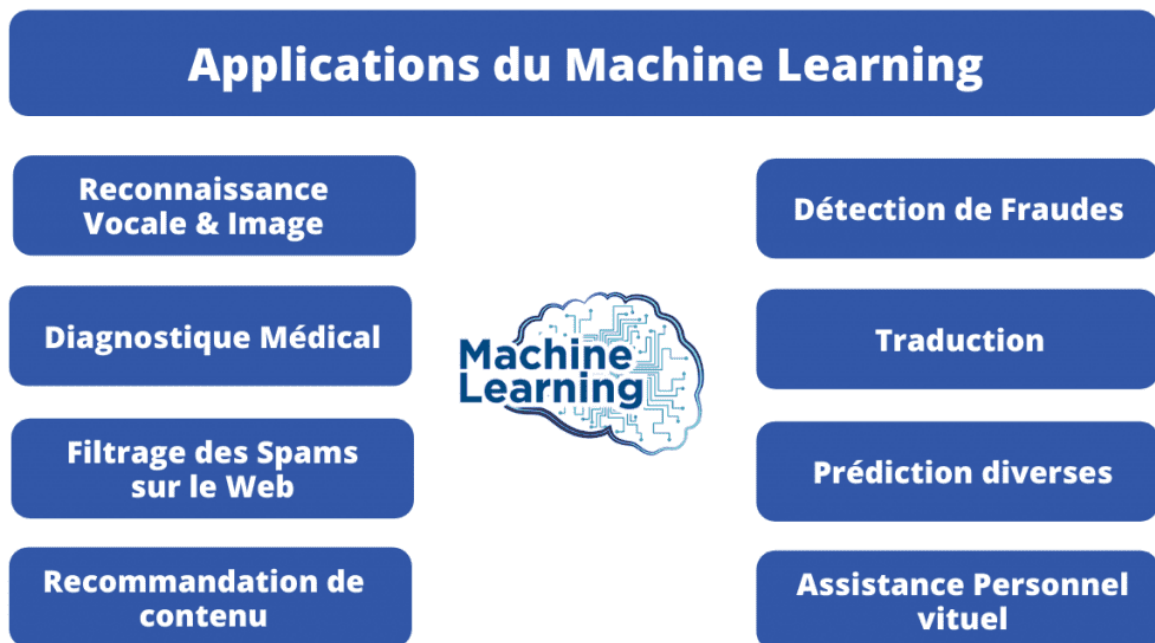


Figure 2: applications du machine learning

3. Deep Learning

a. Introduction to Deep Learning

Le Deep Learning fait partie d'une famille plus large de méthodes de Machine Learning basées sur les réseaux neuronaux artificiels avec apprentissage par représentation. L'apprentissage peut être supervisé, semi-supervisé ou non supervisé.

Les architectures du Deep Learning telles que le Deep neural networks, Recurrent neural networks and Convolutional neural networks ont été appliquées à des domaines tels que computer vision, la reconnaissance vocale, le traitement du langage naturel, la reconnaissance audio, le filtrage des réseaux sociaux, la traduction automatique, la bio-informatique, la conception de médicaments, l'analyse d'images médicales, l'inspection des matériaux et les programmes de jeux de société, où elles ont produit des résultats comparables et, dans certains cas, supérieurs aux performances des experts humains.

Les réseaux de neurones artificiels (ANN) ont été inspirés par le traitement de l'information et les nœuds de communication distribués dans les systèmes biologiques. Les ANN présentent diverses différences par rapport aux cerveaux biologiques. Plus précisément, les réseaux neuronaux ont tendance à être statiques et symboliques, alors que le cerveau biologique de la plupart des organismes vivants est dynamique (plastique) et analogique.

L'adjectif "profond" dans le Deep Learning fait référence à l'utilisation de plusieurs couches dans le réseau. Les premiers travaux ont montré qu'un perceptron linéaire ne peut pas être un classificateur universel, puis qu'un réseau avec une fonction d'activation non polynomiale avec une couche cachée de largeur non limitée peut en revanche l'être. Le Deep Learning est une variante moderne qui concerne un nombre non limité de couches de taille limitée, ce qui permet une application pratique et une mise en œuvre optimisée, tout en conservant l'universalité théorique dans des conditions légères. Dans le Deep Learning, les couches sont également autorisées à être hétérogènes et à s'écarter largement des modèles connexionnistes biologiques, dans un souci d'efficacité, d'apprentissage et de compréhension, d'où le terme "structuré".

b. Difference entre Machine Learning et Deep Learning

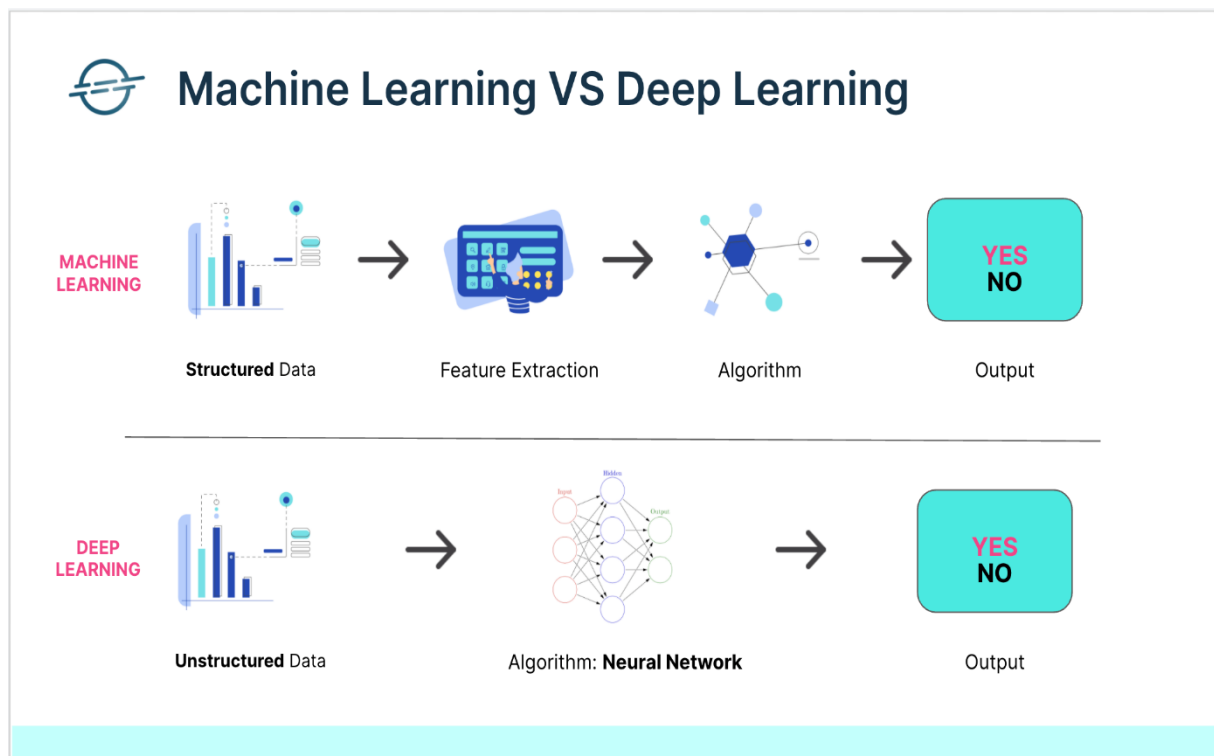


Figure 3 : machine learning vs Deep learning

4. Réseau de Neurones

a. Définition Réseau de Neurones

Un réseau neuronal est une série d'algorithmes qui s'efforcent de reconnaître les relations sous-jacentes dans un ensemble de données par un processus qui imite le fonctionnement du cerveau humain. En ce sens, les réseaux neuronaux font référence à des systèmes de neurones, de nature organique ou artificielle. Les réseaux neuronaux peuvent s'adapter à des entrées changeantes ; ainsi, le réseau génère le meilleur résultat possible sans avoir à redéfinir les critères de sortie. Le concept des réseaux neuronaux, qui trouve ses racines dans l'intelligence artificielle, gagne rapidement en popularité dans le développement des systèmes actuels.

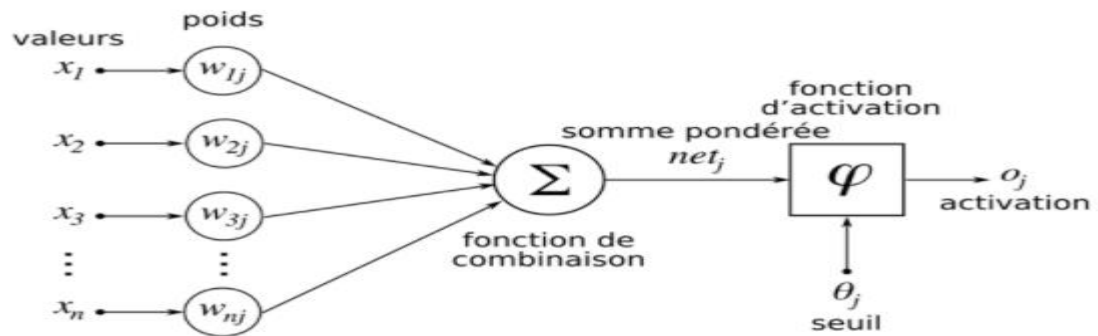


Figure 4 : ARCHITECTURE D'UN RÉSEAU NEURONAL

b. Convolutional Neural Network

En Deep Learning, un réseau neuronal convolutionnel (CNN) est une classe de réseaux neuronaux profonds, le plus souvent appliquée à l'analyse de l'imagerie visuelle. Ils sont également connus sous le nom de réseaux de neurones artificiels invariants en translation ou invariants dans l'espace (SIANN), en raison de l'architecture à poids partagé des noyaux de convolution qui analysent les couches cachées et des caractéristiques d'invariance en translation.

Ils ont des applications dans la reconnaissance d'images et de vidéos, les systèmes de recommandation, la classification d'images, la segmentation d'images, l'analyse d'images médicales, le traitement du langage naturel, les interfaces cerveau-ordinateur et les séries chronologiques financières.

Les CNN sont des versions régularisées des perceptrons multicouches. Les perceptrons multicouches désignent généralement des réseaux entièrement connectés, c'est-à-dire que chaque neurone d'une couche est connecté à tous les neurones de la couche suivante. Le fait que ces réseaux soient "entièrement connectés" les rend enclins à sur adapter les données. Les méthodes typiques de régularisation consistent à faire varier les poids au fur et à mesure que la fonction de perte est minimisée, tout en réduisant la connectivité de manière aléatoire. Les réseaux CNN adoptent une approche différente de la régularisation, ils tirent parti du modèle hiérarchique des données et assemblent des modèles de complexité croissante en utilisant des modèles plus petits et plus simples gravés dans les filtres. Par conséquent, sur l'échelle de la connexité et de la complexité, les CNN se situent à l'extrémité inférieure.

Les réseaux convolutifs ont été inspirés par des processus biologiques dans la mesure où le modèle de connectivité entre les neurones ressemble à l'organisation du cortex visuel animal. Les neurones corticaux individuels ne répondent aux stimuli que dans une région restreinte du champ visuel, appelée champ réceptif. Les champs réceptifs de différents neurones se chevauchent partiellement, de sorte qu'ils couvrent la totalité du champ visuel.

Les CNN utilisent relativement peu de prétraitement par rapport aux autres algorithmes de classification d'images. Cela signifie que le réseau apprend à optimiser les filtres ou les noyaux de convolution qui, dans les algorithmes traditionnels, sont élaborés à la main. Cette indépendance vis-à-vis des connaissances préalables et de l'intervention humaine dans l'extraction des caractéristiques est un avantage majeur.

c. Recurrent neural network

Un réseau neuronal récurrent (RNN) est une classe de réseaux neuronaux artificiels où les connexions entre les nœuds forment un graphe dirigé le long d'une séquence temporelle. Cela lui permet de présenter un comportement dynamique temporel. Dérivés des réseaux neuronaux à anticipation, les RNN peuvent utiliser leur état interne (mémoire) pour traiter des séquences d'entrées de longueur variable. Cela les rend applicables à des tâches telles que la reconnaissance de l'écriture manuscrite non segmentée et connectée ou la reconnaissance vocale.

Le terme "réseau neuronal récurrent" est utilisé sans distinction pour désigner deux grandes classes de réseaux ayant une structure générale similaire, l'une étant à impulsion finie et l'autre à impulsion infinie. Les deux classes de réseaux présentent un comportement dynamique temporel. Un réseau récurrent à impulsions finies est un graphe acyclique dirigé qui peut être déroulé et remplacé par un réseau neuronal à anticipation stricte, tandis qu'un réseau récurrent à impulsions infinies est un graphe cyclique dirigé qui ne peut être déroulé.

Les réseaux récurrents à impulsions finies et à impulsions infinies peuvent avoir des états supplémentaires stockés, et le stockage peut être sous le contrôle direct du réseau neuronal. Le stockage peut également être remplacé par un autre réseau ou graphe, s'il incorpore des délais ou comporte des boucles de rétroaction. De tels états contrôlés sont appelés "gated state" ou "gated memory", et font partie des réseaux de mémoire à long terme (LSTM) et des unités récurrentes gated. On parle également de réseau neuronal à rétroaction (FNN).

d. La différence entre CNN & RNN

La principale différence entre CNN et RNN est **la capacité à traiter des informations temporelles** ou des données qui se présentent sous forme de séquences, comme une phrase par exemple. En outre, les réseaux neuronaux convolutifs et les réseaux neuronaux récurrents sont utilisés à des fins complètement différentes, et il existe des différences dans les structures des réseaux neuronaux eux-mêmes pour s'adapter à ces différents cas d'utilisation.

Les CNN utilisent des filtres dans les couches convolutionnelles pour transformer les données. Les RNN, quant à eux, réutilisent les fonctions d'activation d'autres points de données de la séquence pour générer la sortie suivante d'une série.

CNN	RNN
It is suitable for spatial data such as images.	RNN is suitable for temporal data, also called sequential data.
CNN is considered to be more powerful than RNN.	RNN includes less feature compatibility when compared to CNN.
This network takes fixed size inputs and generates fixed size outputs.	RNN can handle arbitrary input/output lengths.
CNN is a type of feed-forward artificial neural network with variations of multilayer perceptrons designed to use minimal amounts of preprocessing.	RNN unlike feed forward neural networks - can use their internal memory to process arbitrary sequences of inputs.
CNNs use connectivity pattern between the neurons. This is inspired by the organization of the animal visual cortex, whose individual neurons are arranged in such a way that they respond to overlapping regions tiling the visual field.	Recurrent neural networks use time-series information - what a user spoke last will impact what he/she will speak next.
CNNs are ideal for images and video processing.	RNNs are ideal for text and speech analysis.

Table 4 :RNN vs CNN

Chapitre II : notre travail

I. Etude comparative des solutions de détection

1. Algorithme à base de Opencv

En premier temps, nous avons commencé par un algorithme que nous avons trouvé pendant notre recherche. L'objectif du programme donné est de détecter l'objet d'intérêt (voiture) dans les images vidéo et de continuer à suivre le même objet. Ceci est un exemple de détection de véhicules en Python.

First: Install packages « opencv – numpy »

Second : Téléchargement du fichier cars.xml « c'est le modèle déjà entraîne que nous allons utiliser »

Third : Exécuter le code

```
# Loop runs if capturing has been initialized.
while True:
    # reads frames from a video
    ret, frames = cap.read()
    if ret == False:
        break
    # convert to gray scale of each frames
    gray = cv2.cvtColor(frames, cv2.COLOR_BGR2GRAY)

    # Detects cars of different sizes in the input image
    cars = car_cascade.detectMultiScale(gray, 1.1, 1)

    # To draw a rectangle in each cars
    for (x,y,w,h) in cars:
        cv2.rectangle(frames,(x,y),(x+w,y+h),(0,0,255),2)

    # Display frames in a window
    cv2.imshow('video2', frames)

    # Wait for Esc key to stop
    if cv2.waitKey(33) == 27:
        break

# De-allocate any associated memory usage
cv2.destroyAllWindows()
```

```

# OpenCV Python program to detect cars in video frame
# import libraries of python OpenCV
import cv2

# capture frames from a video
cap = cv2.VideoCapture('Downloads/video.avi')

# Trained XML classifiers describes some features of some object we want to detect
car_cascade = cv2.CascadeClassifier('Desktop/cars.xml')

# Loop runs if capturing has been initialized.
while True:
    # reads frames from a video
    ret, frames = cap.read()
    if ret == False:
        break
    # convert to gray scale of each frames
    gray = cv2.cvtColor(frames, cv2.COLOR_BGR2GRAY)

    # Detects cars of different sizes in the input image
    cars = car_cascade.detectMultiScale(gray, 1.1, 1)

    # To draw a rectangle in each cars
    for (x,y,w,h) in cars:
        cv2.rectangle(frames, (x,y), (x+w,y+h), (0,0,255), 2)

```

Figure 5:opencv based code

Après l'exécution du programme on obtient comme sortie une vidéo ou les véhicules sont détecter et suivies. Les résultats de cette détection :

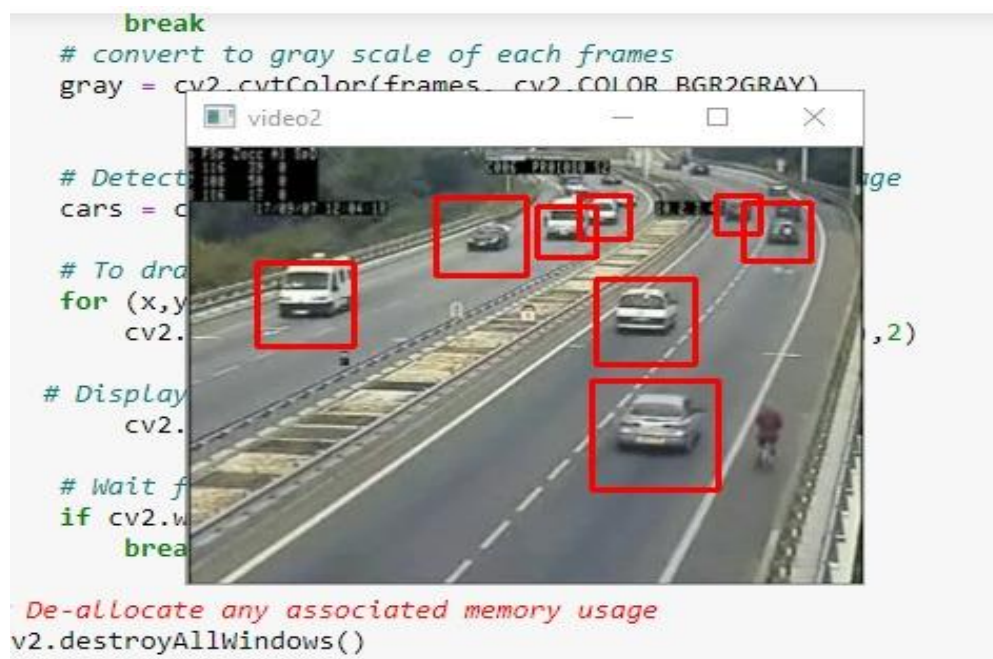


Figure 6:resultat de detection

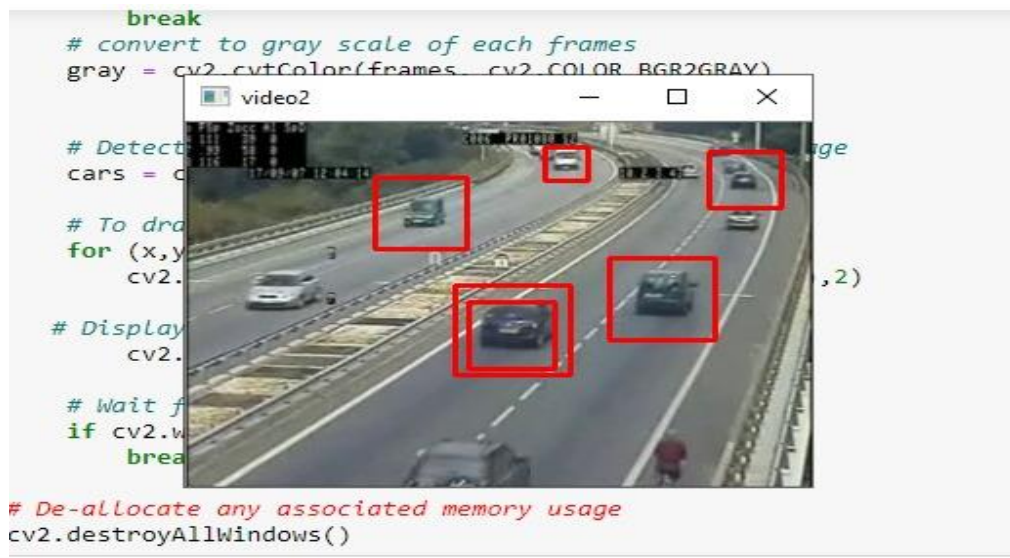


Figure 7: résultat de la detection 2

Ce code a été présenté par Afzal Ansari , Source : (Ansari, n.d.)

2. Detection des object en utilisant Yolo v3

a. Yolo en pratique

La deuxième méthode consiste à utiliser yolo v3, Yolov3 est l'un des algorithmes d'apprentissage profond les plus populaires, il est rapide, précis et différent.

Il utilise une approche différente, qui consiste à faire la détection en un seul passage. Cet algorithme "You only look once" l'image en ce sens qu'il n'a besoin que d'un seul passage de propagation vers l'avant à travers le réseau pour faire des prédictions.

La procédure est la suivante :

First : Nous devons télécharger trois fichiers contenant la configuration, le fichier de poids pré-entraîné et les classes du jeu de données COCO, et les classes de l'ensemble de données COCO.



Figure 8: les fichiers a telecharger

Second : installer : OpenCV et Numpy

Third : Exécuter le code.

```
In [1]: import cv2
```

```
In [2]: import numpy as np
import time
```

```
In [5]: #Load YOLO
net = cv2.dnn.readNet("Desktop/yolov3.weights","Desktop/yolov3.cfg") # Original yolov3
#net = cv2.dnn.readNet("yolov3-tiny.weights","yolov3-tiny.cfg") #Tiny Yolo
classes = []
with open("Desktop/coco.names","r") as f:
    classes = [line.strip() for line in f.readlines()]
```

```
In [6]: print(classes)
```

```
['person', 'bicycle', 'car', 'motorbike', 'aeroplane', 'bus', 'train', 'truck', 'boat', 'traffic light', 'fire hydrant', 'stop
sign', 'parking meter', 'bench', 'bird', 'cat', 'dog', 'horse', 'sheep', 'cow', 'elephant', 'bear', 'zebra', 'giraffe', 'backpa
ck', 'umbrella', 'handbag', 'tie', 'suitcase', 'frisbee', 'skis', 'snowboard', 'sports ball', 'kite', 'baseball bat', 'baseball
glove', 'skateboard', 'surfboard', 'tennis racket', 'bottle', 'wine glass', 'cup', 'fork', 'knife', 'spoon', 'bowl', 'banana',
'apple', 'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza', 'donut', 'cake', 'chair', 'sofa', 'pottedplant', 'be
d', 'diningtable', 'toilet', 'tvmonitor', 'laptop', 'mouse', 'remote', 'keyboard', 'cell phone', 'microwave', 'oven', 'toaste
r', 'sink', 'refrigerator', 'book', 'clock', 'vase', 'scissors', 'teddy bear', 'hair drier', 'toothbrush']
```

```
In [7]: layer_names = net.getLayerNames()
outputlayers = [layer_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]
```

```
In [7]: layer_names = net.getLayerNames()
outputlayers = [layer_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]
```

```
In [8]: colors= np.random.uniform(0,255,size=(len(classes),3))
```

```
In [20]: cap=cv2.VideoCapture("Downloads/cars.mp4") #0 for 1st webcam
font = cv2.FONT_HERSHEY_PLAIN
starting_time= time.time()
frame_id = 0

while True:
    _,frame= cap.read() #
    frame_id+=1

    height,width,channels = frame.shape
    #detecting objects
    blob = cv2.dnn.blobFromImage(frame,0.00392,(320,320),(0,0,0),True,crop=False) #reduce 416 to 320

    net.setInput(blob)
    outs = net.forward(outputlayers)
    #print(outs[1])

    #Showing info on screen/ get confidence score of algorithm in detecting an object in blob
    class_ids=[]
    confidences=[]
    boxes=[]
```

```

#Showing info on screen/ get confidence score of algorithm in detecting an object in blob
class_ids=[]
confidences=[]
boxes=[]
for out in outs:
    for detection in out:
        scores = detection[5:]
        class_id = np.argmax(scores)
        confidence = scores[class_id]
        if confidence > 0.3:
            #object detected
            center_x= int(detection[0]*width)
            center_y= int(detection[1]*height)
            w = int(detection[2]*width)
            h = int(detection[3]*height)

            #cv2.circle(img,(center_x,center_y),10,(0,255,0),2)
            #rectangle co-ordinaters
            x=int(center_x - w/2)
            y=int(center_y - h/2)
            #cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),2)

            boxes.append([x,y,w,h]) #put all rectangle areas
            confidences.append(float(confidence)) #how confidence was that object detected and show that percentage
            class_ids.append(class_id) #name of the object tha was detected

indexes = cv2.dnn.NMSBoxes(boxes,confidences,0.4,0.6)

```

```

indexes = cv2.dnn.NMSBoxes(boxes,confidences,0.4,0.6)

for i in range(len(boxes)):
    if i in indexes:
        x,y,w,h = boxes[i]
        label = str(classes[class_ids[i]])
        confidence= confidences[i]
        color = colors[class_ids[i]]
        cv2.rectangle(frame,(x,y),(x+w,y+h),color,2)
        cv2.putText(frame,label+" "+str(round(confidence,2)),(x,y+30),font,1,(255,255,255),2)

elapsed_time = time.time() - starting_time
fps=frame_id/elapsed_time
cv2.putText(frame,"FPS:"+str(round(fps,2)),(10,50),font,2,(0,0,0),1)

cv2.imshow("Image",frame)
key = cv2.waitKey(1) #wait 1ms the loop will start again and we will process the next frame

if key == 27: #esc key stops the process
    break;

cap.release()
cv2.destroyAllWindows()

```

Figure 9: yolo based code

Après l'exécution du programme on obtient comme sortie une vidéo où les véhicules sont détectés et suivies. Les résultats de cette détection :

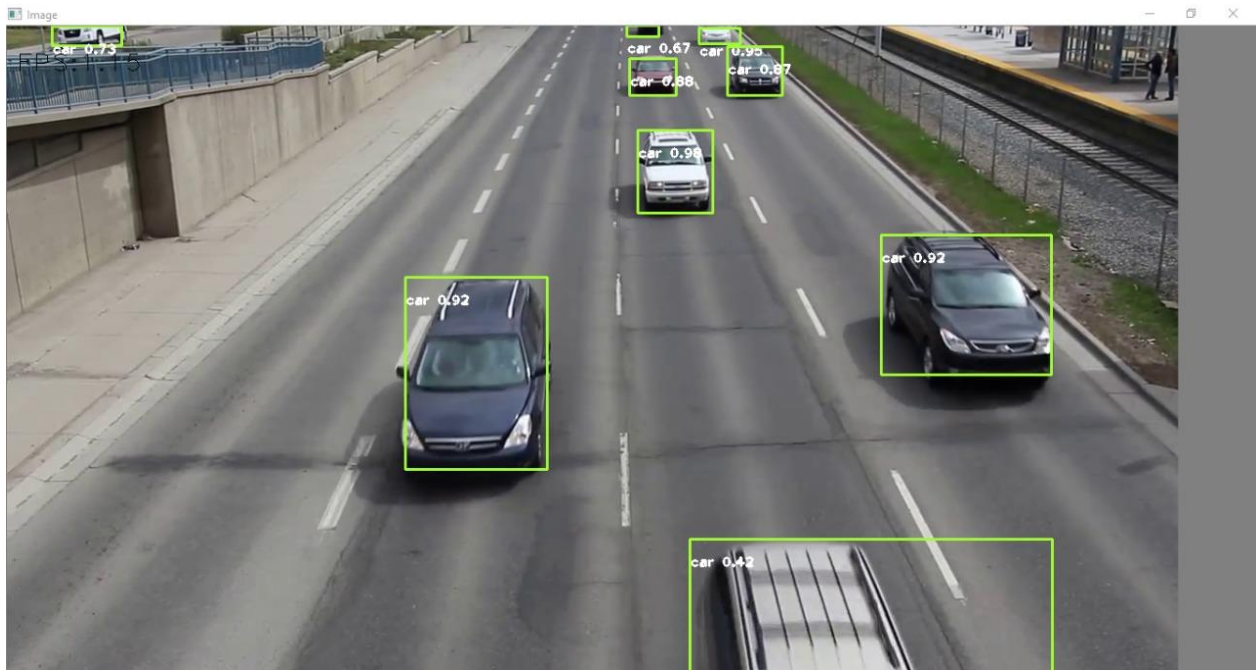


Figure 10: résultat de détection yolo

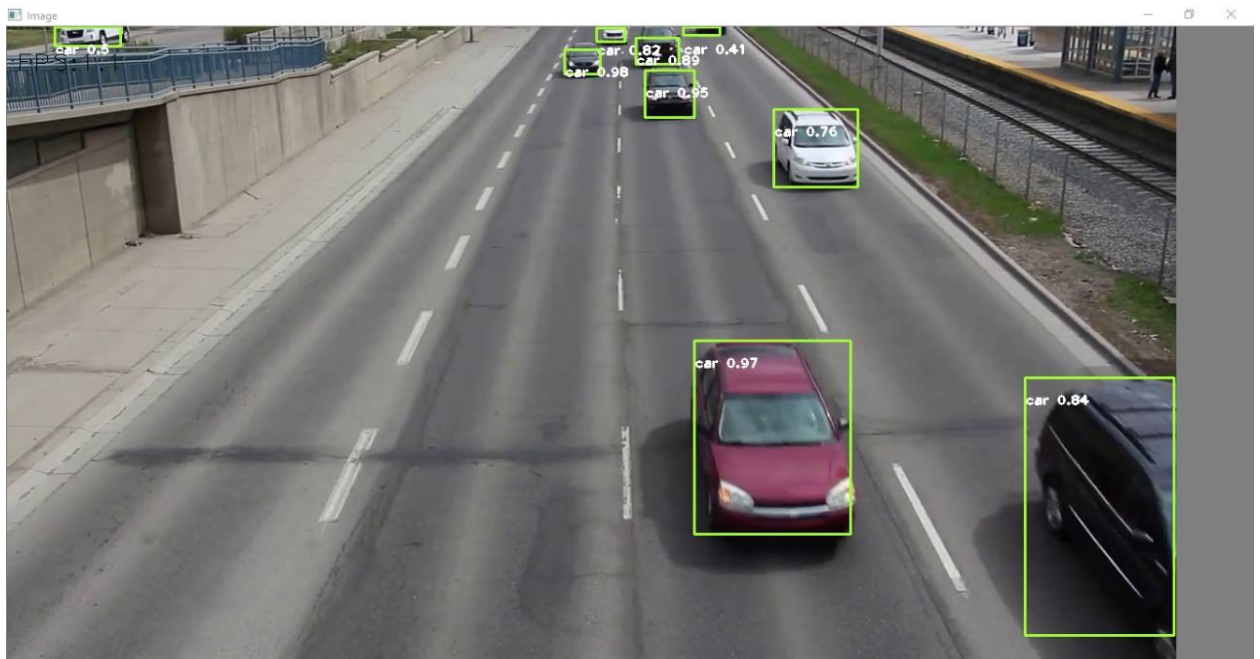


Figure 11: resultat de detection 2

On remarque des deux illustrations qui sont des outputs de notre algorithme, où les objets mobiles sont bien détectés ainsi que suivies à l'aide des BOXES.

b. Yolo en théorie

Nous essayons dans cette section d'expliquer l'algorithme YOLOv3 tel que nous l'avons compris dans l'article. Il s'agit de Détection, Classification et Localisation (Bounding box).



Figure 12: bounding box

Yolo divise l'image d'entrée en une grille $S * S$. Chaque grille ne prédit qu'un seul objet.

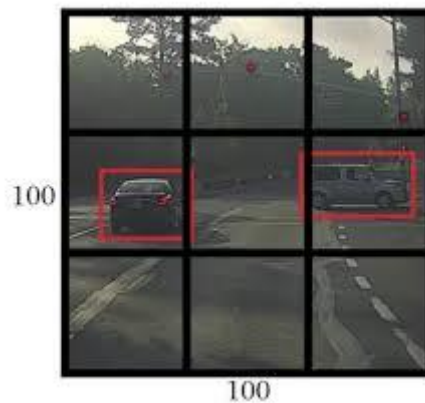


Figure 13: la grille de detection

Le système prédit les boîtes de délimitation en utilisant des groupes de dimensions comme boîtes d'ancrage. Les boîtes d'ancrage sont des boîtes sélectionnées à la main et présentant différents rapports hauteur/largeur (pour les boîtes bidimensionnelles) conçus pour correspondre aux rapports relatifs des classes d'objets à détecter.

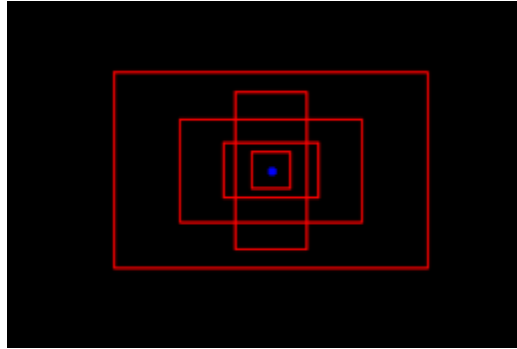


Figure 14: boîtes d'ancrages

Le réseau prédit 4 coordonnées pour chaque boîte de délimitation t_x , t_y , t_w , t_h .

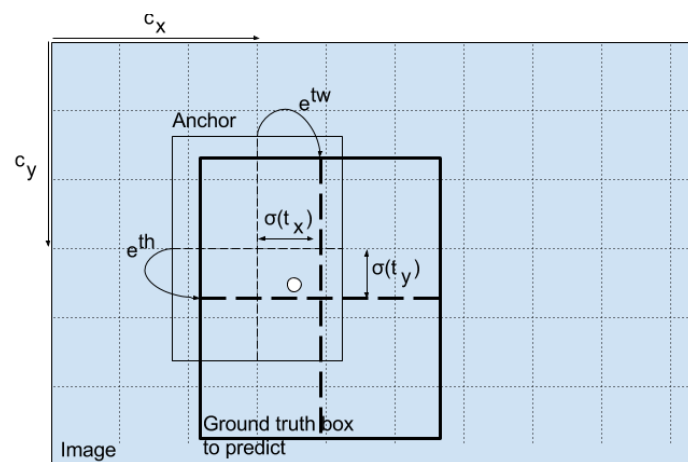


Figure 15: les coordonnées des boîtes

Une fonction sigmoïde est utilisée pour s'assurer que les coordonnées ne dépassent pas (1,1).

-Chaque boîte prédit les classes que la boîte englobante peut contenir en utilisant la classification multi-label.

-La sortie résultante est

$$\mathbf{y} = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 1 \\ 0 \\ 0 \\ 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

} Anchor box 1
 Pedestrian

 } Anchor box 2
 Car

Figure 16: résultat de la classification M-label

Le système utilise le clustering k-means pour déterminer les périeurs de la boîte englobante. Nous avons choisi arbitrairement 9 clusters et 3 échelles, puis nous avons réparti les clusters de manière égale entre les échelles. Sur le jeu de données COCO, les 9 clusters étaient : (10×13) , (16×30) , (33×23) , (30×61) , (62×45) , (59×119) , (116×90) , (156×198) , (373×326) .

L'extracteur de caractéristiques comporte 53 couches convolutives.

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

Figure 17: les couches convolutives

3. Comparaison entre les différents algorithmes

mAP (précision moyenne) pour la détection d'objets

La PA (précision moyenne) est une métrique populaire pour mesurer la précision des détecteurs d'objets tels que Faster R-CNN, SSD, YOLO etc.

La précision moyenne calcule la valeur de précision moyenne pour une valeur de rappel comprise entre 0 et 1. Cela semble compliqué, mais c'est en fait assez simple, comme le montre un exemple. Mais avant cela, nous allons d'abord faire un bref rappel sur la précision, le rappel et l'IoU.

a. Précision et rappel

La **précision** : mesure l'exactitude de vos prédictions, c'est-à-dire le pourcentage de vos prédictions qui sont correctes.

Le **rappel** mesure dans quelle mesure vous trouvez tous les cas positifs. Par exemple, nous pouvons trouver 80% des cas positifs possibles dans nos K meilleures prédictions.

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

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

TP = True positive

TN = True negative

FP = False positive

FN = False negative

Par exemple, dans le test de dépistage du cancer :

$$Precision = \frac{TP}{\text{total positive results}}$$

$$Recall = \frac{TP}{\text{total cancer cases}}$$

b. IoU (Intersection over union)

L'IoU mesure le chevauchement entre deux frontières. Nous l'utilisons pour mesurer le degré de chevauchement entre la frontière prédite et la vérité de terrain (la frontière réelle de l'objet). Dans certains jeux de données, nous prédéfinissons un seuil IoU (par exemple 0,5) pour déterminer si la prédiction est un vrai positif ou un faux positif

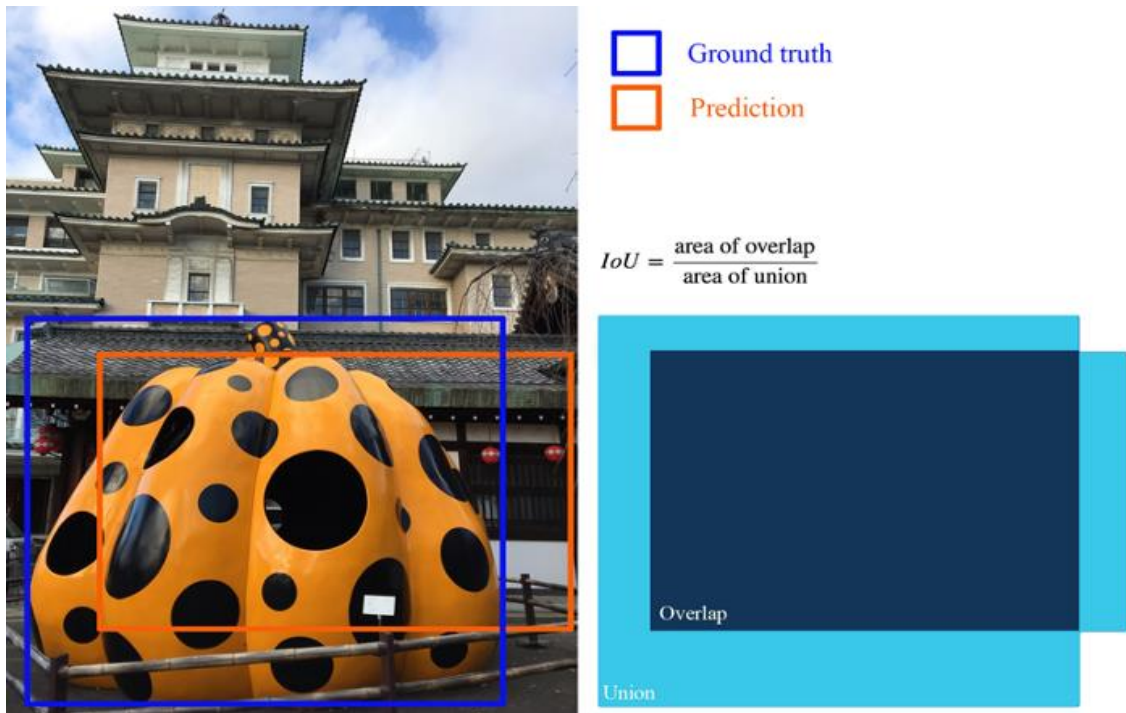


Figure 18: notion de IoU

c. AP (Average Precision)

Nous allons créer un exemple simplifié à l'extrême pour démontrer le calcul de la précision moyenne. Dans cet exemple, l'ensemble de données ne contient que 5 pommes. Nous recueillons toutes les prédictions faites pour les pommes dans toutes les images et les classons par ordre décroissant en fonction du niveau de confiance prédit. La deuxième colonne indique si la prédiction est correcte ou non. Dans cet exemple, la prédiction est correcte si $\text{IoU} \geq 0,5$.

Rank	Correct?	Precision	Recall
1	True	1.0	0.2
2	True	1.0	0.4
3	False	0.67	0.4
4	False	0.5	0.4
5	False	0.4	0.4
6	True	0.5	0.6
7	True	0.57	0.8
8	False	0.5	0.8
9	False	0.44	0.8
10	True	0.5	1.0

Figure 19: Recall / Precision

Prenons la ligne de rang 3 et montrons d'abord comment la précision et le rappel sont calculés.

La précision est la proportion de TP = $2/3 = 0,67$.

Le rappel est la proportion de TP parmi les positifs possibles = $2/5 = 0,4$.

Les valeurs de rappel augmentent à mesure que l'on descend dans le classement des prédictions.

d. Comparatif yolo et les alternatifs

YOLOv3 est extrêmement rapide et précis. En mAP mesuré à 0,5 IOU, YOLOv3 est à égalité avec Focal Loss mais environ 4x plus rapide. De plus, vous pouvez facilement arbitrer entre la vitesse et la précision en changeant simplement la taille du modèle, sans avoir à vous réentraîner

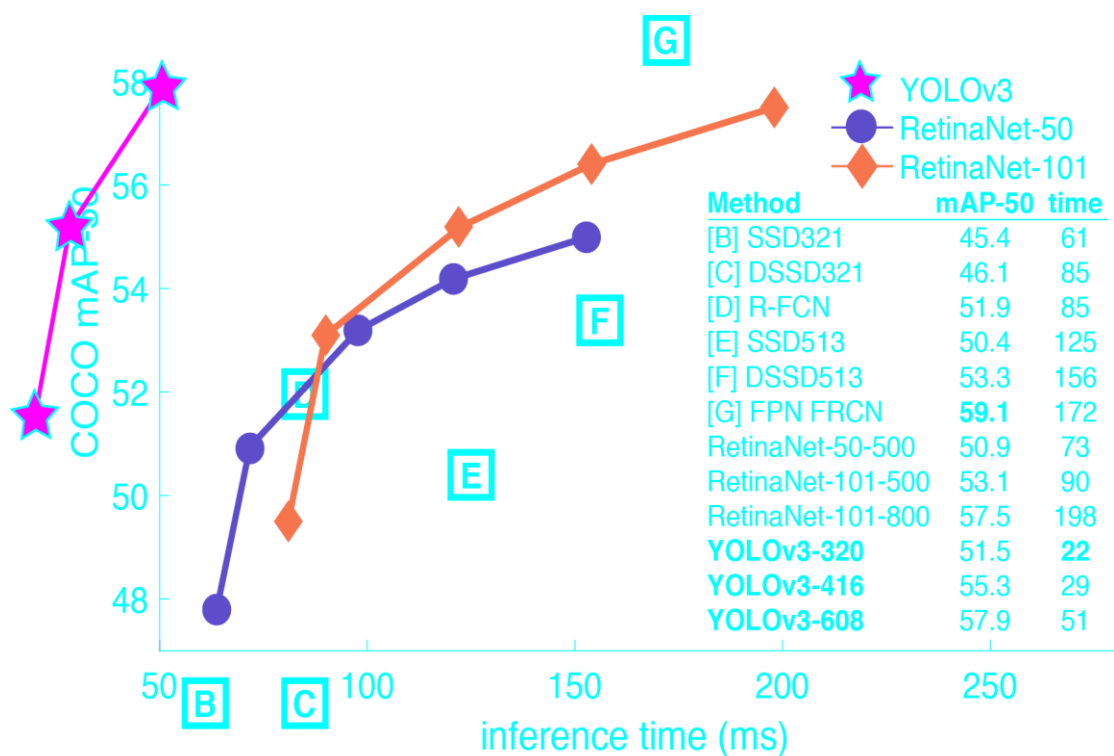


Figure 20: comparaison des modèles

Notre modèle présente plusieurs avantages par rapport aux systèmes basés sur des classificateurs. Il examine l'image entière au moment du test, de sorte que ses prédictions sont informées par le contexte global de l'image. Il fait également des prédictions avec une seule évaluation de réseau, contrairement aux systèmes comme le R-CNN qui en nécessite des

milliers pour une seule image. Cela le rend extrêmement rapide, plus de 1000 fois plus rapide que le R-CNN et 100 fois plus rapide que le R-CNN rapide.

Conclusion

En somme, notre projet comportée deux parties essentiel, une partie théorique où on était devant l'obligation d'effectuer plusieurs recherches bibliographiques pour se familiariser avec notre sujet et aussi chercher l'ensemble des anciennes solutions à notre problématique, les étudier ainsi passer à la partie pratique où on a implémenté plusieurs technique d'intelligence artificiel, les comparer et ainsi conclure la solution la plus optimal pour la résolution de notre problème.

Comme étendu de notre travail, la détection et le suivie sont réalisés, nous continuerons le travail sur la dernière partie de notre travail qui est la prédiction.