Traffic detection:
Vehicle counting and classification

Réalisé par :
Alaoui Belghiti Khaoula
3ACI-GDV N61

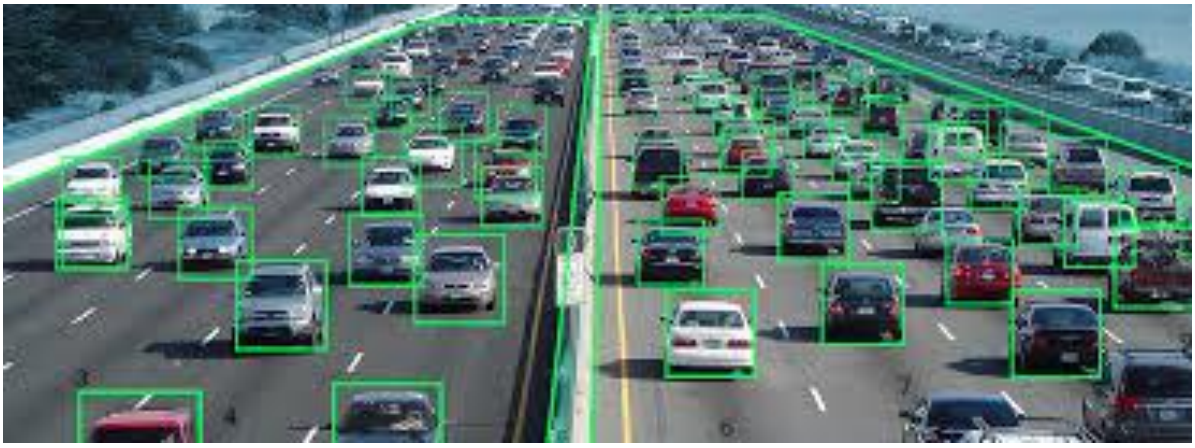
Résumé :

Le rapport suivant synthétise le travail que j'ai effectué au cadre du projet à réaliser sous le module Image Processing.
Le projet intitulé **Traffic Detection : vehicle counting and classification** est un projet de traitement d'image qui se base sur une base de données des feu de circulation prise à partir des caméras de surveillance, contenant plusieurs véhicules de différentes taille et types.
La détection et la classification des véhicules jouent un rôle essentiel dans l'espace du système de gestion du trafic. Il y a un domaine très vaste pour le développement de ce système, car il est associé à la précision et à l'exactitude. En raison de l'augmentation du trafic dans les occasions avancées, il est fondamental d'organiser un cadre gagnant pour garder une trace des véhicules passant par un chemin ou une rue. L'identification spontanée des données des véhicules a été largement utilisée dans le système d'identification et de classification des véhicules. Les applications du système développé sont souvent utiles dans le contrôleur de feux de circulation, le système d'alerte de sortie de voie pour les véhicules. L'objectif de la technique est de fournir des données appropriées sur les véhicules en circulation avec un comptage exact.
L'entrée est donnée sous forme des frames des vidéos de surveillance à partir d'une base de données pour la détection des feux de circulation. Le prétraitement est établie, pour l'exactitude de classification n'est pas encore précise, le projet vise en finalité le comptage des véhicules, la classification et la variété totale des mouvements des véhicules à un moment précis.

Introduction :

La détection et le comptage intelligents des véhicules deviennent de plus en plus importants dans le domaine de la gestion des routes. Cependant, en raison des différentes tailles de véhicules, leur détection reste un défi qui affecte directement la précision du comptage des véhicules. Où la plupart des recherches précédentes ne comptaient que la voiture, ou utilisent des algorithmes pour l'identification des véhicules, la surveillance des infractions au code de la route, la surveillance de la vitesse des véhicules, etc. Mais difficile de contourner le type exacte des véhicules passant et leur quantité. Dans le cadre de ce projet je ne propose pas seulement le comptage des véhicules mais aussi de les classifier qui peut servira à plusieurs finalités d'analyse de circulation.

Lors de l'évaluation des projets de vision par ordinateur, la formation et les données de test sont essentielles. Les données utilisées sont une représentation d'un défi qu'un système proposé doit résoudre. Il est souhaitable de disposer d'une grande base de données avec de grandes variations représentant le défi, pour le cas de mon projet la détection et la reconnaissance des véhicules dans un environnement urbain.



L'étude des travaux existants montre clairement qu'actuellement l'évaluation se limite principalement à de petits ensembles de données locales recueillies par les auteurs eux-mêmes et non à un ensemble de données accessibles au public. Les ensembles de données locales sont souvent de petite taille et ne présentent que peu de variations. Il est donc presque impossible de comparer les travaux et les résultats de différents auteurs, mais il devient également difficile d'identifier l'état actuel d'un domaine. Afin de fournir une base commune pour la comparaison future des recherches sur la reconnaissance des véhicules, une vaste base de données publique est collectée à partir d'images provenant des routes américaines sur les feux de circulation. La base de données comprend des séquences vidéo de test et de formation continue, totalisant 43 007 images et 113 888 feux de circulation annotés. Les séquences sont capturées par une caméra stéréo montée sur le toit d'un véhicule qui roule de nuit comme de jour dans des conditions de lumière et de météo variables.
Ce rapport englobe une description des modèles utiles pour une telle analyse avec le résultat de test, quelques travaux de recherches et solutions liées au sujet traités ainsi que la solution proposé dans ce cas.

Contexte métier :

Le Deep Learning est un algorithme de Machine Learning populaire qui est largement utilisé dans de nombreux domaines de la vie quotidienne actuelle. Ses performances robustes et ses cadres et architectures prêts à l'emploi permettent à de nombreuses personnes de développer divers logiciels ou systèmes basés sur le Deep Learning pour soutenir les tâches et activités humaines. La surveillance de la circulation est un domaine qui utilise le DL à plusieurs fins. En utilisant des caméras installées à certains endroits sur les routes, de nombreuses tâches telles que le comptage des véhicules, l'identification des véhicules, la surveillance des infractions au code de la route, la surveillance de la vitesse des véhicules, etc. peuvent être réalisées. Dans ce rapport, je discute l'implémentation des algorithmes de CNN et DL pour créer un système de comptage et classification des véhicules. Afin d'améliorer les performances du système et de réduire le temps de déploiement de l'architecture de DL. Cette recherche vise à créer un système simple de comptage de véhicules pour aider l'homme à classer et à compter les véhicules qui traversent la rue. Le comptage est basé sur quatre types de véhicules, c'est-à-dire voiture, moto, bus et camion, alors que les recherches précédentes ne comptaient que la voiture. En conséquence, le système que je propose est capable de compter les véhicules qui traversent la route.

Background modèles :

Je pense que je vais utiliser certains de ces modèles et des modèles préformés pour classer et compter les véhicules :

- Le VGG-16 est l'un des modèles préformés les plus populaires pour la classification des images. Présenté lors de la célèbre conférence ILSVRC 2014, il était et reste encore aujourd'hui LE modèle à battre. Développé au sein du Visual Graphics Group de l'Université d'Oxford, le VGG-16 a battu le standard d'AlexNet et les chercheurs et l'industrie pour leurs tâches de classification d'images l'ont rapidement adopté.

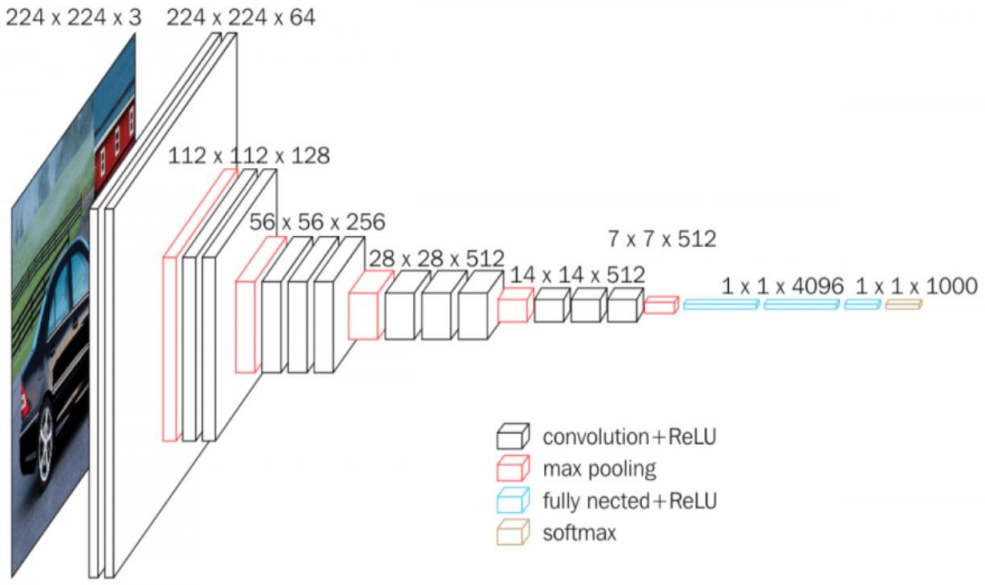


Figure A : VGG16

Le modèle atteint une précision de 92,7 % dans le top 5 des tests d'ImageNet, qui est un ensemble de données de plus de 14 millions d'images appartenant à 1000 classes. L'amélioration par rapport à AlexNet est due au remplacement des grands filtres de la taille d'un noyau (11 et 5 dans la première et la deuxième couche convolutionnelle, respectivement) par plusieurs filtres de la taille d'un noyau 3×3, l'un après l'autre. Le VGG16 a été formé pendant des semaines et utilisait les GPU NVIDIA Titan Black.

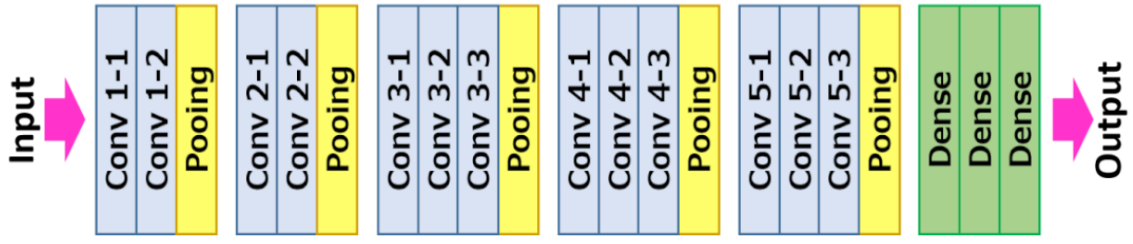


Figure B: VGG16 Architecture

- DS Net : Prévoir le nombre de personnes dans l'image. Détermination avec préservation des informations provenant de différentes parties de l'image. Modèle de bout en bout.

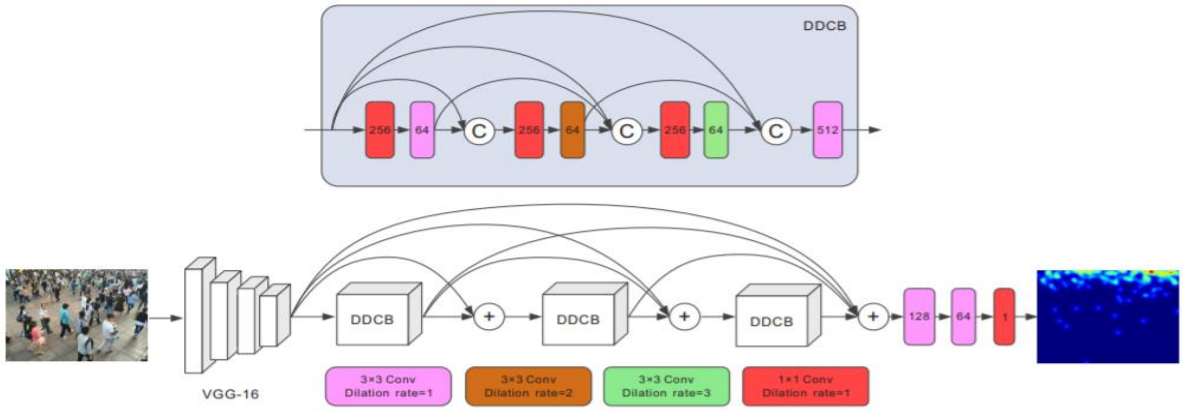
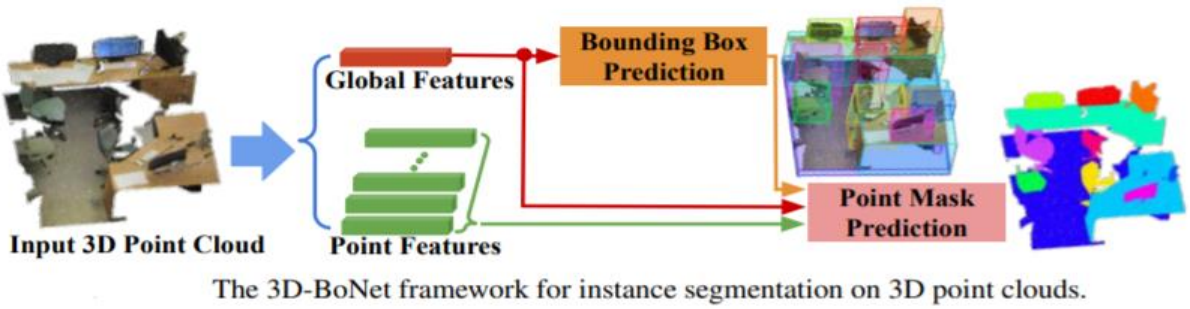


Figure C : DS Net

L'architecture du réseau à échelle dense (DSNet) proposé pour le comptage des foules. Le DSNet se compose d'un réseau de base avec les dix premières couches du VGG-16, trois blocs de convolution dilatés denses (DDCB) avec des connexions résiduelles denses (DRC), et trois couches convolutionnelles pour la régression de la carte de densité de foule. Les DDCB avec DRCs sont utilisés pour élargir la diversité d'échelle et les champs réceptifs des caractéristiques pour traiter les variations à grande échelle afin que les cartes de densité puissent être estimées avec précision.

- 3D-BoNet : Segmentation d'objets en images 3D. La résolution du problème de la segmentation des instances est 10 fois meilleure, en termes de calcul, que les autres approches existantes. Réseau neuronal de bout en bout qui accepte une image 3D en entrée, et donne la limite des objets reconnus en sortie.



The 3D-BoNet framework for instance segmentation on 3D point clouds.

Figure D : 3D BoNet

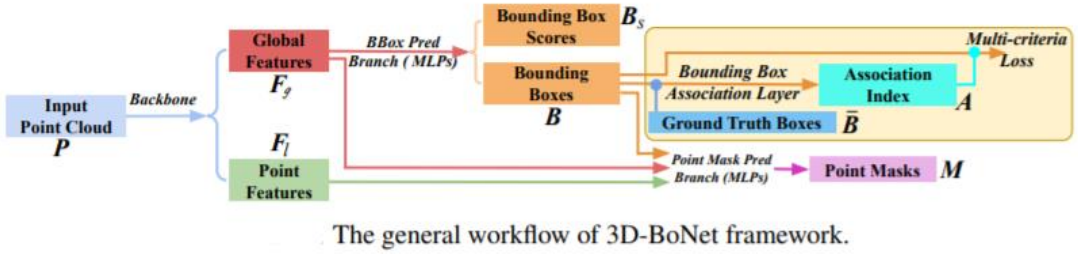


Figure E : 3D BoNet Architecture

- Raisonnement - RCNN : Reconnaissance d'objets parmi des milliers de catégories. Détection d'objets difficiles à voir dans l'image. Une architecture qui vous permet de travailler sur n'importe quel détecteur existant.

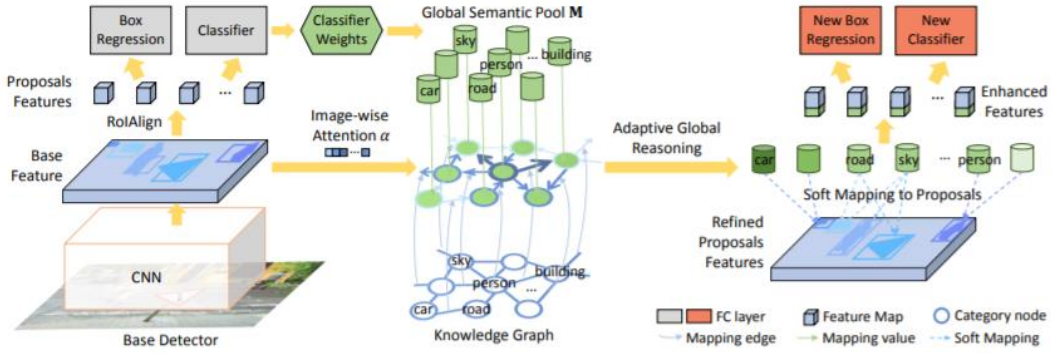


Figure F : RCNN Architecture

Le RCNN de raisonnement peut être empilé sur n'importe quel détecteur de base existant, tel que le RCNN rapide. Les poids du classificateur précédent sont collectés pour générer un pool sémantique global sur toutes les catégories, qui est alimenté dans un module de raisonnement global adaptatif. Les contextes de catégories améliorés (c'est-à-dire les sorties du module de raisonnement) sont mis en correspondance avec les propositions de régions par un mécanisme de mise en correspondance souple. Enfin, les caractéristiques améliorées de chaque région sont utilisées pour améliorer les performances de la classification et de la localisation de bout en bout.

- Mask R-CNN : Modélisation de la forme 3D d'objets à partir d'une image. Prédiction de la forme 3D des objets dans l'image d'entrée. Modèle de bout en bout.

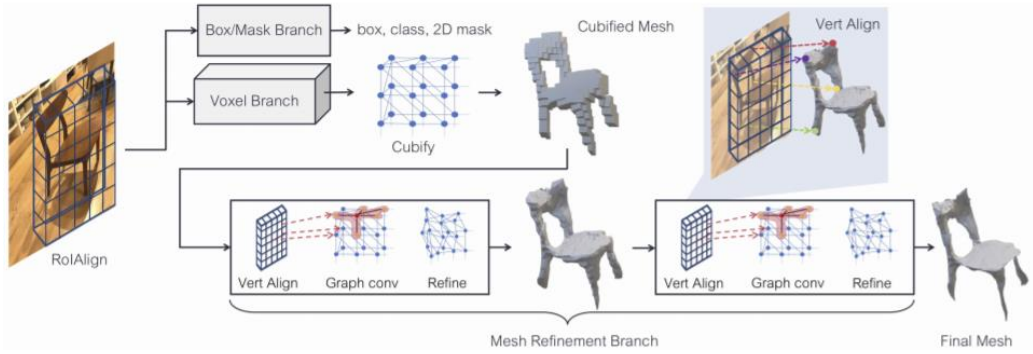


Figure G : Mask R-CNN

Le masque R-CNN est complété par une inférence de forme 3D. La branche de voxel prédit une forme grossière pour chaque objet détecté, qui est ensuite déformée par une séquence d'étapes de raffinement dans la branche de raffinement du maillage.

- L'efficacité de la mise à l'échelle du modèle dépend aussi fortement du réseau de base. Par conséquent, pour améliorer encore les performances, un nouveau réseau de base a été développé en effectuant une recherche d'architecture neurale à l'aide du cadre AutoML MNAS, qui optimise à la fois la précision et l'efficacité (FLOPS). L'architecture qui en résulte utilise la convolution mobile à goulot d'étranglement inversé (MBConv), similaire à MobileNetV2 et MnasNet, mais est légèrement plus grande en raison d'un budget FLOP accru. Nous avons ensuite élargi le réseau de base pour obtenir une famille de modèles, appelés EfficientNet.

Figure H : Efficient Net

Efficient Net : permet d'obtenir une précision et une efficacité bien meilleures que les ConvNets précédents. En particulier, notre EfficientNet-B7 atteint une précision de pointe de 84,3% dans le top 1 d'ImageNet, tout en étant 8,4x plus petit et 6,1x plus rapide en inférence que le meilleur ConvNet existant.

Revue de littérature :

Actuellement, la détection d'objets dans les véhicules basée sur la vision est divisée en deux catégories : les méthodes traditionnelles de vision artificielle et les méthodes complexes d'apprentissage en profondeur. Les méthodes traditionnelles de vision artificielle utilisent le mouvement d'un véhicule pour le séparer d'une image d'arrière-plan fixe. Cette méthode peut être divisée en trois catégories : la méthode d'utilisation de la soustraction de l'arrière-plan, la méthode d'utilisation de la différence de trame vidéo continue et la méthode d'utilisation du flux optique. Avec la méthode de la différence d'images vidéo, la variance est calculée en fonction des valeurs en pixels de deux ou trois images vidéo consécutives. De plus, la région d'avant-plan en mouvement est séparée par le seuil [3]. En utilisant cette méthode et en supprimant le bruit, l'arrêt du véhicule peut également être détecté.

➢ Automatic vehicle counting system for traffic monitoring:

Sur la même base de données cet article se base pour présenter un système pour le comptage des véhicules routiers et classification. Le système est capable de réaliser un comptage avec une très bonne précision même dans des scénarios difficiles liés aux occlusions et/ou à la présence d'ombres. Le principe du système est d'utiliser des caméras déjà installées dans les routes sans aucune procédure de calibrage supplémentaire. Et propose un algorithme de segmentation robuste qui détecte les pixels de premier plan correspondant aux véhicules en mouvement.
La méthode développée dans le cadre de ce travail est capable de gérer les ombres avec une haute résolution. Et a été testé et comparé à une méthode classique. Les résultats expérimentaux basés sur quatre grands ensembles de données montrent que notre méthode peut compter et classer les véhicules en temps réel avec un haut niveau de performance (>98%) dans différentes situations environnementales.
https://core.ac.uk/download/pdf/83533829.pdf

➢ Vehicle counting method based on digital image processing algorithms :

Par Tourani, Ali & Shahbahrami, Asadollah. (2015).
Publié sur et IEEE Xplore : 1-6. 10.1109/PRIA.2015.7161621.
Cet article présente un compteur-classificateur de véhicules basé sur une combinaison de différentes méthodes de traitement d'images vidéo, y compris la détection d'objets, la détection de bords, la différenciation d'images et le filtre de Kalman. Une implémentation de la technique proposée a été réalisée en utilisant le langage de programmation C++. La performance de la méthode en termes de

précision du comptage et de la classification des véhicules a été évaluée, ce qui a permis d'obtenir une précision d'environ 95 % pour la classification et une erreur d'environ 4 % pour les cibles de détection des véhicules.

➢ 2019 International Conference on Emerging Trends in Science and Engineering (ICESE) :

Publié par IEEE lors de cette conférence, le 18-19 Sept. 2019.
Cet article présente un moyen de détecter, de compter et de classer les véhicules à l'aide de techniques de traitement d'images. Bien qu'un grand nombre de recherches aient été menées à ce sujet, il y a toujours une marge d'amélioration.
La tâche de détection et de comptage des véhicules est divisée en six étapes :
1) Acquisition d'images,
2) Analyse d'images,
3) Détection d'objets,
4) Comptage,
5) Classification,
6) Affichage du résultat.
Les algorithmes qui seront utilisés pour effectuer ces tâches comprendront l'algorithme de détection et de comptage des véhicules et l'algorithme de détection du marquage routier. Cela peut également être utilisé pour surveiller les voies en hauteur, détecter les accidents, les arrêts injustifiés de véhicules sur les routes, les infractions au code de la route. La classification des véhicules se fera dans l'une des catégories suivantes :
Bicyclettes et motocyclettes, voitures à moteur, minibus et camionnettes, bus, remorques, camions.
Ces données permettront de déterminer la priorité et le nombre maximum d'usagers d'une route et de concevoir des schémas de circulation qui seront bénéfiques au maximum.
https://ieeexplore.ieee.org/document/9194678

➢ Traffic Light Detection with Google Object Detection API

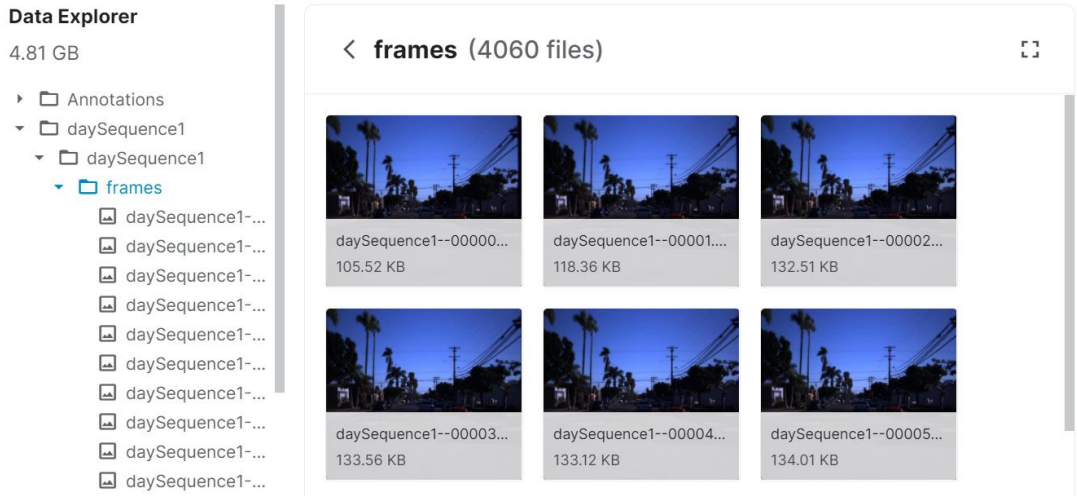
Un projet réalisé sur la base de données que j'utilise pour détecter les Traffic lights était publié sur GitHub.
Il se base sur un pre-trained Model : Faster_RCNN_inception_v2_coco et en utilisant Python et Tenserflow.
Ou l'auteur a choisi de ne pas redimensionner les images d'entrée et de conserver la taille de l'ancre. Pour les objets plus petits, comme les ampoules des feux de circulation. Quant aux détections maximales par classe et au nombre total de détections, 10 est suffisant pour la plupart des cas.
https://github.com/evanchien/LISA_TL_DETECTION

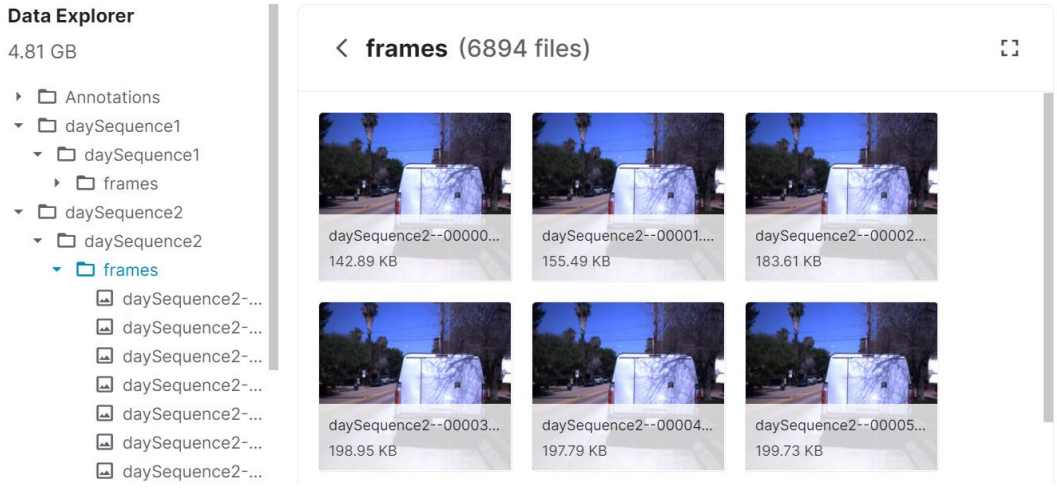
➢ Learning to Remove Rain in Traffic Surveillance by Using Synthetic Data:

La pluie est un problème dans la surveillance automatisée du trafic. Les traînées de pluie occultent les usagers de la route et dégradent la visibilité globale qui, à son tour, diminue les performances de détection des objets.
Un moyen de remédier à cette situation est d'enlever artificiellement la pluie des images. Pour ce faire, il faut connaître les images correspondantes, qu'elles soient pluvieuses ou non. Ces images sont souvent produites en superposant de la pluie synthétique à des images sans pluie.
https://vbn.aau.dk/ws/files/291869643/Bahnsen_LearningToRemove_CameraReady.pdf

Base de données :

La base de données est collectée à San Diego, en Californie, aux États-Unis, et fournit quatre séquences de jour et deux séquences de nuit utilisées principalement pour les tests, offrant **23 minutes et 25 secondes de conduite** à Pacific Beach et La Jolla, San Diego. La stéréo
Les paires d'images sont acquises à l'aide du bourdon XB3 de Point Grey (BBX3-13S2C-60) qui contient trois lentilles qui capturent des images avec **une résolution de 1280 x 960**, chacune avec un champ de vision (FoV) de 66°. La vue de la caméra de gauche est utilisée pour toutes les séquences de test et les clips d'entraînement. Les clips de formation se composent de **13 clips de jour et de 5 clips de nuit**.





Présentation de la solution :

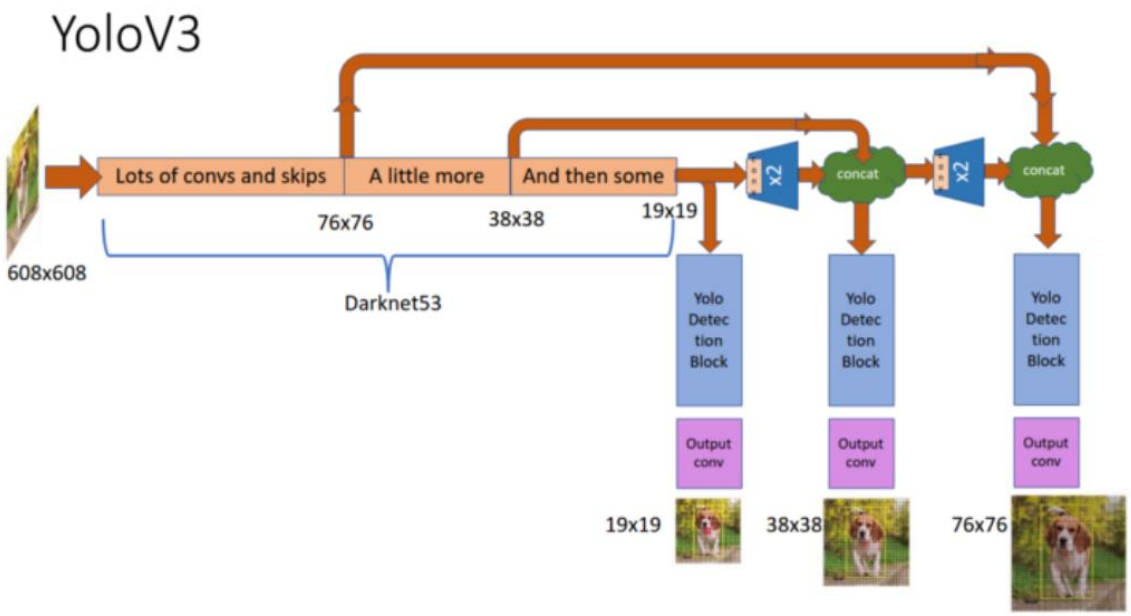
Le modèle YOLO v3 :

YOLO : Détection d'objets en temps réel
You Only look once (YOLO) est un système de pointe de détection d'objets en temps réel. Sur un Pascal Titan X, il traite les images à 30 FPS et a une mAP de 57,9% sur le test COCO.
Les systèmes de détection antérieurs réutilisent les classificateurs ou les localisateurs pour effectuer la détection. Ils appliquent le modèle à une image à plusieurs endroits et à plusieurs échelles. Les régions de l'image ayant un score élevé sont considérées comme des détections.

YOLO utilise une approche totalement différente. il applique un seul réseau de neurones à l'ensemble de l'image. Ce réseau divise l'image en régions et prédit les boîtes englobantes et les probabilités pour chaque région. Ces boîtes limites sont pondérées par les probabilités prédites.
Ce modèle présente plusieurs avantages par rapport aux systèmes basés sur des classificateurs. Il examine l'ensemble de l'image au moment du test, de sorte que ses prédictions sont influencées par le contexte global de l'image. Il fait également des prédictions avec une seule évaluation de réseau, contrairement à des systèmes comme R-CNN qui en nécessitent des milliers pour une seule image. Cela le rend extrêmement rapide, plus de 1000x plus rapide que le R-CNN et 100x plus rapide que le Fast R-CNN.

L'architecture du modèle :

Inspiré des architectures ResNet et FPN (Feature-Pyramid Network), l'extracteur de fonctionnalités YOLO-V3, appelé Darknet-53 (il a 52 convolutions) contient des connexions de saut (comme ResNet) et 3 têtes de prédiction (comme FPN) - chacune traitant l'image à une compression spatiale différente.
Comme son prédécesseur, le Yolo-V3 offre de bonnes performances sur une large gamme de résolutions d'entrée. Testé avec une résolution d'entrée de 608x608 sur l'ensemble de validation COCO-2017, le Yolo-V3 a obtenu un score de 37 mAP (précision moyenne moyenne). Ce score est identique à la version de formation de GluonCV plus rapide-RCNN- ResNet50, (une architecture plus rapide RCNN qui utilise ResNet50 comme son épine dorsale), mais 17 fois plus rapide. Dans ce modèle de zoo, les seuls détecteurs assez rapides pour rivaliser avec le Yolo-V3 (architectures Mobilenet-SSD) ont obtenu un mAP de 30 et moins.



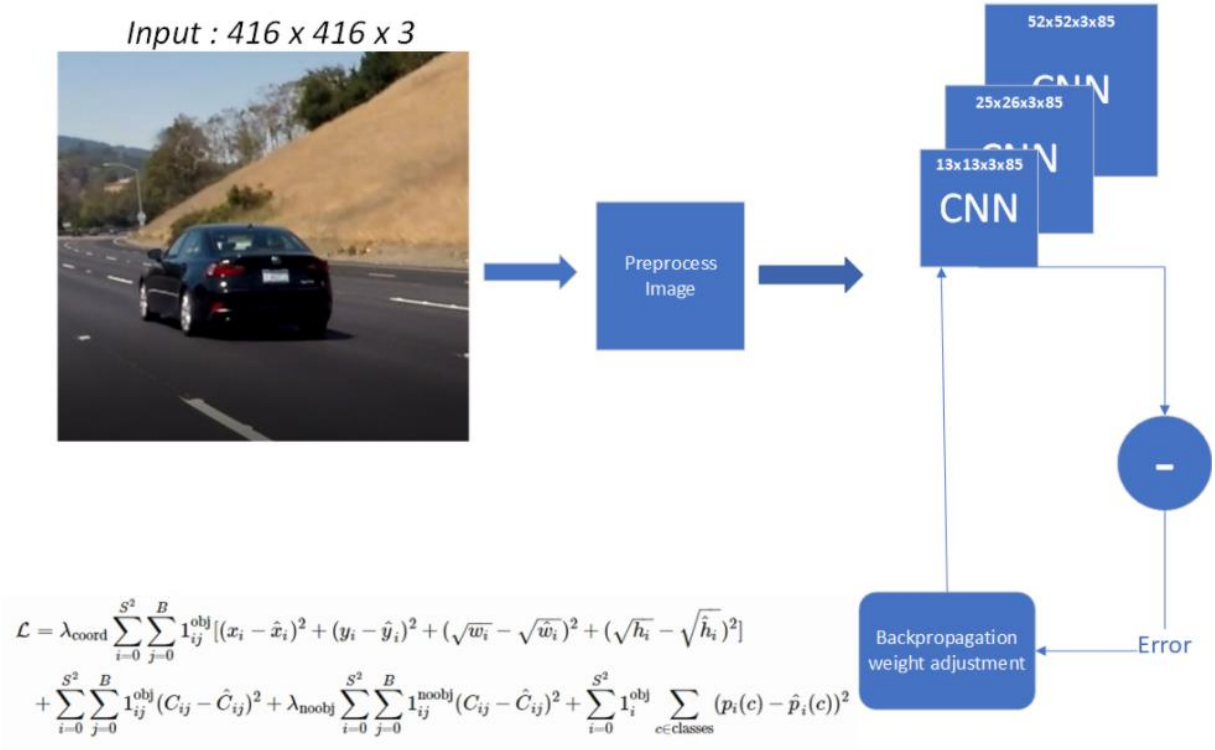
Réseau neuronal io :

Input : (m, 416, 416, 3)
Sortie : confidence de la présence d'un objet dans le rectangle, liste de la position des rectangles et des tailles et classes des objets commencent à être détectés.
Chaque rectangle délimité est représenté par 6 nombres (Pc, Rxd, Ry, Rh, Rd, C1.Cn) comme expliqué ci-dessus. Dans ce cas, n=80, ce qui signifie que nous avons c comme vecteur à 80 dimensions, et la taille finale de la représentation de la boîte englobante est de 85
La première détection est faite par la 82e couche. Pour les 81 premières couches, l'image est échantillonnée par le réseau, de sorte que la 81e couche a une foulée de 32. Si nous avons une image de 416 x 416, la carte de caractéristiques résultante serait de taille 13 x 13. Une détection est faite ici en utilisant le noyau de détection 1 x 1, ce qui nous donne une carte de caractéristiques de détection de 13 x 13 x 3 x 85.

Ensuite, la feature map de la couche 79 est soumise à quelques couches convolutives avant d'être échantillonnée par 2x à des dimensions de 26 x 26. Cette carte de caractéristiques est ensuite concaténée en profondeur avec la carte de caractéristiques de la couche 61. Ensuite, la carte d'entités combinée est à nouveau soumise à quelques couches convolutionnelles 1 x 1 pour fusionner les entités de la couche précédente (61). Ensuite, la deuxième détection est effectuée par la 94e couche, ce qui donne une carte des caractéristiques de détection de 26 x 26 x 3 x 85.

Une procédure similaire est suivie à nouveau, où la carte des caractéristiques de la couche 91 est soumise à quelques couches convolutives avant d'être concaténée en profondeur avec une carte des caractéristiques de la couche 36. Comme auparavant, quelques couches convolutionnelles 1 x 1 suivent pour fusionner les informations de la couche précédente (36). Nous réalisons la dernière des 3 couches à la 106e couche, ce qui donne une carte des caractéristiques de taille 52 x 52 x 3 x 85.

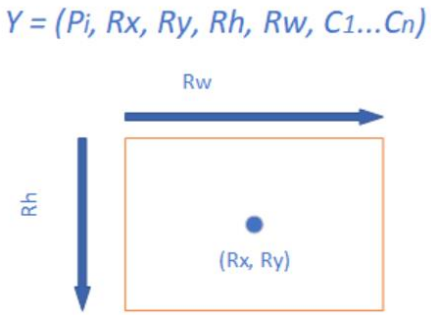
YOLO v3, utilise au total 9 boîtes d'ancrage. Trois pour chaque échelle.



$$\mathcal{L} = \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2]$$
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} (C_{ij} - \hat{C}_{ij})^2 + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{noobj}} (C_{ij} - \hat{C}_{ij})^2 + \sum_{i=0}^{S^2} 1_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

Implémentation et résultats :

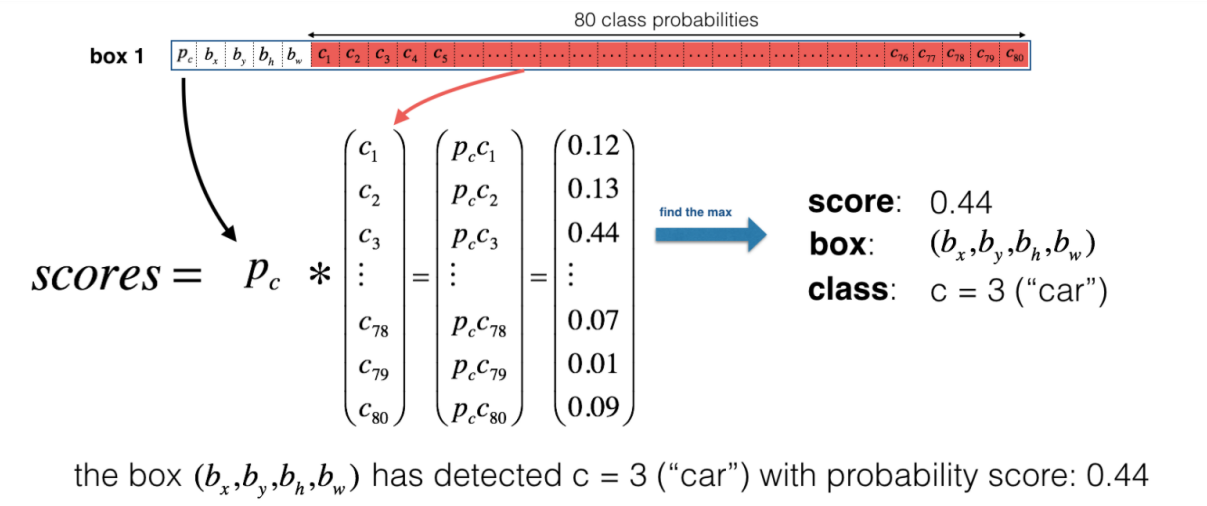
Dans ce projet, j'utilise des poids préentraînés, où nous avons 80 classes de yolo entraînées, pour la reconnaissance.
L'étiquette de la classe est représentée par c et c'est un nombre entier de 1 à 80, chaque nombre représente l'étiquette de la classe en conséquence.
Si c=3, alors l'objet classifié est un car :



Pi : Confidence of an object being present in the rectangle
C1..Cn : Class of the object being detected (Example Pi = 3 for "car")

Pour simplifier les choses, nous allons aplatir les deux dernières dimensions, de (19, 19, 5, 85) la sortie de notre CNN est aplatie à (19, 19, 425).

Maintenant, pour chaque case (de chaque cellule), nous allons calculer le produit élémentaire suivant et extraire une probabilité que la case contienne une certaine classe.



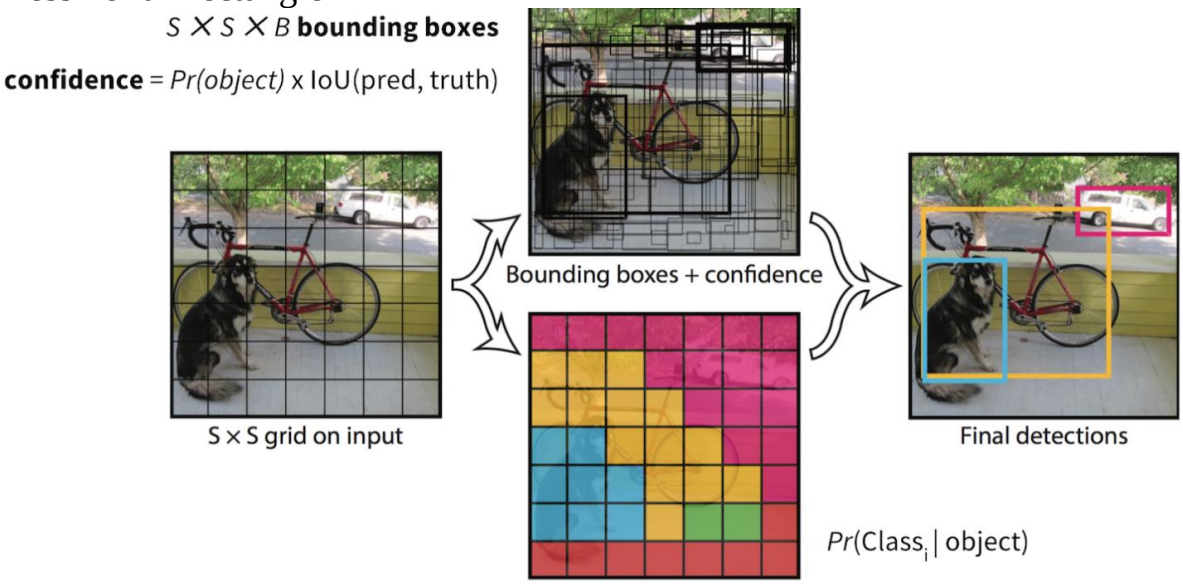
$$scores = p_c * \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_{78} \\ c_{79} \\ c_{80} \end{pmatrix} = \begin{pmatrix} p_c c_1 \\ p_c c_2 \\ p_c c_3 \\ \vdots \\ p_c c_{78} \\ p_c c_{79} \\ p_c c_{80} \end{pmatrix} = \begin{pmatrix} 0.12 \\ 0.13 \\ 0.44 \\ \vdots \\ 0.07 \\ 0.01 \\ 0.09 \end{pmatrix}$$

the box (b_x, b_y, b_h, b_w) has detected c = 3 ("car") with probability score: 0.44

Voici un exemple de visualisation de ce que YOLO peut prédire sur une image :

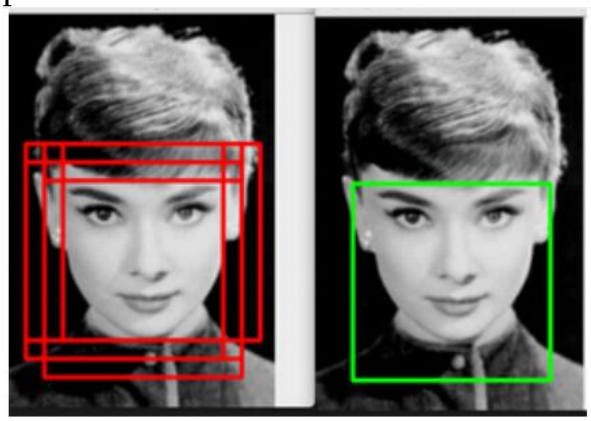
Pour chacune des cellules de la grille SxS , trouvez le maximum des scores de confiance (en prenant un maximum à travers les 5 cases d'ancrage et à travers les différentes classes).
Un score de confiance est : probabilité (contenant un objet) x IoU (préd, vérité).
Si la cellule contient un objet, elle prédit une probabilité que cet objet appartienne à une classe Ci, i=1,2,..., K : probability(the object belongs to the class Ci | containing an object). À ce stade, le modèle ne prédit qu'un seul

ensemble de probabilités de classe par cellule, quel que soit le nombre de cases B.
Coloriez cette cellule de la grille en fonction de l'objet que cette cellule considère comme le plus probable.
Dessiner un rectangle

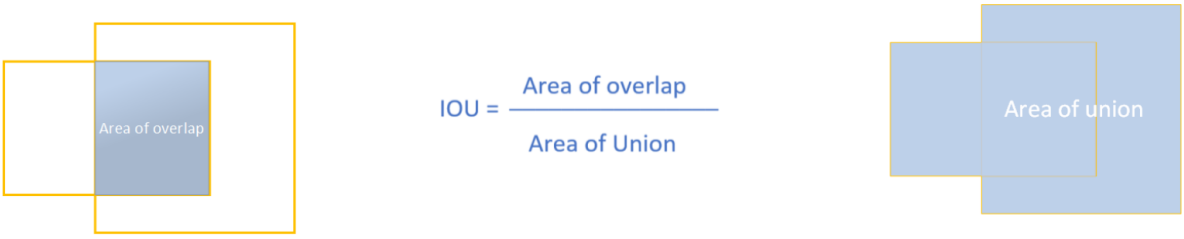


Même après un filtrage yolo par dépassement de seuil, nous avons encore beaucoup de boîtes qui se chevauchent. La deuxième approche et le filtrage est l'algorithme de suppression Non-Max.

Éliminer toutes les boîtes avec Pc<=0.6.
Alors qu'il ne reste plus que des cases :
Choisissez la case avec le plus grand Pc
Produire cela comme une prédiction
Jeter toutes les cases restantes avec IOU>=0,5 avec la sortie de la case à l'étape précédente



La suppression non-max utilise la fonction très importante appelée "Intersection over Union ", ou IoU.



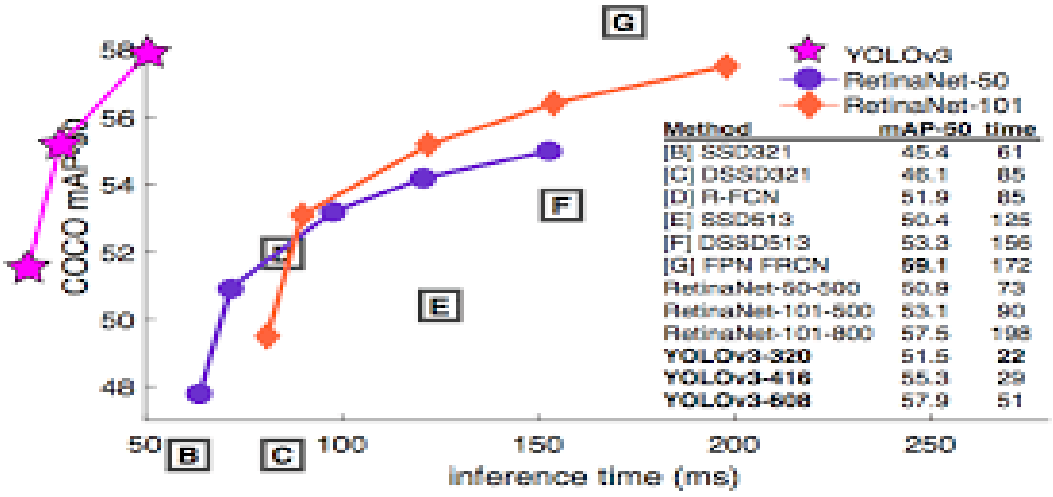
Interprétation des résultats

Analyse des résultats :

Le but du projet est accompli en utilisant OpenCV comme montrée lors de l'exécution test mais l'utilisation du modèle de machine learning YoLov3 qui permettra d'avoir plus d'exactitude et meilleur analyse .

Performance et discussion :

L'utilisation du modèle VGG16 a permis de classifier les images de la base en tant qu'images de traffic lights et n'a pas pu préciser l'existence des véhicules parmi les 5 premières classes prédites, pour l'algorithme de HOG avec le modèle de classification SVM Linear on a pas pu avoir une faible exactitude de cadrage et détection des véhicules mais en utilisant la librairie OpenCV on a pu établir un système de compte et détection des véhicules sur pycharm. Par contre YoLov3 permet une analyse rapide et exacte en termes de détection des véhicules au sein d'une image.



Method	mAP-50	time
[B] SSD321	45.4	61
[C] DSSD321	46.1	85
[D] R-FCN	51.9	85
[E] SSD513	50.4	125
[F] DSSD513	53.3	156
[G] FPN FRCN	59.1	172
RetinaNet-50-500	50.9	73
RetinaNet-101-500	53.1	90
RetinaNet-101-800	57.5	198
YOLOv3-320	51.5	**22**
YOLOv3-416	55.3	29
YOLOv3-608	57.9	51

Conclusion et recommandations

Un système de comptage de véhicules a été développé en utilisant Open CV en suivre les mouvements des véhicules. Le comptage est simplement effectué en évaluant la distance entre le centriole du véhicule et la ligne de frontière. Il a atteint avec succès la plus haute précision de 97,72% en utilisant la vidéo avec zoom frontside-1x.
YOLOv3 joue un rôle important dans la détection des véhicules, puisque le comptage ne s'effectue que par rapport à l'objet détecté. Toute amélioration devrait être faite pour obtenir un meilleur système.
Le count est possible avec des modifications de code de base de YoLov4, alors pour que le projet soit bien établie et complet sur colab et pas seulement sur local avec OpenCv il faut finaliser le travail débuté par YoLo v3 en établissant un système de détection et count avec YoLov 4.

Bibliographie :

- VGG16 : https://www.analyticsvidhya.com/blog/2020/08/top-4-pre-trained-models-for-image-classification-with-python-code/

- DS NET : https://arxiv.org/pdf/1906.09707.pdf

- 3D BO Net: https://arxiv.org/pdf/1906.01140.pdf

- Mask RCNN : https://arxiv.org/pdf/1906.02739.pdf

- -Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi - You Only Look Once: Unified, Real-Time Object Detection https://arxiv.org/abs/1506.02640
- Joseph Redmon, Ali Farhadi - YOLO9000: Better, Faster, Stronger https://arxiv.org/abs/1612.08242
- Allan Zelener https://github.com/allanzelener/YAD2K
- The official YOLO website (https://pjreddie.com/darknet/yolo/)
- Vivek Yadav https://medium.com/@vivek.yadav/part-1-generating-anchor-boxes-for-yolo-like-network-for-vehicle-detection-using-kitti-dataset-b2fe033e5807
- Christopher Bourez's blog Bounding box object detectors: understanding YOLO, You Look Only Once http://christopher5106.github.io/object/detectors/2017/08/10/bounding-box-object-detectors-understanding-yolo.html
- Andrew NG Coursera : https://www.coursera.org/learn/convolutional-neural-networks
- Jonathan Hui : https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088

- Liens au Notebooks :

  YoLov3 : https://colab.research.google.com/drive/160Yqr7h1bqn-GxpeXD9L_6MjK-XIWwyF?usp=sharing

  VGG16 et SVMLinear : https://colab.research.google.com/drive/1amrh0dwnSa1cyvct8WiGgINuoY8waby2?usp=sharing