# Revue de littérature

## I*) Introduction*

*Victor RIOS*

Dans cette revue de littérature, je vais vous présenter l'ensemble des méthodes et des techniques utilisés en vision par ordinateur associé à l'apprentissage profond. C'est un sujet particulièrement en vogue en ce moment dans le monde du Deep Learning appliqué à l'informatique, l'ingénierie, la physique, la biologie… Nous nous focusserons par la suite sur les modèles de détection d'objets comprenant classification et localisation, que j'utiliserai pour mon projet. Premièrement, j'aimerais définir historiquement la computer vision par "the construction of explicit, meaningful descriptions of physical objects from images" (Ballard & Brown, 1982). Ainsi, si l'intelligence artificielle permet aux ordinateurs de penser, la vision par ordinateur leur permet de voir, d'observer et de comprendre.

Il existe néanmoins de nombreux domaines d'applications à la vision par ordinateur tel que :

- La robotique
- Le divertissement
- La sécurité
- La biométrie
- Les voitures intelligentes

**Définition du projet :**

Pour rappel, mon projet consiste en la détection rapide et efficace d'une menace sur une image à l'aide d'un algorithme de vision par ordinateur. Ainsi, mon modèle prendra en entrée une image avec un ou plusieurs objets et devra présenter en sortie un ou plusieurs cadres de délimitation et une étiquette de classe pour chaque cadre de délimitation. On se retrouve alors face à une tâche de machine learning supervisée utilisée pour prédire la classe (catégorie) d'une image, mais qui donne également un cadre englobant là où cette catégorie se trouve dans l'image. C'est donc bien une tâche de classification d'images car je souhaite en sortie de mon modèle une classe (discrète) mais comprenant également une étape de régression qui nous sortira les coordonnées de la boîte englobante.

**Base de données :**

Après de longues recherches, j'ai pu isoler quelques datasets potentiellement intéressantes pour notre projet. Premièrement, je me suis intéressé à Weapon Detection Dataset par Ankan Sharma disponible sur Kaggle (2021). Cette dataset était intéressante car présentait 2 classes (knife/gun) avec leurs boites englobantes associées (xmin, ymin, xmax, ymax). Cependant, l'équilibre entre les classes de données était assez faible et donc l'entrainement semblait trop long pour être efficace. Enfin, les résultats de performance des détecteurs sur cette dataset ne me convenait pas non plus ( <60 %).

En lisant l'article « *Object Detection Binary Classifiers methodology based on deep learning to identify small objects handled similarly » (*Francisco Pérez-Hernandez, 2020), j'ai découvert la dataset Sohas_weapon. Ce jeu de données propose un grand nombre de possibilités aussi bien

en classification qu'en détection, sur des armes tels que couteaux ou pistolets. Cependant, c'est la dataset Weapons and similar handled objects qui a retenu mon attention. En effet, ses ensembles de données incluent des armes et de petits objets qui sont manipulés de la même manière. Il contient six classes différentes telles que **pistolet**, **couteau**, **billet**, **portefeuille**, **smartphone** et **carte**. Les images de détection sont accompagnées des coordonnées de leurs boites englobantes pour chacun des objets à traiter. Elle présente également des données de tests ainsi que des biais, facilitant l'entraînement. Elle correspond ainsi parfaitement à mon projet et permet l'identification d'objets dangereux et d'objets du quotidien, comme souhaité.

| Database- | # img | Pistol | Knife | Smartphone | Bill | Purse | Card |
|---|---|---|---|---|---|---|---|
| Sohas_weapon-Detection | 3255 | 1425 | 1825 | 575 | 425 | 530 | 300 |
| Sohas_weapon-Test | 1170 | 294 | 470 | 115 | 123 | 104 | 64 |

*Figure 1 : Sohas Weapons and similar handled objects dataset*

**Introduction aux réseaux de neurones convolutifs (CNN) :**

Une première approche en vision par ordinateur serait d'utiliser un algorithme de Machine Learning « classique », comme la régression logistique ou bien une forêt aléatoire. Bien que ces approches obtiennent des résultats relativement corrects, ce type d'algorithmes ne pourra pas se généraliser aux images dont l'item se retrouverait dans un coin de l'image plutôt qu'au centre de celle-ci. En d'autres termes, le caractère spatial des éléments caractéristiques de certaines catégories n'est pas pris en compte

Ainsi, on a besoin d'utiliser un algorithme capable de détecter des formes relatives indépendamment de leur position dans l'image : c'est ce que permettent les Convolutionnal Neural Networks (CNN). En effet, les algorithmes de vision par ordinateur modernes sont basés sur des réseaux de neurones convolutifs, qui offrent une amélioration importante des performances par rapport aux algorithmes de traitement d'image traditionnels.

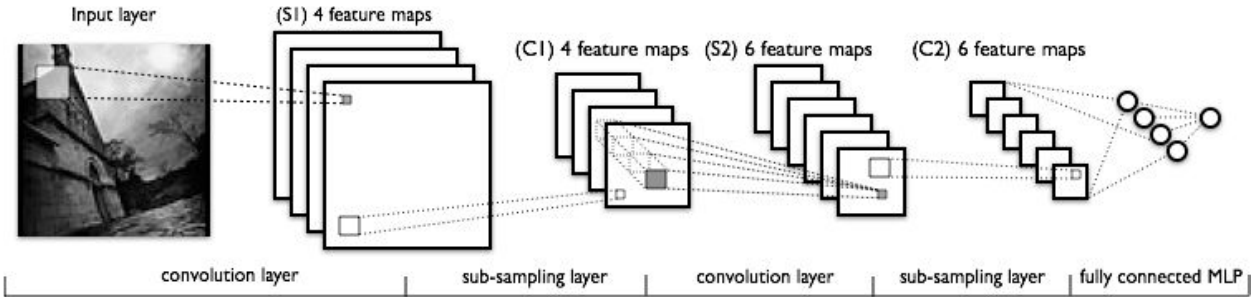


*Figure 2 : Réseaux de neurones convolutifs*

Un CNN classique alterne majoritairement deux types de couches :

- Couches avec filtre convolutif : on réalise un produit entre un filtre et l'input de la couche. La multiplication de ces couches au sein du réseau va permettre d'extraire des features de plus en plus complexes qui permettront enfin de prédire une classe d'appartenance pour l'item présent dans l'image.
- Couches avec pooling : elles permettent un undersampling qui va compresser la dimension de l'image et réduire le coût computationnel des couches suivantes. En général on utilise une fonction maximum ou moyenne.
- Les dernières couches aplatissent les features via une couche Flatten avant d'enchaîner avec des couches dense (FC pour Fully Connected). La dernière couche applique une fonction softmax, afin de déterminer la classe de l'image parmi les dix catégories.

Ainsi, d'après un benchmark effectué par l'entreprise Aquila sur le dataset Fashion MNIST, on peut observer un boost effectif en termes de performance prédictives sur les réseaux de neurones profonds, par rapport aux algorithmes de Machine Learning classique.

| Modèles | Paramètres / Architecture | accuracy |
|---|---|---|
| Régression Logistique | {C = 1, multi_class = ovr, penalty = l1} | 0.84 |
| Forêts Aléatoires | {criterion = entropy, max_depth = 50, n_estimators = 100} | 0.88 |
| Gradient Boosting | {loss = deviance, max_depth = 10, n_estimators = 100} | 0.89 |
| CNN | 2 Conv | 0.92 |
| CNN | 2 Conv + 3 FC | 0.93 |

*Figure 3 : Résultats des prédictions sur Fashion MNIST en fonction du modèle*

Enfin, on peut donc en déduire que l'utilisation d'un réseau de neurones convolutifs sera la solution la plus performante.

## II*) Techniques*

On peut identifier quelques tâches clés en vision par ordinateur quand on s'intéresse à la détection, la localisation et la classification d'image. Celles-ci témoignent des 6 techniques majeurs en computer vision dont je souhaite vous faire part :

- **La classification d'image** consiste à prédire la classe d'un objet dans une image. Plus précisément, elle est capable de prévoir avec précision qu'une image donnée appartient à une certaine classe. L'architecture la plus populaire utilisée pour la classification des images est les réseaux de neurones convolutifs (CNN).

  On peut citer des architectures courantes en classification d'image tel que : AlexNet (2012), GoogleNet (2014), ou encore ResNet (2015).

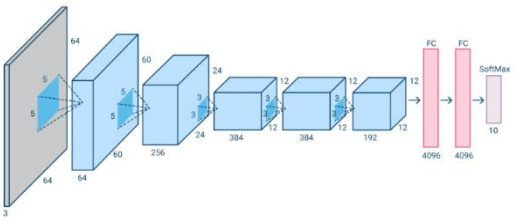


*Figure 4 : Architecture AlexNet*

- **Le suivi d'objet** fait référence au processus de suivi d'un objet d'intérêt spécifique, ou de plusieurs objets, dans une scène donnée. Les méthodes de suivi d'objets peuvent être divisées en 2 catégories selon le modèle d'observation : méthode générative (modèle génératif comme PCA) et méthode discriminative (distinction entre objet et arrière-plan). Le réseau profond le plus populaire pour le suivi des tâches à l'aide des encodeurs automatiques empilés (SAE) est Deep Learning Tracker. On peut citer 2 algorithmes de suivi basés sur CNN et régions d'intérêts (ROI), le suivi de réseau entièrement convolutif (FCNT) et le CNN multi-domaine (MD Net).
- **La segmentation sémantique** repose sur la division d'images entières en groupes de pixels qui peuvent ensuite être étiquetés et classés. Ici, on tente de comprendre sémantiquement le rôle de chaque pixel dans l'image. Par conséquent, contrairement à la classification, nous avons besoin de prédictions denses au niveau des pixels de nos modèles. La solution est les réseaux entièrement convolutifs (FCN) de l'UC Berkeley, qui ont popularisé les architectures CNN de bout en bout pour des prédictions denses sans couches entièrement connectées.

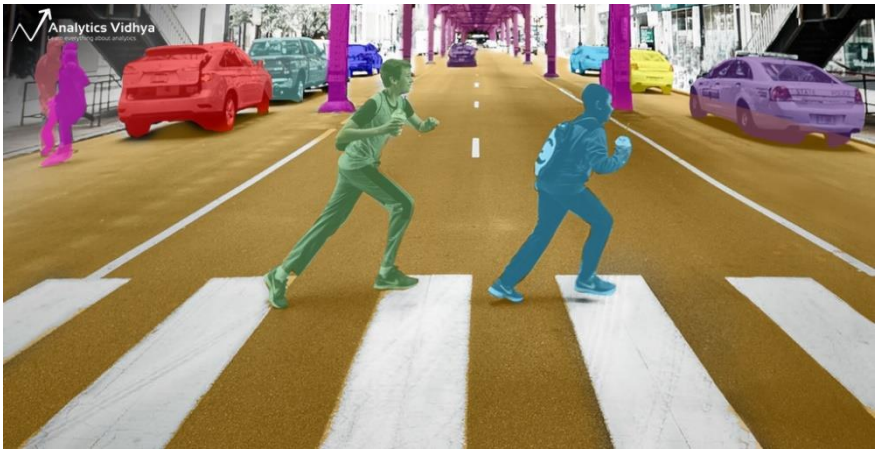


*Figure 5 : Segmentation sémantique d'une image*

- **La segmentation d'instances** consiste à segmenter différentes instances de classes. Nous observons des images avec de multiples objets qui se chevauchent et des arrière-plans différents, et non seulement nous classons ces différents objets, mais nous identifions également leurs limites, leurs différences et leurs relations les uns avec les autres ! Ce problème de segmentation d'instance est exploré chez Facebook AI en utilisant une architecture connue sous le nom de Mask R-CNN.
- **L'estimation de pose** est une méthode utilisée pour déterminer où se trouvent les articulations d'une personne ou d'un objet dans une image et ce que le placement de ces articulations indique. Il peut être utilisé avec des images 2D et 3D. L'architecture principale utilisée pour l'estimation de la pose est PoseNet.

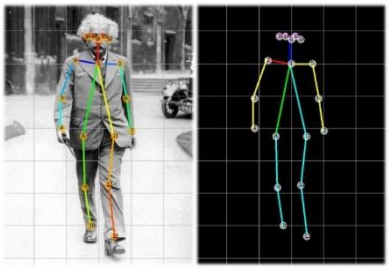

*Figure 6 : Estimation de pose*

Enfin, nous allons nous concentrer sur la détection d'objets, une composante essentielle de la vision par ordinateur et également la technique que j'utiliserai dans mon projet.

# III*) Modèles de détection*

La détection d'objets utilise la classification d'images pour identifier une certaine classe d'images, puis détecter et localiser leur emplacement dans une boîte englobante (bounding box). On peut distinguer deux éléments principaux constituant la détection d'objets :

- Le positionnement de chaque objet dans l'image dans des bounding boxes
- L'affiliation de labels à chaque objet afin d'identifier sa classe au fil du flux d'images

Il existe deux types courants de détection d'objets effectués via des techniques de vision par ordinateur :

- **Détection d'objets en deux étapes** - la première étape nécessite un réseau de proposition de région (RPN), fournissant un certain nombre de régions candidates pouvant contenir des objets importants. La deuxième étape consiste à transmettre les propositions de régions à une architecture de classification neuronale, généralement un algorithme de regroupement hiérarchique basé sur R-CNN, ou un regroupement de régions d'intérêt (ROI) dans Fast R-CNN. Ces approches sont assez précises, mais peuvent être très lentes. Les méthodes à deux étapes donnent la priorité à la précision de la détection.

- **Détection d'objets en une étape** - avec le besoin de détection d'objets en temps réel, des architectures de détection d'objets en une étape ont émergé, telles que YOLO, SSD et RetinaNet. Celles-ci combinent l'étape de détection et de classification, en régressant les prédictions de la boîte englobante. Chaque boîte englobante est représentée avec seulement quelques coordonnées, ce qui facilite la combinaison de l'étape de détection et de classification et accélère le traitement. Les méthodes à une étape donnent la priorité à la vitesse d'inférence.

## A) Détection d'objets en deux étapes

**R-CNN** a été le premier algorithme à appliquer le deep learning à la tâche de détection d'objets. Il bat les précédents de plus de 30 % par rapport au VOC2012 (Visual Object Classes Challenge) et constitue donc une amélioration considérable dans les domaines de la détection d'objets.
Comme mentionné précédemment, la détection d'objets présente deux difficultés : trouver des objets et les classer.

C'est le but de R-CNN : diviser la tâche difficile de la détection d'objets en 3 modules :

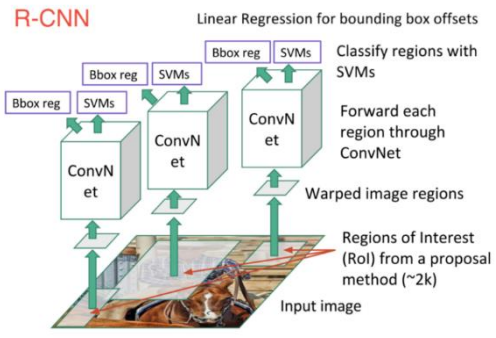


*Figure 7 : Architecture R-CNN*

- Génération de propositions régionales : indépendantes de la catégorie, qui définissent l'ensemble des détections candidates disponibles pour notre détecteur.
- Extraction de caractéristiques : le deuxième module est un grand réseau neuronal convolutif qui extrait un vecteur caractéristique de longueur fixe de chaque région.

- Classification et localisation : Le troisième module est un ensemble de support vector machine (SVM) linéaires spécifiques à chaque classe.

R-CNN prend en entrée les Regions Proposal (ou d'objets ou de boîtes, environ 2000 pour une image standard) et l'objectif de R-CNN est de trouver quelles régions sont significatives et les objets qu'elles représentent.

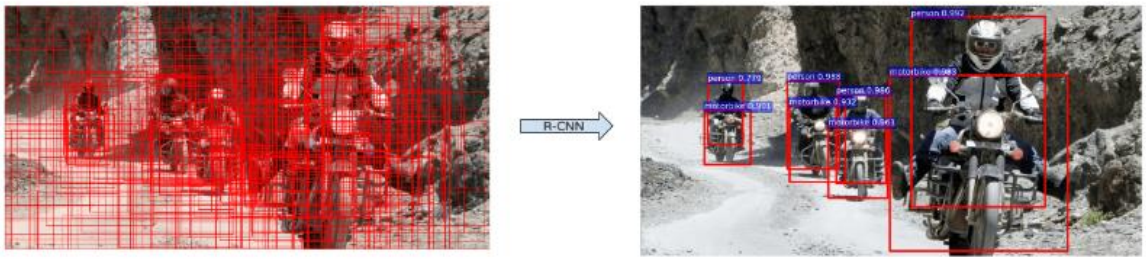


*Figure 8 : Etape de proposition de régions dans un RCNN*

La partie convolutionnelle d'Alexnet est utilisée pour calculer les caractéristiques de chaque région, puis les SVM utilisent ces caractéristiques pour classer les régions. Les propositions de régions, qui sont des rectangles de différentes formes possibles, sont transformées en carrés de 227×227 pixels. Elles sont ensuite traitées par le réseau et les valeurs obtenues sur la dernière carte de caractéristiques sont sorties. En conservant le meilleur score parmi tous les classificateurs binaires, nous obtenons la classe d'objet détectée correspondante.

Il y a ensuite une étape de régression des boîtes englobantes afin de corriger la localisation des propositions de régions. Cette phase de régression produit des facteurs de correction aux coordonnées de la boîte englobante.

Ainsi, cet algorithme est puissant et fonctionne très bien, mais il présente plusieurs inconvénients. Premièrement, il est très long. En effet, la proposition de région prend de 0,2 à plusieurs secondes selon la méthode, puis l'extraction et la classification des caractéristiques prennent à nouveau plusieurs secondes. De plus, ce n'est pas un algorithme fluide, il possède trois étapes différentes qui sont presque indépendantes et qui nécessitent donc une formation séparée.

**Fast R-CNN** propose un nouvel algorithme d'apprentissage qui corrige les inconvénients du R-CNN. Le réseau traite d'abord l'image entière avec plusieurs couches convolutionnelles et de mise en commun maximale pour produire une carte de caractéristiques.

Ensuite, pour chaque proposition d'objet, une couche de mise en commun des régions d'intérêt (RoI) extrait un vecteur de caractéristiques de longueur fixe de la carte de caractéristiques. Chaque vecteur de caractéristiques est introduit dans une séquence de couches entièrement connectées (FC) qui se ramifient finalement en deux couches de sortie sœurs :

✓ Une couche qui produit des estimations de probabilité softmax sur K classes d'objets plus une classe de "fond".

✓ Une couche qui produit quatre nombres à valeur réelle pour chacune des K classes d'objets.

Chaque ensemble de 4 valeurs code les positions raffinées de la boîte de liaison pour l'une des K classes.

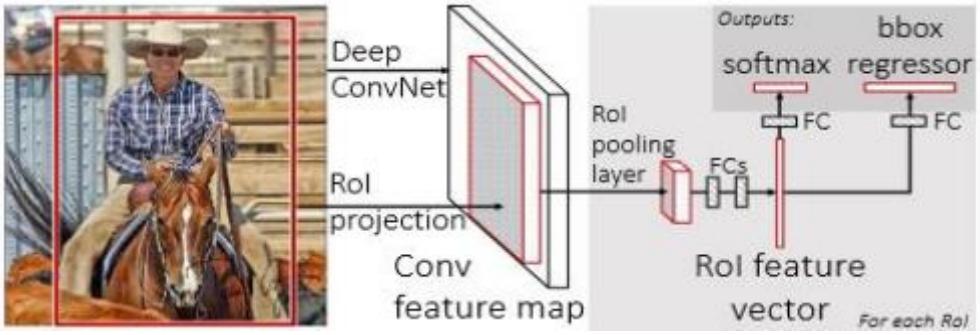


*Figure 9 : Architecture Fast R-CNN*

Enfin, le modèle **Faster R-CNN** en 2 modules, le premier est un réseau convolutif profond qui créé la carte de caractéristiques convolutives en utilisant un module RPN et produit un ensemble de propositions d'objets rectangulaires, chacune avec un score de précision. Le second module est le détecteur Fast R-CNN qui utilise les régions proposées. Il est ainsi encore plus rapide que Fast R-CNN !

### B) Détection d'objets en une étape

**Single Shot MultiBox Detector (SSD)** est basé sur un réseau convolutif feed-forward qui produit une collection de boîtes englobantes de taille fixe et des scores de présence d'instances de classes d'objets dans ces boîtes, suivi d'une étape de suppression non maximale pour produire les détections finales. Les premières couches du réseau sont basées sur une architecture standard utilisée pour la classification d'images de haute qualité (VGG-16) qui s'appelle réseau de base.

Ensuite une structure auxiliaire au réseau produisant des détections avec les caractéristiques clés suivantes :

✓ Cartes de caractéristiques multi-échelles pour la détection

✓ Prédicteurs convolutifs pour la détection (location et classification avec un filtre 3x3)

✓ Boîtes et aspect par défaut (union de boîtes différentes autour d'un point d'intérêt)

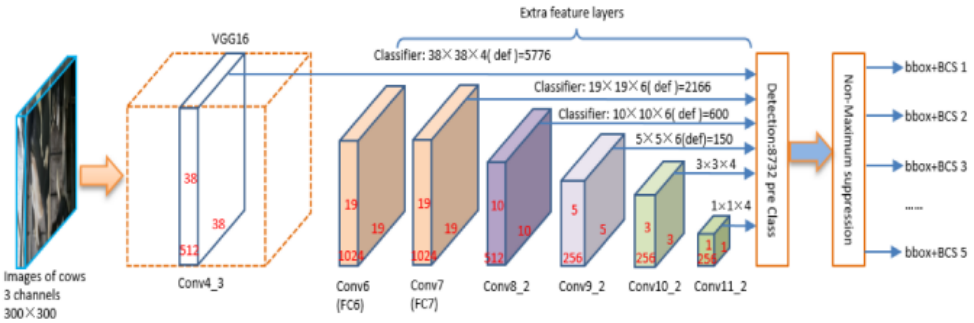


*Figure 10 : Architecture SSD*

**You Only Look Once** ou **YOLO** est l'un des algorithmes les plus populaires de détection d'objets utilisé par les chercheurs du monde entier. Il a été décrit pour la première fois en 2015 dans l'article de Joseph Redmon et al.

Le réseau utilise les caractéristiques de l'image entière pour prédire chaque boîte englobante. Il prédit également toutes les boîtes englobantes de toutes les classes d'une image simultanément. La conception YOLO permet un apprentissage de bout en bout et des vitesses en temps réel tout en maintenant une précision moyenne élevée.

On peut diviser l'algorithme YOLO en 3 étapes :

- **Division de l'image en cellules de taille SxS** : Cela nous donne N cellules, qui sont chacune responsable de la prédiction d'un cadre de délimitation et d'une prédiction de classe dans leur région.

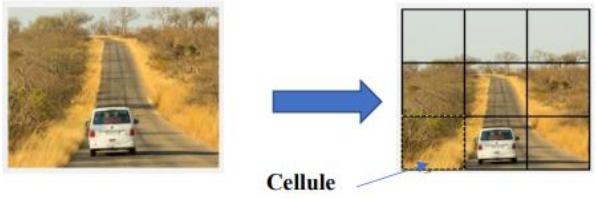


*Figure 11 : Division de l'image*

- **Prédiction de boîtes englobantes :** Chaque cellule de la grille prédit des boîtes englobantes et des scores de confiance associés. On prédit alors (x,y) les coordonnées du centre de la boîte, (w,h) largeur et hauteur de celle-ci, ainsi qu'un indicateur de confiance représenté par l'Intersection Over Union (IOU) entre la boîte prédite et la boîte effective.

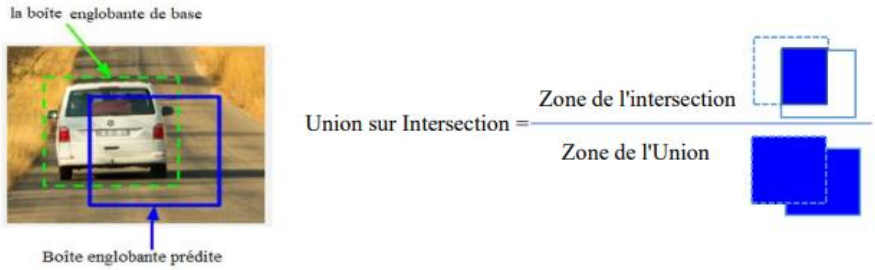


*Figure 12 : Union sur Intersection*

Les carrés de la grille sont utilisés pour générer un certain nombre d'anchor boxes. La première étape, tout naturellement, est de se débarrasser de toutes les anchor boxes qui ont une faible probabilité qu'un objet soit détecté. Cela peut être fait en construisant un masque booléen et en ne gardant que les cases qui ont une probabilité supérieure à un certain seuil. Cette étape élimine les détections anormales d'objets mais il reste une dernière étape pour conserver une unique boîte.

- **Suppression non maximale :** Cela consiste à supprimer les anchor boxes ayant une probabilité inférieure à un certain seuil, puis on sélectionne l'anchor avec la probabilité de détection la plus élevé. Et enfin, on supprime les anchor boxes trop proches les unes des autres dans cette étape car elles labelisent le même objet. On répète ces 2 étapes jusqu'à avoir une anchor pour chaque objet car si IoU > 0.5 entre 2 anchor boxes, cela signifie qu'elles labellisent le même objet.

*Figure 13 : Différentes étapes de l'algorithme YOLO*

*Il y a eu 6 versions du modèle jusqu'à présent, chaque nouvelle version améliorant la précédente en termes de vitesse et de précision.* Pour mon projet, j'ai choisi d'utiliser un modèle de détection d'objet en une étape, tel que YOLO, car je souhaite avoir un bon compromis entre efficacité et rapidité.

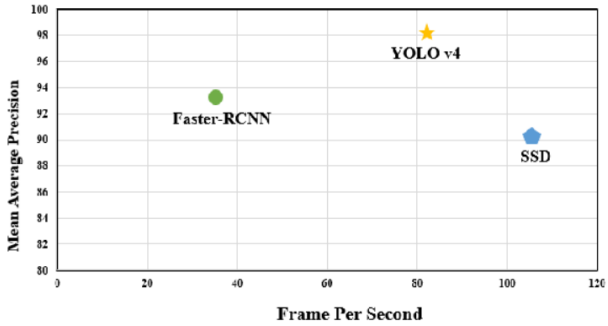


*Figure 14 : Comparaison de performance entre modèles sur une dataset identique*

De plus j'aimerais que mon application puisse tourner en temps réel si je le désire, c'est ainsi tout naturellement que je vais étudier YOLO. Ici, l'idée est d'appliquer une méthode de *transfert learning*. Cette méthode permet de s'appuyer sur un modèle pré-entraîné sur une tâche semblable à la nôtre, pour pouvoir réaliser un apprentissage qui va converger plus rapidement.

J'ai décidé de m'intéresser au modèle le plus récent, afin de le découvrir par moi-même et tester ses limites encore inconnues.

**Architecture de YOLO*v7* :**

L'article officiel YOLOv7 intitulé "YOLOv7 : Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors" a été publié en juillet 2022 par Chien-Yao Wang, Alexey Bochkovskiy et Hong-Yuan Mark Liao.

Il offre une précision de détection d'objets en temps réel grandement améliorée sans augmenter les coûts d'inférence. Par rapport à d'autres détecteurs d'objets connus, YOLOv7 peut réduire efficacement environ 40 % des paramètres et 50 % du calcul des détections d'objets

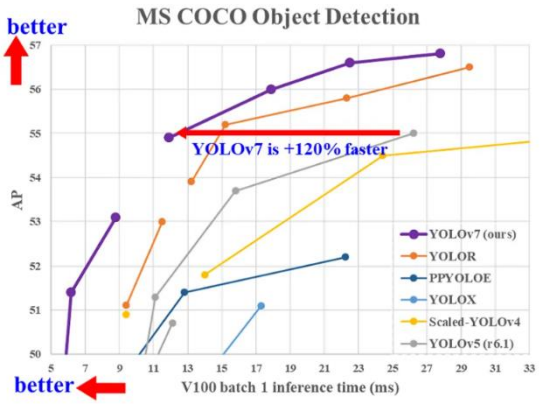


*Figure 15 : YOLO architectures comparaison*

L'architecture YOLOv7 est basée sur les architectures de modèles YOLOv4. Dans cette nouvelle version, on peut remarquer l'ajout de certaines fonctionnalités :

- Extended Efficient Layer Aggregation Network (E-ELAN) est le nouveau bloc de calcul du backbone au modèle de mieux apprendre sans détruire le chemin du gradient
- Mise à l'échelle du modèle composé permet d'optimiser la largeur (nombre de canaux), la profondeur (nombre d'étages) et la résolution du modèle (taille de l'image)

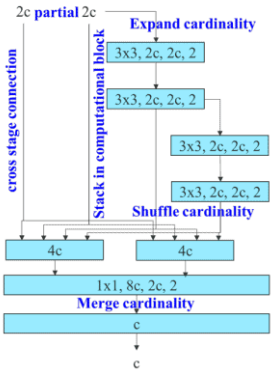


*Figure 16 : E-ELAN architecture*

L'avantage le plus intéressant est l'utilisation de bag of freebies features qui est une structure de réseau plus optimisé, fonction de coût optimisé… et qui permet l'augmentation significative de la précision sans perte de vitesses.

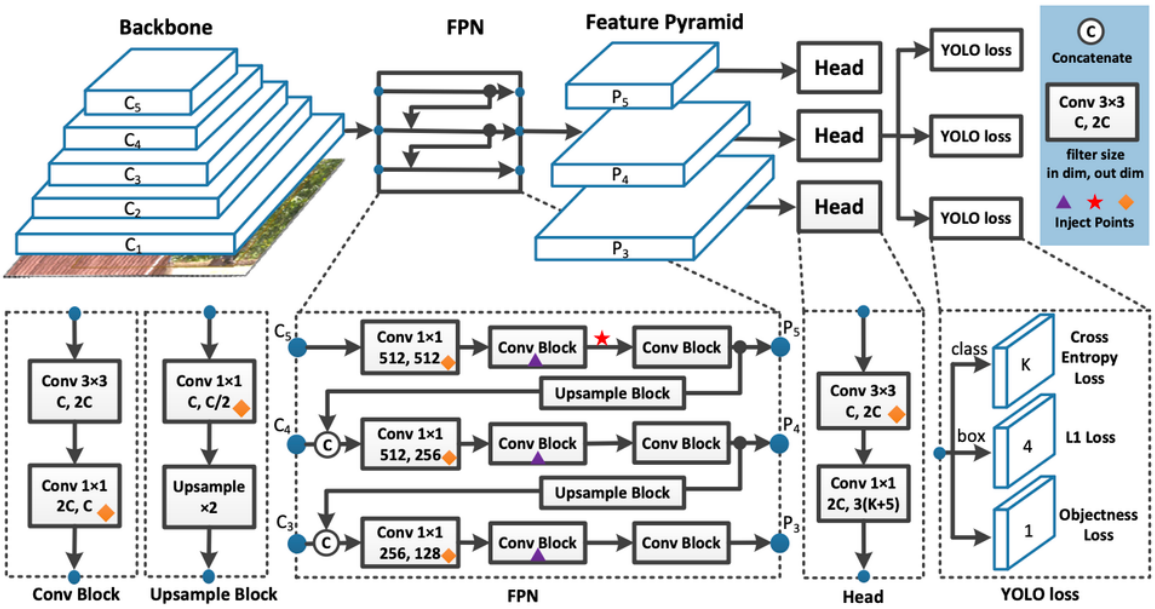


*Figure 17 : YOLOv7 architecture*

**Mesure de la performance / métrique d'évaluation :**

La mAP (mean Average Precision) est une mesure de performance la plus utilisée pour évaluer les modèles d'apprentissage automatique. Le calcul de **mAP** nécessite **IoU**, **Precision**, **Recall**, **Precision Recall Curve** et **AP.**

Ainsi, l'intersection sur l'union (IoU) est la métrique d'évaluation de facto utilisée dans la détection d'objets. Elle est utilisée pour déterminer les vrais positifs et les faux positifs dans un ensemble de prédictions. Ainsi, les boîtes englobantes prédites qui se chevauchent fortement avec les boîtes englobantes de base ont des scores plus élevés que celles qui se chevauchent moins

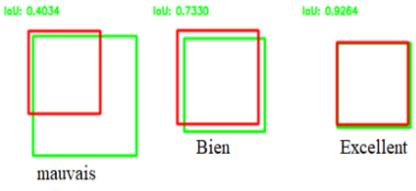


*Figure 18 : Exemple IoU*

La précision est la proportion d'exemples classifiés correctement parmi les exemples classifiés positifs.

$$pr = \frac{VP}{VP+FP}$$

Le rappel est la proportion d'exemples classifiés correctement parmi les exemples vraiment positifs

$$rap = \frac{VP}{VP+FN}$$

Ensuite, on trouve la précision moyenne (AP) en traçant la courbe Précision-Rappel, puis en appliquant une méthode d'interpolation.

Enfin, en calculant AP pour chaque classe, on peut isoler la **mAP** et déterminer efficacement la performance d'un modèle. En temps réel, le nombre de FPS (frame per second) influencera également la performance du modèle selon ses critères.

## IV*) Conclusion*

Ainsi, nous avons pu découvrir l'ensemble des méthodes et des techniques utilisés en vision par ordinateur associé à l'apprentissage profond. Nous avons pu fixer les la tâche et les limites de notre projet de **détection**. Notre documentation nous a permit d'identifier la **base de données** la plus pertinente ainsi que la technique et la méthode les plus performantes pour notre application. Enfin, nous avons compris en détail comment fonctionne le **modèle YOLO** que nous utiliserons et nous savons également comment évaluer sa **performance**.

### *Bibliographie :*


- Ian Goodfellow & Yoshua Bengio & Aaron Courville, (2016), *Deep Learning Book, « Convolutional Networks. Chp.9 »* : http://www.deeplearningbook.org
- Zewen Li, Wenjie Yang, Shouheng Peng, Fan Liu, (2021), *A Survey of Convolutional Neural Networks : Analysis, Applications, and Prospects* : https://ieeexplore.ieee.org/document/9451544
- Marco Pedersoli, (Automne 2022), *Cours SYS-819 Apprentissage profond* : https://ena.etsmtl.ca/course/view.php?id=17653
- James Le, (12 avril 2018), *The 5 Computer Vision Techniques That Will Change How You See The World : https://heartbeat.comet.ml/the-5-computer-vision-techniques-that-will-change-how-you-see-the-world-1ee19334354b*
- IBM, (2022), *Qu'est-ce que la Computer Vision ?* : *https://www.ibm.com/ca-fr/topics/computer-vision*
- Simon Code, (26 février 2020), *CLASSIFICATION D'IMAGES ET DÉTECTION D'OBJETS PAR CNN : http://www.aquiladata.fr/insights/classification-dimages-et-detection-dobjets-par-cnn/*
- RunAI, (2022), *Deep Learning for Computer Vision : https://www.run.ai/guides/deep-learning-for-computer-vision*
- Touraya EL HASSANI, (26 juillet 2021), *You Only Look Once – un réseau de neurones pour la détection d'objets : https://blog.octo.com/you-only-look-once-un-reseau-de-neurones-pour-la-detection-dobjets/*

- Francisco Pérez-Hernandez, Siham Tabik, Alberto Lamas, .al, (22 avril 2020), *Object Detection Binary Classifiers methodology based on deep learning to identify small objects handled similarly: Application in video surveillance :* https://doi.org/10.1016/j.knosys.2020.105590
- Alberto Castillo Llama , (Juin 2020), *Weapon Detection :* *https://dasci.es/transferencia/open-data/24705/*
- Saagie, (13 novembre 2020), *Qu'est-ce que la détection d'objet ? :* *https://www.saagie.com/fr/blog/quest-ce-que-la-detection-dobjet/*
- Mesbah Fethia, (2021), *Détection d'objets par Deep Neural Network à l'aide du modèle YOLO en temps réel :* *https://dspace.univ-guelma.dz/jspui/bitstream/123456789/11668/1/MESBAH_FETHIA_F5.pdf*
- Chien-Yao Wang, Alexey Bochkovskiy et Hong-Yuan Mark Liao, (juillet 2022), *YOLOv7 : Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors :* *https://arxiv.org/pdf/2207.02696.pdf*
- Kukil, Sovit Rath, (2 août 2022), *YOLOv7 Object Detection Paper Explanation and Inference :* *https://learnopencv.com/yolov7-object-detection-paper-explanation-and-inference/#YOLOv7-Object-Detection-Inference*
- WongKinYiu, (2022), Official YOLOv7 Github : https://github.com/WongKinYiu/yolov7
- Viso.ai, (2022), *YOLOv7 : The Most Powerful Object Detection Algorithm :* *https://viso.ai/deep-learning/yolov7-guide/*
- Kukil, (9 août 2022), *Mean Average Precision (mAP) in Object Detection :* *https://learnopencv.com/mean-average-precision-map-object-detection-model-evaluation-metric/*