الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et populaire

Ministère de l'enseignement supérieur et de ma recherche scientifique

Université de 8 Mai 1945 - Geulma -

Faculté des Mathématiques d'Informatique et des Sciences de la matière

Département d'Informatique

Mémoire de Master

Filière : Informatique

Option : Système Informatiques

Thème :

Détection et Classification des Véhicules par Réseaux de Neurones à Convolution CNN

Encadré Par :

Dr.BENCHERIET Chemesse ennahar

Présenté Par :

TOUAHRI Islam

2019/2020

Remerciement

Dieu, merci pour nous avoir donné la force et la volonté de mener à bien ce travail.

Nous tenons à remercier tout particulièrement notre encadreur, Madame ***Bencheriet Chemesse Enahar*** *pour nous avoir fait confiance et nous avoir suggéré ce thème.*

Nos remerciements les plus sincères, ainsi que nos gratitudes vont à nos parents ainsi que nos frères et soeurs, qui nous ont accompagné et soutenu tout au long de notre parcours.

Nous remercions les membres de jury de soutenance d'avoir accepté d'évaluer ce modeste travail.

A vous tous nous exprimons notre reconnaissance et notre gratitude car encore une fois c'est grâce à vous tous et à votre abnégation et à votre altruisme qu'on a pu poursuivre nos études.

Résumé

La sécurité routière et autoroutière est devenue un problème de haute priorité pour les autorités publiques à cause de l'augmentation massive des accidents de la route par année.

Notre projet entre dans le cadre du 'monitoring' routier et autoroutier dont l'objectif principale est la détection et la classification des véhicules sur des images du trafic routier ou autoroutier. Pour cela nous avons utilisé deux réseaux convolutifs travaillant séquentiellement où la première phase dédiée au réseau 'Alex Net' (modifié) consiste en la détection des véhicules dans toute l'image. La fin de la tâche du premier réseau déclenche le démarrage du deuxième (VGG16) pré entrainé sur 4 classes de véhicules qui décidera par la suite de la classe de chaque véhicule détecté précédemment par 'Alex Net' (modifié).

Le système mise en place est capable de détecter (véhicule, non-véhicule) et classifier les véhicules selon leur type 'bus, car, motocycle, truck', et présente une base solide pour un développement futur du domaine du 'monitoring' routier et autoroutier en Algérie.

Mots clés : Détection des véhicules, classification des véhicules, Apprentissage profond, Réseaux de neurones, convolutions, 'Alex Net', 'VGG16'.

Abstract

Road safety has become a high priority issue for public authorities due to the massive increase in road accidents per year.

Our project falls within the framework of road monitoring, the main objective of which is the detection and classification of vehicles on images of road traffic.

For this we used two convolutional networks working sequentially where the first phase dedicated to the "Alex Net" (modified) network consists of the detection of vehicles in the whole image. The end of the task of the first network triggers the start of the second (VGG16) pre-trained on 4 classes of vehicles which will then decide the class of each vehicle previously detected by “Alex Net” (modified).

The system put in place is capable of detecting (vehicle, non-vehicle) and classifying vehicles according to their type (bus, car, motorcycle, truck), and presents a solid basis for a future development of the field of road monitoring in Algeria.

Keywords: Vehicle detection, Vehicle classification, Deep Learning, Neural networks, convolutions, ‘Alex Net’, ‘VGG16’.

ملخص

أصبحت السلامة على الطرق السريعة قضية ذات أولوية فائقة بسبب الزيادة الهائلة في حوادث المرور سنويًا حيث نلاحظ تسجيل احصائيات مخيفة في حالات الموت جراء حوادث المركبات خاصة على الطريق السيار، وقد أدى الارتفاع الرهيب في معدل الوفيات سنويا بسبب السرعة بالعلماء الى دق ناقوس الخطر والتشمير عن سواعدهم لإيجاد حل لهذه المعضلة.

التطور التكنولوجي الذي يشهده العالم في مجال الأجهزة الرقمية خاصة في قسم الكاميرات حيث نلاحظ في السنوات القليلة الماضية ولادة العديد من أجيال الكاميرات الرقمية عالية الدقة أدى بالباحثين الى استغلال هذه الوسائل لمراقبة الطرق و لإيجاد حلول عاجلة للحد من الخسائر البشرية.

يدخل مشروعنا هذا في إطار المراقبة الرقمية للطرق السريعة، والهدف الأساسي له هو رصد وتصنيف المركبات في الصور المأخوذة من كاميرات المراقبة على الطرق.

ولهذا قمنا باستعمال شبكتين من الشبكات الالتفافية تعملان تسلسليا في مرحلتين، حيث المرحلة الأولى مخصصة للشبكة المعدلة 'أليكس نات' التي تتكون من الكشف على المركبات في كل الصور.

نهاية المهمة أو المرحلة الأولى يفعل بداية المرحلة الثانية تتألف من شبكة 'في جي جي 16' المعاد تدريبها على أربعة تصنيفات للمركبات، والتي تقوم بعملها اعتمادا على ما سترصده الشبكة السابقة 'أليكس نات' المعدلة.

النظام الذي تتناوله هاته الأطروحة يستطيع رصد المركبات في أية صورة (مركبة أو ليست مركبة) وتصنيفها حسب نوعها (سيارة، حافلة، دراجة نارية، شاحنة).

وعليه يمكننا اعتبار هذا البرنامج أساس لبناء نظام ذكي لمراقبة الطرقات، بالإضافة الى أنه يقدم أساسًا قويًا للتطور المستقبلي لأنظمة مراقبة الطرق السريعة في الجزائر.

الكلمات المفتاحية: رصد المركبات، تصنيف المركبات، التعلم العميق، الشبكات العصبية، الالتفافية، 'أليكس نات'، 'في جي جي 16'.

Sommaire

Liste des figures

page

Liste des tableaux

INTRODUCTION GENERALE

Introduction générale

La sécurité routière et autoroutière est devenue un problème de haute priorité pour les autorités publiques à cause de l'augmentation massive des accidents de la route par année.

La révolution technologique des capteurs et processeurs a engendré la disponibilité d'une large gamme de caméra numérique à haute résolution. Cette disponibilité à induit une orientation des chercheurs vers l'utilisation de ces moyens pour trouver des solutions à divers problèmes de la route tell que la congestion, la mobilité et la sécurité routière.

Notre projet entre dans le cadre du 'monitoring' routier et autoroutier dont l'objectif principale est la détection et la classification des véhicules sur des images du trafic routier ou autoroutier. Pour ce faire nous avons utilisé les réseaux de neurones convolutifs connus par leurs performances dans ce type d'apprentissage.

Pour cela nous avons utilisé deux réseaux convolutifs travaillant séquentiellement où la première phase dédiée au réseau 'Alex Net' (modifié) consiste en la détection des véhicules dans toute l'image. La fin de la tâche du premier réseau déclenche le démarrage du deuxième (vgg16) pré entrainé sur 4 classes de véhicules qui décidera par la suite de la classe de chaque véhicule détecté précédemment par 'Alex Net' (modifié).

Notre mémoire est composée de quatre chapitres :

- **Chapitre I :** fait l'objet d'une étude détaillée sur les systèmes de transport intelligent suivi d'un état de l'Art sur quelques études récentes sur la détection et classification des véhicules.
- **Chapitre II :** consacré à une étude détaillée sur les réseaux de neurones convolutifs et l'apprentissage profond.
- **Chapitre III :** dédié à la concept et l'architecture détaillée du système proposé.
- **Chapitre IV :** les principaux résultats obtenus et leurs interprétations seront détaillés dans ce chapitre.

Et nous clôturons cette étude par une conclusion et des perspectives pour des travaux futurs qui seront élaboré par d'autre étudiants.

Chapitre I :
Transport Intelligent & Détection de véhicules

1. Introduction

Au cours des dernières années une forte augmentation des accidents de la route a été noté. Une augmentation due principalement : à la violation du code de la route, la conduite sous l'influence de stupéfiants, la fatigue et la somnolence, l'excès de vitesse, la congestion …. etc.

Avec le grand développement du secteur des médias automatise, au cours de la dernière décennie, surtout dans le domaine de l'intelligent artificielle ce phénomène peut être réduit en créant des systèmes de transport intelligents(STI) qui interviennent dans plusieurs champs d'activité à savoir l'optimisation de l'utilisation des infrastructures de transport, l'amélioration de la sécurité routière et le respect de l'environnement.

2. Systèmes de transport intelligent

Le système de transport intelligent (STI) est l'application des technologies de détection, d'analyse, de contrôle et de communication au transport terrestre afin d'améliorer la sécurité, la mobilité et l'efficacité. Les STI comprennent une large gamme d'applications qui traitent et partagent des informations pour réduire la congestion, améliorer la gestion du trafic, minimiser l'impact environnemental et augmenter les avantages du transport pour les utilisateurs commerciaux et le public en général.

Les STI ont un effet significatif sur le transport dans des applications telles que la perception des péages électroniques, les compteurs de rampe, les caméras de feux de signalisation, la coordination des feux de circulation, la priorité des signaux de transit et les systèmes d'information des voyageurs.

L'adoption des STI devrait augmenter dans les applications telles que la surveillance de la flotte (fleet monitoring), la gestion des péages, la gestion des billets, la tarification des transports, la télématique et la surveillance du trafic, les principaux bénéficiaires des améliorations de la sécurité des STI ainsi que la disponibilité d'informations et d'analyses en temps réel sont les voyageurs, les entreprises et les agences de transport. Les données des STI ont également des applications de sécurité intérieure. [w1]

3. Quelques applications des systèmes de transport intelligents

Les systèmes intelligents révolutionnent une variété d'industries, notamment le transport et la logistique, la sécurité et la fabrication. Ils contribuent à améliorer l'efficacité énergétique, la qualité et la flexibilité de ces systèmes. Les systèmes intelligents sont complexes et utilisent un

large éventail de technologies - intelligence artificielle, cyber sécurité, traitement du langage naturel, apprentissage en profondeur, processeurs intégrés, stockage distribué, réseau sans fil et signalisation graphique. [w2]

4. Système de transport intelligent de l'Algérie

En raison de la crise du trafic en l'Algérie, en particulier dans les wilayas : Alger, Oran, Constantine, Annaba, les pouvoirs publics ont décidé d'installer des systèmes de contrôle de la circulation des 25 carrefours au niveau de la wilaya d'Alger. En partenariat algéro- espagnole doté d'une enveloppe de 19 millions DA, ce projet dont la réalisation en 3 phases, est prévu sur 55 mois permettra de connaitre le Traffic routier en temps réel, d'améliorer les conditions de déplacement, réduire la durée de voyages et tenir informé les usagers de la route via Internet. [1]

5. Détection de véhicules

5.1. Définition

La détection d'objets est une technique de vision par ordinateur permettant de localiser des instances d'objets dans des images ou des vidéos. Les algorithmes de détection d'objets exploitent généralement l'apprentissage automatique ou l'apprentissage profond pour produire des résultats significatifs. Lorsque les humains regardent des images ou des vidéos, nous pouvons reconnaître et localiser des objets d'intérêt en quelques instants. Le but de la détection d'objets est de reproduire cette intelligence à l'aide d'un ordinateur. [w3]

5.2. Schéma générale de détection de véhicule

La détection de véhicules dans une ou plusieurs images est compose principalement des modules illustrés par la figure (1.1).

5.2.1. Prétraitement

Plusieurs approches de détection de changement sont précédées de prétraitements nécessaires pour la suppression de changement jugés moins importants qui impliquent, généralement, des rectifications géométriques et des réajustements d'intensité. [1]

5.2.2 Détection

La détection de véhicule en mouvement est la segmentation consistant à séparer ou classer les Pixels en deux classes distinctes, l'arrière-plan ‘background’ et l'avant-plan ‘foreground’. Les zones de l'arrière-plan font référence à toute structure ou objet situé dans le champ de vision de la caméra et ne subissant pas de changements au cours du temps tandis que les régions de l’avant-plan correspondent aux objets en mouvement. [1]

5.2.3. Extraction de caractéristiques

Cette étape consiste à extraire les paramètres pertinents représentant les caractéristiques des véhicules et permettant leurs identifications. Récemment beaucoup de méthodes ont été proposées (SURF[2], SIFT[3], NWFE[4],….etc.) pour l’extraction des caractéristiques, mais les plus utilisées en raison de leur fiabilité et de la pertinence des informations porté sont : les caractéristiques ‘Pseudo-Haar’ ‘Haar-Like’, Motif Binaire Local ‘Local Binary Pattern « LBP »’, Transformé de Hough, Analyse en Composantes Principales ‘Principal Component Analysis « PCA »’, Histogramme de Gradient Orienté ‘Histogram of oriented gradients « HOG»’. [1]

5.2.4. Base des modèles d’apprentissage

Représente l’ensemble des exemples positifs (véhicules) et de exemples négatifs
(Non-véhicules) utilisés pour l’apprentissage du classifieur.

5.2.5. Classification

La classification c’est l’étape final dans le processus de détection véhicule, l’objectif de cette étape est classé les objets détectés en deux classes (classe véhicule) ou (classe non véhicule), récemment beaucoup de méthodes ont été proposées (SAR[5], FUZZY[6], Maximum-likelihood[7], …etc.) pour la classification, mais les plus utilisées en raison de leur fiabilité de classification sont : ‘K-means’, séparateurs à vaste marge ‘Support Vector Machine’ et les réseaux de neurones classique et convolutifs. [1]

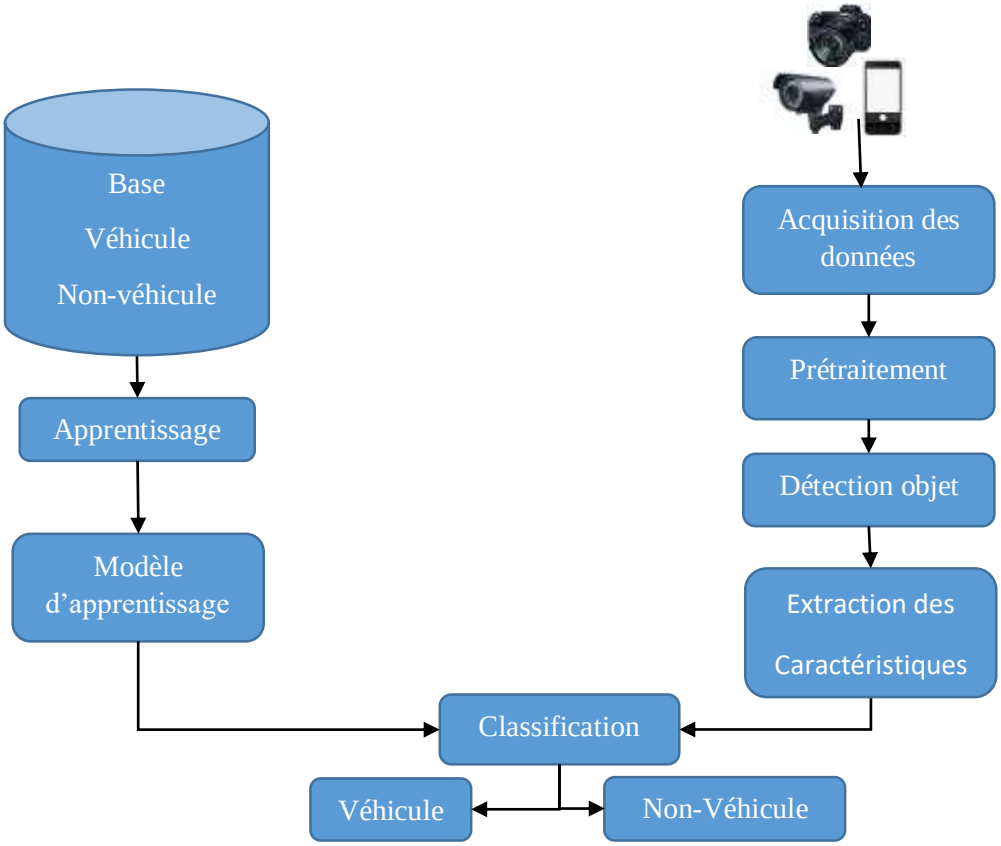


Figure 1.1 : *Schéma générale d'un système de détection des véhicules [1].*

6. Base d'images des véhicules

C'est un jeu de données utilisé dans l'entrainement du classifieur. Nous présentons dans ce qui suit les 'data sets' les plus utilisés dans l'état de l'Art.

6.1. Base 'stanford cars'

L'ensemble de données Cars contient 16 185 images dont 196 classes de voitures. Les données sont divisées en 8 144 images d'entraînement et 8 041 images de test, où chaque classe a été divisée à peu près à 50-50. Les classes sont généralement au niveau de la marque, du modèle, de l'année, ex. Tesla Model S 2012 ou BMW M3 coupé 2012. [w4]

Figure 1.2 : *Base 'Stanford Cars'.*

6.2. Base 'BOXY VEHICLES'

Un grand ensemble de données de détection de véhicules avec près de deux millions de véhicules annotés pour la formation et l'évaluation des méthodes de détection d'objets pour les voitures autonomes sur autoroutes dont les caracteristiques sont illustrés dans le tableau 1.1 [w5]

Nombre des images	véhicules annotés	Résolution	Condition	Caractéristiques
200 000	1 990 000	5 mégapixels	- Soleil - pluie - crépuscule - nuit	- Autoroutes - dégagées - circulation - dense - embouteillages

Tableau 1.1 : *Caractéristiques des véhicules de 'BOXY VEHICLES'.*

Figure 1.3 : *Base 'BOXY VEHICLES'.*

6.3. Base 'TME Motorway'

La base de données d'autoroute Toyota Motor Europe (TME) est composée de 28 clips pour un total d'environ 27 minutes (plus de 30000 images) avec annotation du véhicule. L'annotation a été générée semi-automatiquement à l'aide des données du scanner laser. Les séquences d'images ont été sélectionnées à partir de l'acquisition réalisée sur les autoroutes du nord de l'Italie en décembre 2011. Cette sélection comprend des situations de circulation variables, le nombre de voies, la courbure de la route et l'éclairage, couvrant la plupart des conditions présentes dans l'acquisition complète. [w6]

Figure 1.4 : *Base 'TME Motorway'[s26].*

6.4. Base 'BIT Vehicles'

La base de données BIT-Vehicle contient 9 850 images de véhicules. Il existe des images de tailles 1600 * 1200 et 1920 * 1080 capturées à partir de deux caméras à des moments et à des endroits différents dans l'ensemble de données. Les images contiennent des changements dans les conditions d'éclairage, l'échelle, la couleur de la surface des véhicules et le point de vue. Les parties supérieures ou inférieures de certains véhicules ne sont pas incluses dans les images en raison du délai de capture et de la taille du véhicule. Il peut y avoir un ou deux véhicules

dans une image, de sorte que l'emplacement de chaque véhicule est pré-annoté. L'ensemble de données peut également être utilisé pour évaluer les performances de détection de véhicules. [w7]

Figure 1.5 : *Base 'BIT vehicle'.*

6.5. Base 'GTI Vehicle'

La base de données comprend 3425 images d'arrière de véhicules prises de différents points de vue, et 3900 images extraites de séquences routières ne contenant pas de véhicules. Les images sont sélectionnées pour maximiser la représentativité de la classe de véhicule, ce qui implique une variabilité naturellement élevée.[w8]

Figure 1.6 : *base 'GTI vehicle'.*

6.6. Base ‘CompCar’

L'ensemble de données Comprehensive Cars (CompCars) contient des données de deux scénarios, y compris des images de nature Web et de nature Surveillance. Les données de nature Web contiennent 163 marques de voitures avec 1716 modèles de voitures. Il y a un total de 136726 image capturant les voitures entières et 7618 images capturant les pièces automobiles. Les images complètes de la voiture sont étiquetées avec des cadres et des points de vue. Chaque modèle de voiture est étiqueté avec cinq attributs, y compris la vitesse maximale, le déplacement, le nombre de portes, le nombre de sièges et le type de voiture. Les données de surveillance-nature contiennent 50000 images de voiture capturées en vue de face.[w9]

Figure 1.7 : *Base‘CompCar’.*

6.7. Base ‘Kitti’

Kitti contient une suite de tâches de vision construites à l'aide d'une plate-forme de conduite autonome. Le benchmark complet contient de nombreuses tâches telles que la stéréo, le flux optique, l'odométrie visuelle, etc. Cet ensemble de données contient l'ensemble de données de détection d'objet, y compris les images monoculaires et les cadres de délimitation. L'ensemble de données contient 7481 images d'entraînement annotées avec des cadres de délimitation 3D.[w10]

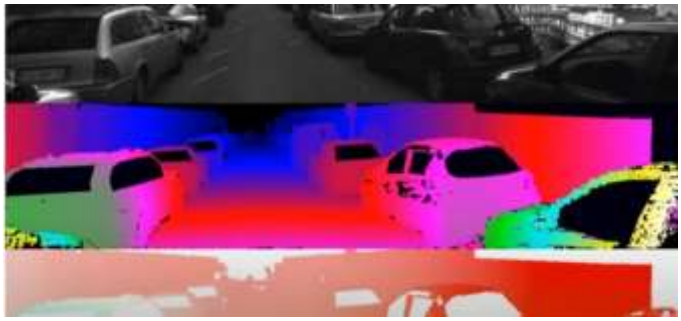


Figure 1.8 : *Base ‘kitti’.*

6.8. Base ‘MOI-TCD vehicle classification’ [w11]

le tableaux suivant contient les noms des classes et le nombre des images dans chaque classe de cet ‘data base’.

car	bus	bicycle	motocycle	pedestrian	Pickup truck	Non motorize vehicle	work van	articulated truck	Non motorized vehicle	Single unit truck
6238	5072	2254	1952	6232	6260	1721	5425	4961	1721	5090

Tableau 1.2 : *Nombre de vehicules de chaque classe ‘MOI-TCD’.*

Figure 1.9 : *base ‘MOI-TCD’.*

	Nom de base	Taille de base	Extension	Référence
01	Véhicules de Stanford	16185	jpg	[w4]
02	BOXY VEHICLES	200000	jpg	[w5]
03	TME Motorway	28 clips 1 clips = (30000 frame)	Mp4	[w6]
04	BIT vehicle	9850	jpg	[w7]
05	GTI Vehicle image	7 325	jpg	[w8]
06	CompCar	136 726	jpg	[w9]
07	Kitti	7481	jpg	[w10]
08	MOI-TCD	46926	jpg	[w11]

Tableau 1.3 : *Tableau repulatif de quelques bases de véhicules.*

7. Quelques travaux sur la détection des véhicules

Dans le cadre de notre projet nous avons fait l'étude de quelques travaux récents de la litérature scientifique afin de positionne notre travail.

7.1. Article 1 : Détection et classification des véhicules par apprentissage profond 'Detection and classification of vehicles using deep learning' [8]

7.1.1. Résumé

Dans cet article, les auteurs se concentrent sur la détection et la reconnaissance des véhicules à partir d'un flux vidéo. Comparées aux méthodes traditionnelles de détection et de classification d'objets, les méthodes d'apprentissage en profondeur sont un nouveau concept dans le domaine de la vision par ordinateur. Ce modèle fonctionne en deux temps : une étape de préparation des données, il consiste à appliquer des Traitements sur les images composant l'ensemble de données afin d'en extraire les caractéristiques, la deuxième étape est d'appliquer le concept de réseaux de neurones convolutifs pour classer les véhicules.

7.1.2. Base de donnée utilisée

L'ensemble de données utilisé comprend deux fichiers (fichier véhicule et fichier non véhicule). Cette base de données contient des images extraites de séquences vidéo (obtenues par une caméra frontale montée sur une voiture). Pour assurer un bon apprentissage des données, les images sont capturées dans différentes conditions de route (loin, près, gauche, droite). Le fichier véhicule comprend 8798 images et le fichier non véhicule comprend 8971 images. Chaque image est de dimension : (64x64) pixels.

7.2. Article 2 : Détection et classification des véhicules basés sur le réseau de neurones profonds pour les applications de transport intelligent

'Vehicle Detection and Classification based on Deep Neural Network for Intelligent Transportation Applications' [9]

7.2.1. Résume

Cet article propose une méthode optimisée de détection et de classification des véhicules basée sur une technologie d'apprentissage en profondeur pour les applications de transport intelligent. Les auteurs optimisent l'architecture CNN (Convolutional Neural Network) en peaufinant l'architecture CNN existante pour les applications de transport intelligent. La conception proposée (CNN) atteint un taux de détection de faux positifs autour de 10% lorsque FPPI (False

Positive Per Image) est 0,1. Réalisé sur le GPU nVidia Titan-X, le design proposé peut atteindre les performances de la vidéo 720x480 dans différentes conditions météorologiques (jour, nuit, pluie) à 25fps (frame per sec).

Le modèle proposé peut atteindre une précision de 90% sur trois catégories de véhicules cibles, y compris les petits véhicules (berline, VUS, fourgonnette), les gros véhicules (bus) et les camions.

Name	Type	Stride
Conv1	7x7 C.ReLU	2
Pool1	3x3 max-pool	2
Conv2_1	3x3 C.ReLU	1
Conv2_2	3x3 C.ReLU	1
Conv2_3	3x3 C.ReLU	1
Conv3_1	3x3 C.ReLU	2
Conv3_2	3x3 C.ReLU	1
Conv3_3	3x3 C.ReLU	1
Conv3_4	3x3 C.ReLU	1
Conv4_1	Modified Inception	2
Conv4_2	Modified Inception	1
Conv4_3	Modified Inception	1
Conv4_4	Modified Inception	1
Conv5_1	Modified Inception	2
Conv5_2	Modified Inception	1
Conv5_3	Modified Inception	1
Conv5_4	Modified Inception	1
Downscale	3x3 max-pool	2
Upscale	4x4 deconv	2
Concat	Concat	X
Convf	1x1 conv	1

Tableau 1.4 : *Le concept du CNN proposé.*

7.2.2. Base de donnée utilisé

Les ensembles de données sur les véhicules que les auteurs ont utilisés pour former le modèle proposé sont IVS-1 et IVS-2 constitues respectivement de 316733 véhicules et de 599277 véhicules.

7.3. Article 3 : Un algorithme de détection de véhicules routiers basé sur un réseau de neurones convolutionnels

‘An algorithm for highway vehicle detection based on convolutional neural network’ [10]

7.3.1. Résume

Dans cet article, les auteurs présentent un système efficace de détection et de classification des véhicules à partir de caméras de surveillance du trafic. Tout d'abord, ils ont regroupé les échelles et les proportions des véhicules dans les jeux de données des véhicules. Ils utilisent ensuite le réseau neuronal à convolution ‘CNN’ pour détecter un véhicule. Ils utilisent des techniques de fusion de fonctionnalités pour concaténer des fonctionnalités de haut niveau et des fonctionnalités de bas niveau et détecter différentes tailles de véhicules sur différentes fonctionnalités.

Afin d'améliorer la vitesse, ils adoptent naturellement une architecture entièrement convolutive au lieu de couches entièrement connectées ‘FC’. En outre, des avancées complémentaires récentes telles que ‘batch-norm’, ‘hard example mining’ et ‘inception’ ont été adoptées. Des expériences approfondies sur « JiangSuHighway Dataset » (JSHD) démontrent les performances compétitives de cette méthode.

Le système obtient une amélioration significative par rapport au Faster R-CNN de 6,5% de (mAP : mean average precision). Avec une mémoire GPU de 1,5 G en phase de test, la vitesse du réseau est de 15 FPS (frame per sec), trois fois plus rapide que le Faster R-CNN.

Models	Overall (mAP)	Car	Bus	Minibus	Truck
Fast R-CNN	67.2	53.6	83.2	62.5	69.5
Faster R-CNN	69.2	55.2	85.4	64.7	71.3
YOLO	58.9	46.6	75.2	53.4	60.5
SSD300	68.8	54.4	85.1	64.3	71.5
SSD512	71.2	57.4	87.2	66.8	73.4
RON320 [25]	73.6	60.2	89.5	69.1	75.5
cet article	75.7	62.4	91.8	71.3	77.3

Tableau 1.5 : *résultat de comparaison.*

7.3.2. Base de donnée utilisé

La base de données construite à partir de 25 vidéos de l'autoroute JiangSu (une ville dans la chine), province du Jiangsu, appelé JSHD, elle contient 5000 images capturées à partir des vidéos.

Chaque véhicule est classé en quatre catégories (bus, minibus, voiture, camion).

7.4. Article 4 : Système de détection de véhicule et d'identification de type de voiture utilisant l'apprentissage profond et le véhicule aérien sans pilote

‘Vehicle Detection and Car Type Identification System using Deep Learning and Unmanned Aerial Vehicle’ [11]

7.4.1. Résume

Cet article propose la conception et la mise en œuvre d'une méthode de détection et d'identification de type de véhicules à l'aide du Deep Learning et d'UAV (*Unmanned Aerial Vehicle*) dans des images vidéo 4K UHD (ultra-haute définition,>= 3840 pixels).

La présente étude propose la méthode de mise en œuvre d'un système de détection et de classification qui peut être accompli pour la classification des véhicules selon le plan d’AUSTROADS. Le système proposé est basé sur deux processus principaux : la détection et la classification.

Cet article présente la méthode de formation, de détection et de classification des données du véhicule en appliquant un Darknet-53 pour les véhicules trouvés dans les images UHD (ultra HD). L’auteur propose également la méthode de classification variable en fonction des voitures stationnées et arrêtées pour la surveillance des flux de trafic. Il a donc considéré les trois conditions de conduite, d'arrêt et de stationnement.

Les résultats de l'expérience montrent que l'approche proposée a entraîné des erreurs deux fois plus faibles que les méthodes conventionnelles qui utilisent une zone de recherche fixe.

7.4.2. Base de donnée utilisée

Les images aériennes utilisées dans cet article sont des images de véhicules sur la route, enregistrées à moins de 120 m d'altitude à l'aide du drone (Phantom3 Professional de DJI). La taille des images expérimentales est de 3940 × 2160, 30 FPS en résolution UHD (ultra HD).

7.5. Article 5 : Détection de véhicules à l'aide de 3D-LIDAR et ConvNet

‘Vehicle Detection Using 3D-LIDAR and ConvNet’ [12]

7.5.1. Résumé

Cet article aborde le problème de la détection des véhicules en utilisant le réseau neuronal à convolution profonde (ConvNet) et les données 3D-LIDAR (détecteur de mouvement des objets) avec une application dans les systèmes avancés d'aide à la conduite et la conduite autonome. Un système de détection de véhicules basé sur les paradigmes de génération d'hypothèses (HG) et de vérification (HV) est proposé. Les données entrées dans le système sont un nuage de points obtenu à partir d'un 3D-LIDAR monté à bord d'un véhicule instrumenté, qui est transformé en une carte de profondeur dense (DM). La solution proposée commence par la suppression des points au sol, suivie de la segmentation des nuages de points. Ensuite, des obstacles segmentés (hypothèses d'objet) sont projetés sur le DM. Les boîtes englobantes sont ajustées aux objets segmentés comme hypothèses de véhicule (étape HG). Enfin, les cadres de délimitation sont utilisés comme entrées dans un ConvNet pour classer / vérifier les hypothèses d’appartenance à la catégorie « véhicule » (étape HV). Dans cet article, les auteurs présentent une évaluation de ConvNet utilisant des DM basés sur LIDAR ainsi que l'impact de l'augmentation des données spécifiques au domaine sur les performances de détection des véhicules. Pour former et évaluer le système de détection de véhicules proposé, le KITTI Benchmark Suite a été utilisé.

7.5.2. Base de donnée utilisée (KITTI Dataset for DM-based Vehicle Detection page 21**).**

8. Conclusion

Ces dernières années, Les système intelligent sont devenus une partie de notre quotidien. Il nous donne des solutions à certains problèmes comme l’amélioration de la sécurité routière, économie d'énergie électrique, la gestion de la circulation ……etc.

La détection des véhicules est le cœur des systèmes de transport intelligent car tout le processus en dépend. Il est donc important d’étudier attentivement la fiabilité de cette partie du système.

Chapitre II : Réseaux du Neurones Convolutifs et Apprentissage Profond

1. Réseaux de neurones

1.1. Introduction

Un réseau de neurones artificiel est un système de calcul qui tente d'imiter ou du moins s'inspire de connections neuronales de notre système nerveux. Les réseaux de neurones artificiels sont également appelés réseaux de neurones.

1.2. Système de neurone artificiel

Pour qu'un système soit considéré comme un NN (neural network), il doit contenir une structure de graphe orientée étiquetée où chaque nœud du graphe effectue un calcul simple. D'après la théorie des graphes, un graphe orienté se compose d'un ensemble de nœuds (c'est-à-dire de sommets) et d'un ensemble de connexions (c'est-à-dire d'arêtes) qui relient des paires de nœuds. Dans (figure 2.1). Chaque nœud effectue un calcul simple. Chaque connexion transporte alors un signal (c'est-à-dire la sortie du calcul) d'un nœud à un autre, étiqueté par un poids indiquant la mesure dans laquelle le signal est amplifié ou diminué. Certaines connexions ont des poids positifs importants qui amplifient le signal, indiquant que le signal est très important lors de la classification. D'autres ont des poids négatifs, diminuant la force du signal, spécifiant ainsi que la sortie du nœud est moins importante dans la classification finale. Un système est un réseau neuronal artificiel s'il consiste en une structure de graphe (comme dans la figure 2.1) avec des poids de connexion modifiables à l'aide d'un algorithme d'apprentissage. [13]

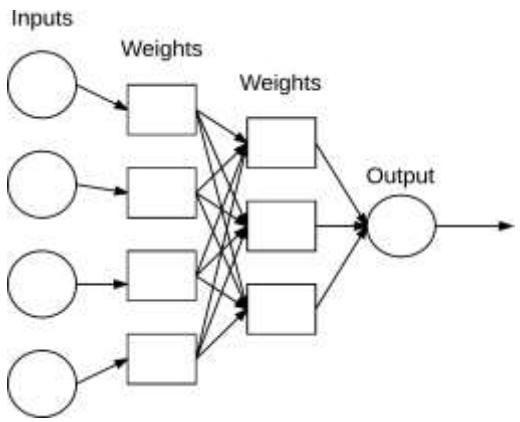


Figure 2.1 : *Architecture de réseau neuronal simple.*

1.3. Relation avec la biologie

Nos cerveaux sont composés d'environ 10 milliards de neurones, chacun étant connecté à environ 10 000 autres neurones. Le corps cellulaire du neurone est appelé le soma, où les entrées (dendrites) et les sorties (axones) connectent le soma à d'autres soma (Figure 2.2).

Chaque neurone reçoit des entrées électrochimiques d'autres neurones au niveau de leurs dendrites. Si ces entrées électriques sont suffisamment puissantes pour activer le neurone, alors le neurone activé transmet le signal le long de son axone, le transmettant aux dendrites d'autres neurones. Ces neurones attachés peuvent également se déclencher, poursuivant ainsi le processus de transmission du message.

La clé à retenir ici est qu'un neurone ne se déclenchera que si le signal total reçu au niveau du soma dépasse un seuil donné.

Cependant, gardez à l'esprit que les ANN sont simplement inspirés par ce que nous savons sur le cerveau et son fonctionnement. [13]

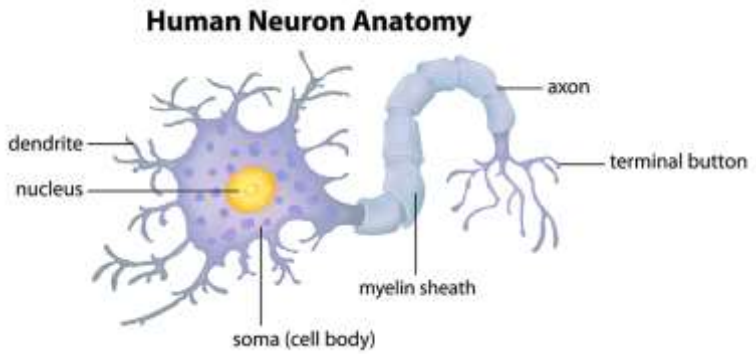


Figure 2.2 : *Structure d'un neurone biologique.*

1.4. Réseaux de neurones artificiels

Un réseau de neurones artificiels peut être modéliser par le graphe de la (figure 2.3) ou les valeurs x1, x2, et x3 sont les entrées de notre NN et correspondent généralement à une seule ligne (c'est-à-dire, point de données) de notre matrice de conception. La valeur constante 1 est notre biais qui est supposé être intégré dans la matrice de conception. Nous pouvons considérer ces entrées comme les vecteurs de caractéristiques d'entrée du NN.

En pratique, ces entrées pourraient être des vecteurs utilisés pour quantifier le contenu d'une image de manière systématique et prédéfinie (par exemple, histogrammes de couleur, histogramme de gradient orientés……. etc.). Dans le cadre du 'deep learning', ces entrées sont les intensités de pixels brutes des images elles-mêmes.

Chaque x est connecté à un neurone via un vecteur de poids **W** constitué de (w1, w2.... wn), ce qui signifie que pour chaque entrée x nous avons également un poids associé w.

Enfin, le nœud de sortie à droite de la Figure 4 prend la somme pondérée, applique une fonction d'activation f (utilisée pour déterminer si le neurone «se déclenche » ou non) et délivre une valeur. [13]

L'expression mathématique de la sortie rencontre généralement les trois formes suivantes :

$$f(w_1x_1 + w_2x_2 + \cdots + w_nx_n) \qquad (2.1)$$

$$f(\sum_{i=1}^{n} w_i\, x_i) \qquad (2.2)$$

$$ou\ simplement\ f(net), net = \sum_{i=1}^{n} w_i\, x_i \qquad (2.3)$$

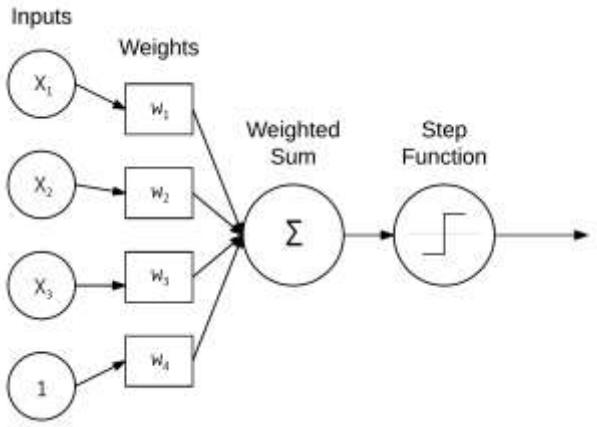


Figure 2.3 : *Un RNA simple qui prend la somme pondérée de l'entrée x et des poids w.*

1.5. Fonctions d'activation [13]

1.5.1. Fonction à seuil 'step'

La fonction d'activation la plus simple est la 'step function' (figure 2.5.a)

Il s'agit d'une fonction de seuil très simple. Si la somme pondérée $\sum_{i=1}^{n} w_i \; x_i > 0$, nous sortons 1, sinon, nous sortons zeros.

$$f(net) = \begin{cases} 1, & if \; net > 0 \\ 0, & else \end{cases} \qquad (2.4)$$

1.5.2. Fonction 'sigmoid'

La fonction d'activation la plus courante utilisée dans l'histoire de la littérature NN est la fonction sigmoïde (Figure 2.5.b), décrite par l'équation suivante :

$$t = \sum_{i=1}^{n} w_i \, x_i \quad (2.5) \qquad s(t) = 1/(1 + e^{-t}) \quad (2.6)$$

1.5.3. La tangente hyperbolique

La tangente hyperbolique, ou tanh (avec une forme similaire du sigmoide) a également été largement utilisée comme fonction d'activation jusqu'à la fin des années 1990 (Figure 2.5.c) L'équation de tanh est la suivante :

$$f(z) = \tanh(z) = (e^{z} - e^{-z})/(e^{z} + e^{-z}) \quad (2.7)$$

1.5.4. Fonction 'ReLU'

'ReLU' sont également appelées (fonctions de rampe) en raison de leur apparence lorsqu'elles sont tracées (Figure 2.5.d). Fonction est nulle pour les entrées négatives, mais augmente ensuite linéairement pour les valeurs positives. La fonction 'ReLU' n'est pas saturable et est également extrêmement efficace en termes de calcul. Empiriquement, la fonction d'activation 'ReLU' a tendance à surpasser les fonctions sigmoïde et tanh dans presque toutes les applications.

1.5.5. Fonction ‘Leaky ReLUs’

Une variante des ‘ReLUs’ (figure 2.5.e), appelée ‘Leaky ReLUs’ permet un petit gradient non nul lorsque l'unité n'est pas active :

$$f(net) = \begin{cases} net, & if\ x \geq 0 \\ \alpha * net, & otherwise \end{cases} \quad (\alpha = 0.3) \qquad (2.8)$$

1.5.6. Fonction Unités linéaires exponentielles ‘Exponential Linear Units ELUs’

La valeur de α est constante et définie lorsque l'architecture réseau est instanciée. Une valeur typique pour α est 1.0 (figure 2.5.f).

$$\mathrm{f(net)} = \begin{cases} \mathrm{net}, & \mathrm{if\ net} \geq 0 \\ \alpha * (\exp(\mathrm{net}) - 1), & \mathrm{otherwise} \end{cases} \qquad (2.9)$$

1.5.7. Fonction ‘Softmax’ :

Nous utilisons ‘softmax’ comme fonction de sortie de la dernière couche dans les réseaux neuronaux (si le réseau a n couches, la n-i ème couche est la fonction softmax). [w12]

$$softmax(z_j) = \frac{e^{z_j}}{\sum_{k=1}^{k} e^{z_k}}\ Pour\ j = 1, \ldots\ldots, k \quad (2.10)$$

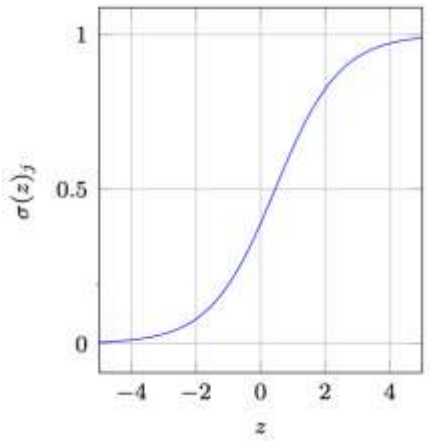


Figure 2.4: *graphe de ‘softmax’.*

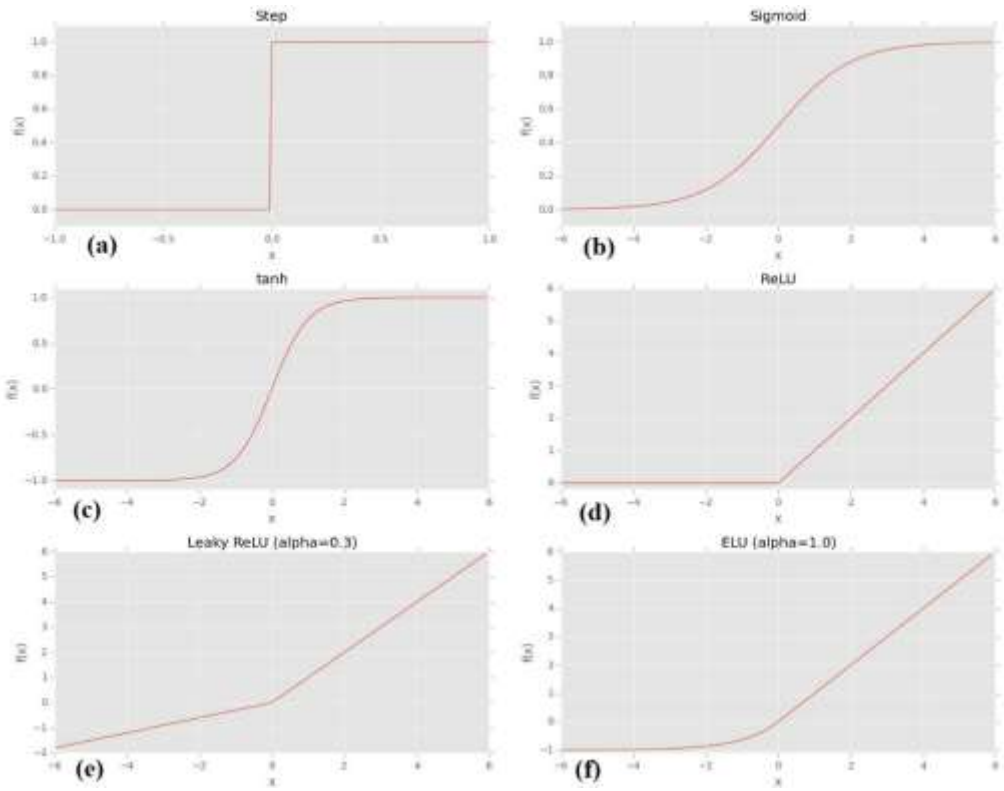


Figure 2.5 : *Fonctions d'activation [13].*

1.6. Architectures des réseaux de neurones

Il existe de très nombreuses architectures NN (neural network) différentes, l'architecture la plus courante étant le réseau à réaction, comme le montre la (figure 2.6).

Dans ce type d'architecture, une connexion entre nœuds n'est autorisée que depuis les nœuds de la couche (layer) i vers les nœuds de la couche (layer) i + 1. Aucune connexion arrière ou inter-couche (inter-layer) n'est autorisée. Lorsque les réseaux à action directe incluent des connexions de rétroaction (connexions de sortie qui alimentent les entrées), ils sont appelés réseaux neuronaux récurrents. [13]

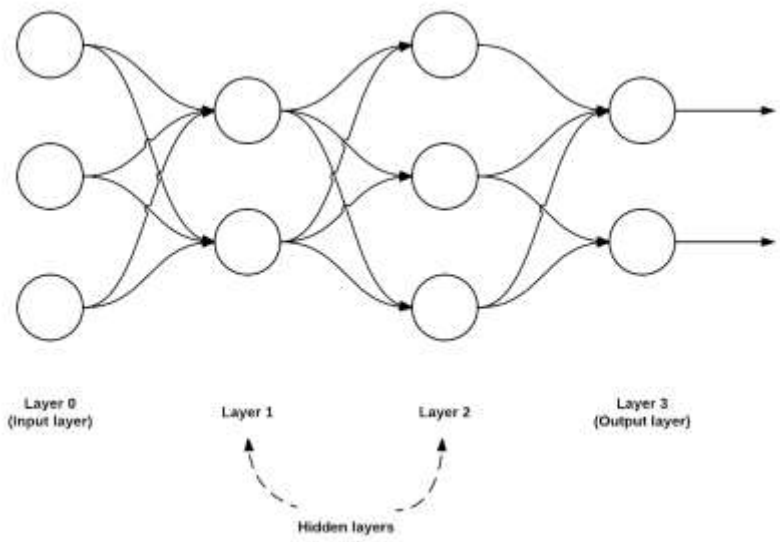


Figure 2.6 : *Un exemple de réseau neuronal (NN) à anticipation.*

1.7. Réseaux de neurones à réaction 'feed-forward'

Un réseau de neurones à rétroaction (feed-forward) permet aux informations de circuler uniquement dans le sens aller, des nœuds d'entrée, à travers les couches cachées et vers les nœuds de sortie. Il n'y a pas de cycles ou de boucles dans le réseau (figure 2.7).

Dans un réseau neuronal à réaction directe (feed-forward), les décisions sont basées sur l'entrée actuelle. Il ne mémorise pas les données passées et il n'ya pas de portée future. Les réseaux de neurones à réaction directe sont utilisés dans les problèmes généraux de régression et de classification. [w13]

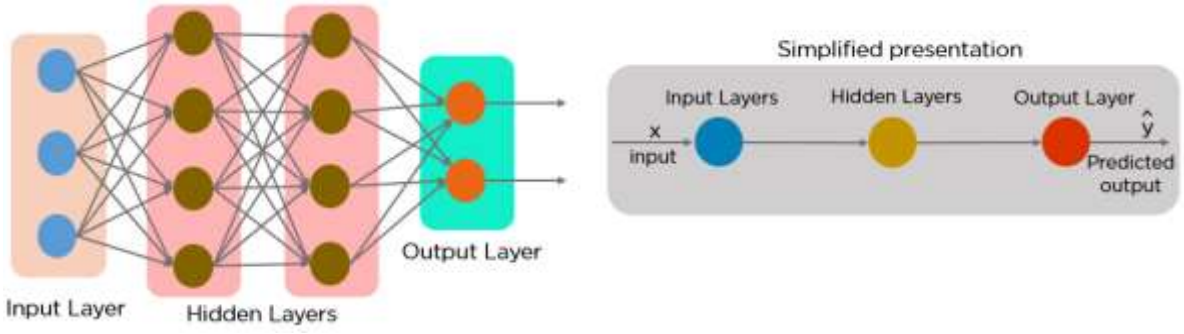


Figure 2.7: *Réseaux de neurones à réaction.*

1.8. Réseaux de neurones récurent

Un réseau neuronal récurrent fonctionne sur le principe de sauvegarder la sortie d'une couche particulière et de la renvoyer à l'entrée afin de prédire la sortie de la couche.

Les nœuds des différentes couches (layers) du réseau neuronal sont compressés pour former une seule couche de réseaux neuronaux récurrents. A, B et C sont les paramètres du réseau (figure 2.8)

Ici, « x » est la couche d'entrée, « h » est la couche cachée et « y » est la couche de sortie. A, B et C sont les paramètres réseau utilisés pour améliorer la sortie du modèle. A tout instant t donné, l'entrée courante est une combinaison d'entrée à x (t) et x (t-1). La sortie à un moment donné est récupérée sur le réseau pour améliorer la sortie. (Figure 2.9) [w13]

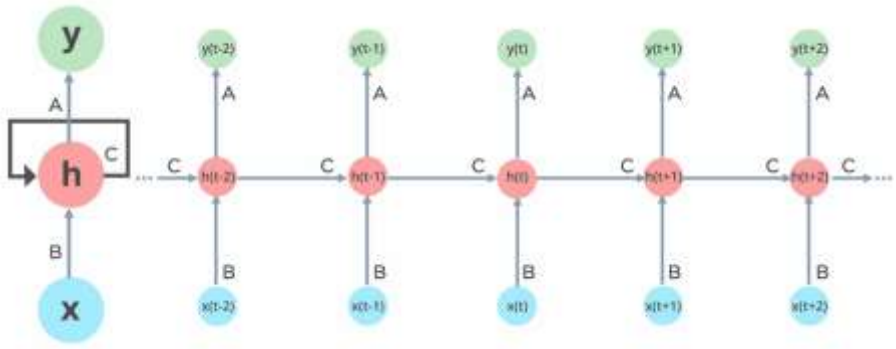


Figure 2.8: *réseau de neurons recurrent 'Recurrent Neural Network'.*

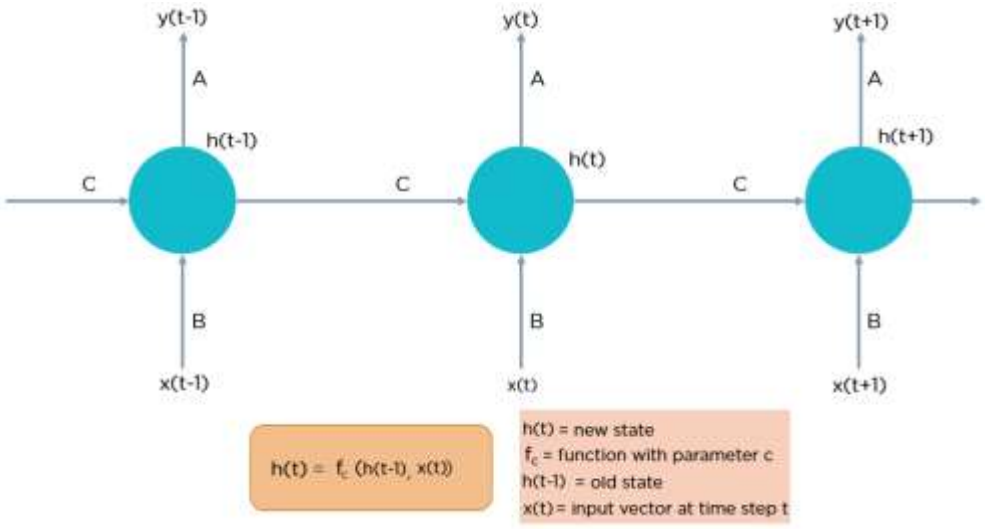


Figure 2.9*: explication réseau de neurons recurrent 'Recurrent Neural Network'.*

1.9. Différence entre ‘reccurent neural network’ et ‘feed-forward neural network’

Des réseaux de neurones récurrents ‘reccurent neural network’ ont été créés en raison de quelques problèmes dans le réseau de neurones à réaction (feed-forward neural network) :

- Impossible de gérer les données séquentielles
- Ne prend en compte que l'entrée actuelle
- Impossible de mémoriser les entrées précédentes

La solution à ces problèmes est le réseau neuronal récurrent (RNN). Un RNN peut gérer des données séquentielles, accepter les données d'entrée actuelles et les entrées reçues précédemment. Les RNN peuvent mémoriser les entrées précédentes grâce à leur mémoire interne. [w13]

2. L’apprentissage automatique et l’apprentisage profond ‘Machine learning and deep learning’

2.1. Introduction

L'apprentissage profond est un sous-domaine de l'apprentissage automatique, qui est à son tour un sous-domaine de l'intelligence artificielle ‘IA’ (figure 2.10) L'objectif central de l'IA est de fournir un ensemble d'algorithmes et de techniques qui peuvent être utilisés pour résoudre des problèmes que les humains exécutent de manière intuitive et quasi automatique, mais qui sont par ailleurs très difficiles pour les ordinateurs. Un excellent exemple d'une telle classe de problèmes d'IA est l'interprétation et la compréhension du contenu d'une image - cette tâche est quelque chose qu'un humain peut faire avec peu ou pas d'effort, mais il s'est avéré extrêmement difficile à accomplir pour les machines.

Alors que l'IA incarne un ensemble important et diversifié de travaux liés au raisonnement automatique de la machine (inférence, planification, heuristique, etc.), le sous-champ d'apprentissage automatique a tendance à être spécifiquement intéressé par la reconnaissance des formes et l'apprentissage à partir des données.

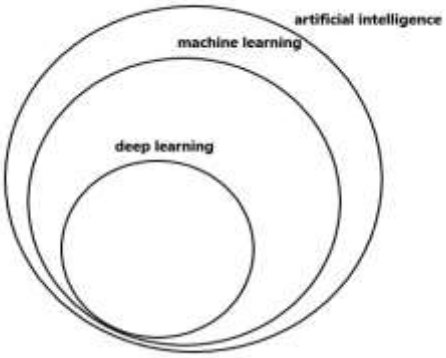


Figure 2.10 : *Relation entre 'AI, ML, DL'.*

2.2. Apprentissage automatique 'Machine Learning'

Les algorithmes d'apprentissage automatique sont des algorithmes mathématiques qui permettent aux machines d'apprendre en imitant la façon dont les humains apprennent, bien que l'apprentissage automatique ne soit pas seulement des algorithmes, c'est aussi l'approche à partir de laquelle le problème est abordé. L'apprentissage automatique est essentiellement un moyen d'obtenir de l'intelligence artificielle. [w14]

2.3. Apprentissage profond 'deep learning'

L'apprentissage profond 'DL' fait partie de l'apprentissage automatique. En fait, il peut être décrit comme la nouvelle évolution de l'apprentissage automatique. C'est un algorithme automatique qui imite la perception humaine inspirée de notre cerveau et de la connexion entre les neurones.

'Deep Learning' est la technique qui se rapproche le plus de la façon dont les humains apprennent.

La plupart des méthodes d'apprentissage en profondeur utilisent une architecture de réseau neuronal. C'est pourquoi l'apprentissage en profondeur est souvent appelé "réseaux de neurones profonds". Il est appelé "profond" en référence aux couches que ces réseaux de neurones ont. [w14]

2.4. Diférence entre 'Machine Learning' et 'deep learning'

L'apprentissage automatique et l'apprentissage profond imitent la façon dont le cerveau humain apprend. Sa principale différence réside donc dans le type d'algorithmes utilisés dans chaque cas, bien que l'apprentissage profond soit plus similaire à l'apprentissage humain car il fonctionne avec des neurones. L'apprentissage automatique utilise généralement des arbres de décision et des

réseaux de neurones d'apprentissage en profondeur, qui sont plus évolués. De plus, les deux peuvent apprendre de manière supervisée ou non. [w14]

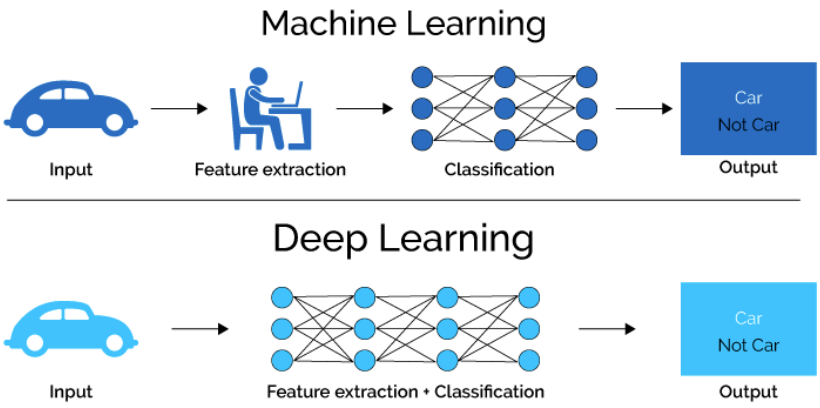


Figure 2.11 : *Apprentissage automatique et apprentissage profond.*

2.5. Réseau de neurones à convolution 'Convolution neural network CNN'

2.5.1. Convolution

En mathématiques, la convolution est une opération effectuée sur deux fonctions (f et g) pour produire une troisième fonction. La convolution est l'une des opérations les plus importantes du traitement du signal et de l'image. Il peut fonctionner en 1D (par exemple, traitement de la parole), 2D (par exemple, traitement d'image) ou 3D (traitement vidéo). [w15]

2.5.2. Convolution sur image

Dans le traitement d'image, la convolution est le processus de transformation d'une image en appliquant un noyau sur chaque pixel et ses voisins locaux sur toute l'image. Le noyau est une matrice de valeurs dont la taille et les valeurs déterminent l'effet de transformation du processus de convolution.

Le processus de convolution implique ces étapes :

(1) Il place la matrice du noyau sur chaque pixel de l'image (en s'assurant que le noyau complet est dans l'image), multiplie chaque valeur du noyau par le pixel correspondant sur lequel il se trouve.

(2) Ensuite, additionne les valeurs multipliées résultantes et renvoie la valeur résultante en tant que nouvelle valeur du pixel central.

(3) Ce processus est répété sur toute l'image. [w16]

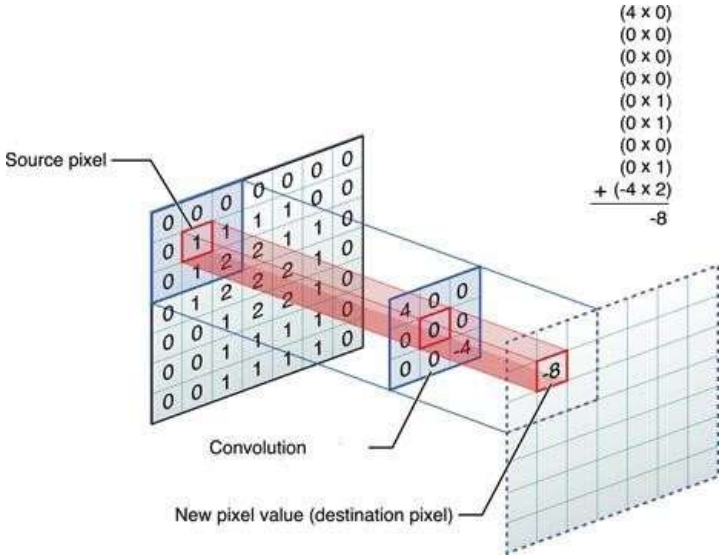


Figure 2.12 : *Illustration de l'opération de convolution.*

2.5.3. Qu'est-ce que le réseau neuronal convolutif (CNN ou ConvNet) ?

Convolutional Neural Network est un algorithme d'apprentissage en profondeur qui est utilisé pour reconnaître les images. Cet algorithme regroupe les images par similitude et effectue la reconnaissance d'objets dans les scènes. CNN utilise une caractéristique unique des images (par exemple la queue et les oreilles du chat, l'aile et le moteur de l'avion, etc.) pour identifier l'objet placé sur l'image. En fait, ce processus est très similaire à ce que fait notre cerveau pour identifier les objets. [w16]

2.5.3.1. Classification d'images avec réseaux de neurones à convolution

Les classifications d'images CNN prennent une image d'entrée, la traitent et la classent dans certaines catégories (par exemple, chien, chat, tigre, lion). Les ordinateurs voient une image d'entrée comme un tableau de pixels et cela dépend de la résolution de l'image. En fonction de la résolution de l'image, il verra h x l x d (h = hauteur, w = largeur, d = dimension). Par exemple, une image d'un tableau 6 x 6 x 3 de matrice RGB (3 se réfère aux valeurs RGB) et une image d'un tableau 4 x 4 x 1 de matrice d'image en niveaux de gris (figure 2.13).

Techniquement, l'apprentissage en profondeur des modèles CNN pour former et tester, chaque image d'entrée la passera à travers une série de couches de convolution avec des filtres (noyaux), regroupement, couches entièrement connectées (FC) et appliquera la fonction Softmax pour classer un objet avec des valeurs probabilistes comprises entre 0 et 1. (La figure 2.14) est un flux complet de CNN pour traiter une image d'entrée et classer les objets en fonction des valeurs.

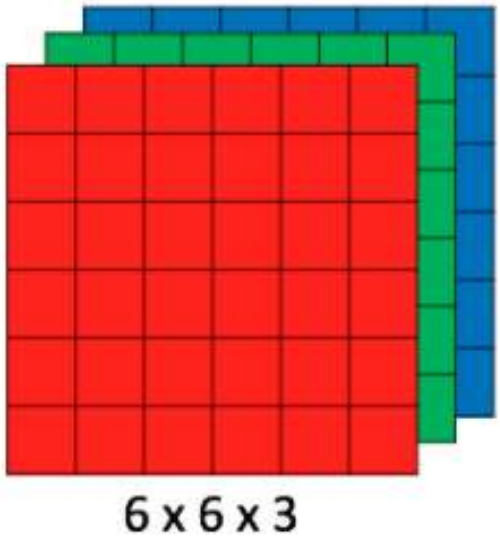


Figure 2.13 : *Tableau de matrice RVB (rouge, vert, bleu).*

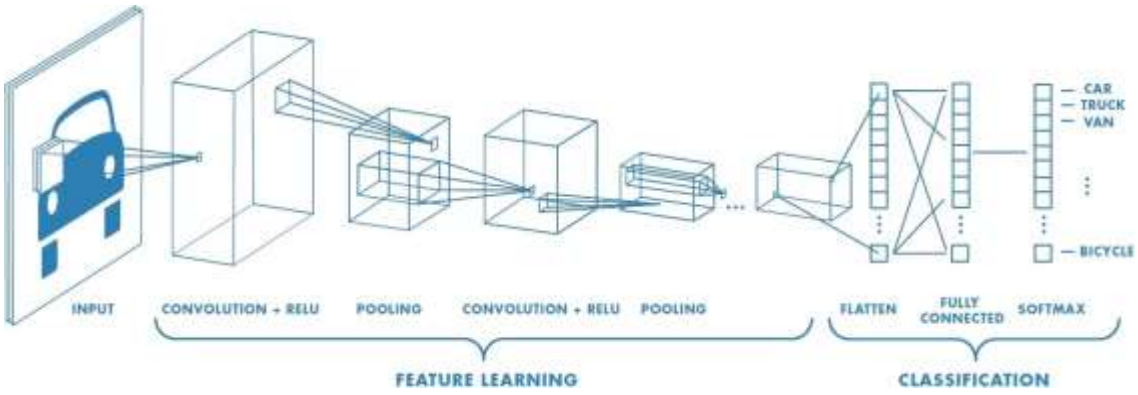


Figure 2.14 : *Réseau de neurones avec de nombreuses couches convolutifs.*

2.5.3.2. Couche de convolution ‘convolution layer’

La convolution est la première couche à extraire des entités d'une image d'entrée. La convolution préserve la relation entre les pixels en apprenant les caractéristiques de l'image à l'aide de petits carrés de données d'entrée. C'est une opération mathématique qui prend deux entrées telles qu'une matrice d'image et un filtre ou un noyau.

- Matrice d’image (volume) de dimension (h * w * d).
- Filtre (f_h * f_w * d).
- Sortie de volume de dimension (h – f_h + 1) * (w –f_w +1) * 1.

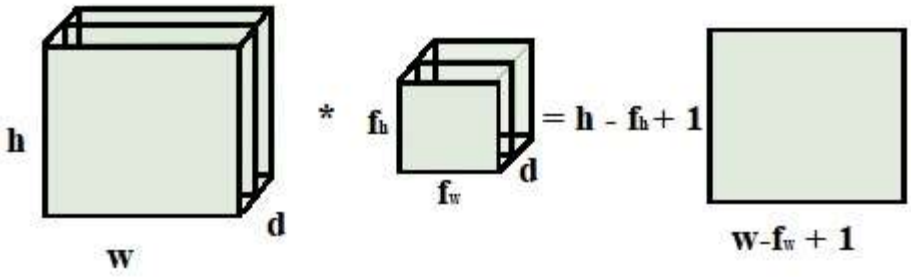


Figure 2.15 : *La matrice d'image multiplie le noyau ou la matrice de filtre.*

Considérons un 5 x 5 dont les valeurs de pixel d'image sont 0, 1 et la matrice de filtre 3 x 3 comme indiqué ci-dessous :

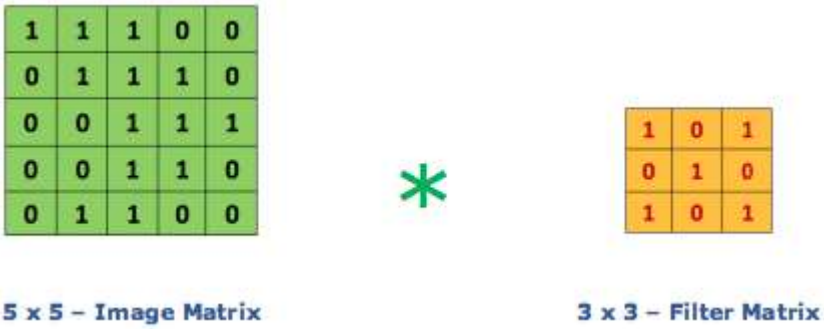


Figure 2.16 : *La matrice d'image multiplie le noyau ‘kernel’ ou la matrice de filtre.*

Ensuite, la convolution de la matrice d'image 5 x 5 multipliée par la matrice de filtre 3 x 3 qui est appelée « Carte des caractéristiques » comme la sortie montrée ci-dessous

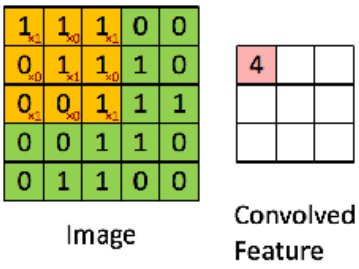


Figure 2.17 : *Matrice de sortie 3 x 3.*

La convolution d'une image avec différents filtres peut effectuer des opérations telles que la détection des contours, le flou et la netteté en appliquant des filtres. L'exemple ci-dessous montre diverses images de convolution après avoir appliqué différents types de filtres (noyaux).

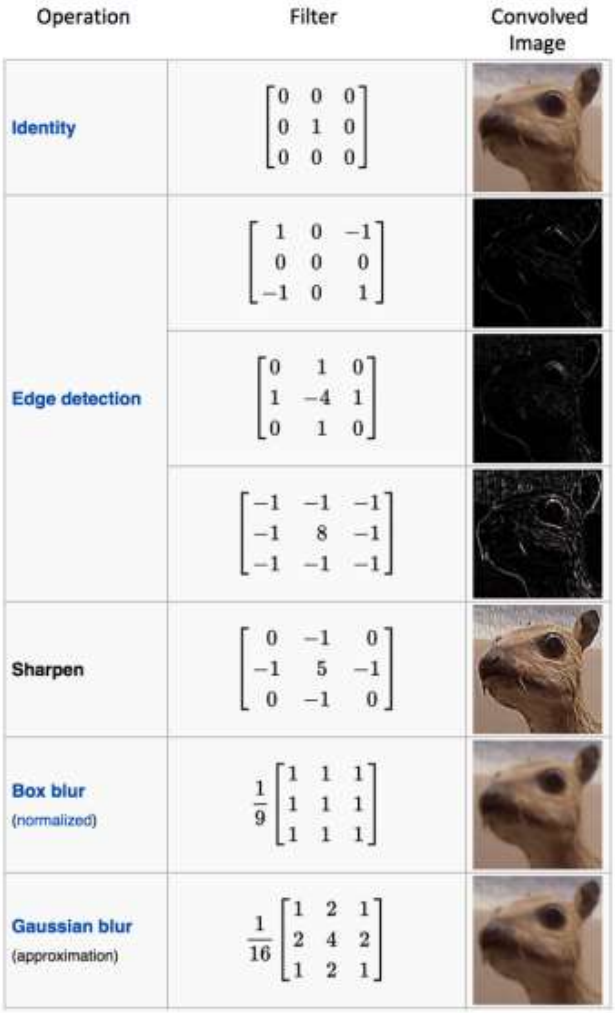


Figure 2.18 : *Quelques filtres courants [14].*

2.5.3.3. Le pas ‘Strides’

Le pas ‘stride’ est le nombre de pixels décalés sur la matrice d'entrée. Lorsque le pas est de 1, nous déplaçons les filtres sur 1 pixel à la fois. Lorsque le pas est de 2, nous déplaçons les filtres sur 2 pixels à la fois et ainsi de suite. La figure ci-dessous montre que la convolution fonctionnerait avec un pas de 2.

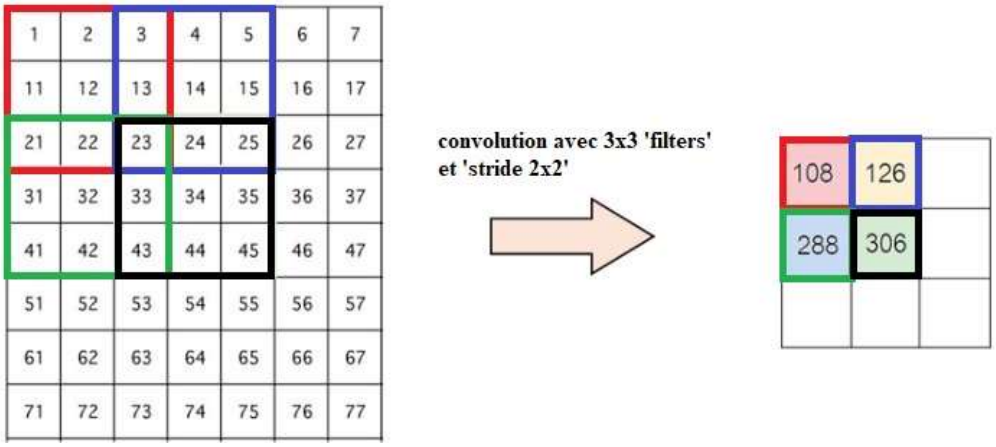


Figure 2.19 : *illustration du pas de 2 pixels.*

2.5.3.4. La marge à zéros ‘Padding’

Parfois, le filtre ne correspond pas parfaitement à l'image d'entrée, pour résoudre ce problème, nous remplissons l'image avec des zéros (zéro-padding), La figure (2.20) illustre l’opération de marge à zéros ‘padding’.

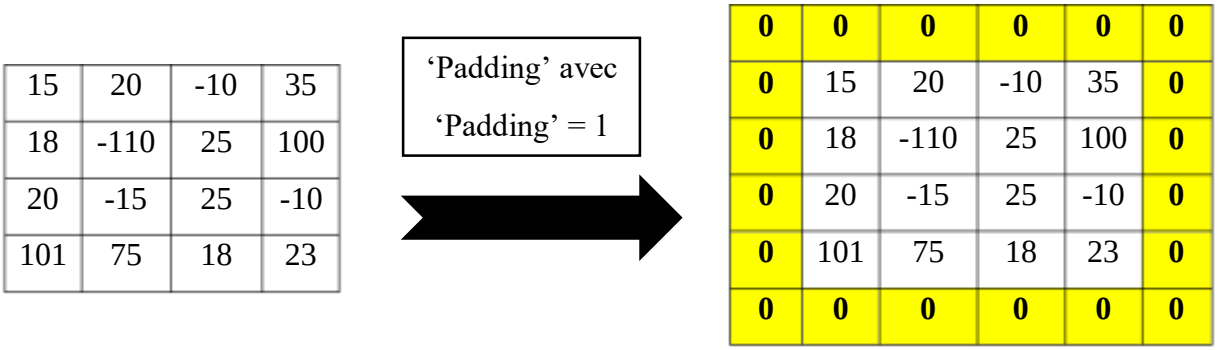


Figure 2.20 : *Opération de marge ‘padding’.*

2.5.3.5. La couche de correction ‘ReLU’

‘ReLU’ signifie Rectified Linear Unit pour une opération non linéaire.

La sortie est $f(x) = \max(0, x)$. (2.11)

Pourquoi ReLU est important : Le but de ReLU est d’introduire la non-linéarité dans notre ConvNet. Puisque, les données du monde réel voudraient que notre ConvNet apprenne seraient des valeurs linéaires non négatives.

Il existe d'autres fonctions non linéaires telles que tanh ou sigmoïde …etc. (dans partie fonctions d’activation page …).

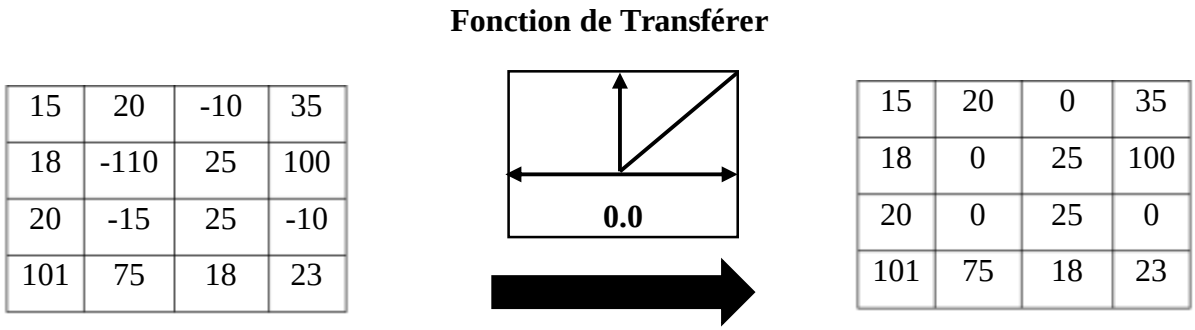


Figure 2.21 : *opération de ReLU.*

2.5.3.6. Couche de pooling ‘Pooling layer’

La section des couches de regroupement ‘pooling layer’ réduirait le nombre de paramètres lorsque les images sont trop grandes. Le regroupement spatial est également appelé sous-échantillonnage ou sous-échantillonnage qui réduit la dimensionnalité de chaque carte mais conserve des informations importantes. La mise en commun spatiale peut être de différents types :

- Regroupement maximale ‘max pooling’ : La regroupement maximale ‘max pooling’ prend le plus grand élément de la carte des caractéristiques ‘feature map’ rectifiée.
- Regroupement moyenne ‘average pooling’ : La regroupement moyenne ‘average pooling’ prend la moyenne de tous les éléments de la carte des caractéristiques ‘feature map’ rectifiée.

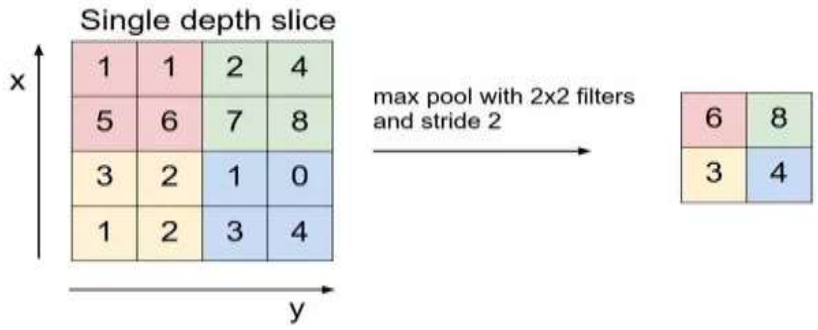


Figure 2.22 : *Illustration du 'Max Pooling'.*

2.5.3.7. Couche entièrement connectée 'fully connected layer'

La couche que nous appelons couche 'fully connected layer', aplatit la matrice en vecteur et la nourrit dans une couche entièrement connectée comme un réseau neuronal.

Dans le diagramme ci-dessus (figure 2.23), la matrice de la carte des caractéristiques sera convertie en vecteur (x1, x2, x3…). Avec les couches entièrement connectées, combinez ces fonctionnalités pour créer un modèle. Enfin, une fonction d'activation telle que softmax ou sigmoïde pour classer les sorties en chat, chien, voiture, camion …… etc.

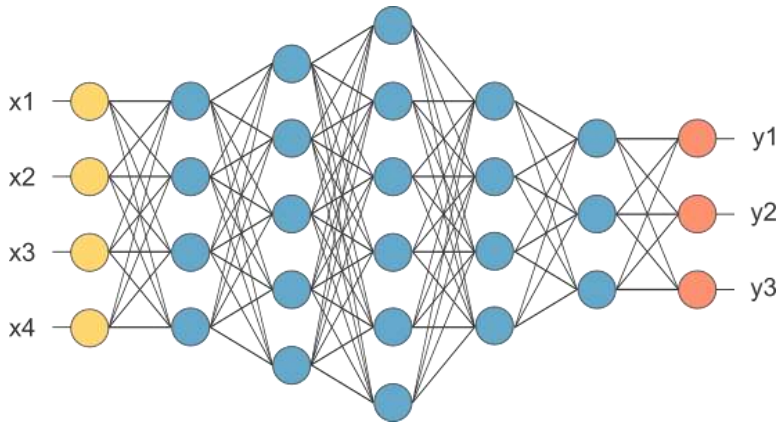


Figure 2.23 : *réseaux de neurones (NN).*

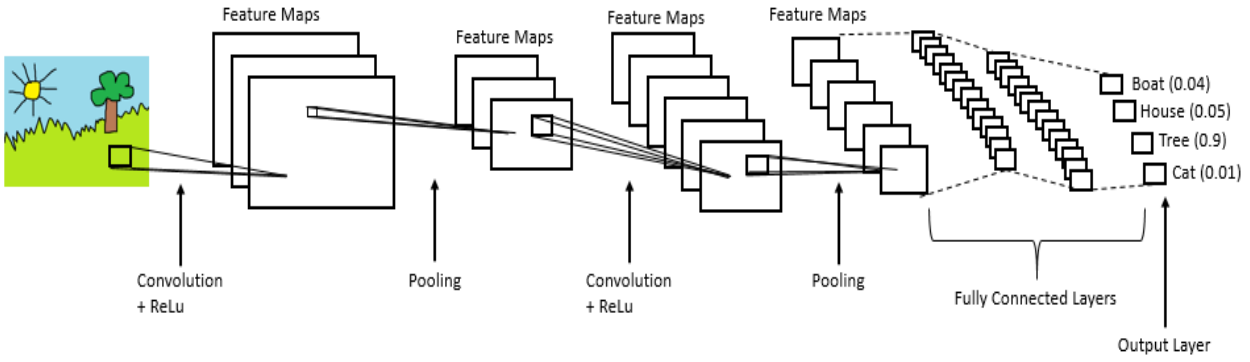


Figure 2.24 : *Architecture complète d'un CNN.*

2.6. Types d'apprentissage

Il existe quatre types d'apprentissage : l'apprentissage supervisé, l'apprentissage non supervisé, l'apprentissage semi-supervisé et l'apprentissage en transfert.

2.6.1. Apprentissage supervisé

L'apprentissage supervisé est sans doute le type d'apprentissage automatique le plus connu et étudié. Compte tenu de nos données d'entraînement, un modèle est créé via un processus d'apprentissage où des prédictions sont faites sur les données d'entrée, puis corrigées lorsque les prédictions sont fausses. Ce processus d'apprentissage se poursuit jusqu'à ce que le modèle atteigne un critère d'arrêt souhaité, tel qu'un faible taux d'erreur ou un nombre maximal d'itérations d'apprentissage.

2.6.2. Apprentissage non supervisé

Contrairement à l'apprentissage supervisé, l'apprentissage non supervisé (parfois appelé apprentissage autodidacte) n'a pas d'étiquettes associées aux données d'entrée et nous ne pouvons donc pas corriger notre modèle s'il fait une prédiction incorrecte.

2.6.3. Apprentissage semi-supervisé

Le petit peu de données étiquetées fournies aux systèmes sert de point de départ pour les systèmes informatiques. Après cela, les systèmes doivent accepter et apprendre de grands volumes de données non étiquetées. Cependant, les données étiquetées fournies peuvent être utiles pour classer le type général de données non étiquetées que le système peut recevoir. Par exemple, selon les données étiquetées, les températures supérieures à 104 ° F devraient être traitées comme un cas de

forte fièvre est donné, mais en réalité, une telle température élevée peut également être due à d'autres complications. Il appartient aux systèmes d'utiliser les données étiquetées de base et d'en apprendre davantage sur les grands volumes de données non étiquetées qu'ils reçoivent. Théoriquement, l'apprentissage semi-supervisé peut être considéré comme une meilleure méthode de formation pour les systèmes que l'apprentissage supervisé ou non supervisé. [w17]

2.6.4. Apprentissage par transfert ‘transfert Learning’

Dans l'apprentissage par transfert, nous formons d'abord un réseau de base sur un ensemble de données et une tâche de base, puis nous réutilisons les fonctionnalités apprises ‘learned features’, ou les transférons, vers un deuxième réseau cible pour être entraînées sur un ensemble de données et une tâche cibles ‘task’.

Ce processus aura tendance à fonctionner si les fonctionnalités sont générales, c'est-à-dire adaptées aux tâches de base et cibles ‘target tasks’, au lieu d'être spécifiques à la tâche de base [w18].

2.7. Construction d’un modèle d'apprentissage en profondeur

Pour construire un modèle d’apprentissage en profondeur il est indispensable de suivre minsieusent les étapes suivantes :

2.7.1. Etape 1 : Rassembler l'ensemble de données

La première composante de la construction d'un réseau d'apprentissage en profondeur consiste à rassembler notre ensemble de données initial. Nous avons besoin des images elles-mêmes ainsi que des étiquettes associées à chaque image. Ces étiquettes doivent provenir d'un ensemble fini de catégories, telles que : les catégories (chien, chat, souris).

De plus, le nombre d'images pour chaque catégorie doit être approximativement uniforme (c'est-à-dire le même nombre d'exemples par catégorie). Si nous avons deux fois plus d'images de chats que d'images de chiens et cinq fois plus d'images de souris que d'images de chats, alors notre classificateur deviendra naturellement biaisé ou sur-ajuster dans ces catégories fortement représentées.

Le déséquilibre des classes est un problème courant dans l'apprentissage automatique et il existe un certain nombre de façons de le surmonter. [13]

2.7.2. Etape 2 : Fractionner l'ensemble de données

Divisez l'ensemble de données en deux parties : un ensemble d'entraînement, un ensemble de test.

Un ensemble de formation est utilisé par notre classificateur pour « apprendre » à quoi ressemble chaque catégorie en faisant des prédictions sur les données d'entrée, puis se corriger lorsque les prédictions sont fausses. Une fois le classificateur formé, nous pouvons évaluer les performances sur un ensemble de tests.

Il est extrêmement important que l'ensemble de formation et l'ensemble de tests soient indépendants l'un de l'autre.

Les tailles divisées communes pour les ensembles de formation et de test incluent [(66.6% | 33.3%) ; (75% | 25%) ; (90% | 10%)] respectivement. [13]

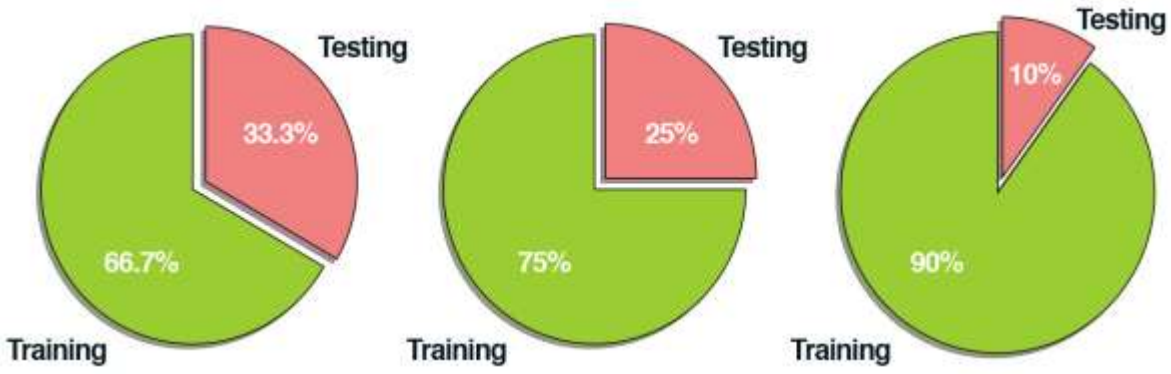


Figure 2.25 : *Exemples de fractionnement commun des données de formation et de test[w19].*

2.7.3. Etape 3 : Former le réseau

Création d'une architecture de modèle en définissant le nombre de nœuds dans chaque couche ‘layer’, le nombre de couches cachées ‘hidden layers’, le nombre de nœuds dans la dernière couche ‘output layer’, le type de fonction d'activation …. etc. [13]

2.7.4. Etape 4 : Evaluation

Faire un test pour chacune des images de l'ensemble de test. Puis tabule les prédictions du modèle pour une image dans l'ensemble de test. Enfin, ces prédictions de modèle sont comparées aux étiquettes de vérité terrain de l'ensemble de test. Les étiquettes de vérité terrain représentent ce qu'est réellement la catégorie d'image. À partir de là, ou peut calculer le nombre de prédictions que notre classificateur a obtenu correctement et calculer des rapports agrégés tels que la précision, le rappel et la f-mesure (mesures des performances), qui sont utilisés pour quantifier les performances du réseau dans son ensemble. [13]

2.8. Mesures de performances

Les évaluations de réseaux mise en œuvre nécessite le calcule d'un certain nombre de paramètres, on note :

a. Vrai positif 'True positive TP' : Le modèle prédit correctement la classe positive.

b. Vrai négatif 'True negative TN' : Le modèle prédit correctement la classe négative

c. Faux positif 'False positive FP' : Le modèle prédit incorrectement la classe positive.

d. Faux négatif 'False negative FN' : Le modèle prédit incorrectement la classe négative.

2.8.1. Matrice de confusion

La matrice de confusion d'un système de classification binaire est:

	Classe réelle positive	***Classe réelle négative***
Classe prédite positive	*vrai positif (VP)*	*Faux positif (FP)*
Classe prédite négative	*Faux négatif (FN)*	*Vrai négatif (VN)*

Tableau 2.1 : *matrice de confusion.*

2.8.2. Exactitude 'Accuracy' [w20]

Taux de bonnes prédictions.

$$Accuracy = \frac{VP+VN}{VP+VN+FN+FP} \qquad (2.12)$$

2.8.3. Taux vrai positif ou Sensitivité 'Recall' w20]

Probabilité d'un cas positif si la prédiction est positive.

$$Taux\ vrai\ positif = \frac{VP}{VP+FP} = \frac{VP}{P} \qquad (2.13)$$

2.8.4. Taux vrai négatif ou Spécificité [w20]

Probabilité d'une prédiction négative dans un cas négatif.

$$Taux\ vrai\ negatif = \frac{VN}{VN+FP} = \frac{VN}{N} \qquad (2.14)$$

2.8.5. Taux faux positif [w20]

$$Taux\ faux\ positif = 1 - \frac{VN}{VN+FP} = 1 - \frac{VN}{N} = 1 - taux\ vrai\ negatif \qquad (2.15)$$

2.8.6. Taux faux négatif [w20]

$$Taux\ faux\ negatif = 1 - \frac{VP}{VP+FP} = 1 - \frac{VP}{P} = 1 - taux\ vrai\ positif \qquad (2.16)$$

2.8.7. F-mesure 'moyenne harmonique' [w20]

C'est la moyenne harmonique entre le rappel et la précision.

$$FM = \frac{1}{\frac{1}{P}+\frac{1}{R}} = 2 * \frac{P*R}{P+R} \qquad (2.17)$$

Pour l'apprentissage de plusieurs classes, on généralise :

- rappel = somme des rappels de chaque classe/nombre de classes

- précision = somme des précisions de chaque classe/nombre de classes

2.8.8. La courbe ROC

La courbe ROC 'Receiver Operating Characteristic' permet d'analyser l'influence d'un paramètre du modèle : on fait varier le paramètre et on trace la courbe du rappel en fonction de 1-spécificité (taux de vrais positifs en fonction du taux de faux positifs). [w21]

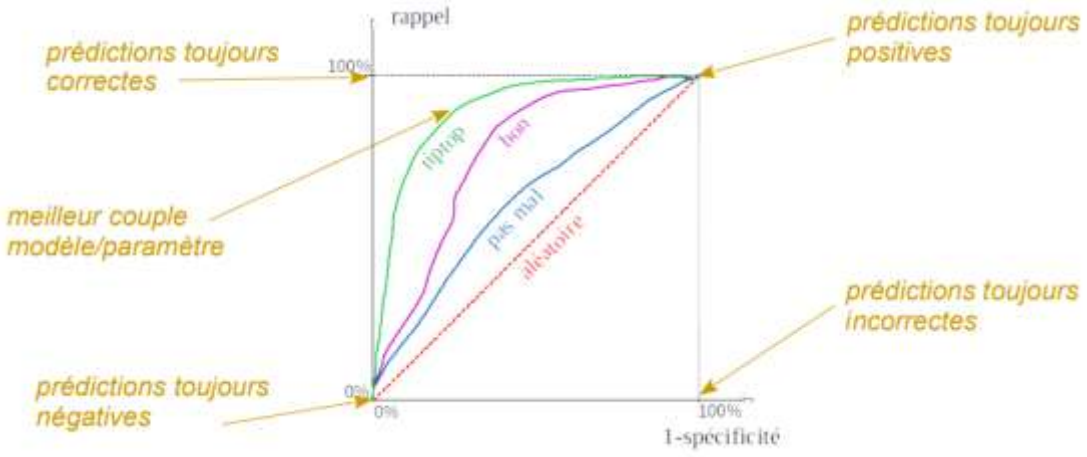


Figure 2.26 : *la courbe ROC.*

3. Conclusion

De ce chapitre, nous avons conclu que 'CNN' est plus précis que 'simple neural network' dans l'apprentissage des modèles de classification car 'CNN' contient une architecture complexe et de nombreuses couches multidimensionnelles travaillent à l'extraction de caractéristiques mais 'Neural Network' fonctionne avec des vecteurs (un vecteur d'une dimension pour chaque couche).

Chapitre III : Conception

1. Introduction

L'un des domaines de la vision par ordinateur qui ont eu récemment un énorme intérêt, est la détection et la classification d'objets en images, car il est le cœur de nombreux système important comme les systèmes de surveillance des routes, les systèmes de sécurité, les systèmes de véhicules autonomie.

Devant le nombre croissant de caméras et l'impossibilité de placer un opérateur humain derrière chacune d'elles, la demande et le besoin d'outils d'analyse automatique des données récupérées ont fortement augmenté.

Notre projet entre dans le cadre du 'monitoring' routier et autoroutier dont l'objectif principale est la détection et la classification des véhicules sur des images du trafic tourier ou autoroutier. Pour ce faire nous avons utilisé les réseaux du neurones convolutifs connus car leur performance dans ce type d'application.

2.Architecture du système

L'architecture détaillée de notre système est illustrée la (figure 3.1).

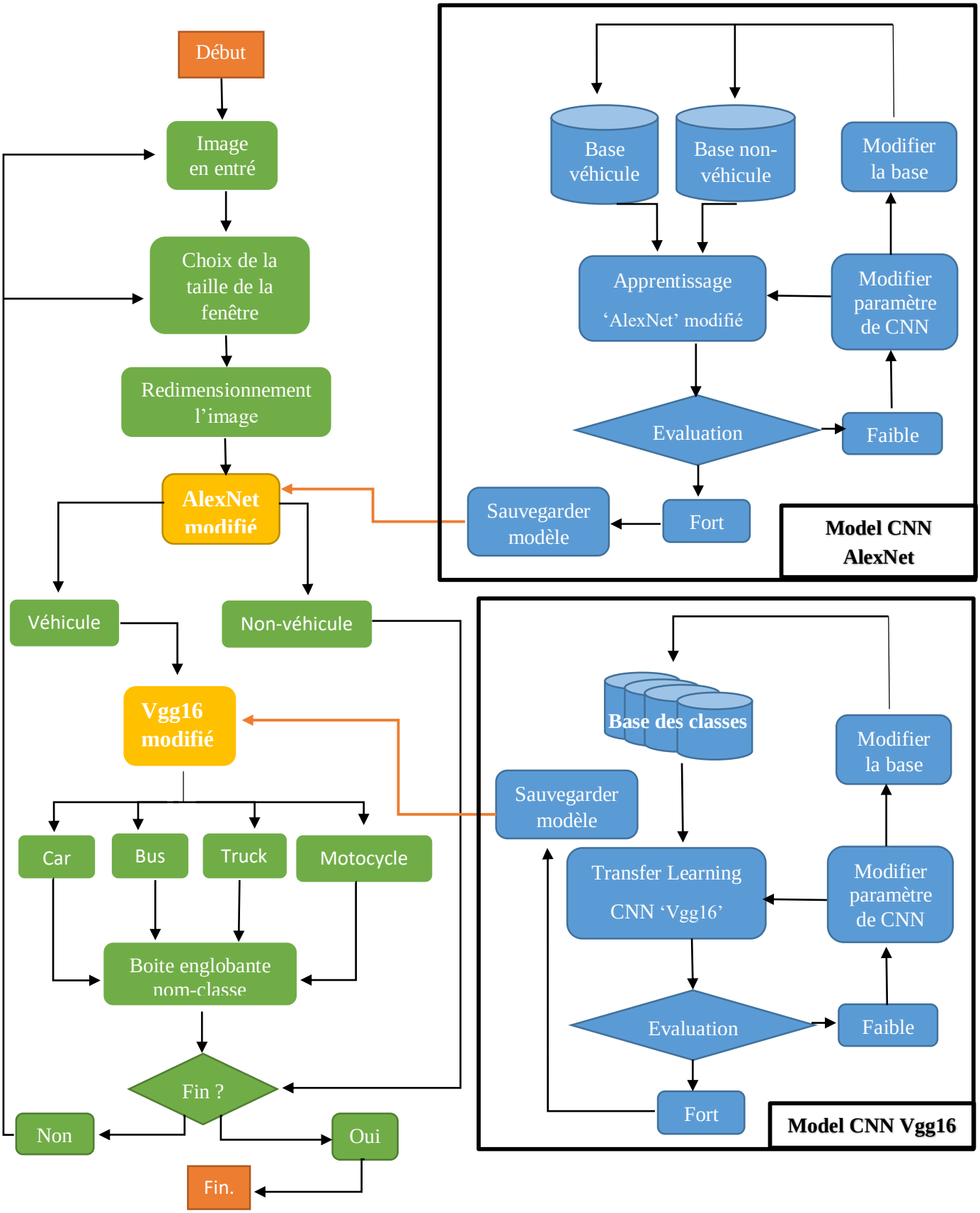


Figure 3.1 : *Architecture de notre système.*

3. Choix de la taille de la fenêtre

Dans notre travail, pour faire une détection de véhicule, nous devons faire un scan général pour l'image avec une fenêtre. La taille de cette boîte et le pas de déplacement déterminent si on va détecter tous les véhicules ou non. Donc, nous devons choisir la bonne taille pour chaque vue différente.

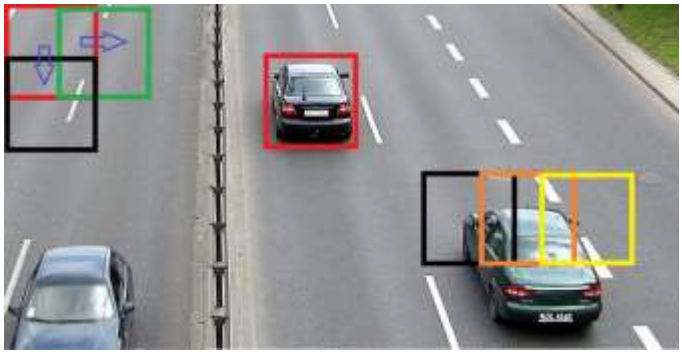


Figure 3.2 : *déplacement d'une fenêtre.*

4. Redimensionnement de l'image

Après avoir choisi une taille pour la boîte, nous coupons cette fenêtre et faisons un redimensionnement avant nous l'introduire à notre model 'alexNet' parce que il n'accepte que les entrées de taille fixe (100x100).

5. Réseaux CNN

5.1. Alex Net

5.1.1. Préparation de base de données

Nous avons prévu plusieurs préparations nécessaires pour le bon entrainement de notre model, et accélérer ce processus.

- Convertir tous les images de la base au couleur gris (véhicule et non-véhicule).
- Redimensionner les images à (100x100x1).
- Annoter la base de données 'Label data set', marquer chaque image dans la base d'apprentissage 0 ou 1 (véhicule, non-véhicule).
- Normaliser l'intensité des images (diviser matrice d'image sur 255).

5.1.2. Architecture du réseaux ‘AlexNet’

Afin d’adapter le réseaux ‘AlexNet’ à notre application et obtenir un taux de classification considérable nous avons apporté un changement sur l’architecture du réseau, la figure (3.3) illustre l’architecture originale du réseaux ‘AlexNet’ qui se compose de cinq couches convolutifs, dont certaines sont suivies de couches de ‘pooling’ maximum, puis de trois couches entièrement connectées, enfin un classificateur ‘softmax’ à 1000 classes.

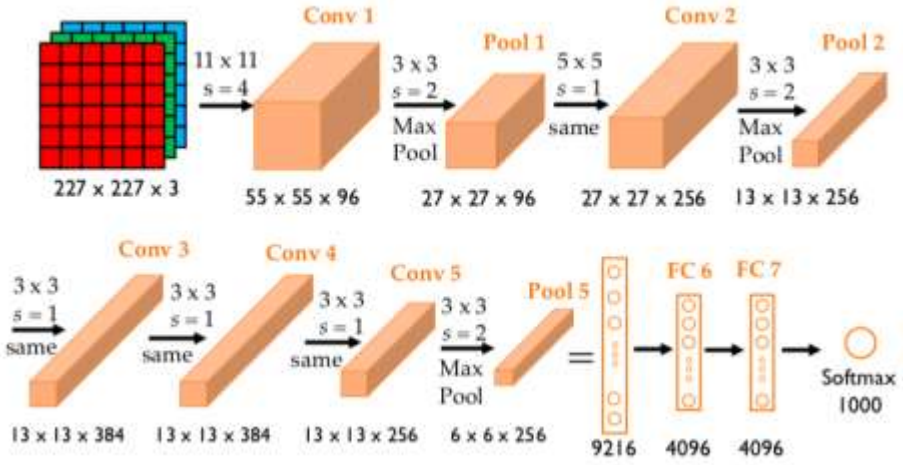


Figure 3.3 : *Schéma de l’architecture de ‘Alex Net’[w22].*

Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	227x227x3	-	-	-
1	Convolution	96	55 x 55 x 96	11x11	4	relu
	Max Pooling	96	27 x 27 x 96	3x3	2	relu
2	Convolution	256	27 x 27 x 256	5x5	1	relu
	Max Pooling	256	13 x 13 x 256	3x3	2	relu
3	Convolution	384	13 x 13 x 384	3x3	1	relu
4	Convolution	384	13 x 13 x 384	3x3	1	relu
5	Convolution	256	13 x 13 x 256	3x3	1	relu
	Max Pooling	256	6 x 6 x 256	3x3	2	relu
6	FC	-	9216	-	-	relu
7	FC	-	4096	-	-	relu
8	FC	-	4096	-	-	relu
Output	FC	-	1000	-	-	Softmax

Tableau 3.1 : *architecture originale de réseaux ‘Alex Net’ [15].*

Tableau 3.2 illustre l'architecture modifié que nous avons proposée après une série de tests.

Layer		Feature map	size	Kernel size	stride	activation
input	image	1	**100x100x1**	-	-	-
1	convolution	96	90x90x96	11x11	**1**	Relu
	Max pool	96	45x45x96	2x2	**1**	Relu
2	convolution	256	35x35x256	11x11	**1**	Relu
	Maxpool	256	17x17x256	2x2	**1**	Relu
3	Convolution	384	15x15x384	3x3	**1**	Relu
4	Convolution	384	13x13x384	3x3	**1**	Relu
5	Convolution	256	11x11x256	3x3	**1**	Relu
	maxpool	256	5x5x256	2x2	**1**	Relu
6	FC	-	4096	-	-	Relu
7	FC	-	4096	-	-	Relu
8	FC	-	1000	-	-	Relu
Output	FC	-	**1**	-	-	**Sigmoid**

Tableau 3.2 : *architecture modifie de réseaux Alex Net.*

Notons que les principaux changements apportés

a- La taille de la couche d'entrer (input layer) de (227x227x3) à (100x100x1).

b- Nombre de nœuds de couche de sortie (output layer) de 1000 nœud à 1 noud.

c- La fonction d'activation on dernière couche de softmax a sigmoid.

d- La valeur de 'stride' ont toutes les couches de convolution à 1.

e- La valeur de 'stride' ont toutes les couches de (max pooling) à 1.

5.2. Vgg16

5.2.1. Préparation de base de données

- Redimensionner les images à (224x224x3).
- Annoter la base de données (Label data set), marquer chaque image dans la base d'apprentissage 0, 1, 2, 3, (car, bus, motocycle, truck).

5.2.2. Architecture du réseaux 'Vgg16'

Afin d'adapter le réseaux 'Vgg16' à notre application et obtenir un taux de classification considérable nous avons apporté un changement sur l'architecture du réseau, la figure (3.6) illustre l'architecture originale du réseaux 'Vgg16' qui se compose de treize couches convolutifs et cinq couche de 'pooling' maximum, puis de trois couches 'fully connected' entièrement connectées, enfin un classificateur 'softmax' à 1000 classes.

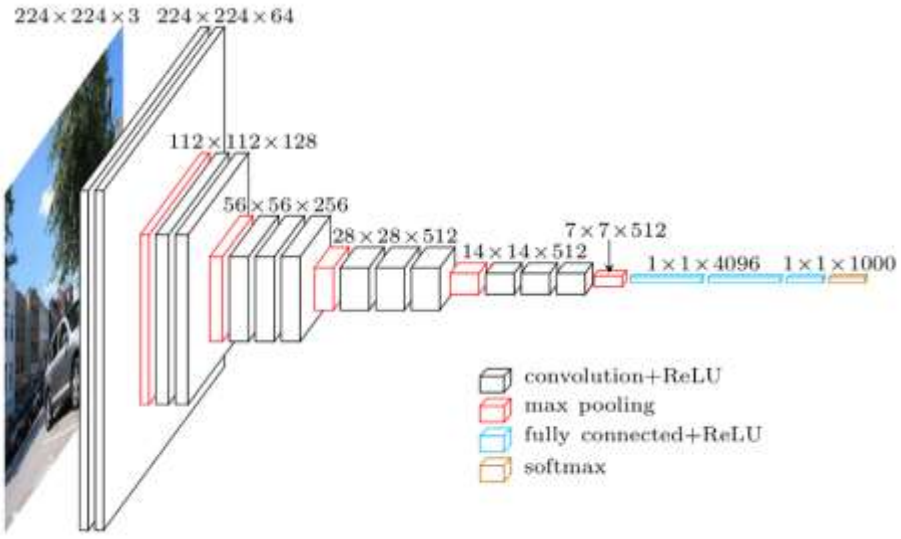


Figure 3.4 : *Schéma de l'architecture 'vgg16' [16].*

Layer		Feature map	size	Kernel size	stride	activation
input	image	1	224x224x3	-	-	-
1	Convolution	64	224x224x64	3x3	1x1	Relu
2	Convolution	64	224x224x64	3x3	1x1	Relu
	Maxpool	128	112x112x128	2x2	2x2	Relu
3	Convolution	128	112x112x128	3x3	1x1	Relu
4	Convolution	128	112x112x128	3x3	1x1	Relu

	Maxpool	256	56x56x256	2x2	2x2	Relu
5	Convolution	256	56x56x256	3x3	1x1	Relu
6	Convolution	256	56x56x256	3x3	1x1	Relu
7	Convolution	256	56x56x256	3x3	1x1	Relu
	Maxpool	512	28x28x512	2x2	2x2	Relu
8	Convolution	512	28x28x512	3x3	1x1	Relu
9	Convolution	512	28x28x512	3x3	1x1	Relu
10	Convolution	512	28x28x512	3x3	1x1	Relu
	Maxpool	512	14x14x512	2x2	2x2	Relu
11	Convolution	512	14x14x512	3x3	1x1	Relu
12	Convolution	512	14x14x512	3x3	1x1	Relu
13	Convolution	512	14x14x512	3x3	1x1	Relu
	Maxpool	512	7x7x512	2x2	2x2	Relu
	FC	-	4096	-	-	Relu
	FC	-	4096	-	-	Relu
	FC	-	4096	-	-	Relu
	Output	-	1000	-	-	Softmax

Tableau 3.3 : *architecture originale de 'vgg16' [16].*

Tableau 3.4 illustre l'architecture modifié que nous avons proposée.

Layer		**Feature map**	**size**	**Kernel size**	**stride**	**activation**
Input	image	1	224x224x3	-	-	-
1	Convolution	64	224x224x64	3x3	1x1	Relu
2	Convolution	64	224x224x64	3x3	1x1	Relu
	Maxpool	128	112x112x128	2x2	2x2	Relu
3	Convolution	128	112x112x128	3x3	1x1	Relu
4	Convolution	128	112x112x128	3x3	1x1	Relu
	Maxpool	256	56x56x256	2x2	2x2	Relu
5	Convolution	256	56x56x256	3x3	1x1	Relu
6	Convolution	256	56x56x256	3x3	1x1	Relu
7	Convolution	256	56x56x256	3x3	1x1	Relu
	Maxpool	512	28x28x512	2x2	2x2	Relu
8	Convolution	512	28x28x512	3x3	1x1	Relu
9	Convolution	512	28x28x512	3x3	1x1	Relu
10	Convolution	512	28x28x512	3x3	1x1	Relu
	Maxpool	512	14x14x512	2x2	2x2	Relu
11	Convolution	512	14x14x512	3x3	1x1	Relu
12	Convolution	512	14x14x512	3x3	1x1	Relu
13	Convolution	512	14x14x512	3x3	1x1	Relu
	Maxpool	512	7x7x512	2x2	2x2	Relu
	FC	-	4096	-	-	Relu
	FC	-	4096	-	-	Relu
	FC	-	4096	-	-	Relu
	Output	-	**4**	-	-	Softmax

Tableau 3.4 : *architecture modifié 'vgg16'.*

5.2.3. ‘Transfer Learning’ avec ‘Vgg16’

Pour l’apprentissage de model ‘vgg16’ modifié nous avons utilisé le ‘Transfer Learning’.

Pour ce faire, nous avons fait

a- télécharger le poids pour un modèle ‘Vgg16’ pré-entraîné.

b- créer l’architecture de ‘vgg16’ et faire quelques modification sur la couche de sortie.

```
out = Dense(num_of_classes, activation='softmax', name='output')(last_layer)
```

Figure 3.5 : *code couche de sortie de ‘vgg16’.*

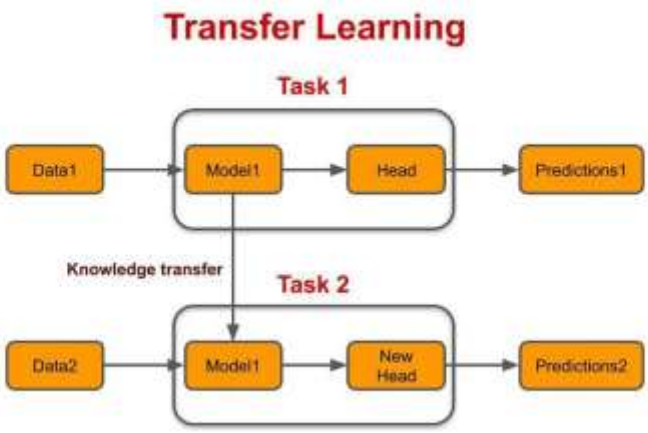


Figure 3.6: *explication de ‘transfer learning’[w23].*

c- fixer l’entrainement Just sur le dernière couche ‘output_layer.trainable = True else False’.

```
for layer in custom_vgg16.layers[:-1]:
    layer.trainable = False

print(custom_vgg16.layers[3].trainable)
custom_vgg16.summary()
```

Figure 3.7 : *code de modification l’état de chaque couche.*

5.3. Classification

Après avoir terminé la formation des deux modèles, nous les utilisons dans cet ordre

D'abord, nous alimentons l'image d'entrée vers le modèle 'Alex Net', si le résultat est 0 cela signifie (non-véhicule) alors nous allons à la fin sinon si 1 (véhicule), nous passons au modèle suivant 'vgg16'. La sortie positive du modèle 'Alex Net' représente une image d'entrée pour le modèle 'vgg16' qui fait la classification et le résultat sera 0 ou 1 ou 2 ou 3.

5.4. Boîte englobante avec nom de classe

On trouve dans la boîte englobante avec nom de classe le résultat réel pour la classification qui est selon notre dictionnaire est l'une de ces quatre classes (0 : bus, 1 : voiture, 2 : moto, 3 : camion).

6. Conclusion

La fiabilité des réseaux de neurones convolutifs dépend en grande partie du jeu de données utilisé 'data set' et de l'apprentissage.

L'utilisation des réseaux pré-entrainés tell que le 'vgg16' est très fiable mais nécessite un renforcement de l'apprentissage (Apprentissage par transfert) sur des classes désignées selon l'application.

.

Chapitre IV : Implémentation

1.Introduction

Il est important de valider l'architecture du système proposé par une implémentation avec un langage adéquat (dans notre cas python) et une série de tests sur des benchmarks de renommé.

2. Environnement

2.1. Google colab

2.1.1. Définition

Google Colab ou Colaboratory est un service cloud, offert par Google (gratuit), basé sur 'Jupyter Notebook' et destiné à la formation et à la recherche dans l'apprentissage automatique. Cette plateforme permet d'entraîner des modèles de Machine Learning directement dans le cloud. Sans donc avoir besoin d'installer quoi que ce soit sur l'ordinateur à l'exception d'un navigateur[w24].

Figure 4.1 : *Interface de google colab.*

2.1.2. Entrainer sur GPU

Pour passer en mode GPU, dans la barre des options choisir "exécution" puis "modifier le type d'exécution" et mettre l'option accélérateur matériel en mode GPU.

Nous serons obligés de ré-exécuter tout le notebook. Cette fois ci l'exécution du précédent prendra environ 4 minutes. Pour cet exemple l'entrainement sur un GPU est 60 fois plus rapide qu'un entrainement classique sur un CPU.

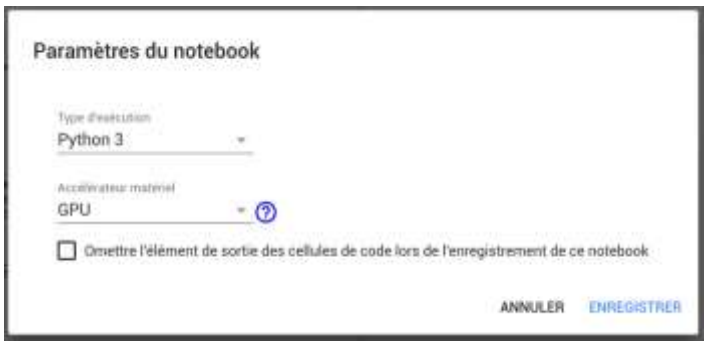


Figure 4.2 : *Paramètre du notebook.*

2.2. Bibliothèque utilisée

2.2.1. Python

Python est un langage de programmation de haut niveau à usage général et largement utilisé. Il a été créé par 'Guido van Rossum' en 1991 et développé par 'The Python Software Foundation'. Il a été conçu en mettant l'accent sur la lisibilité du code et sa syntaxe permet aux programmeurs d'exprimer leurs concepts en moins de lignes de code.

Python est un langage de programmation qui vous permet de travailler rapidement et d'intégrer les systèmes plus efficacement.

Il existe deux versions majeures de Python : Python 2 et Python 3. Les deux sont assez différents. [17]

2.2.2. OpenCV-Python

'OpenCV-Python' est une bibliothèque de liaisons Python conçue pour résoudre les problèmes de vision par ordinateur.

Comparé à des langages comme « C / C ++ », 'Python' est plus lent. Cela dit, Python peut être facilement étendu avec « C / C ++ », ce qui nous permet d'écrire du code intensif en calcul en « C / C ++ » et de créer des wrappers Python qui peuvent être utilisés comme modules 'Python'. Cela nous donne deux avantages : premièrement, le code est aussi rapide que le code « C / C ++ » original (puisque c'est le code C ++ réel travaillant en arrière-plan) et deuxièmement, il est plus facile de coder en Python qu'en « C/C++ ». 'OpenCV-Python' est un 'Python wrapper' pour l'implémentation 'OpenCV C ++' d'origine [18].

2.2.3. Numpy

‘NumPy’ est une bibliothèque python « Numerical Python » utilisée pour travailler avec des tableaux. Il a également des fonctions pour travailler dans le domaine de l'algèbre linéaire, de la transformée de Fourier et des matrices.

‘NumPy’ a été créé en 2005 par ‘Travis Oliphant’. C'est un projet open source et nous pouvons l'utiliser librement [19].

2.2.4. Tensorflow

‘TensorFlow’ est une plate-forme Open Source de bout en bout dédiée au ‘machine learning’. Elle propose un écosystème complet et flexible d'outils, de bibliothèques et de ressources communautaires permettant aux chercheurs d'avancer dans le domaine du ‘machine learning’, et aux développeurs de créer et de déployer facilement des applications qui exploitent cette technologie [20].

2.2.5. Keras

‘Keras’ est une API de réseaux de neurones de haut niveau, écrite en Python et interfaçable avec ‘TensorFlow’, ‘CNTK’ et ‘Theano’. Elle a été développée avec pour objectif de permettre des expérimentations rapides. Être capable d’aller de l’idée au résultat avec le plus faible délai possible étant la clef d’une recherche efficace [21].

2.2.6. Tkinter

Le package ‘tkinter’ « Tk interface » est l'interface Python standard du ‘Tk GUI toolkit’. ‘Tk’ et ‘tkinter’ sont disponibles sur la plupart des plates-formes Unix, ainsi que sur les systèmes Windows. (Tk lui-même ne fait pas partie de Python, il est maintenu à ActiveState.) [22]

2.2.7. Pickle

Le module pickle implémente des protocoles binaires pour la sérialisation et désérialiser une structure d'objet Python. «Pickling» est le processus par lequel une hiérarchie d'objets Python est convertie en un flux d'octets, et «unpickling» est l'opération inverse, par laquelle un flux d'octets (à partir d'un fichier binaire ou d'un objet de type octets) est reconverti en une hiérarchie d'objets.[w25]

2.2.8. Os

Le module OS en python fournit des fonctions d'interaction avec le système d'exploitation.

'OS' fait partie des modules utilitaires standard de Python. Ce module fournit un moyen portable d'utiliser les fonctionnalités dépendant du système d'exploitation. Les modules 'os' et 'os.path' incluent de nombreuses fonctions pour interagir avec le système de fichiers.[23]

3. Base d'apprentissage

Nous avons utilisé la base 'MOI-TCD-Vehicle classification', pour l'adapter avec notre réseau, nous avons fait quelques modifications sur la forme de cette base, Tableau 4.1 et Tableau 4.2 illustrent les nouvelles architectures pour l'apprentissage de 'AlexNet' et 'Vgg16'.

3.1. Base de 'AlexNet'

Classes	***Apprentissage '80%'***	***Test '20%'***
Vehicule ['bus'+'car'+'truck'+'motorcycle']	4000	1000
Non-Vehicule ['background'+ 'pedestrian']	4000	1000
Somme	8000	2000

Tableau 4.1 : *base 'MOI-TCD' adaptée pour Alex Net.*

3.2. Base de 'Vgg16'

Classes	***Apprentissage '80%'***	***Test '20%'***
'Bus'	3550	800
'Car'	4366	800
'Motocycle'	1366	800
'Truck'	3563	800
somme	16045	3200

Tableau 4.2 : *base 'MOI-TCD' adaptée pour Vgg16.*

4. Apprentissage et Test

4.1. Apprentissage de réseaux 'Alex Net'

Après plusieurs test et changements des caractéristiques du réseaux 'Alex Net', Nous avons obtenu le meilleur apprentissage avec la configuration suivante :

optimiser= 'Adam', loss= 'binary_crossentropy', epochs=40', 'batch_size = 32'

(La figure 4.5) illustré les résultats d'apprentissage et de test représente par les graphes de précision 'Accuracy' et de perte 'Loss'.

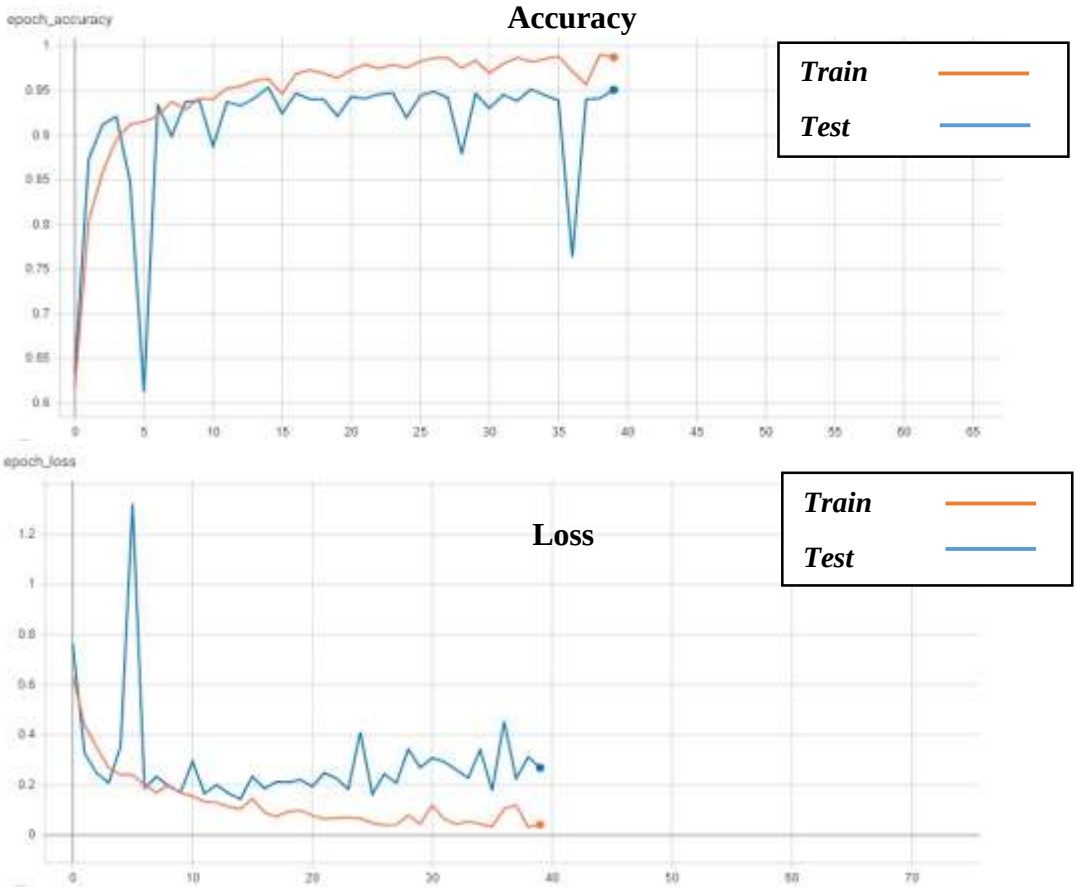


Figure 4.3. : *les graphes de 'accracy' et 'loss' pour apprentissage de 'AlexNet'.*

Où nous remarquons que le réseau a atteind une précession de (98%) durant l'apprentissage et (95%) durant le test. Et une perte de (0.05%) durant l'apprentissage et (0.28%) pour le test. Considérés comme acceptable pour lancer les tests sur des images de scène complète (tableau 4.1)

4.2. Test de réseaux ‘Alex Net’

Nous avons fait une expérience sur 2000 images qui n'appartenaient pas à la base de données utilisée dans l’entraiment, et le résultat était le suivant :

Taux	Pourcentage
VP	92%
VN	95%
FP	05%
FN	08%

Tableau 4.3 *: tableau de taux de test.*

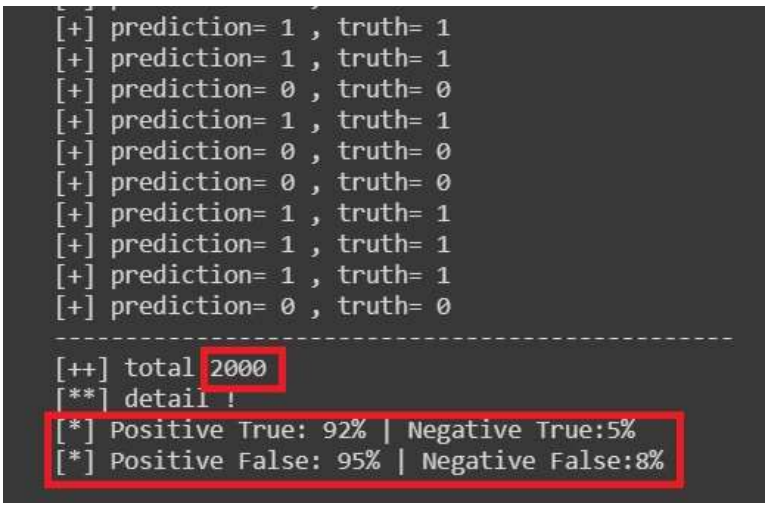


Figure 4.4 : *résultat de test ‘Alex Net’.*

4.3. Apprentissage de réseaux ‘Vgg16’

Après plusieurs test et changements des caractéristiques du réseaux ‘Vgg16’, Nous avons obtenu le meilleur apprentissage avec la configuration suivante :

optimiser= ‘Adam’, loss= ‘categorical_crossentropy’, epochs=30’, ‘batch_size = 32’

(La figure 4.5) illustré les résultats d’apprentissage et de test représente par les graphes de précision ‘Accuracy’ et de perte ‘Loss’.

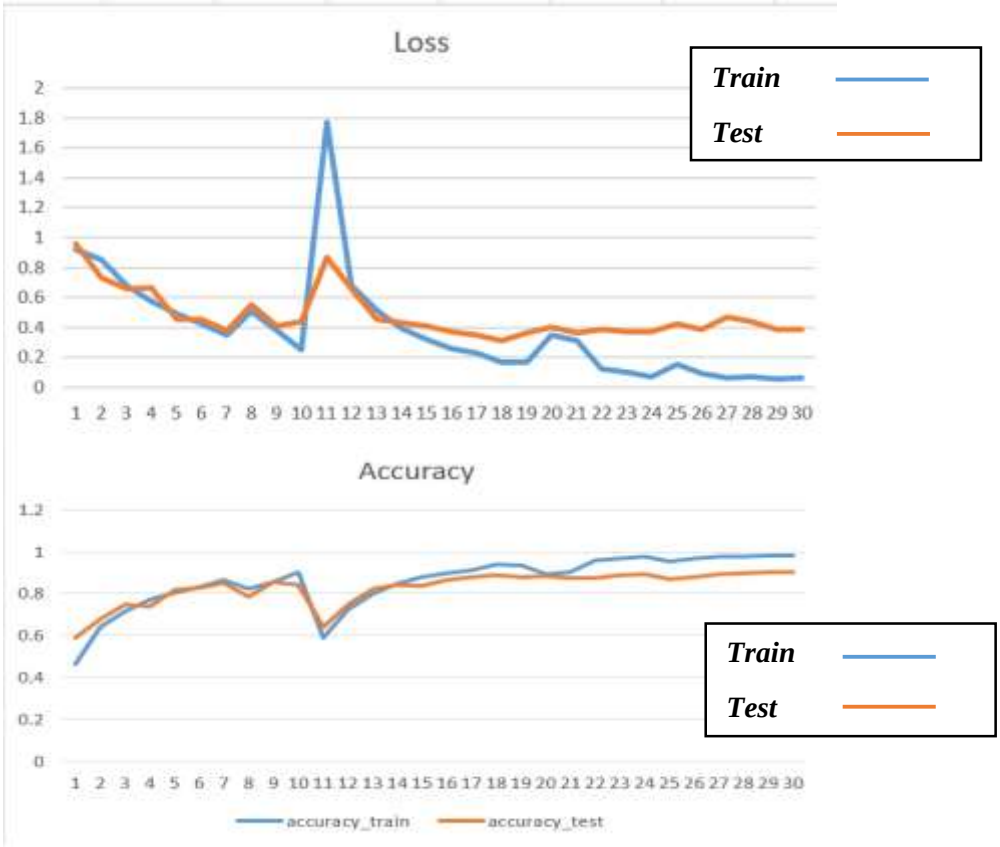


Figure 4.5 : *les graphes de ‘accracy’ et ‘loss’ pour apprentissage de ‘Vgg16.*

Où nous remarquons que le réseau a atteind une précession de (98%) durant l’apprentissage et (92%) durant le test. Et une perte de (0.05%) durant l’apprentissage et (0.4%) pour le test.

Considérés comme acceptable pour lancer les tests sur des images de scène complète (tableau 4.1)

4.4. Test de réseaux ‘Vgg16'

Nous avons fait une expérience sur 3200 images qui n'appartenaient pas à la base de données utilisée dans l’entraiment, Tableau 4.4 illustré le résultat de ce test.

True Class

Predicted Class	*Bus*	*Car*	*Motocycle*	*Truck*
Bus	**94.62%**	01.00%	00.13%	04.25%
Car	00.25%	**98.63%**	00.25%	00.87%
Motocycle	00.00%	03.63%	**92.50%**	03.87%
Truck	06.25%	03.38%	02.62%	**87.75%**

Tableau 4.4: *tableau de matrice de confusion.*

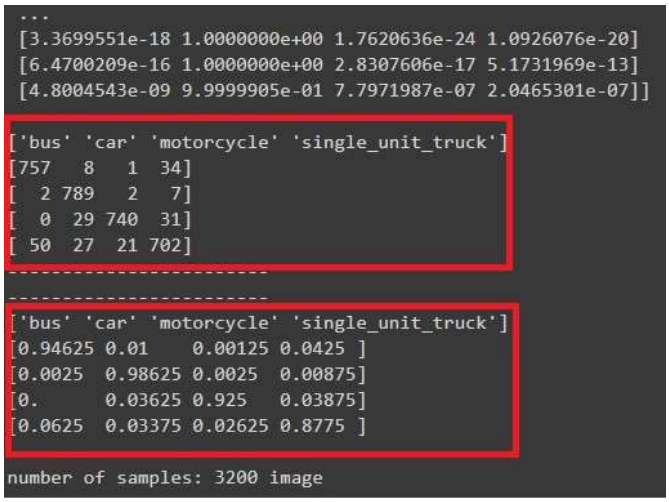


Figure 4.6 *: résultat de test ‘Vgg16’.*

5. Quelque tests sur des images aléatoires

Originale	Détection 'Alex Net'	Classification 'Vgg16'
		
	Alex net résultat : 1/1	**Vgg16 resultat :1/1**
		
	Alex net résultat : 3/3	**Vgg16 resultat :3/3**
		
	Alex net résultat : 2/2	**Vgg16 resultat :2/2**
		
	Alex net résultat : 1/2	**Vgg16 resultat :1/2**
		
	Alex net résultat : 1/2	**Vgg16 resultat :1/1**

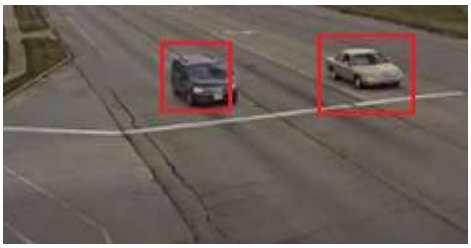
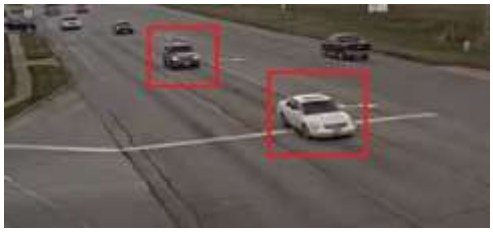
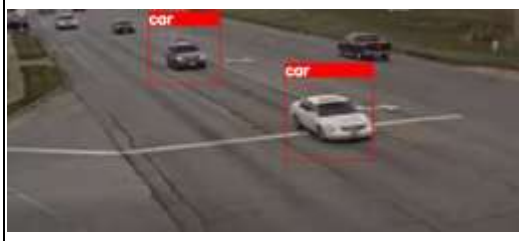
	 **Alex net résultat : 2/2**	 **Vgg16 resultat :2/2**
	 **Alex net résultat : 2/5**	 **Vgg16 resultat :2/2**

Tableau 4.5 : *illustration pour les tests et les résultats de système.*

6. Conclusion

Les résultats de notre système sont très prometteurs dans le domaine la détection et la classification des véhicules. Les performances de ce système peuvent être optimisé en améliorant la qualité du 'hardware' comme (GPU).

L'application permet de détecter (véhicule, non-véhicule) et classifier les véhicules selon leur type (bus, car, motocycle, truck), et présente une base solide pour un développement futur du domaine du 'monitoring' routier et autoroutier en Algérie.

Conclusion générale

Au cours de ces dix dernières années, le monde a vu les plus grandes statistiques d'accidents de la route, et avec le grand développement dans le domaine de l'IA (Intelligent Artificiel), les chercheurs ont développé des systèmes destinés à limité ce fléau. Appelés les systèmes de Transport Intelligents (STI). Ces systèmes ont été adapté dans les réseaux routiers de beaucoup de pays développés mais dont L'Algérie ne dispose pas encore dans ses autoroutes.

C'est dans ce cadre que nous avons élaboré notre projet entre dans le cadre du 'monitoring' routier et autoroutier et dont l'objectif principal est la détection et la classification des véhicules sur des images du trafic routier ou autoroutier.

Pour ce faire nous avons utilisé deux réseaux convolutifs travaillants séquentiellement où la première phase dédiée au réseau 'Alex Net' (modifié) dont l'architecture a été modifié afin l'adapter à notre application qui consiste en la détection des véhicules dans une image arbitraire. La fin de la tâche du premier réseau déclenche le démarrage du deuxième (VGG16) pré-entrainé sur 4 classes de véhicules (voiture, camion, bus, moto) qui décidera par la suite de la classe de chaque véhicule détecté précédemment par 'Alex Net' (modifié).

Les résultats de notre système sont très prometteurs dans le domaine la détection et la classification des véhicules. Les performances de ce système peuvent être optimisé, pour cet objectif nous proposons le suivant :

- Améliore la qualité de matériel de traitement 'hardware' (lancé le programme avec 'GPU' de haut performance).
- Change l'angle de capture des scènes et développe la base d'apprentissage.
- Laisse les modèles s'entraîner plus longtemps.
- Ajouter des modifications sur l'architecture des modèles comme (ajouter de nouvelles couches d'apprentissage, changer les types des fonctions d'activation, ……etc.)

Enfin, l'application peut être un premier pas dans la réalisation d'un système de 'monitoring' routier destiné à la détection et la classification des véhicules et qui mérite d'être améliorer et adapter aux autoroutes de notre pays.

Bibliographie

[1] Boukemoum, Zakarya. "Détection des véhicules par Histogramme orienté gradient." Mémoire de Master 2 département informatique université 8 mai 1945, Guelma (2019).

[2] Pang, Yanwei, et al. "Fully affine invariant SURF for image matching." Neurocomputing 85 (2012): 6-10.

[3] Geng, Cong, and Xudong Jiang. "Face recognition using sift features." 2009 16th IEEE international conference on image processing (ICIP). IEEE, (2009).

[4] Kuo, Bor-Chen, Cheng-Hsuan Li, and Jinn-Min Yang. "Kernel nonparametric weighted feature extraction for hyperspectral image classification." IEEE Transactions on Geoscience and Remote Sensing 47.4 (2009): 1139-1155.

[5] Feng, Jie, et al. "Bag-of-visual-words based on clonal selection algorithm for SAR image classification." IEEE Geoscience and remote sensing letters 8.4 (2011): 691-695.

[6] Lizarazo, Ivan, and Paul Elsner. "Fuzzy segmentation for object-based image classification." International Journal of Remote Sensing 30.6 (2009): 1643-1649.

[7] Wei, Wen, and Jerry M. Mendel. "Maximum-likelihood classification for digital amplitude-phase modulations." IEEE transactions on Communications 48.2 (2000): 189-193.

[8] Zohra, Adami Fatima, et al. "Detection and Classification of Vehicles Using Deep Learning. " International Journal of Computer Science Trends and Technology (IJCST) 6.3 (2018).

[9] Tsai, Chia-Chi, et al. "Vehicle detection and classification based on deep neural network for intelligent transportation applications." Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC). IEEE, (2018).

[10] Chen, Linkai, et al. "An algorithm for highway vehicle detection based on convolutional neural network." Eurasip Journal on Image and Video Processing 2018.1 (2018): 109.

[11] Seo, Chang-Jin. "Vehicle Detection in Dense Area Using UAV Aerial Images." Journal of the Korea Academia-Industrial cooperation Society 19.3 (2018): 693-698.

[12] Asvadi, Alireza, et al. "Depthcn: vehicle detection using 3d-lidar and convnet." (2017) IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). IEEE, (2017).

[13] Rosebrock, Adrian. Deep Learning for Computer Vision with Python: Starter Bundle. PyImageSearch, (2017).

[14] Ye, Chengxi, et al. "Lightnet: A versatile, standalone matlab-based environment for deep learning." *Proceedings of the 24th ACM international conference on Multimedia*. (2016).

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

[16] Gebrehiwot, Asmamaw, et al. "Deep convolutional neural network for flood extent mapping using unmanned aerial vehicles data." *Sensors* 19.7 (2019): 1486.

[17] Van Rossum, Guido. "Python programming language." USENIX annual technical conference. Vol. 41. (2007).

[18] Howse, Joseph. OpenCV computer vision with python. Packt Publishing Ltd, (2013).

[19] Walt, Stéfan van der, S. Chris Colbert, and Gael Varoquaux. "The NumPy array: a structure for efficient numerical computation." Computing in science & engineering 13.2 (2011): 22-30.

[20] Abadi, Martín, et al. "Tensorflow: A system for large-scale machine learning." 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16). (2016).

[21] Ketkar, Nikhil. "Introduction to keras." Deep learning with Python. Apress, Berkeley, CA, (2017). 97-111.

[22] Grayson, John E. Python and Tkinter programming. Manning Publications Co. Greenwich, (2000).

[23] Rossum, Guido. "Python reference manual." (1995).

Webographie

[w1] https://whatis.techtarget.com/definition/intelligent-transportation-system

[w2] https://online.lewisu.edu/mscs/resources/what-are-intelligent-systems

[w3] https://www.mathworks.com/discovery/object-detection.html

[w4] https://www.kaggle.com/jessicali9530/stanford-cars-dataset

[w5] https://boxy-dataset.com/boxy/

[w6] http://cmp.felk.cvut.cz/data/motorway/

[w7] https://programmersought.com/article/7654351045/;jsessionid=3CECBB5C3A435EC24D27AB30F9149CF9

[w8] https://www.kaggle.com/iamprateek/vehicle-images-gti?

[w9] http://mmlab.ie.cuhk.edu.hk/datasets/comp_cars/index.html

[w10] http://www.cvlibs.net/datasets/kitti/

[w11] https://www.kaggle.com/yash88600/miotcd-dataset-50000-imagesclassification?

[w12] https://mc.ai/understanding-the-softmax-activation-function/

[w13] https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn

[w14] https://blog.bismart.com/en/difference-between-machine-learning-deep-learning

[w15] https://medium.com/@bdhuma/6-basic-things-to-know-about-convolution-daef5e1bc411

[w16]https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148

[w17] https://www.techopedia.com/definition/32056/semi-supervised-learning

[w18] https://machinelearningmastery.com/transfer-learning-for-deep-learning/

[w19] https://www.cnblogs.com/paladinzxl/p/9491633.html

[w20] https://projeduc.github.io/intro_apprentissage_automatique/introduction.html

[w21] https://home.mis.u-picardie.fr/~furst/docs/Apprentissage_artificiel.pdf

[w22] https://anhreynolds.com/blogs/alexnet.html

[w23] https://www.topbots.com/transfer-learning-in-nlp/

[w24] https://ledatascientist.com/google-colab-le-guide-ultime/

[w25] https://docs.python.org/3/library/pickle.html