République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Benyoucef BENKHEDDA-Alger1



Faculté des Sciences

Département des Mathématiques et Informatique

Projet de Fin d’étude pour l’obtention du diplôme de Licence en Informatique

Option : Systèmes Informatiques

Thème

Une approche d’apprentissage profond

Pour la détection d’objet

**Encadré par : Réalisé par :**

- Aiouez Sabrina. - Hamitouche Anis

- Belmadoui Mohamed Sabri

2020/2021

Remerciement

**Résume**

En vision par ordinateur on désigne par détection d’objet une méthode permettant de détecter la présence d’une instance ou d’une classe d’objets dans une image numérique.

Dans cette étude, nous avons élaboré une application mobile pour la détection et la reconnaissance des postures de la main de la langue des signes arabes basée sur l’apprentissage profond.

**Abstract**

En vision par ordinateur on désigne par détection d’objet une méthode permettant de détecter la présence d’une instance ou d’une classe d’objets dans une image numérique.

Dans cette étude, nous avons élaboré une application mobile pour la détection et la reconnaissance des postures de la main de la langue des signes arabes basée sur le Deep Learning.

# Table des matières :

[Table des matières : 4](#_Toc75645526)

[Introduction générale : 5](#_Toc75645527)

[Chapitre 1 : Détection d’objets dans les images : 6](#_Toc75645528)

[1. Introduction : 6](#_Toc75645529)

[2. Composants d’un système de détection d’objets : 6](#_Toc75645530)

[2.1 La localisation D’objets : 7](#_Toc75645531)

[2.2 La classification d’image : 7](#_Toc75645532)

[3. Problème liée à la détection d’objets : 7](#_Toc75645533)

[4. Domaine d’application : 11](#_Toc75645534)

[5. Reconnaissance des postures de la main de la LSA : 13](#_Toc75645535)

[6. Conclusion : 13](#_Toc75645536)

[Chapitre 2 : Réseaux de Neurones Convolutifs : 13](#_Toc75645537)

[1. Introduction : 13](#_Toc75645538)

[2. Les CNNs : 13](#_Toc75645539)

[2.1 Couche convolution : 14](#_Toc75645540)

[2.2 Couche pooling : 16](#_Toc75645541)

[2.3 Couche fully connected : 17](#_Toc75645542)

[3. Les architecture utilisées pour la détection d’objets : 18](#_Toc75645543)

[3.1 R-CNN : 18](#_Toc75645544)

[3.2 Fast R-CNN : 19](#_Toc75645545)

[3.3 Faster R-CNN : 21](#_Toc75645546)

[3.4 Yolo (you look only once): 22](#_Toc75645547)

[4. Conclusion : 23](#_Toc75645548)

[Chapitre 3 : Un système de vison pour la reconnaissance des signes de la LSA : 24](#_Toc75645549)

[1. Introduction : 24](#_Toc75645550)

[2. Présentation de l’architecture du système de détection des signes de LSA : 24](#_Toc75645551)

[3. Datasets : 24](#_Toc75645552)

[3.1 Construction des données : 24](#_Toc75645553)

[3.2 Prétraitements des données : 24](#_Toc75645554)

[3.3 Annotation des images : 24](#_Toc75645555)

[4. Mesure des performances : 24](#_Toc75645556)

[5. Outils de développement : 24](#_Toc75645557)

[Voici les différents environnements d'exécution fournis par Google Colab : 25](#_Toc75645558)

[6. Résultats expérimentaux : 26](#_Toc75645559)

[7. Interfaces de l’application : 26](#_Toc75645560)

[Conclusion : 27](#_Toc75645561)

[Bibliographie 28](#_Toc75645562)

# Introduction générale :

La détection d’objets dans des flux vidéo est une étape essentielle pour de nombreux algorithmes de vision par ordinateur. En effet, avec le développement des interfaces homme machine multimodales, on trouve aujourd’hui de plus en plus d'applications dans le domaine de la reconnaissance de la gestuelle de la main.

Dans ce cadre, la communauté des malentendants qui utilisent les gestes de la main pour communiquer trouvent beaucoup de difficultés aux quotidiens pour se faire comprendre par les autres, qui ne maitrisent pas toujours la langue des signes. Ces gestes sont en effet utilisés pour épeler le nom des personnes, adresse et d'autres mots. Par conséquent, le développement de systèmes de traduction automatique entre la langue des signes et la langue parlée, pour éliminer les obstacles liés à l'intégration des personnes sourdes dans la société, est apparue comme un besoin urgent pour cette communauté.

Ces dernières années, les réseaux de neurones convolutionnels ont permis d’obtenir de performances élevées dans la reconnaissance visuelle.

L'objectif de notre projet et de développer une application mobile capable de reconnaitre les signes de la main, en temps réel. Nous nous intéresserons dans ce travail aux postures de la main représentant les lettres de l'alphabet. Pour cela nous avons construit une base d’images de trente catégories correspondant aux lettres de la langue des signes arabes. Le système développé utilise un réseau de neurones convolutionnels pour détecter la main dans les images données par une caméra pour ensuite reconnaître les signes exprimés par ces postures.

Ce mémoire est organisé en 3 chapitres :

* Dans le premier nous présenterons les différents composants d’un système de détection d’objets. Ensuite nous aborderons le domaine de la reconnaissance des postures de la main de la LSA.
* Le deuxième décrit les réseaux de neurones convolutifs et les diffèrent architectures utilisé pour la détection d’objets.
* Et dans le dernier chapitre on présente les étapes de réalisation de notre application de reconnaissance des postures de la main pour la LSA.
* Nous clôturons ce mémoire par une conclusion générale et quelques perspectives.

# Chapitre 1 : Détection d’objets dans les images :

## Introduction :

La détection d’objet est une méthode base sur l’apprentissage profond **(Deep Learning)** vise à déterminer s’il existe des instances d’un objet d’une catégories donnée (telles que : des humains, animaux, voitures, arbres ………) dans une image. La détection d’objets s’appuie sur deux étapes intermédiaires : la **localisation d’objets** et **la classification d’image**.

Pour avoir une bonne méthode de détection d’objets, il est nécessaire d’avoir un algorithme solide de détection de régions et un bon algorithme de classification d’images, Cette méthode qui utilise le **DL** sert à résoudre des taches complexe et de haut niveau de la vision tel que la segmentation, la poursuite des objets, la détection des évènements, reconnaissance des activités, compréhension des scène, comptages des objets, etc. [1]

## Composants d’un système de détection d’objets :

On peut classer la détection d’objet en deux types : la détection d'instances spécifiques et détection d’une catégorie. Le premier type consiste à détecter un objet spécifique (par exemple : visage de Bill Gates, le monument Maqam E'chahid, les cartoons tom and Jerry …). Par contre le but de l’autre type est de détecter une instance d’une catégorie d’objet prédéfinies (comme un humain ou une voiture ou chat …) [1].



Figure 1 Instance d’objet spécifique

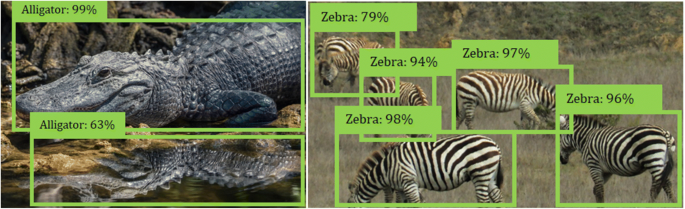
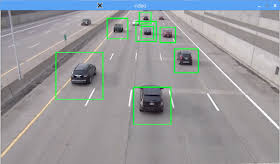
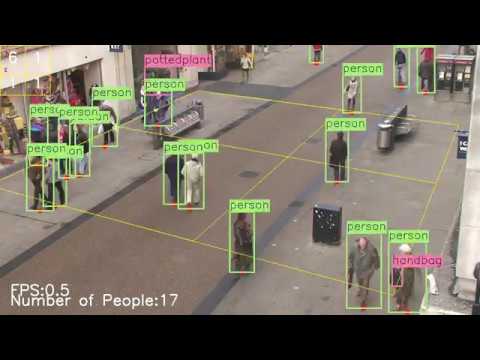


Figure 2 Détectiond’une catégorie

### La localisation D’objets :

La première étape de détection d’objet consiste à trouver la position d’un ou plusieurs objets dans une image et de dessiner leurs contours. Donc on génère des valeurs contenues qui constituent les cordonnées et les dimensions du cadre (**Bounding Box**) qui inclure l’objet unique détecté dans l’image.

### La classification d’image :

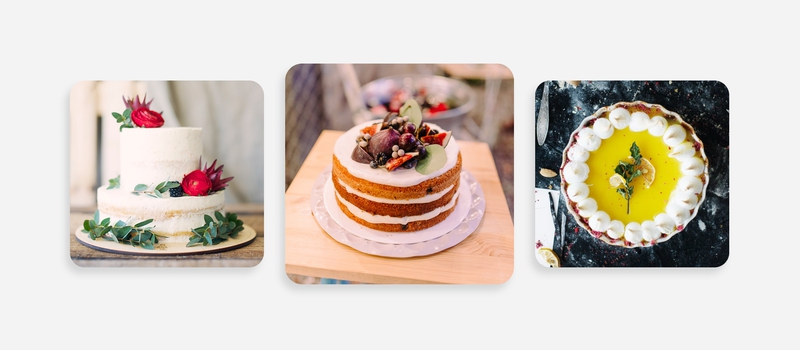
C’est la méthode qui consiste à prédire et attribuer à chaque instance une étiquette de classe correspondante. Classer les images en fonction des données qui sont des images d’entrée (images : inputs) avec leurs classes respectivement (objets visée : outputs). Il extrait toutes les **features** (caractéristiques) des images pour les utiliser lors de la prédiction. Pour les chiffres manuscrits, nous pourrions avoir dix classes, correspondant aux chiffres de 0 à 9.

La forme la plus simple de classification est lorsqu'il n'y a que deux classes, un problème que nous appelons classification binaire (binary classification). Par exemple, notre ensemble de données pourrait être constitué d'images d'animaux et de nos étiquettes pourrait être les classes {chat, chien}. Lorsque nous avons plus de deux classes possibles, nous appelons le problème de classification multi classe (multiclass classification). Les exemples courants incluent la reconnaissance de caractères manuscrits {0, 1, 2, ... 9, a, b, c, ...}. [2]

## Problème liée à la détection d’objets :

Les systèmes intelligents équipés de vision par ordinateur sont confrontés à un certain nombre de difficultés, en prêtant attention à ce qui ne fait que rendre le système plus précis. Parmi les challenges que on peut confronte on regarde ça :

**Variation du point de vue**L'une des plus grandes difficultés de la détection d'objets est qu'un objet vu sous différents angles peut sembler complètement différent. Par exemple, les images des gâteaux que vous pouvez voir ci-dessous diffèrent les unes des autres car elles montrent l'objet de différents côtés. Ainsi, le but des détecteurs est de reconnaître des objets sous différents points de vue. [3]

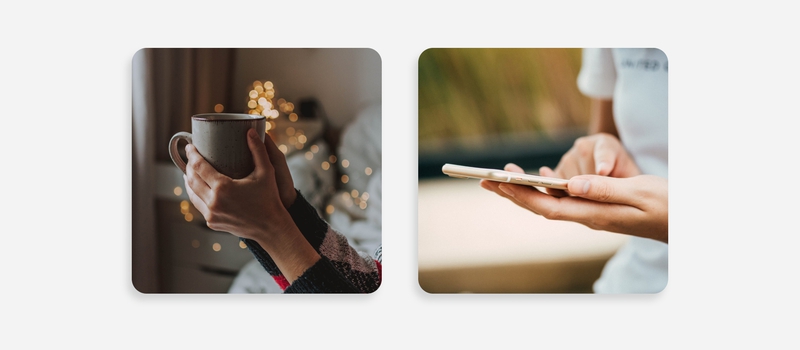


**Déformation**Le sujet de l'analyse de vision par ordinateur n'est pas seulement un objet solide, mais aussi des corps qui peuvent être déformés et changer de forme, ce qui ajoute une complexité supplémentaire à la détection d'objets.



Regardez les images de joueurs de football dans différentes poses. Si le détecteur d'objets est entraîné à trouver une personne uniquement en position debout ou en train de courir, il peut ne pas être en mesure de détecter un joueur allongé sur le terrain ou se préparant à effectuer une manœuvre en se penchant. [3]

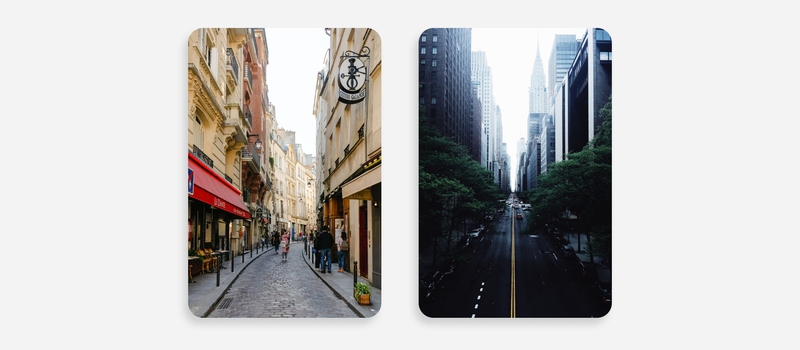
**Occlusion**Parfois, les objets peuvent être masqués par d'autres choses, ce qui rend difficile la lecture des signes et l'identification de ces objets. Par exemple, dans la première image ci-dessous, une tasse est recouverte par la main de la personne qui tient cette tasse. [3]



Dans la deuxième image, une personne tient également un téléphone portable de telle manière que les mains obstruent l'objet. De telles situations créent des difficultés supplémentaires pour déterminer le sujet. [3]

**Conditions d'éclairage**

L'éclairage a une très grande influence sur la définition des objets. Les mêmes objets auront un aspect différent selon les conditions d'éclairage. Regardez les images ci-dessous : moins l'espace est éclairé, moins les objets sont visibles. Tous ces facteurs affectent la capacité du détecteur à définir des objets. [3]



**Arrière-plan encombré ou texturé**Les objets qui doivent être identifiés peuvent se fondre dans l'arrière-plan, ce qui rend leur identification difficile. Par exemple, l'image ci-dessous montre de nombreux éléments, dont l'emplacement est déroutant lors de l'identification de ciseaux ou d'autres éléments d'intérêt. Dans de tels cas, le détecteur d'objet rencontrera des problèmes de détection. [3]



**Variété**Le même objet peut avoir des formes et des tailles complètement différentes. La vision par ordinateur doit faire beaucoup de recherches pour lire un objet et comprendre ce qu'il signifie.



Toutes les images ci-dessus représentent différents types de maisons. Un bon détecteur doit détecter ces objets et les affecter à la même classe. [3]

**Vitesse**Les algorithmes de détection d'objets doivent non seulement classer et localiser avec précision les objets importants, mais ils doivent également être incroyablement rapides au moment de la prédiction pour répondre aux exigences en temps réel du traitement vidéo. Plusieurs améliorations clés au fil des ans ont augmenté la vitesse de ces algorithmes, améliorant le temps de test de 0,02 images par seconde (ips) de R-CNN aux impressionnants 155 ips de YOLO. [4]

**Données limitées**

La quantité limitée de données annotées actuellement disponibles pour la détection d'objets s'avère être un autre obstacle important. Les ensembles de données de détection d'objets contiennent généralement des exemples de vérité terrain pour une douzaine à une centaine de classes d'objets, tandis que les ensembles de données de classification d'images peuvent inclure jusqu'à 100 000 classes. De plus, le crowd sourcing produit souvent des balises de classification d'images gratuitement (par exemple, en analysant le texte des légendes de photos fournies par l'utilisateur). La collecte d'étiquettes de vérité terrain *avec* des cadres de délimitation précis pour la détection d'objets reste cependant un travail incroyablement fastidieux. [4]

**Problèmes de caméra**

En plus des défis liés aux objets, il existe également des problèmes liés aux limitations de la caméra. La vidéo peut être capturée par des caméras tremblantes ou des caméras avec une faible résolution ou des informations de couleur limitées. Par conséquent, une séquence vidéo peut contenir des artefacts de bloc causé par la compression ou un flou causé par les vibrations. Tous ces artefacts peuvent dérouter les algorithmes de détection d'objets en mouvement s'ils ne sont pas entraînés à gérer des vidéos de mauvaise qualité. Une grande variété de solutions ont été proposées pour surmonter les défis de la caméra, mais ce problème reste ouvert. [5]

**Fond complexe**

Les environnements extérieurs naturels peuvent être trop complexes pour de nombreux algorithmes de détection d'objets en mouvement. La raison en est que l'arrière-plan peut être très texturé ou contenir des parties mobiles qui ne doivent pas être détectées comme des objets. Par exemple, les fontaines, les nuages, les vagues et les arbres oscillants créent des mouvements irréguliers ou périodiques en arrière-plan. Il est difficile de gérer une telle dynamique en arrière-plan. [5]

## Domaine d’application :

La détection d'objets fait son entrée dans un large éventail d'industries, avec des cas d'utilisation allant de la sécurité personnelle à la productivité sur le lieu de travail. La détection et la reconnaissance d'objets sont appliquées dans de nombreux domaines de la vision par ordinateur, notamment la récupération d'images, la sécurité, la surveillance, les systèmes automatisés de véhicules et l'inspection des machines. Des défis importants restent dans le domaine de la reconnaissance d'objets. Les possibilités sont infinies en ce qui concerne les futurs cas d'utilisation pour la détection d'objets. Ici, nous présentant quelque domaine d’applications actuelles et futures.

**Voiture autonome**

L'un des meilleurs exemples de pourquoi vous avez besoin de la détection d'objets pour la conduite autonome est Pour qu'une voiture décide quoi faire à l'étape suivante, accélérer, freiner ou tourner, elle doit savoir où se trouvent tous les objets autour de la voiture et quels sont ces objets Cela nécessite la détection d'objets et nous entraînerions essentiellement la voiture à détecter un ensemble connu d'objets tels que les voitures, les piétons, les feux de circulation, les panneaux de signalisation, les vélos, les motos, etc.

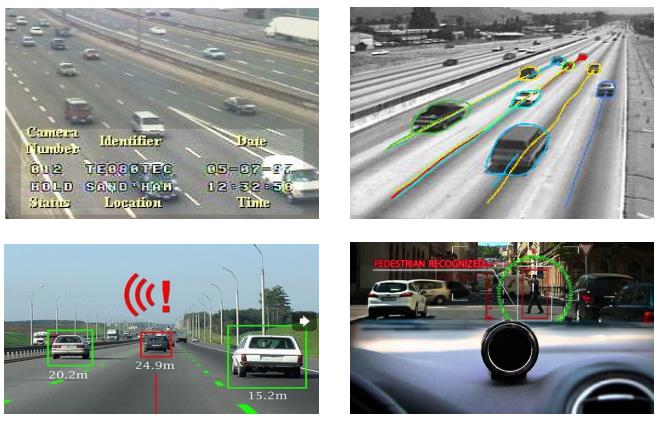


Figure 3 Detection automobiles

**L'IMAGERIE MÉDICALE**

Les outils de traitement d'images médicales jouent un rôle de plus en plus important dans l'aide aux cliniciens pour le diagnostic, la planification du traitement et les interventions guidées par l'image. Demande le suivi précis, robuste et rapide d'objets anatomiques déformables tels que le cœur est une tâche cruciale dans l'analyse d'images médicales.



Figure 4 Image medical

**RECONNAISSANCE D'ACTIVITÉ**

La reconnaissance d'activité vise à reconnaître les actions et les objectifs d'un ou plusieurs agents à partir d'une série d'observations sur les actions des agents et les conditions environnementales. Ce domaine de recherche a attiré l'attention de plusieurs communautés informatiques en raison de sa capacité à fournir un soutien personnalisé pour de nombreuses applications différentes et de sa connexion à de nombreux domaines d'études tels que l'interaction homme-machine ou la sociologie.

**ROBOTIQUE**

Les robots d'assistance autonomes doivent être dotés de la capacité de traiter des données visuelles en temps réel afin qu'ils puissent réagir de manière adéquate pour s'adapter rapidement aux changements de l'environnement. La détection et la reconnaissance fiables d'objets sont généralement une première étape nécessaire pour atteindre cet objectif.



Figure 5 Robots

## Reconnaissance des postures de la main de la LSA :

## Conclusion :

# Chapitre 2 : Réseaux de Neurones Convolutifs :

## Introduction :

Il existe quelques algorithmes différents pour la détection d'objets et ils peuvent être divisés en deux groupes :

* Algorithmes basés sur la classification. Ils sont mis en œuvre en deux étapes. Tout d'abord, ils sélectionnent les régions d'intérêt dans une image. Deuxièmement, ils classent ces régions à l'aide de CNN. Cette solution peut être lente car nous devons exécuter des prédictions pour chaque région sélectionnée.
* Algorithmes basés sur la régression - au lieu de sélectionner des parties intéressantes d'une image, ils prédisent les classes et les Bounding box pour l'image entière en une seule exécution de l'algorithme. Les exemples les plus connus de ce groupe sont les algorithmes de la famille YOLO (You Only Look Once). Et SSD (Détecteur Multibox Single Shot).

## Les CNNs :

Les CNNs est un type particulier de réseaux de neurones artificiels basé sur le concept de **Deep Learning**, qui ont conçue pour le traitement d’image, la reconnaissance vocale et pour le traitement des langages naturelles (NLP), Ils sont apparus en 1980 par le chercheur LuCan Yann.et depuis ce temps les CNN ont révolutionner le domaine de reconnaissance des images jusqu’au ces dernières années ils ont pu atteindre des performances élevées sur certains taches visuel complexe de l’humain tel que la classification et la segmentation des images, la reconnaissance faciale, la reconnaissance des objets et etc. Donc ils ont remporté un grand succès dans le monde de vision par ordinateur. [4]

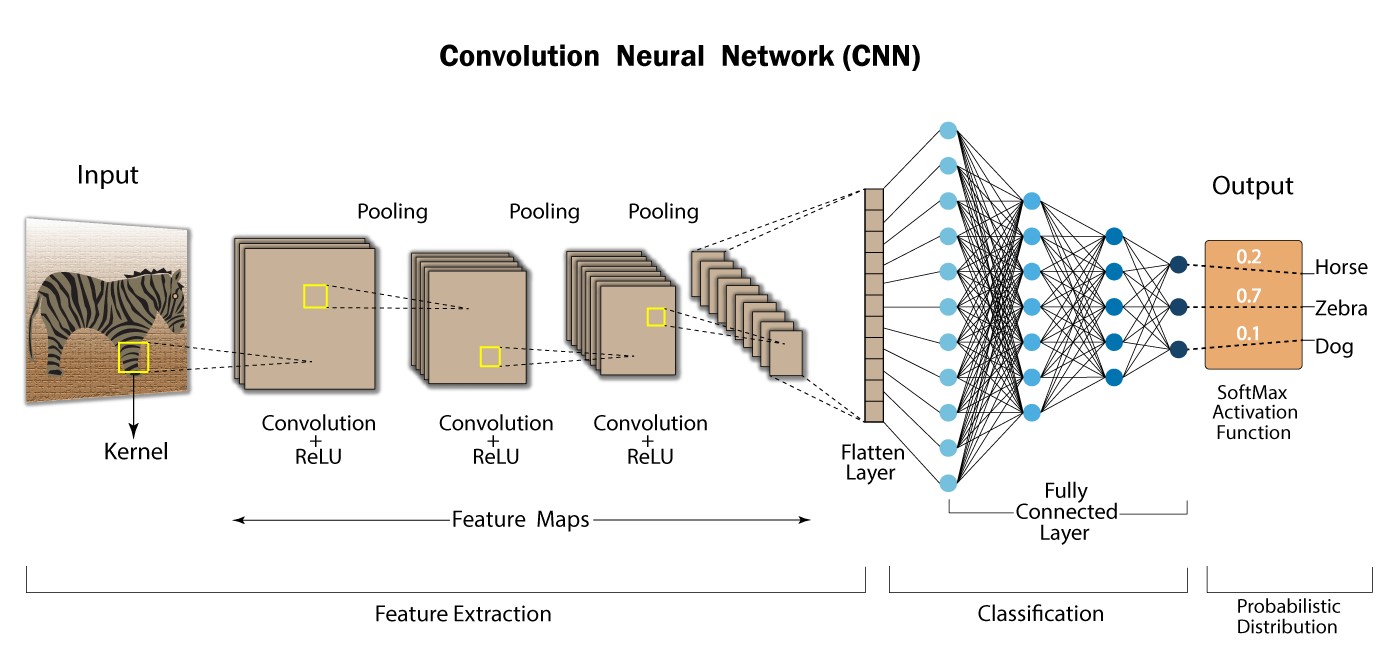


Figure 6 ARCHITECTURE CNN.

L’architecture CNN est une séquence de couche lier qui forme des réseaux hiérarchiques à plusieurs niveaux, elle est composée de trois types de couche (La couche de convolution ; La couche d’activation Relu ; La couche pooling ; Couche entièrement connecter (Fully connected))

Les deux premier qui ont rôle de l’extraction des caractéristiques dans une image, par contre la couche fully connected qui est responsable de la classification de ses **features.**

### Couche convolution :

Les couches convolution c’est les couches les plus important qui caractérise les CNN, elle consiste d’une suite des opérations linéaire et non linéaire.

#### Convolution :

Les couches convolutionnelles fait la particularité des réseaux CNN puisqu’elle fonctionne comme un extracteur de caractéristique, C’est est un type des fonctions linéaires, où on utilise un ensemble de filtres (**Kernel**) pour chaque couche convolution. Les filtres sont des vecteurs générés aléatoirement dans le réseau, constitués de poids et de biais. Les mêmes poids et biais sont partagés entre divers neurones dans CNN au lieu de poids et biais uniques pour chaque neurone. De nombreux filtres peuvent être générés où chaque filtre capture une caractéristique unique à partir de l'entrée [5]. Nous glissons (plus précisément, convolons) chaque filtre sur la largeur et la hauteur du volume d'entrée et calculons les produits scalaires entre les entrées du filtre et l'entrée à n'importe quelle position (**tensor**) puis on calcule la somme des valeurs obtenu pour avoir une valeur correspondant à la positionne actuelle de l’image, On applique cette procédure sur tous les entrées de la couche convolution pour former à la sortie ce qu’on appelle carte des caractéristiques (**feature map)** *(Voir figure 3)*. Les CNNs aide à augmenter l'efficacité du modèle en réduisant le nombre de paramètres à apprendre par rapport aux réseaux de neurones. D'autre part, les hyper paramètres qui doivent être définis avant la taille des filtres, le nombre de filtres, le padding le pas. [6]

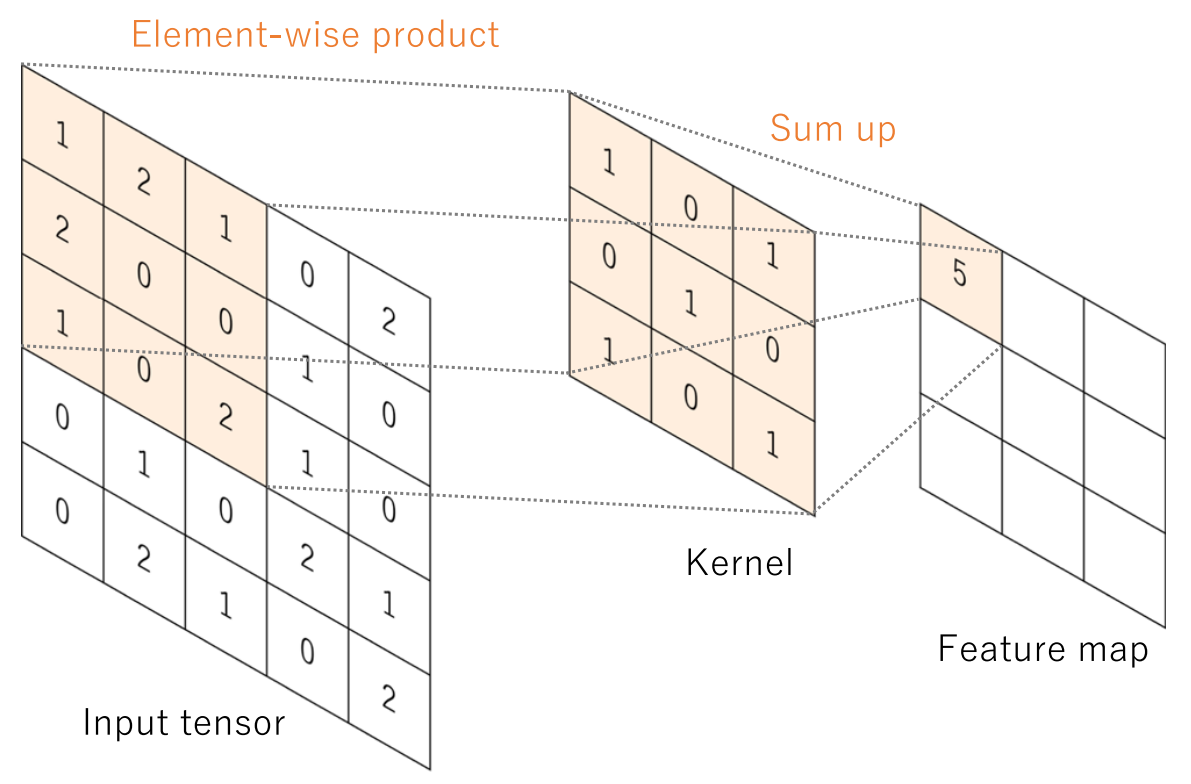


Figure 7 exemple d’une opération de convolution avec un filtre de 3 x 3, pas =1, padding =0.[6]

#### La fonction d’activation :

Tout sorties (feature map) d'une opération linéaire telle que la convolution sont ensuite passées par une fonction d'activation non linéaire ont été utilisées précédemment car ce sont des représentations mathématiques du comportement d'un neurone biologique, la fonction d'activation non linéaire la plus couramment utilisée actuellement est l'unitélinéaire rectifié (ReLU), [4] elle consiste à remplacer toutes les valeurs négatives par zéro. [Fei-Fei Li, 2020].

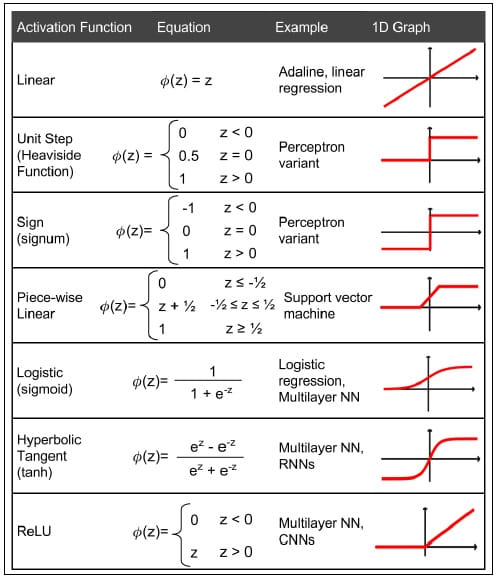


Figure 8 les Fonction d’activation les plus connus. [7]

### Couche pooling :

On trouve souvent Dans les CNN, les couche de convolution suivis des couches de mis en commun (pooling). Qui sert à réduire progressivement la dimension spatiale de la représentation, donc réduire la quantité de paramètres et les calculs dans le CNN. En utilisant le « max Pooling » et le « Average Pooling ». [6]

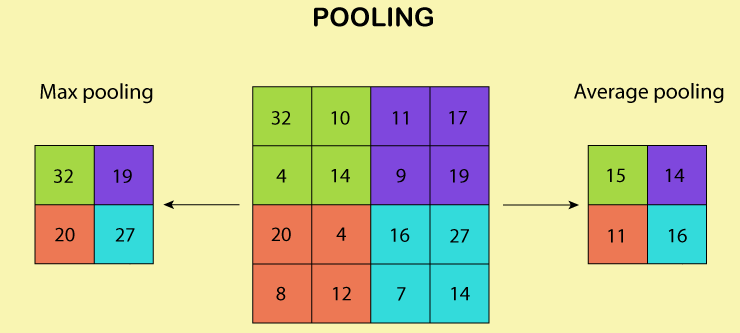


Figure 9 les résultats des opér*ations de Pooling (Avg/Max)*.[8]

#### Max pooling :

La forme la plus populaire d'opération de sous-échantillonnage (pooling) est la Max pooling, qui extrait les sous-échantillonnage des cartes d'entités d'entrée (feature map), en faisant sortie la valeur maximale de chaque carte caractéristique et supprime toutes les autres valeurs (figure 5). Un pooling max avec un filtre de taille 2×2 avec un pas de 2 est couramment utilisée dans la pratique. Cela sous-échantillonne la dimension dans le plan des cartes d'entités par un facteur de 2. Contrairement à la hauteur et la largeur, la dimension de profondeur des cartes d'entités reste inchangée. [6]

#### Average pooling :

Une autre opération de mise en commun à noter est le Average pooling. Qui effectue un type extrême de sous-échantillonnage, où une carte d'entités (feature map) avec la taille de H × L est sous-échantillonnée dans un tableau 1 × 1 en prenant simplement la moyenne de tous les éléments de chaque carte de caractéristiques, alors que la profondeur des cartes de caractéristiques est retenue. Cette opération n'est généralement appliquée qu'une seule fois avant les couches entièrement connectées. Les avantages de l'application de la mise en commun moyenne globale sont les suivants :

- réduit le nombre de paramètres prenables.

- active le CNN pour accepter des entrées de taille variable. [6]

### Couche fully connected :

## Les architecture utilisées pour la détection d’objets :

### R-CNN :

Les R-CNN (Region-Based Convolutional Neural Network) est une méthode basée sur les régions d’une image proposée par Ross Girshick dont l’idée se compose de deux étapes [9]:

1. Utiliser un algorithme de la recherche sélective qui sert à extraire environ 2k régions sur l’image avec les cadres de limitations d’objet (“region of interest” ou “RoI”).
2. Et puis il extrait les caractéristiques avec CNN de chaque région indépendamment pour la classification.

Le fonctionnement de R-CNN peut être résumé comme suit [9]:

1. **Pré-entrainer** un réseau CNN aux tâches de classification d'images qui implique N classes.
2. Proposez des **region of interest** indépendantes de la catégorie par la recherche sélective (~ 2k régions par image). Ces régions peuvent contenir des objets cibles et elles sont de tailles différentes.
3. Les candidats de région sont **déformés** pour avoir une taille fixe compatible avec l’entrée de CNN.
4. Application de **fine-tuning** de CNN sur les régions de proposition déformées pour les classes N + 1 ; La classe supplémentaire fait référence à l'arrière-plan (aucun objet d'intérêt). Dans la phase de **fine-tuning**, nous devrions utiliser un taux d'apprentissage beaucoup plus faible et le mini-batch sur échantillon des cas positifs parce que la plupart des régions proposées ne sont que d'arrière-plan.
5. Étant donné chaque région d'image, une propagation vers l'avant à travers le CNN génère un vecteur de caractéristiques. Ce vecteur de caractéristiques est ensuite consommé par un **SVM binaire** formé pour **chaque classe** indépendamment.  
   Les échantillons positifs sont des régions proposées avec un seuil de chevauchement IoU (intersection sur union)> = 0,3, et les échantillons négatifs ne sont pas pertinents pour les autres.
6. Pour réduire les erreurs de localisation, un modèle de régression est entraîné pour corriger la fenêtre de détection prévue sur le décalage de correction de la boîte englobante à l'aide des fonctionnalités CNN.

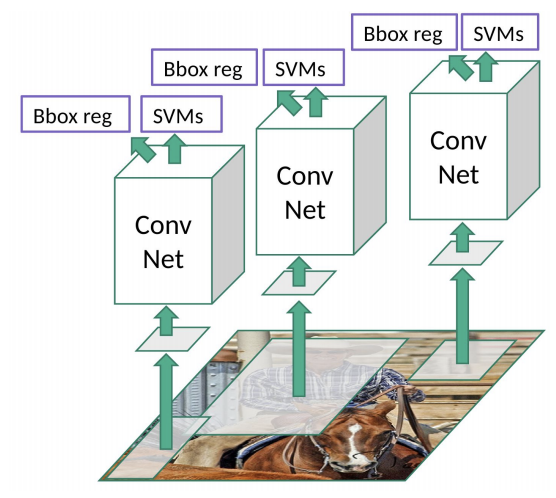


Figure 10 FONCTIONNEMENT de R-CNN. [10]

#### Les inconvénients Des R-CNN :

Jusqu'à présent, nous avons vu comment RCNN peut être utile pour la détection d'objets. Mais cette technique a ses propres limites. La formation d'un modèle RCNN est coûteuse et lente grâce aux étapes ci-dessous :

* Extraction de 2000 régions pour chaque image en fonction de la recherche sélective
* Extraction de caractéristiques à l'aide de CNN pour chaque région d'image. Supposons que nous ayons N images, alors le nombre de caractéristiques CNN sera N \* 2000
* L'ensemble du processus de détection d'objets à l'aide de R-CNN comporte trois modèles :
  1. CNN pour l'extraction de caractéristiques.
  2. Classificateur SVM linéaire pour identifier les classes des objets.
  3. Modèle de régression pour resserrer les cadres des limitations (**bBox**).

Tous ces processus se combinent pour rendre RCNN très lent. Il faut environ 40 à 50 secondes pour faire des prédictions pour chaque nouvelle image, Donc nous avons une autre technique de détection d'objets qui corrige la plupart des limitations que nous avons vues dans R-CNN. [11]

### Fast R-CNN :

Le même créateur du modèle précédent (R-CNN) a résolu certains des inconvénients de R-CNN en construisant un algorithme de détection d'objet plus rapide appelé FR-CNN. En unifiant trois modèles indépendants dans un modèle unique qui extrait les entités des régions, les divise en différentes classes et renvoie simultanément les boîtes de délimitation des classes identifiées.

Au lieu d'extraire les vecteurs de caractéristiques CNN indépendamment pour chaque 2000 région, ce modèle les regrouper en un seul passage CNN sur toute l'image et les region proposals partagent cette matrice des features. Ensuite, la même matrice de features est utilisée pour la classification d'objets et le régresseur de boîte englobante. En conclusion, le partage des calculs accélère R-CNN. [9]

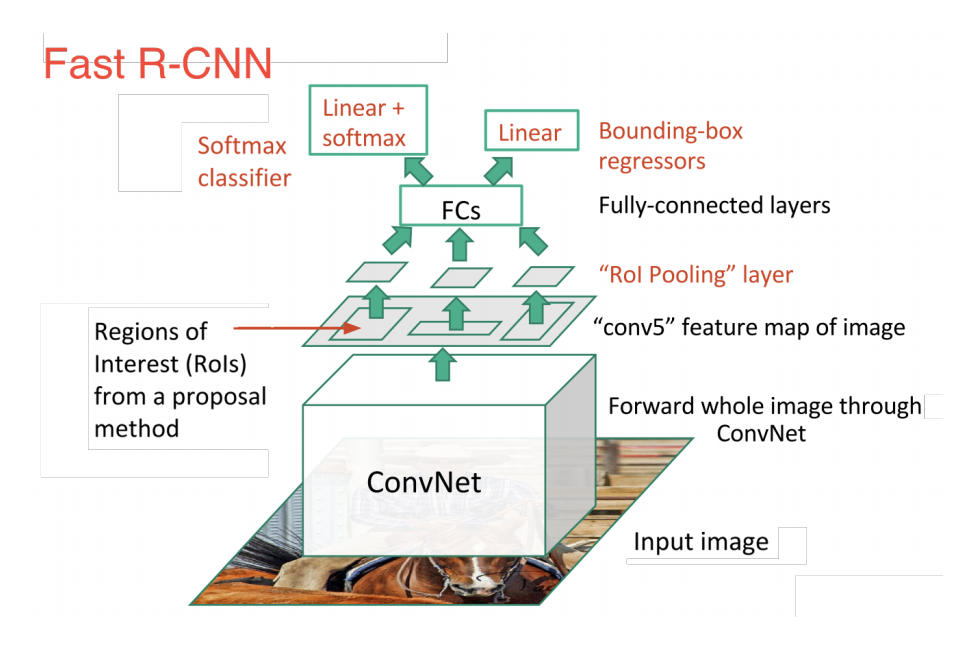


Figure 11 FONCTIONNEMENT de Fast R-CNN. [10]

Décomposons cela en étapes pour simplifier le concept :

1. Nous prenons une image comme entrée.
2. Cette image est transmise à un CNN qui à son tour génère les **region of interest**.
3. Une couche de **pooling RoI** est appliquée sur toutes ces régions pour les remodeler selon l'entrée du ConvNet. Ensuite, chaque région est transmise à un réseau entièrement connecté.
4. Une couche softmax est utilisée pour la classification des objets, Et une couche de régression linéaire est également utilisée parallèlement à la sortie pour les coordonnées de la boîte englobante pour les classes prédites.

#### RoI pooling :

Il s'agit d'un type diffèrent de max pooling pour convertir les entités de la région projetée de l'image de n'importe quelle taille, h x l, en une petite fenêtre fixe, H x W. La région d'entrée est divisée en grilles H x W, environ chaque sous-fenêtre de taille h / H x l / l. Ensuite, appliquez le max pooling dans chaque grille. [9]

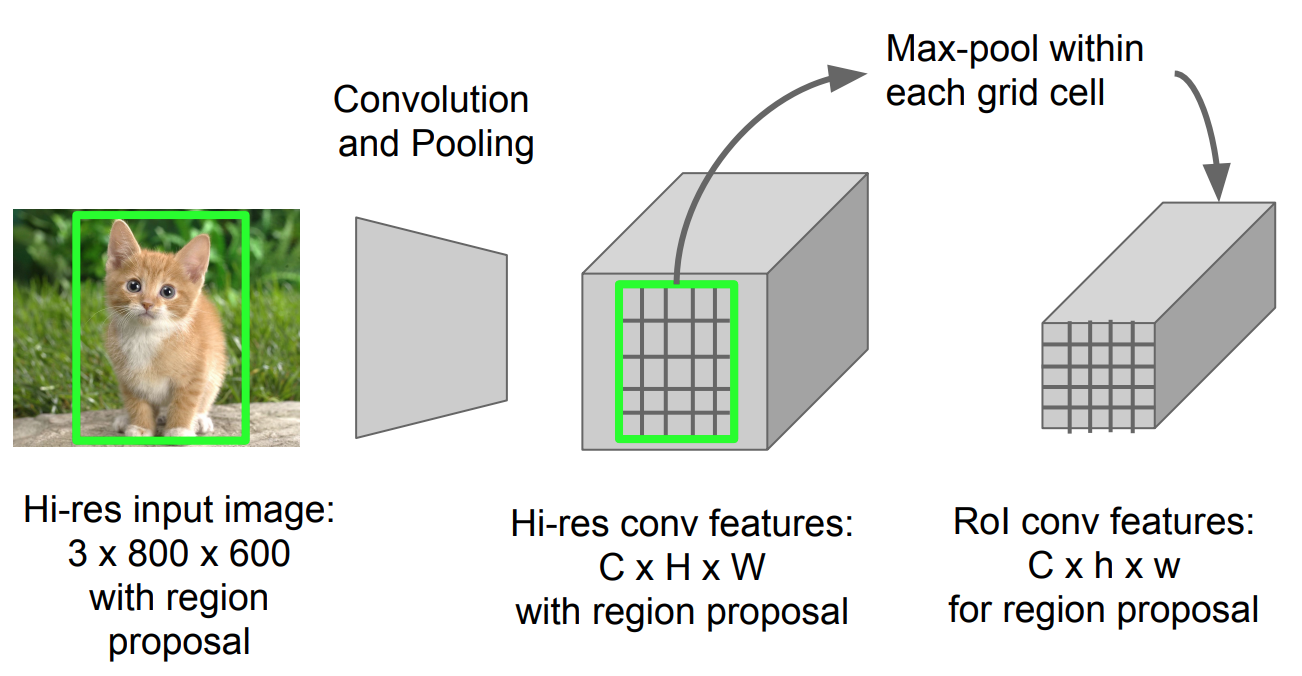


Figure 12 RoI pooling. [12]

#### Les inconvénients Des Fast R-CNN :

Le Fast R-CNN a certains problèmes. Il utilise les algorithmes da recherche sélective pour trouver les régions d'intérêt, ce qui est un processus lent dont Il faut environ 2 secondes par image pour détecter les objets. Mais lorsque nous utilisons des grands ensembles de données, donc le Fast R-CNN ne devient plus si rapide. [11]

### Faster R-CNN :

Faster R-CNN est la version mieux optimisée. La principale différence entre eux est que Fast R-CNN utilise la recherche sélective pour générer des régions de propositions, tandis que Faster RCNN utilise « Region Proposal Network ». RPN prend en entrée les feaetures maps d'image de CNN et génère un ensemble de region of interest. [11]

Voici des étapes d’une approche Faster R-CNN :

1. Nous prenons une image comme entrée et la transmettons au CNN qui renvoie la carte des caractéristiques de cette image.
2. Le réseau de RPN est appliqué sur ces cartes de caractéristiques. Cela renvoie les régions de propositions.
3. Une couche de pooling RoI est appliquée sur ces propositions pour les réformes une taille fixée.
4. Enfin, les propositions sont transmises à une couche entièrement connecté qui a une couche softmax et une couche de régression linéaire en son sommet, pour classer et sortir les cadres englobants des objets. [11]

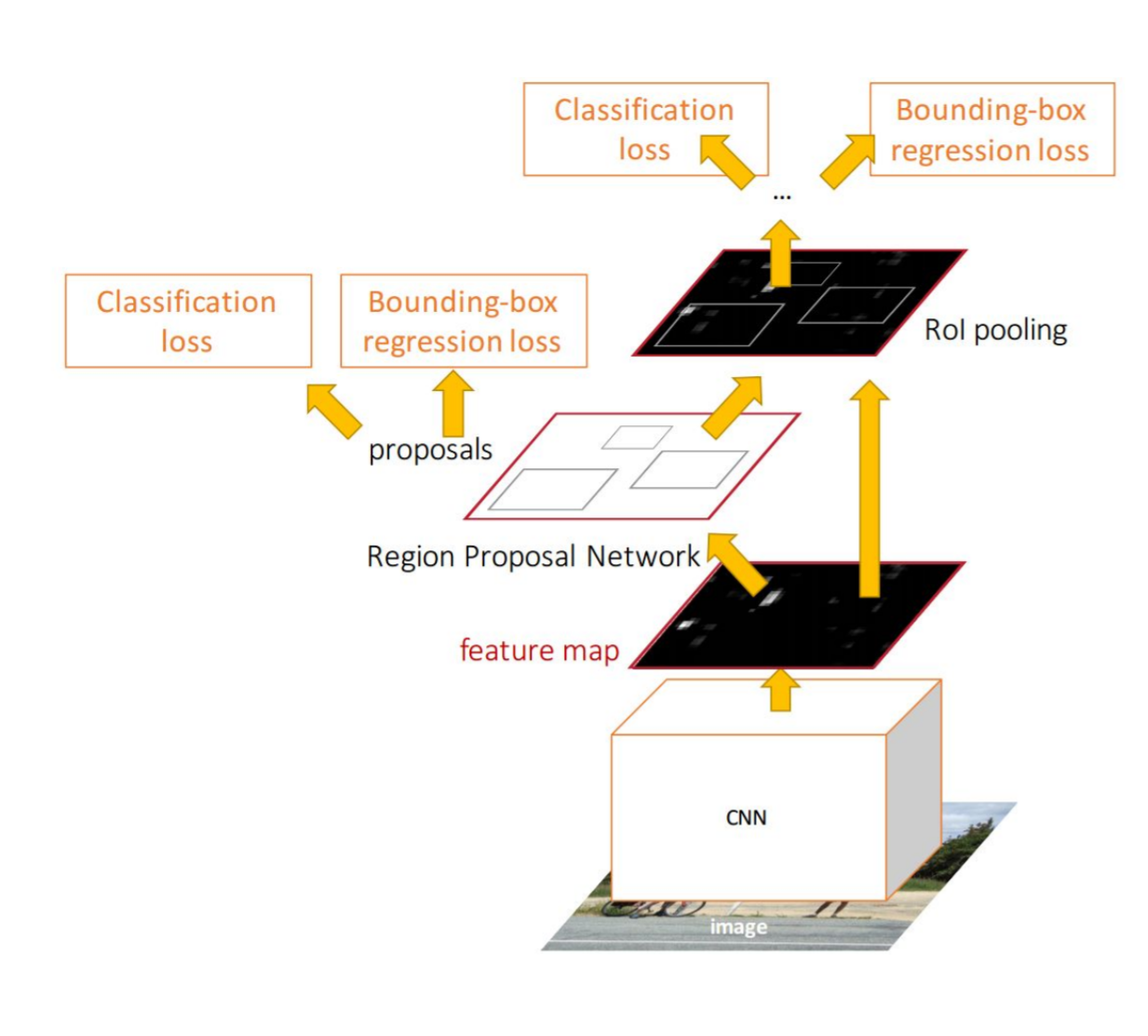


Figure 13 FONCTIONNEMENT de Faster R-CNN. [12]

### Yolo (you look only once):

A l’apparitions de YOLO en 2016, Il est considéré comme le modèle le plus rapide utilisés dans la détection d’objets en temps réel. Étant donné que YOLO évite l'étape de proposition de région et ne prédit que sur un nombre limité de cadre englobante, il est capable de faire des prédictions très rapidement.

En résume les étapes qui compose le modèle YOLO :

* **Pré-entrainer** un réseau CNN sur la tâche de classification d'images.
* Diviser une image en S× S cellules. Si le centre d'un objet trouver dans une cellule, cette cellule est « responsable » de la détection de l'existence de cet objet. Chaque cellule prédit l'emplacement de B boîtes englobants, Un score de confidence, et une probabilité de classe d'objet conditionnée à l'existence d'un objet dans la boîte englobant.
  + Les **coordonnées de** Bounding Box sont définies par la quadruplé (centre-X, centres-y, largeur, hauteur).
  + Un **score de confiance** indique la probabilité que la cellule contienne un objet : Pr (objet exister) x IoU (pred, vrai) ; où Pr= probabilité et IoU= **intersection over union**.
  + Si la cellule contient un objet, elle prédit une **probabilité** que cet objet appartienne à chaque classe Ci, i = 1, …, K : Pr (l’objet appartient à la classe Ci | contient un objet). A ce stade, le modèle ne prédit qu'un seul ensemble de probabilités de classe par cellule, quel que soit le nombre des Bounding Box B.
  + Au total, une image contient S × S × B boîtes englobant, chaque cadre correspondant à **4** prédictions de localisation, **1** score de confidence (au total **5** valeur donc 5xB) et K probabilités conditionnelles pour la classification des objets. Les valeurs de prédiction totales pour une image sont S × S × (5B + K), qui est la forme du tenseur de la couche de convolution finale du modèle.
* La couche finale du CNN pré-entraîné est modifiée pour produire un tenseur de prédiction de taille S × S × (5B + K).

Pour faire un test de la base de données Pascal Voc sur YOLO, On utilise ses paramètres S=7, B=2. Pascal Voc content 20 classe donc K=20. Alors on a 7 x 7 x (5 x 2 + 20). Donc on produire 98 tenseurs par le ConvNet au lieu de 2000 de la famille R-CNN.

## Conclusion :

# Chapitre 3 : Un système de vison pour la reconnaissance des signes de la LSA :

## Introduction :

## Présentation de l’architecture du système de détection des signes de LSA :

## Datasets :

### Construction des données :

### Prétraitements des données :

### Annotation des images :

## Mesure des performances :

## Outils de développement :

Les réseaux de neurones profonde (CNN) comporte des millions de paramètres à entraîner afin de construire le modèle correspondant ce qui est ce qui est recuit une forte capacité de calcul ainsi qu’une mémoire vive suffisamment grande qui ne sont pas disponibles dans les ordinateurs classiques, et pour cela nous avons est était obligé de recourir à utiliser un environnement de développement virtuel qui nous permettant de bénéficier de ses capacités. À l’instant des offres Cloud Computing payante telles que Microsoft azure et Amazon EC2, nous avons choisi d’utiliser les services open source (gratuite) espace de GOOGLE COLAB.

**GOOGLE COLAB :**



Figure 14 GOOGLE COLAB

Google Colab est une plate-forme puissante pour l'apprentissage et le développement rapide de modèles d'apprentissage automatique et profond en Python. Il est basé sur le notebook Jupyter et prend en charge le développement collaboratif. Les membres de groupes peuvent partager et modifier simultanément les blocs-notes, même à distance. Les cahiers peuvent également être publiés sur GitHub et partagés avec le grand public. Colab prend en charge de nombreuses bibliothèques de ML et DL populaires telles que PyTorch, TensorFlow, Keras et OpenCV. Il existe également une limitation des sessions (12h maximum) et de la taille. Compte tenu des avantages, ce sont de petits sacrifices que l'on doit faire.

### **Voici les différents environnements d'exécution fournis par Google Colab :**

* Processeur : Processeur Intel(R) Xeon(R) à 2,30 GHz.
* RAM : 12 GB.
* GPU : **NVIDIA** Tesla K80 / P100 / T4 / P4 avec VRAM au minimum 12 GB.
* TPU : Cloud TPU avec 180 Téra flops de calcul.

**PyTorch** :



Figure 15 Pytorch

PyTorch est une bibliothèque logicielle Python open source d'apprentissage développée par Facebook. Il permet d'effectuer les calculs tensoriels nécessaires notamment pour l'apprentissage profond (*Deep Learning*). Ces calculs sont optimisés et effectués soit par le processeur (CPU) ou par un processeur graphique (GPU) supportant [CUDA](https://fr.wikipedia.org/wiki/CUDA).

PyTorch utilise le langage Python qui permet de :

* Manipuler des tenseurs (tableaux multidimensionnels), de les échanger facilement avec Numpy et d'effectuer des calculs efficaces sur CPU ou GPU (par exemple, des produites de matrices ou des convolutions).
* Calculer des gradients pour appliquer facilement des algorithmes d'optimisation par descente de gradient.
* PyTorch utilise la bibliothèque Autograd.

**TensorBoard :**

L’apprentissage profond implique invariablement la comprèhension de paramètres clès tels que la perte et la manière dont ils changent au fur et à mesure que la formation progresse. Ces métriques peuvent nous aider à comprendre si nous sommes en sur-apprentissage par exemple, ou si nous nous entraînons inutilement pendant trop longtemps. Afin de comparer ces métriques sur diffèrentes exécutions d’entraînement ou bien de savoir les valeurs d’accuracy ou mAp atteint nous avons utilisè TensorBoard qui est un outil integré de manière native à Pytorch permettant de fournir les mesures et les visualisations nècessaires des résultats des entraînements de réseaux de neurones. Il permet de suivre les mètriques d’expèrimentation telles que la perte et l’accuracy.

**Flutter :**

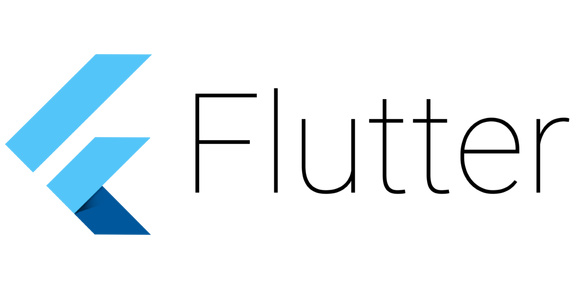
****

Figure 16 Flutter

Après avoir construit notre système, nous avions besoin de concrétiser notre travail par le développement d’une interface graphique interactive, et pour cela nous avons choisis de développer une application mobile. **Flutter** est un outil de développement de logiciel (SDK) d'interface utilisateur open-source créé par Google. Utiliser Dart comme langue de programmation. Il offre plusieurs avantages, Comme créer des applications natives et cross plateforme (iOS, Android, “web”) à partir d’un seul code.

La stratégie de Flutter **« Tout est un widget »** intègre de manière systématique la programmation orientée objet jusque dans l’interface utilisateur : l’interface d’un programme se compose de différents widgets pouvant être imbriqués les uns dans les autres. Chaque bouton et texte affiché est un widget. Ces widgets présentent diverses caractéristiques modifiables. Ils peuvent s’influencer les uns les autres et réagir à des modifications d’état extérieures grâce à des fonctions intégrées.

Flutter donne la flexibilité pour développer notre application utilisant les diffèrent widgets pour l’interface et intégrer notre modèle entrainé pour la détection des postures de la main utilisant la ressource ‘Camera’.

**Roboflow :**

****

On a utilisé Roboflow pour le traitement et les annotations des données. Roboflow est un Framework de développement Computer Vision pour une meilleure collecte de données pour le prétraitement et les techniques de formation de modèles. Roboflow dispose d'ensembles de données publics facilement accessibles aux utilisateurs et permet également aux utilisateurs de télécharger leurs propres données personnalisées. Roboflow accepte divers formats d'annotation. Le prétraitement des données comprend des étapes telles que l'orientation de l'image, le redimensionnement, le contraste et l'augmentation des données.

## Résultats expérimentaux :

## Interfaces de l’application :

# Conclusion :

# Bibliographie

|  |  |
| --- | --- |
| [1] | «Deep Learning for Generic Object Detection: A Survey,» International Journal of Computer Vision (2020) 128:261–318, 31 October 2019. |
| [2] | Z. C. L. M. L. a. A. J. S. Aston Zhang, Dive into Deep Learning, Jan 19, 2021. |
| [3] | A. Bryk, «Deep Learning for Overcoming Challenges of Detecting Moving Objects in VideoIt was originally published on https://www.apriorit.com/,» [En ligne]. Available: https://www.apriorit.com/dev-blog/607-deep-learning-moving-object-detection-video. |
| [4] | Hands-On Machine Learning with Scikit-Learn and TensorFlow, O’Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.. |
| [5] | «CS231n Convolution Neural Networks for Visual Recognition,» [En ligne]. Available: https://cs231n.github.io/convolutional-networks/. |
| [6] | Togashi, Rikiya Yamashita & Mizuho Nishio & Richard Kinh Gian Do & Kaori, «Convolutional neural networks: an overview and application in radiology,» [En ligne]. Available: https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9. |
| [7] | «simplilearn,» [En ligne]. Available: https://www.simplilearn.com/ice9/free\_resources\_article\_thumb/list-of-activation-functions-used-with-perceptron.jpg. |
| [8] | «Convolution neural network deep learning,» [En ligne]. Available: https://developersbreach.com/convolution-neural-network-deep-learning/. |
| [9] | «Lil'Log,» [En ligne]. Available: https://lilianweng.github.io/lil-log/2017/12/31/object-recognition-for-dummies-part-3.html. |
| [10] | «towards Data Science,» [En ligne]. Available: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e. |
| [11] | «Analytics Vidhya,» [En ligne]. Available: https://www.analyticsvidhya.com/blog/2018/10/a-step-by-step-introduction-to-the-basic-object-detection-algorithms-part-1/?utm\_source=blog&utm\_medium=computer-vision-implementing-mask-r-cnn-image-segmentation. |
| [12] | F.-F. L. &. A. K. &. J. Johnson, «cs231n,» [En ligne]. Available: http://cs231n.stanford.edu/slides/2016/winter1516\_lecture8.pdf. |