Ingénieur Machine Learning

Projet 7

Développez une preuve de concept

II. État de l’art de la détection d’objets

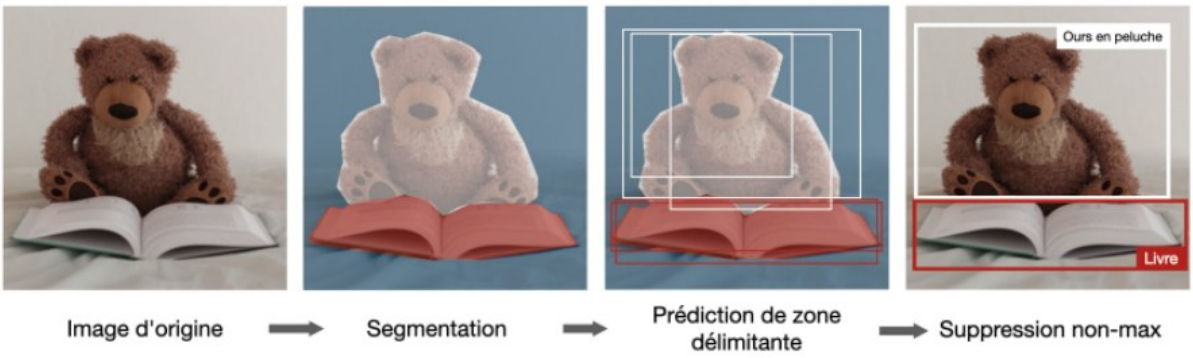
Il y a 3 principaux types d'algorithme de reconnaissance d'objet, pour lesquels la nature de ce qui est prédit est différent. Ils sont décrits dans la table ci-dessous :

Classification d'image	Classification avec localisation	Détection
		
• Classifie une image • Prédit la probabilité d'un objet	• Détecte un objet dans une image • Prédit la probabilité de présence d'un objet et où il est situé	• Peut détecter plusieurs objets dans une image • Prédit les probabilités de présence des objets et où ils sont situés
CNN traditionnel	YOLO simplifié, R-CNN	YOLO, R-CNN

La détection d’objet sur une image regroupe la classification de cet objet, ainsi que sa localisation. Cette dernière se visualise grâce à un rectangle d’ancrage (anchor box) encadrant la cible. Le jeu de données se compose souvent d’images et de fichiers textes regroupant la classe ainsi que les coordonnées du centre du rectangle d’ancrage, ainsi que sa longueur et sa hauteur.

Différentes architectures d’algorithme ont permis d’effectuer de la détection d’objet. Dans un premier temps, il y a eu l’utilisation de CNN de classification, auxquels se greffait une partie supplémentaire permettant de prédire les valeurs relatives à la localisation. Les algorithmes de type R-CNN, Fast R-CNN et Faster R-CNN ont permis d’obtenir des résultats de plus en plus impressionnants

❐ **R-CNN** — L'algorithme de région avec des réseaux de neurones convolutionnels (en anglais *Region with Convolutional Neural Networks*) (R-CNN) est un algorithme de détection d'objet qui segmente l'image d'entrée pour trouver des zones délimitantes pertinentes, puis fait tourner un algorithme de détection pour trouver les objets les plus probables d'apparaître dans ces zones délimitantes.



Remarque : bien que l'algorithme original soit lent et coûteux en temps de calcul, de nouvelles architectures ont permis de faire tourner l'algorithme plus rapidement, tels que le Fast R-CNN et le Faster R-CNN.

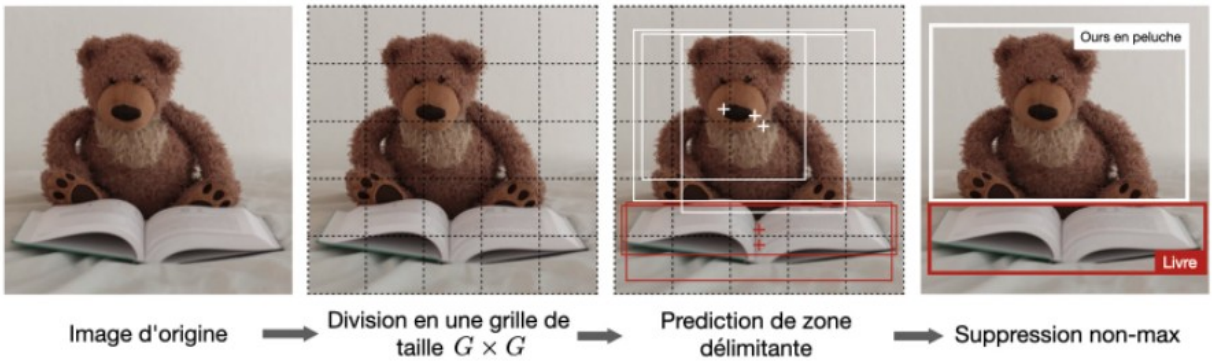
Puis, des réseaux de neurones entièrement convolutifs ont été développés. On parle alors de R-FCN. Ces réseaux de neurones ont surpassé les performances des meilleurs Faster R-CNN.Enfin, une avancé majeure a été d'utiliser un R-FCN en appliquant une méthode dite singleshot.
Les algorithmes plus anciens faisaient une première passe afin de trouver des régions d'intérêt,puis les analysaient. Ici, l'algorithme effectue la détection en une seule passe. Les algorithmes utilisant cette méthodologie sont SSD ou YOLO.
Afin de mesurer les performances d'un algorithme de détection d'objet, nous pouvons utiliser les métriques classiques pour la classification et la métrique mAP (avec seuil IoU)

❒ **YOLO** — L'algorithme You Only Look Once (YOLO) est un algorithme de détection d'objet qui fonctionne de la manière suivante :

• Étape 1 : Diviser l'image d'entrée en une grille de taille $G \times G$.
• Étape 2 : Pour chaque cellule, faire tourner un CNN qui prédit y de la forme suivante :

$$y = \left[\underbrace{p_c, b_x, b_y, b_h, b_w, c_1, c_2, ..., c_p, ...}_{\text{répété } k \text{ fois}}\right]^T \in \mathbb{R}^{G \times G \times k \times (5+p)}$$

où p_c est la probabilité de détecter un objet, b_x, b_y, b_h, b_w sont les propriétés de la zone délimitante détectée, $c_1, ..., c_p$ est une représentation binaire (en anglais *one-hot representation*) de l'une des p classes détectée, et k est le nombre de zones d'accroche.
• Étape 3 : Faire tourner l'algorithme de suppression non-max pour enlever des doublons potentiels qui chevauchent des zones délimitantes.



Remarque : lorsque $p_c = 0$, le réseau ne détecte plus d'objet. Dans ce cas, les prédictions correspondantes $b_x, ..., c_p$ doivent être ignorées.

Nous allons donc essayer de prouver l efficacité des ces nouveaux modèles par rapport aux meilleurs modèles du précédent projet (Xception), nous allons les comparer sur plusieurs dataset

III. Les jeux de données

Le modele Xception a été pré-entraîné sur ImageNet.
Or les images de StandfordDog dataset en font partie, cela veut dire que l expérience du précédent projet était biaisé , en effet le modèle connaissait déjà les images que nous lui avons donner, ce dataset n est pas un bonne «arbitre» pour comparer nos modèles.
Pour ne pas biaisé l expérience , nous allons utiliser deux autres dataset qu'aucun de nos modèles ne connaissent .

Le premier est un dataset sur les panneaux routier, et l autre un dataset sur les globules blanc (permet de détecter les globules blanc défaillant)

- standforddog dataset (http://vision.stanford.edu/aditya86/ImageNetDogs/)
- road signs (https://arcraftimages.s3-accelerate.amazonaws.com/Datasets/RoadSigns/RoadSignsPascalVOC.zip)
- white cells (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7182702/#!po=70.0000)

Avoir un dataset conséquent est indispensable en machine learning , heureusement la communauté deep learning a bien compris cela. Et en 2022 , nous pouvons récupérer pas mal de données dans divers domaine .

PASCAL VOC to YOLO :

Un travail de tuyauterie a du être nécessaire afin de convertir certaines annotations au format PASCAL VOC vers le format YOLO (standford dog et roadsign), (cf notebook annexes)

Concernant les dataset globules blancs :

La détection d objet révolutionne le monde médical aujourd'hui , nous avons récupérer différent dataset mis a disposition par des hôpitaux ou laboratoires de recherches.

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7182702/#!po=70.0000
https://data.mendeley.com/research-data/
https://raabindata.com/free-data/
http://dl.raabindata.com/WBC/Cropped_double_labeled/
https://www.nature.com/articles/s41598-021-04426-x#Sec5
https://github.com/zxaoyou/segmentation_WBC
https://data.mendeley.com/datasets/snkd93bnjr/1
https://www.whitebloodcellclassifier.com/dataset.html
https://github.com/jacob-sycoff/aml-white-blood-cell-classifier
https://drive.google.com/drive/folders/1K-QqzRpd9G15DctrHG5_EKQC9V7mtReH
https://wiki.cancerimagingarchive.net/pages/viewpage.action?pageId=61080958#:~:text=The%20Munich%20AML%20Morphology%20Dataset,without%20signs%20of%20hematological%20malignancy.
https://www.kaggle.com/datasets/andrewmvd/bone-marrow-cell-classification

Afin de réaliser les annotations sur les images , nous avons du, dans un premier temp développer un modèle de machine learning qui permet de détecter un globule blanc (n importe quel type) (grâce aux dataset roboflow https://public.roboflow.com/object-detection/bccd/)
Une fois notre modèle prêt , nous l utilisons pour effectuer la détection de globule blanc sur un nouveau dataset (https://data.mendeley.com/datasets/snkd93bnjr/1) qui elle est classé par type de globule blanc.

Nous récupérons les résultats de toute nos bounding box, et nous mettons a jours les classes pour chaque types de globules .
Le nouveau dataset est ensuite tester sur nos différents modèles

IV. Modèles de référence

Nous allons utiliser comme modèle de référence un CNN utilisant les poids d un modèle Xception avec le transfert learning

RoadSigns :

	precision	recall	f1-score	support
crosswalk	0.73	0.61	0.67	18
speedlimit	0.96	0.94	0.95	138
stop	0.60	0.75	0.67	8
trafficlight	0.73	0.92	0.81	12
accuracy			0.90	176
macro avg	0.76	0.80	0.77	176
weighted avg	0.90	0.90	0.90	176

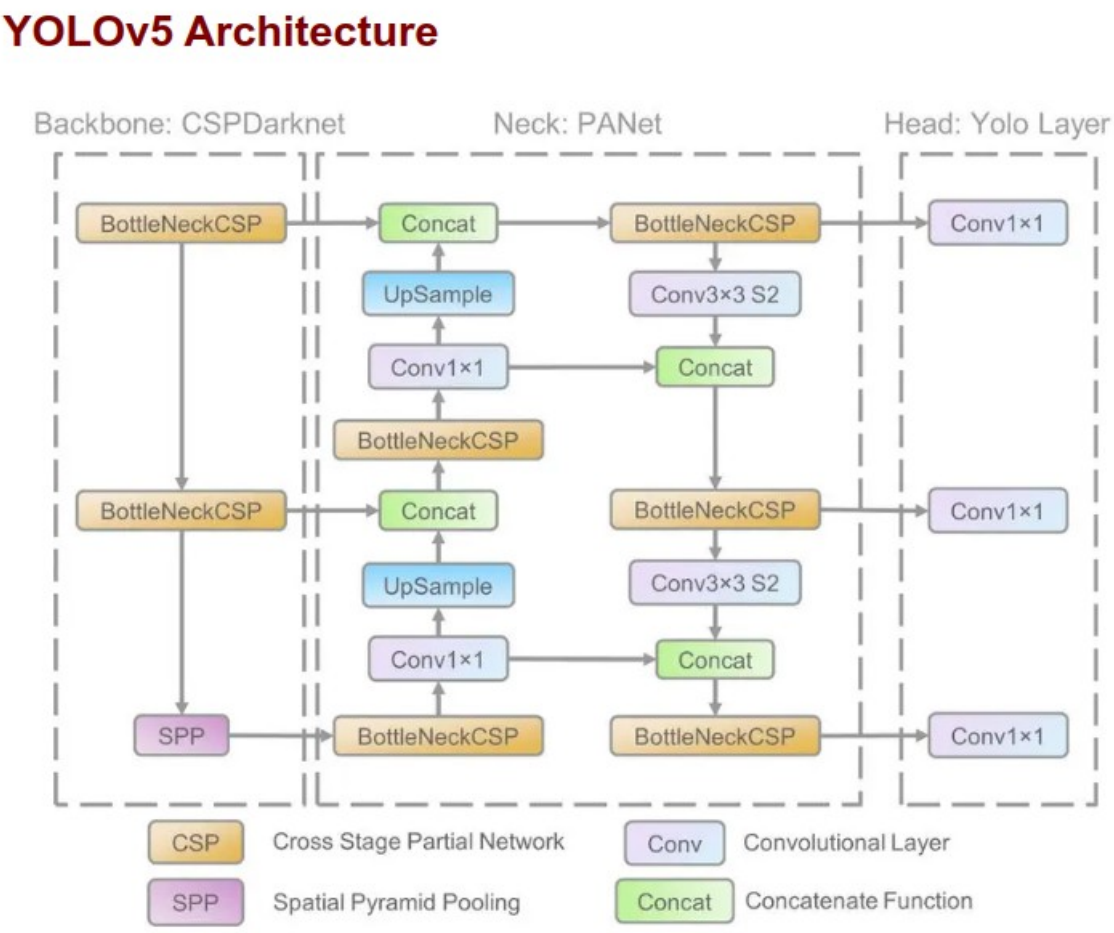
WhiteCells :

	precision	recall	f1-score	support
basophil	0.97	0.93	0.95	245
eosinophil	0.94	0.98	0.96	623
erythroblast	0.89	0.98	0.93	316
lymphocyte	0.88	0.99	0.93	228
monocyte	0.95	0.94	0.94	299
neutrophil	0.98	0.87	0.92	659
accuracy			0.94	2370
macro avg	0.94	0.95	0.94	2370
weighted avg	0.94	0.94	0.94	2370

XCEPTION	StandfordDogs	Roadsigns	Whitebloods
Loss	0.3741	0.1204	0.1576
Accuracy	0.89	0.76	0.94

Yolov5

Yolov5 est un concurrent sérieux dans le domaine de la détection , créer par la société ultralytics qui a été créer récemment , cette année une multitude de start up voit le jour avec comme objectif d avoir le meilleur modèles du marché.



Configuration de l algorithme :

Le projet YOLOv5 est récupérable à cette adresse : https://github.com/ultralytics/yolov5.

Le projet est développé sous le framework Pytorch
Afin d'utiliser le Transfer Learning, nous téléchargeons des poids pré-entraînés
La configuration du projet permet de modifier le nombre de classes à détecter, mais aussi de paramétrer la data augmentation à utiliser , ainsi qu une liste d hyperparametres dans les fichiers de configuration,
Le projet github yolov5 est très complet et orienté production , il est facilement implémentable par rapport aux autres modèles

Nous pouvons également fine tuner le modèle en utilisant le paramètre –freeze nblayers, faire du early stopping –patience nb etc.,

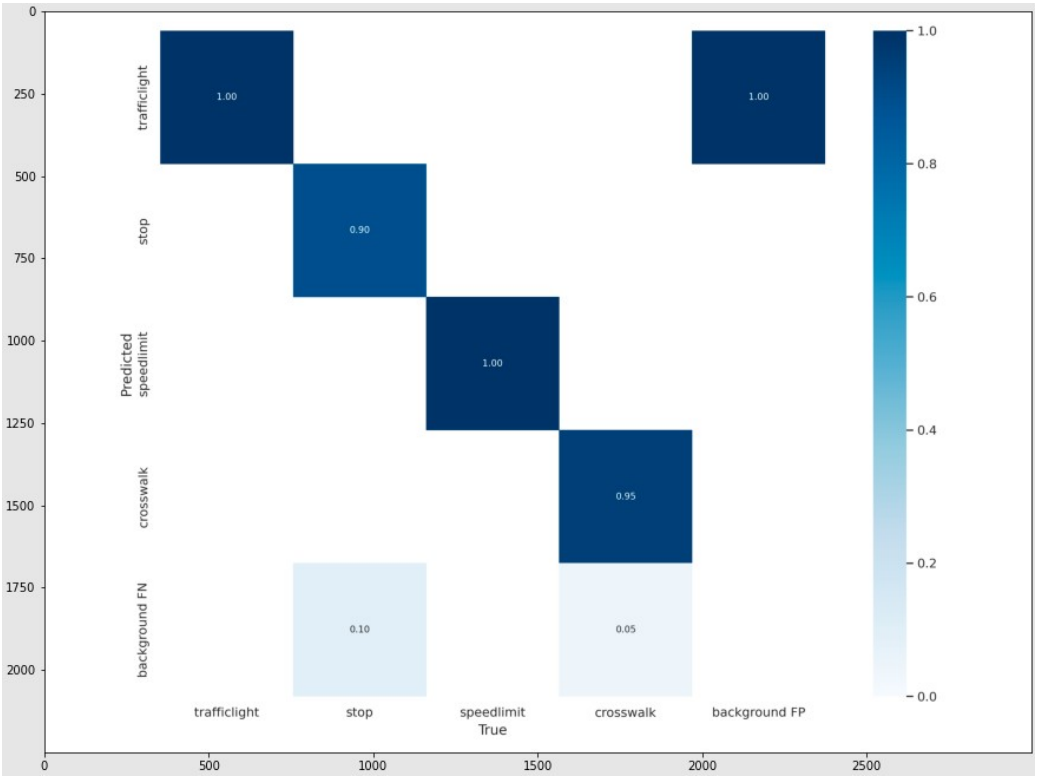
```
# YOLOv5  by Ultralytics, GPL-3.0 license
# Hyperparameters for medium-augmentation COCO training from scratch
# python train.py --batch 32 --cfg yolov5m6.yaml --weights '' --data coco.yaml --img 1280 --epochs 300
# See tutorials for hyperparameter evolution https://github.com/ultralytics/yolov5#tutorials

lr0: 0.01  # initial learning rate (SGD=1E-2, Adam=1E-3)
lrf: 0.1  # final OneCycleLR learning rate (lr0 * lrf)
momentum: 0.937  # SGD momentum/Adam beta1
weight_decay: 0.0005  # optimizer weight decay 5e-4
```

warmup_epochs: 3.0 # warmup epochs (fractions ok)
warmup_momentum: 0.8 # warmup initial momentum
warmup_bias_lr: 0.1 # warmup initial bias lr
box: 0.05 # box loss gain
cls: 0.3 # cls loss gain
cls_pw: 1.0 # cls BCELoss positive_weight
obj: 0.7 # obj loss gain (scale with pixels)
obj_pw: 1.0 # obj BCELoss positive_weight
iou_t: 0.20 # IoU training threshold
anchor_t: 4.0 # anchor-multiple threshold
anchors: 3 # anchors per output layer (0 to ignore)
fl_gamma: 0.0 # focal loss gamma (efficientDet default gamma=1.5)
hsv_h: 0.015 # image HSV-Hue augmentation (fraction)
hsv_s: 0.7 # image HSV-Saturation augmentation (fraction)
hsv_v: 0.4 # image HSV-Value augmentation (fraction)
degrees: 0.0 # image rotation (+/- deg)
translate: 0.1 # image translation (+/- fraction)
scale: 0.9 # image scale (+/- gain)
shear: 0.0 # image shear (+/- deg)
perspective: 0.0 # image perspective (+/- fraction), range 0-0.001
flipud: 0.0 # image flip up-down (probability)
fliplr: 0.5 # image flip left-right (probability)
mosaic: 1.0 # image mosaic (probability)
mixup: 0.1 # image mixup (probability)
copy_paste: 0.0 # segment copy-paste (probability)

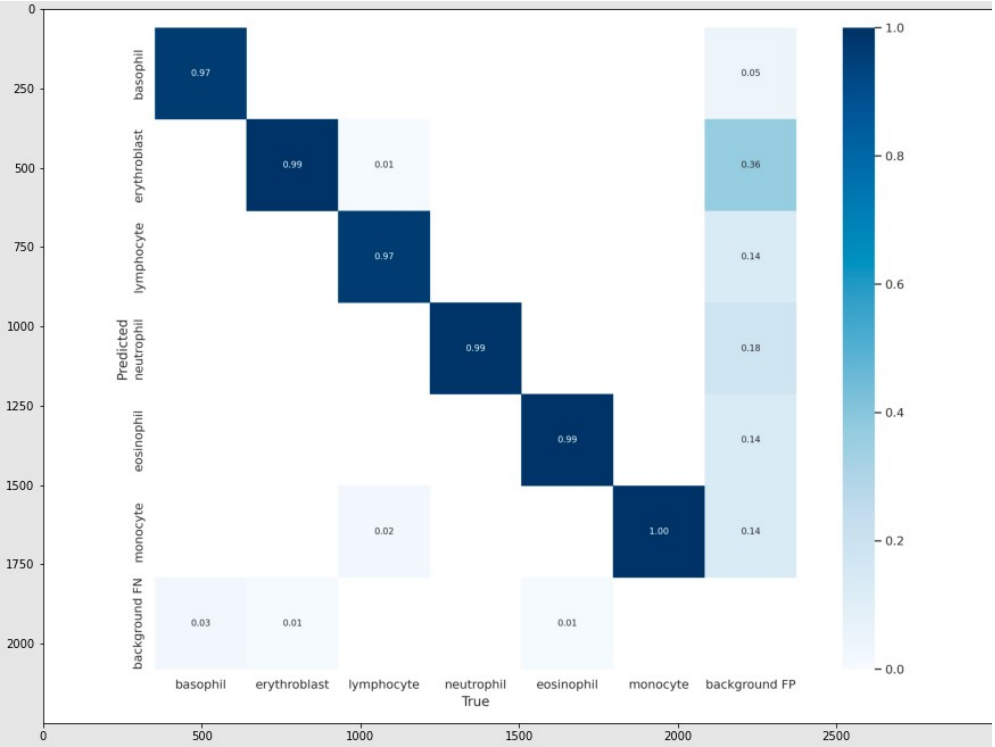
RoadSigns

```
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 3/3
[00:03<00:00,  1.08s/it]
                 all         88        126       0.98      0.914      0.942      0.796
         trafficlight         88         20      0.983       0.75      0.798      0.557
                 stop         88          7      0.984          1      0.995      0.901
           speedlimit         88         76          1      0.993      0.995      0.912
            crosswalk         88         23      0.955      0.913       0.98      0.812
```



WhiteCells

```
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 38/38
[00:09<00:00,  4.16it/s]
                 all       1185       1249      0.991      0.989      0.994      0.968
            basophil       1185        125       0.99      0.992      0.995      0.946
        erythroblast       1185        175      0.987      0.971       0.99      0.965
          lymphocyte       1185        115          1      0.986      0.995      0.973
          neutrophil       1185        334      0.991      0.994      0.995      0.984
          eosinophil       1185        341      0.994      0.997      0.995      0.989
            monocyte       1185        159      0.981      0.994      0.994      0.948
```

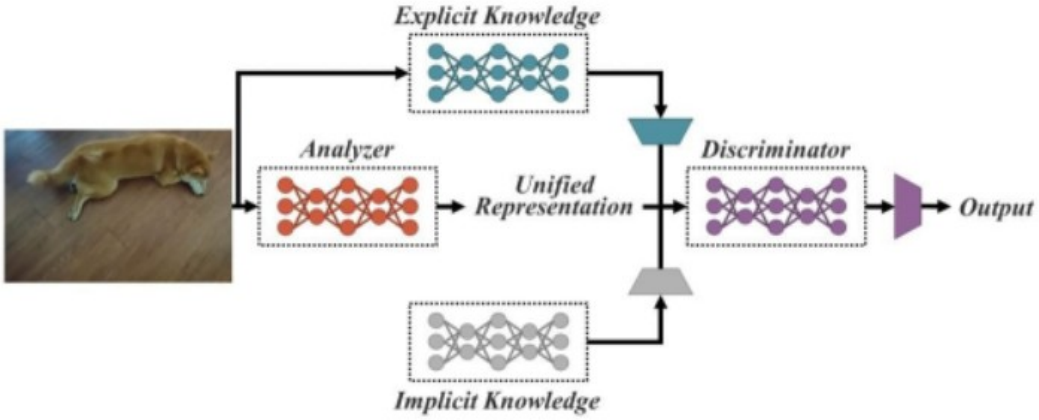


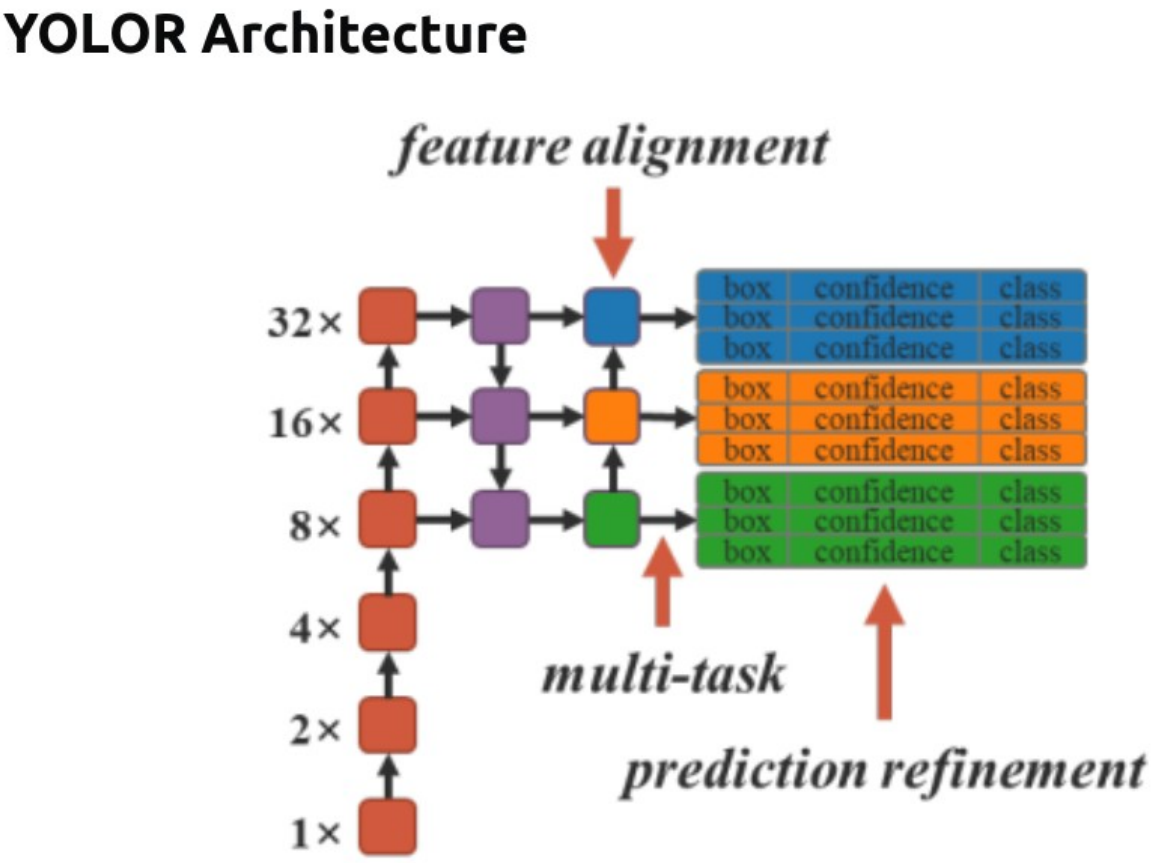
YOLOV5	StandfordDogs	Roadsigns	Whitebloods
Loss	0.00999	0.002799	0.00153
Accuracy	0.812	0.98	0.991

YoloR

https://github.com/WongKinYiu/yolor

YOLOR (You Only Learn One Representation) est un réseau unifié qui intègre des connaissances implicites et des connaissances explicites.Il pré-entraîne un réseau de connaissances implicites avec toutes les tâches présentes dans l'ensemble de données COCO pour apprendre une représentation générale, c'est-à-dire des connaissances implicites. YOLOR forme ensuite un autre ensemble de paramètres qui représentent des connaissances explicites pour une tâche spécifique.Les connaissances implicites et explicites sont utilisées pour l'inférence.





Roadsigns:

Class	Images	Targets	P	R	mAP@.5	mAP@.5:.95: 100%
all	88	132	0.854	0.947	0.948	0.762
trafficlight	88	19	0.785	0.947	0.941	0.587
stop	88	10	0.904	0.9	0.9	0.852
speedlimit	88	81	0.914	0.988	0.988	0.891
crosswalk	88	22	0.814	0.955	0.961	0.719

WhiteCells:

Class	Images	Targets	P	R	mAP@.5	mAP@.5:.95: 100%
all	1.18e+03	1.25e+03	0.962	0.992	0.994	0.969
basophil	1.18e+03	108	0.978	0.981	0.994	0.942
erythroblast	1.18e+03	195	0.961	0.99	0.992	0.967
lymphocyte	1.18e+03	135	0.928	0.993	0.995	0.978
neutrophil	1.18e+03	371	0.983	0.997	0.997	0.988
eosinophil	1.18e+03	313	0.985	0.99	0.997	0.99
monocyte	1.18e+03	131	0.939	1	0.989	0.952

YOLOR	Roadsigns	Whitebloods
Loss	0.01113	0.003741
Accuracy	0.85	0.96

Resultat final :

accuracy	StandfordDogs	Roadsigns	Whitebloods
Xception	0.89	0.76	0.94
YOLOR	X	0.85	0.96
YOLOV5	0.812	0.98	0.99

Conclusion:

Il est important de connaître les données d entraînement d un modèle (coco dataset, imagenet, google open image…), avant de pouvoir les comparer, en effet un modèle risque d être plus avantager qu'un autre si ce modèle connais déjà les images…

C est pour cela que le modele Xception dépasse le modèle Yolov5 sur le dataset standford dogs car le test est biaisé, alors que sur les deux autres dataset, yolov5 et yolor surclasse Xception en terme de performance et précision .

Nous avons donc prouver le concept d'etat de l art pour nos modèles Yolov5 et Yolor sur nos dataset de tests par rapport a Xception, le modèle Yolov5 se démarque du lot avec des résultats proches de 1 .

Sources bibliographiques et autres

Livre :

Deep Learning avec Keras et TensorFlow 2nd édition, A. Géron (2020)

Papier de recherche :

TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios 26 Aug 2021
Xingkui Zhu1 Shuchang Lyu1 Xu Wang Qi Zhao1
https://arxiv.org/pdf/2108.11539.pdf

YOLO5 Face: Why Reinventing a Face Detector

https://arxiv.org/pdf/2105.12931v3.pdf

27 May 2021·**Delong Qi**,**Weijun Tan**,**Qi Yao**,**Jingfeng Liu**

YOLOR You Only Learn One Representation: Unified Network for Multiple Tasks
https://paperswithcode.com/paper/you-only-learn-one-representation-unified
https://arxiv.org/pdf/2105.04206v1.pdf

Blogs :
Deep Learning for Object Detection : A Comprehensive Review , Joyce Xu
https://towardsdatascience.com/deep-learning-for-object-detection-a-comprehensive-review-%0A73930816d8d9 (2017)
R-CNN, Fast R-CNN, Faster R-CNN, YOLO – Object Detection Algorithms, R. Gandhi
https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-%0A36d53571365e (2018)

How to Train YOLOv5 On a Custom Dataset, Jacob Solawetz, Joseph Nelson
https://blog.roboflow.com/how-to-train-yolov5-on-a-custom-dataset/ (2020)

https://medium.com/augmented-startups/top-yolo-variants-of-2021-19dddc23043c

https://towardsdatascience.com/the-practical-guide-for-object-detection-with-yolov5-algorithm-74c04aac4843

https://medium.com/augmented-startups/yolov5-controversy-is-yolov5-real-20e048bebb08

https://blog.paperspace.com/train-yolov5-custom-data/#download-the-data

https://blog.roboflow.com/train-yolor-on-a-custom-dataset/