

# PROJET DEEP LEARNING

Détection d'objets dans les images de  
télédétection avec Mask R-CNN

Année universitaire : 2023/2024

# NOTRE EQUIPE



Abir Kharmachi



Iheb Oueslati



Mohamed Yassine  
Kassem



Anis Farah

# SOMMAIRE

**1. CRISP-DM Méthodologie**

**2. Compréhension métier**

**3. Compréhension des données**

**4. Modélisation & Evaluation**

# CRISP-DM

1- Compréhension du problème métier

2- Compréhension des données

3- Préparations des données

4- Modélisation

5- Evaluation

6- Déploiement

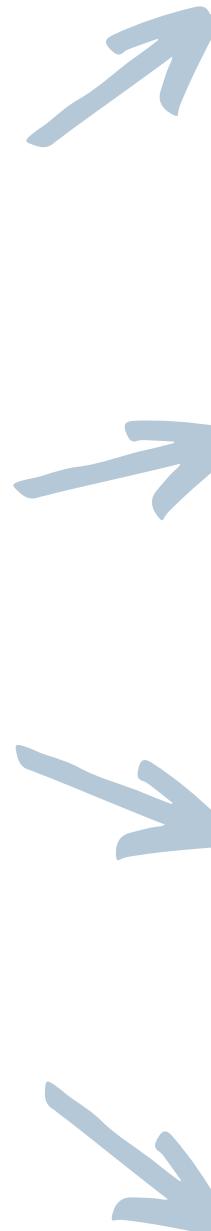


# COMPRÉHENSION MÉTIER



# LES DOMAINES D'APPLICATION

## Détection Des Objets

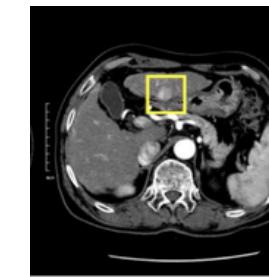


### Sécurité et surveillance



Identifier les comportements suspects et améliorer la sécurité des espaces publics.

### Diagnostic médical



Identifier et analyser les anomalies, comme les tumeurs ou les lésions.



### Robotique industrielle

Identifier et manipuler les pièces ou les produits, améliorant ainsi l'efficacité



### Télédétection

Cartographier les terrains, surveiller les changements environnementaux et identifier les caractéristiques géographiques .

# DÉTECTION D'OBJETS DANS LES IMAGES DE TÉLÉDÉTECTION AVEC MASK R-CNN



L'objectif principal de cette étude est de concevoir une méthode rapide et précise de détection d'objets adaptée aux images de télédétection. Cette méthode est cruciale pour les applications civiles et militaires qui dépendent d'une identification précise et rapide des objets dans les images satellites haute résolution.

# **COMPRÉHENSION DES DONNÉES**



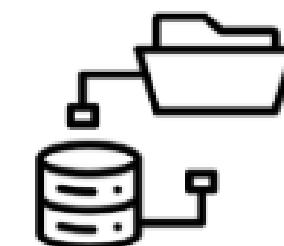
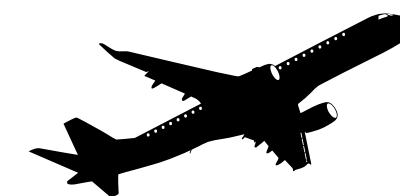
# NWPU-VHR-10



NWPU VHR-10 (Northwestern Polytechnical University, Very High Resolution 10-class dataset)



Il s'agit d'un ensemble de données destiné à la recherche en vision par ordinateur, spécialement pour la détection d'objets dans des images aériennes haute résolution.



Les images proviennent de Google Earth et du jeu de données Vaihingen, fournies par la Société allemande de photogrammétrie, télédétection et géoinformation (DGPF). Ces images sont annotées manuellement par les experts.



# NWPU-VHR-10



## Images

- Ensemble Positif : **650 images**
- Ensemble Négatif : **150 images**



## Annotations

les boîtes de délimitation des objets cibles.  
Format  $(x_1, y_1), (x_2, y_2)$ , classe.

# POSITIVE VS NEGATIVE

## Positive Image Set

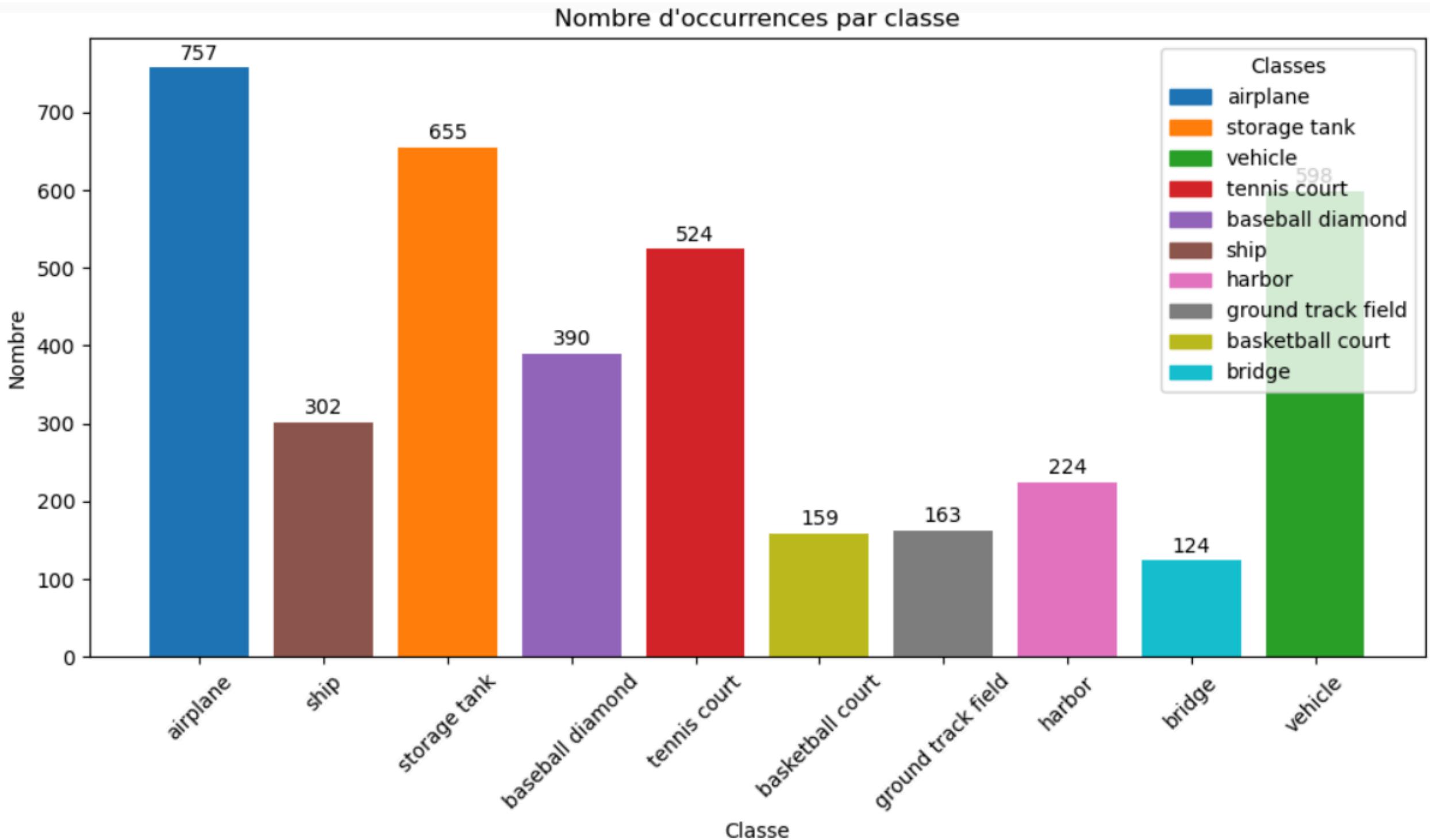
- **Au moins une cible d'intérêt parmi les dix classes**
- **Il sert principalement à entraîner les modèles et à évaluer leur capacité à détecter les objets cibles.**
- **Evaluer les performances du modèle en termes de précision et de rappel**
- **Vérifier combien d'objets sont correctement détectés parmi les cibles annotées.**

## Negative Image Set

- **Ne comportent aucune des cibles d'intérêt.**
- **Evaluer la capacité du modèle à éviter les faux positifs (erreurs de détection)**
- **Ne pas prédire des objets là où il n'y en a pas**
- **Assure que le modèle ne généralise pas trop et maintient une bonne spécificité.**

## 10 classes :

- **Avion**
- **Bateau**
- **Réservoir de stockage**
- **Diamant de baseball**
- **Court de tennis**
- **Terrain de basketball**
- **Champ de piste au sol**
- **port**
- **Pont**
- **Véhicule**



# PRÉPARATION DES DONNÉES



# REDIMENTIONNER LES IMAGES

## Redimensionnement des Images :

**Dimension Utilisée :** Les images sont redimensionnées à 1024x1024 pixels.

**Avantage :** Assure une uniformité dans les dimensions pour chaque image, simplifiant ainsi le traitement par les modèles de détection.

## Ajustement des Boîtes de Délimitation :

**Échelle de Conversion :** Les dimensions des boîtes de délimitation (bounding boxes) sont réajustées selon l'échelle du redimensionnement pour correspondre aux nouvelles dimensions des images.

**Avantage :** Maintient la précision des annotations malgré la modification des dimensions d'image.



✓ Amélioration des Performances : Des dimensions cohérentes permettent aux modèles de généraliser et de détecter les objets plus efficacement.

✓ Uniformisation : La taille standard facilite le traitement batch par les modèles de détection d'objets.

# EXEMPLE DE REDIMENTION D'UNE IMAGE

Image 1

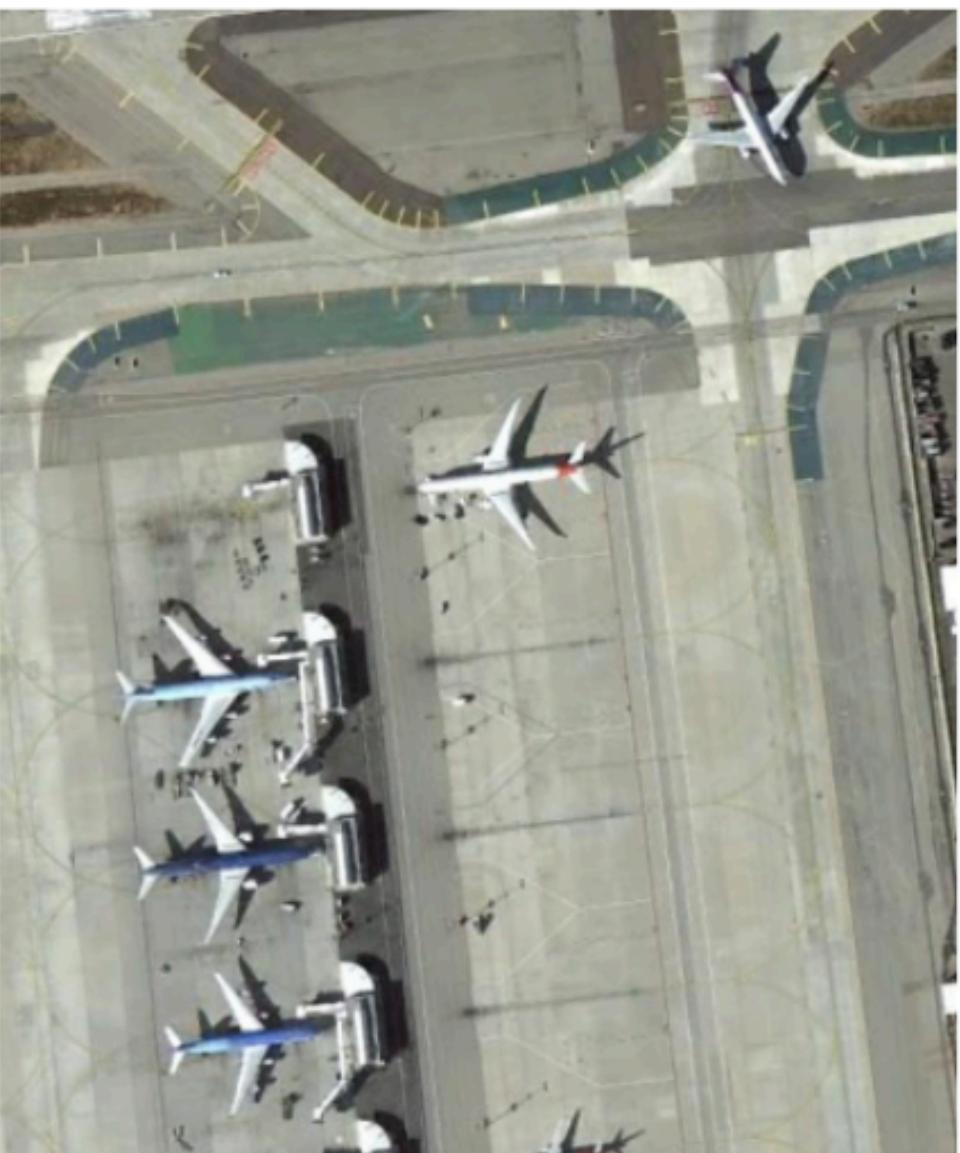
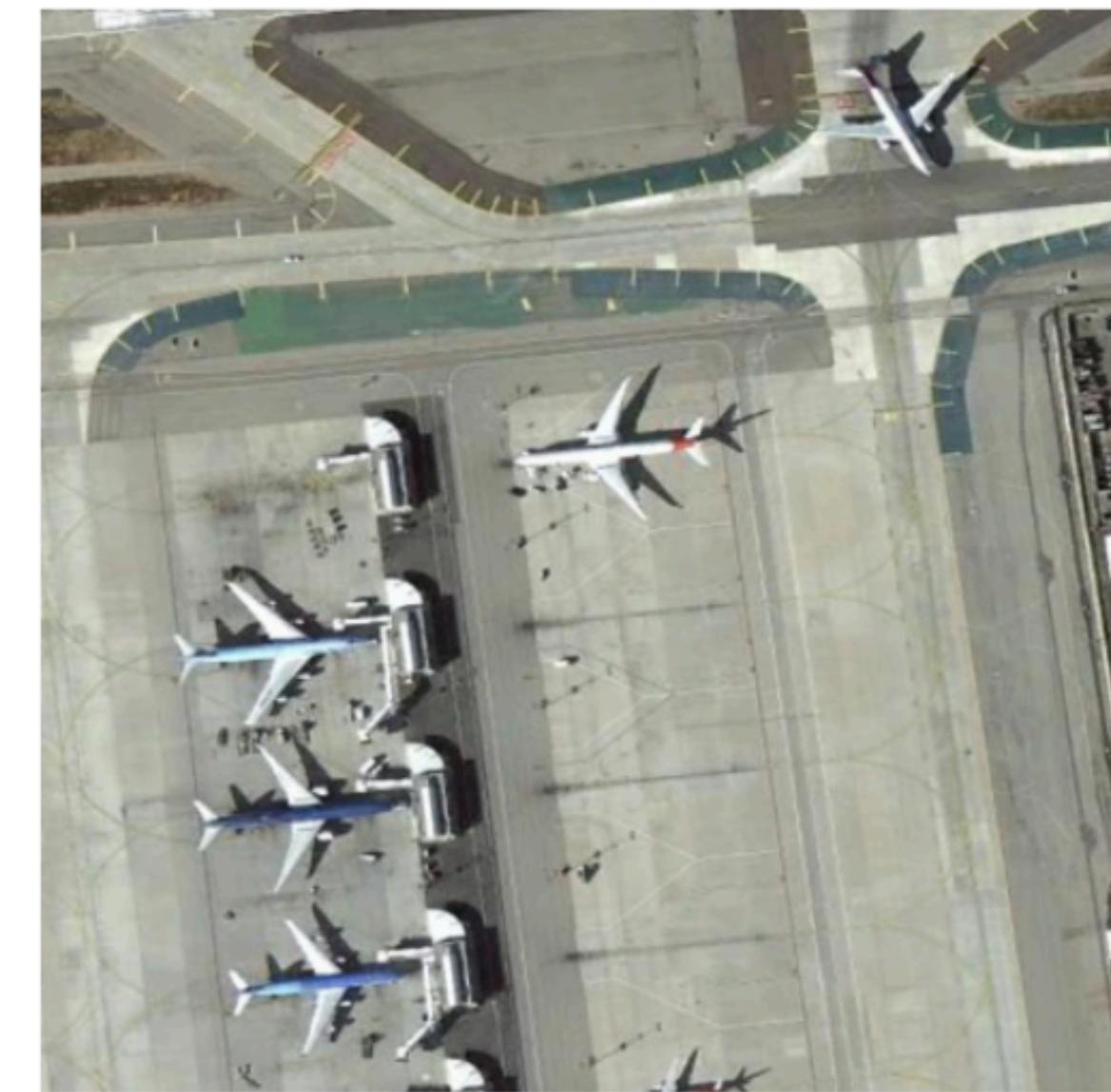


Image 2



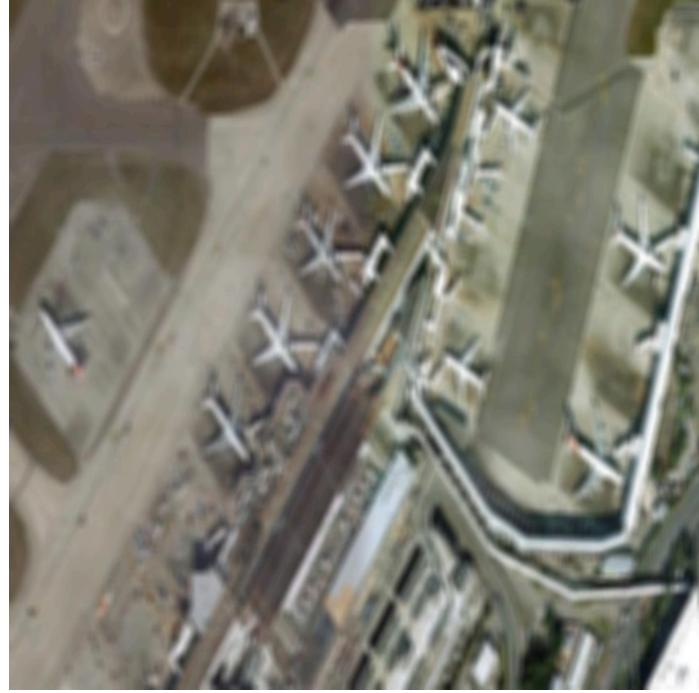
( 69,427),(172,514),1

(132,686),(330,826),1

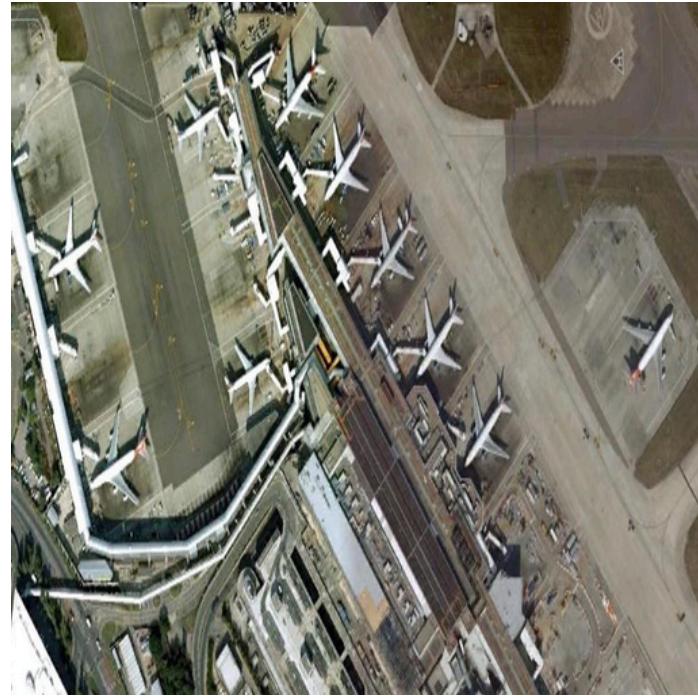
# DATA AUGMENTATION



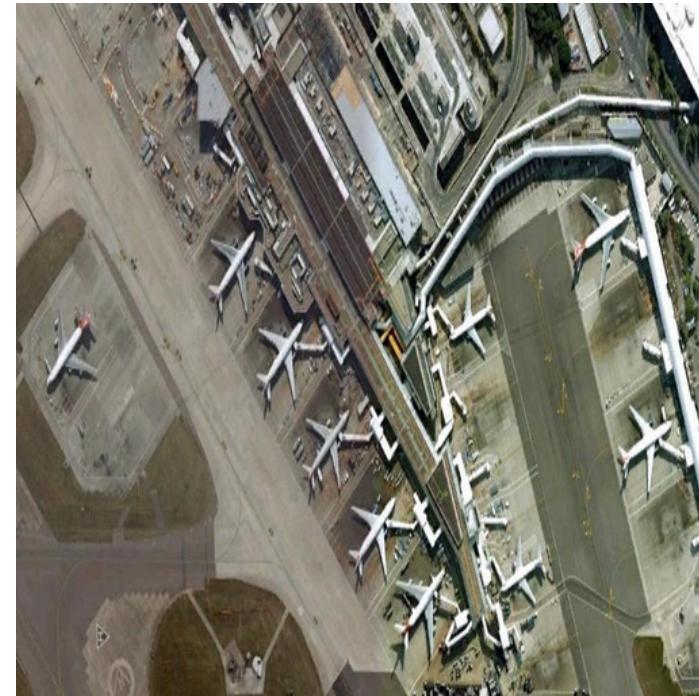
Contraste



Fog



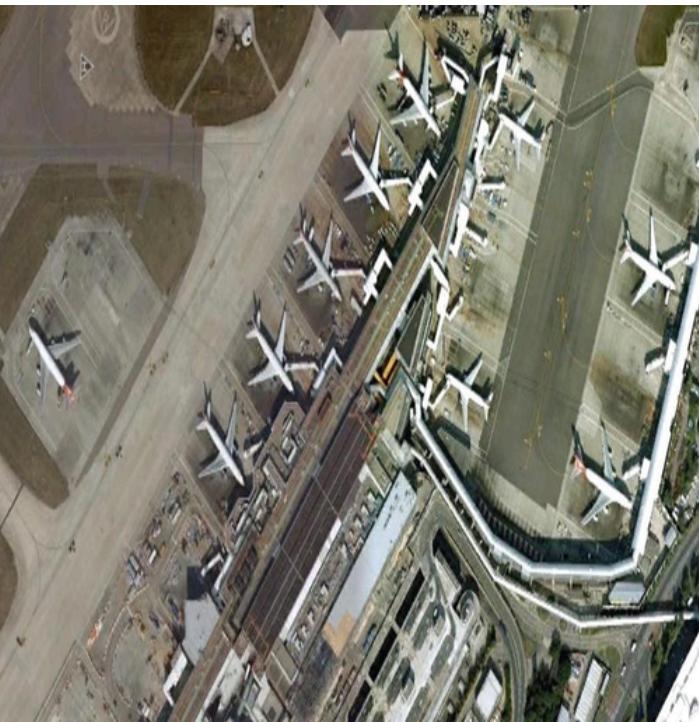
Horizontal Flip



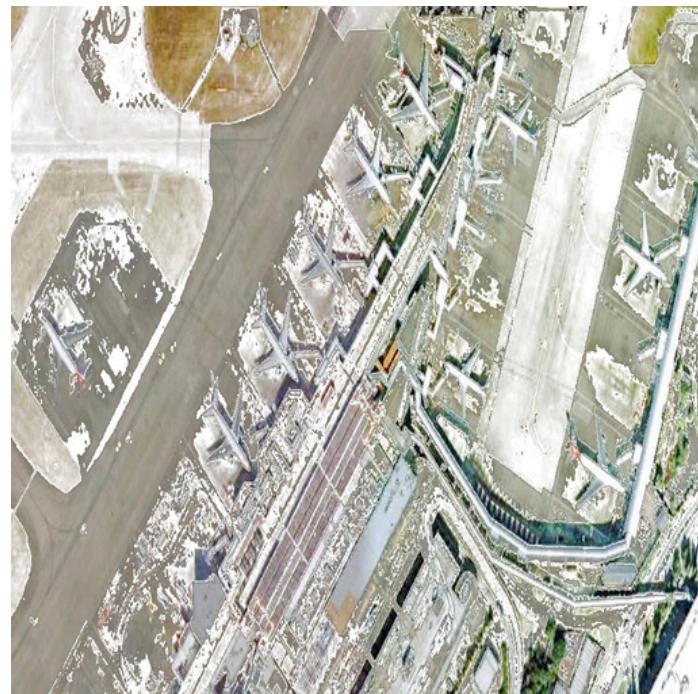
Vertical Flip



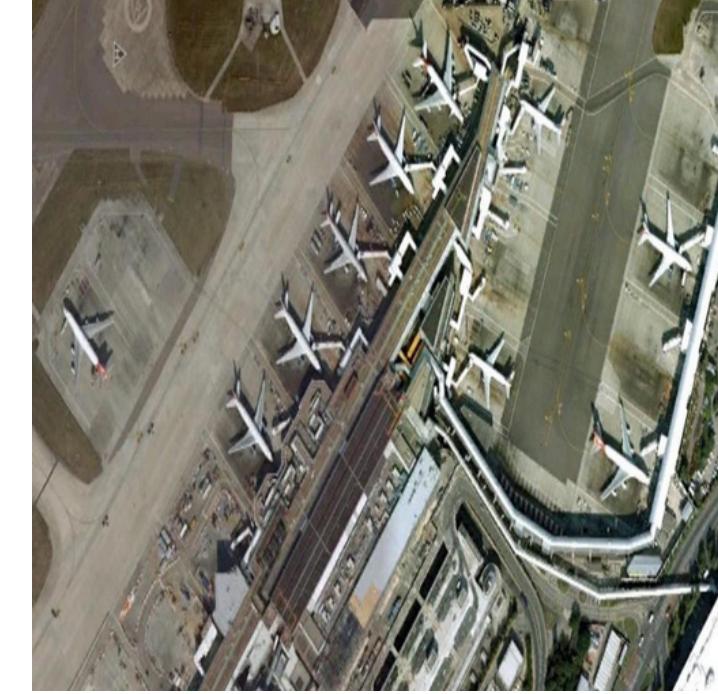
Rotation



Luminosité



Snow



Blur

# RÉPARTITION DU DATA

Ensemble Positif : **650 images**

+

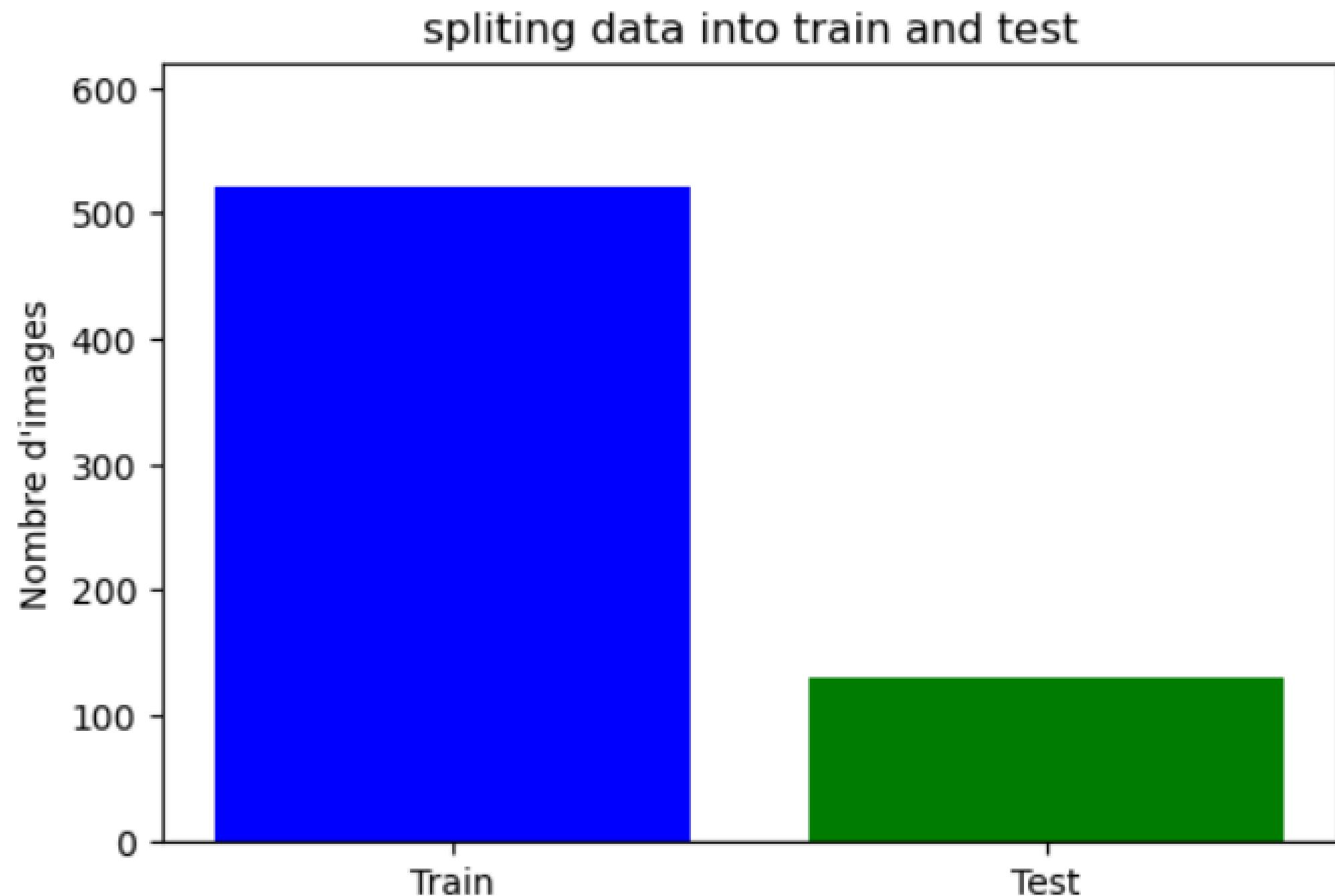
**Annotations**

↓

Train  
**520**

↓

Test  
**130**



# **MODELISATION**

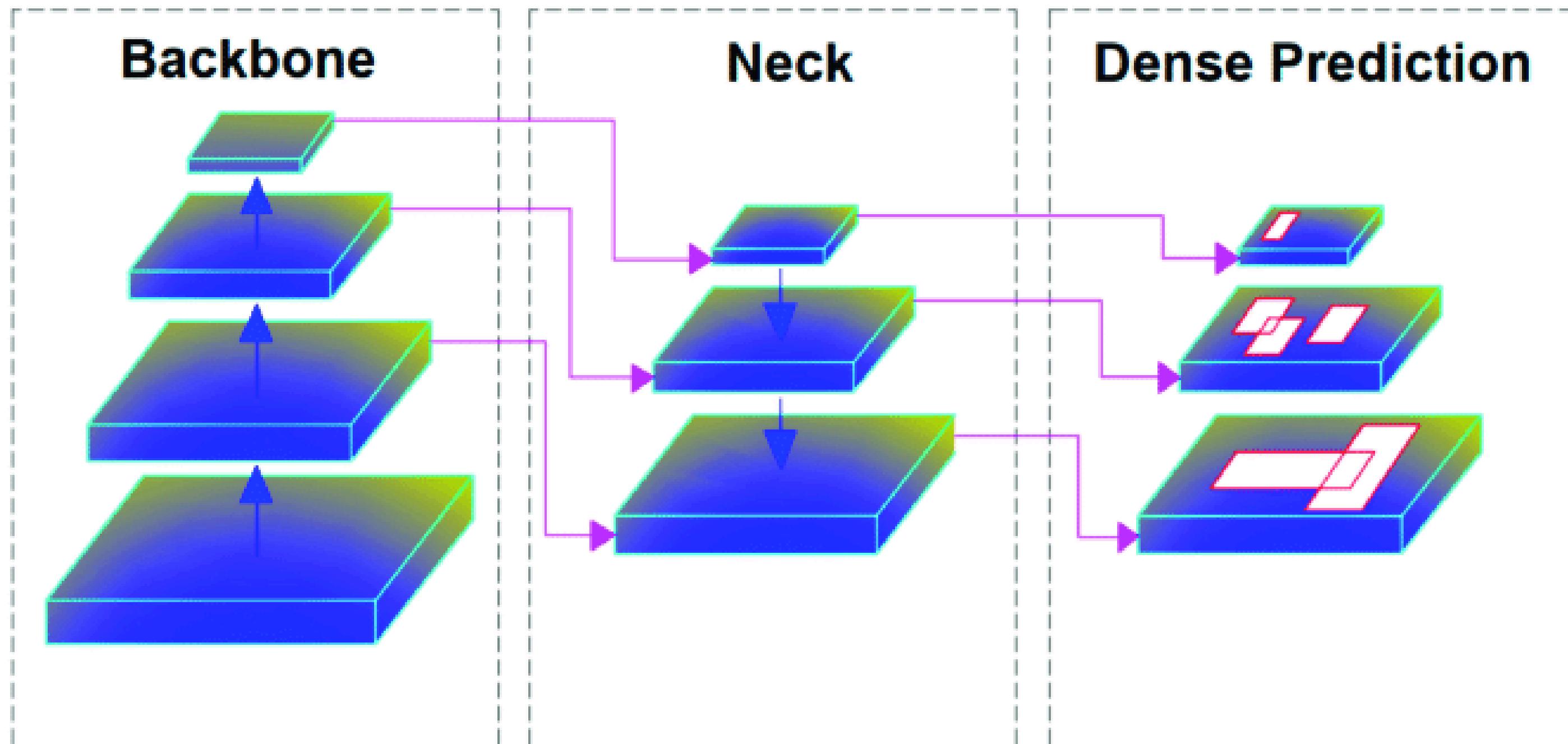
**&**

# **EVALUATION**



# ARCHITECTURE

Yolo



réseau de convolution profond, fournit des représentations riches en caractéristiques qui alimentent le reste du réseau.

Fusionne les caractéristiques de différentes couches pour améliorer la détection des objets à diverses échelles.

La tête du réseau contient plusieurs couches prédictives qui produisent les boîtes englobantes et les scores de classe pour chaque objet détecté.

# YOLO V5

**Réutilisation des Couches :** Lors du fine-tuning, les couches initiales (le backbone) sont gelées (on ne les entraîne plus) tandis que les couches supérieures (têtes de détection) sont ajustées aux nouvelles données.

**Re-calibrage des Classes :** Les prédictions de classes doivent être adaptées au nombre exact de classes du nouveau dataset.

## Choix hyperparamètres :

Lr = 0.01

Epochs = 50

Optimizer = SGD

Batch size = 8

## data.yaml

```
train: /content/data_images/train
val: /content/data_images/test
nc: 10

names: [
    'airplane',
    'ship',
    'storage tank',
    'baseball diamond',
    'tennis court',
    'basketball court',
    'ground track field',
    'harbor',
    'bridge',
    'vehicle'
]
```

## Ultralytics

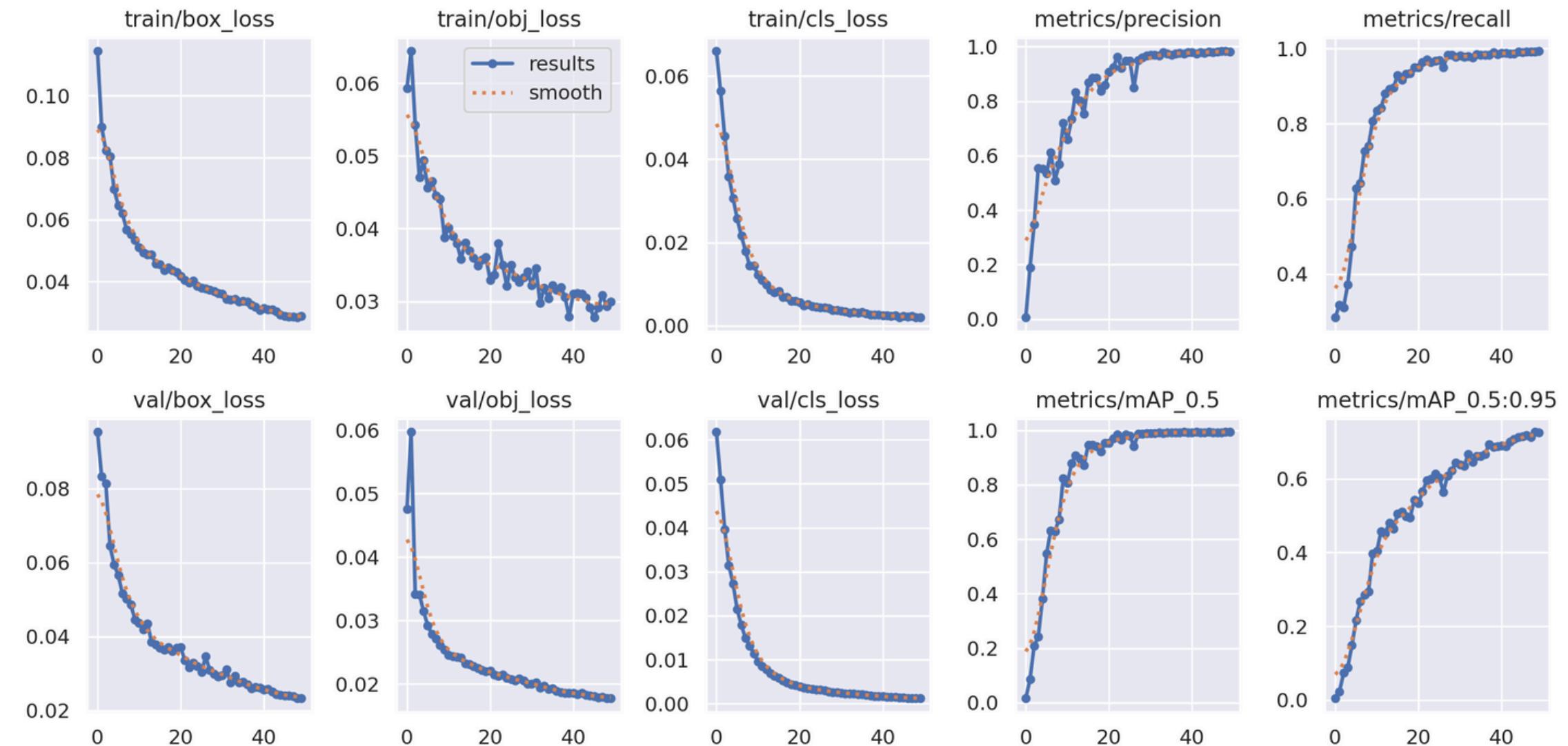
Biblio Python pour la détection d'objets, basée sur PyTorch



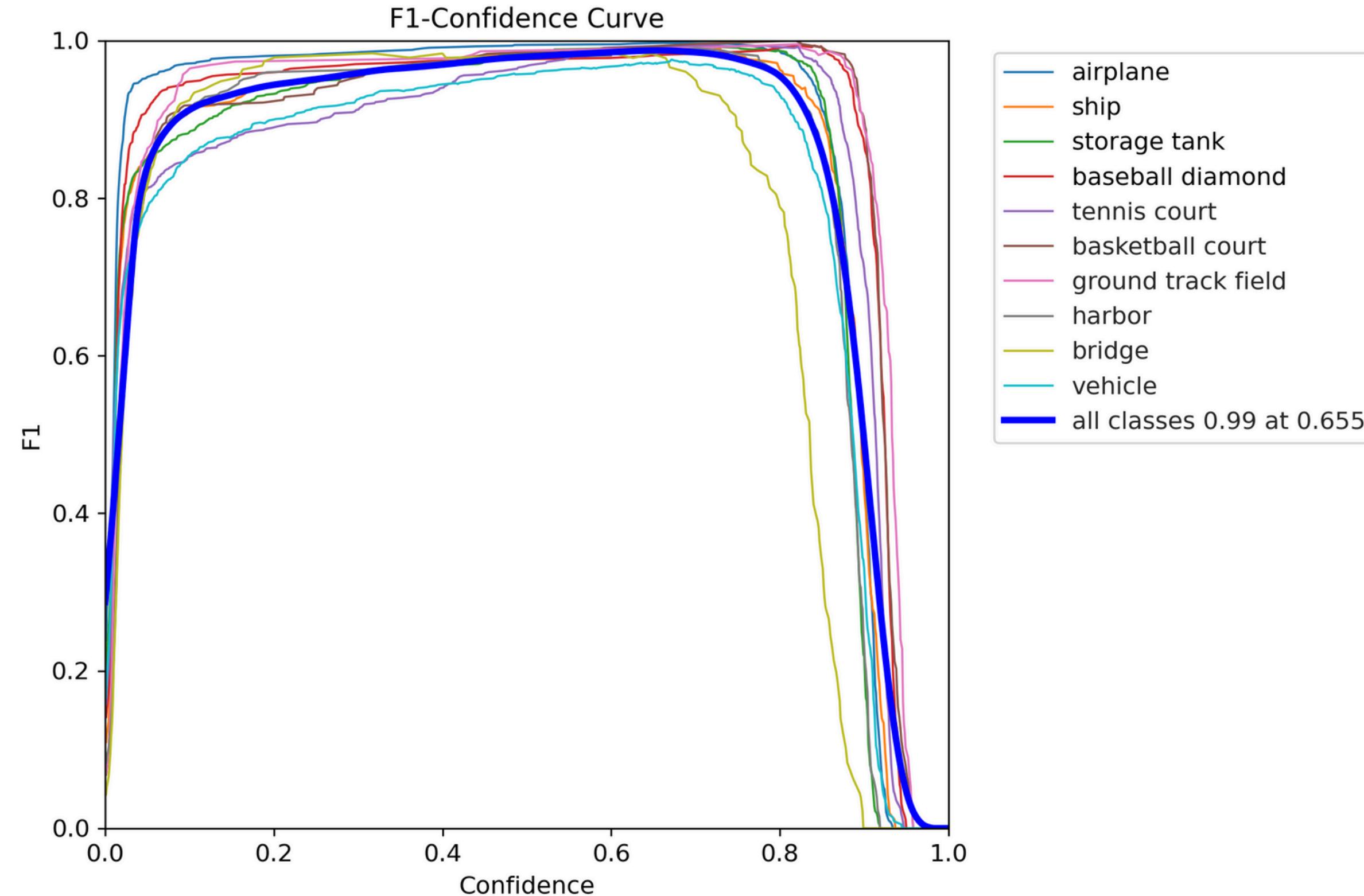
# EVALUATION

YOLOv5s summary: 157 layers, 7039792 parameters, 0 gradients, 15.8 GFLOPs

Class	Images	Instances	P	R	mAP50	mAP50-95: 100% 29/29 [00:16<00:00, 1.81it/s]
all	455	2672	0.984	0.992	0.994	0.727
airplane	455	547	0.994	1	0.995	0.741
ship	455	197	0.988	0.995	0.994	0.694
storage tank	455	442	0.987	0.995	0.993	0.68
baseball diamond	455	244	0.964	1	0.993	0.779
tennis court	455	325	0.986	1	0.995	0.744
basketball court	455	117	0.986	1	0.995	0.777
ground track field	455	110	0.98	1	0.995	0.853
harbor	455	157	0.987	0.994	0.993	0.737
bridge	455	92	1	0.966	0.994	0.584
vehicle	455	441	0.967	0.975	0.993	0.683

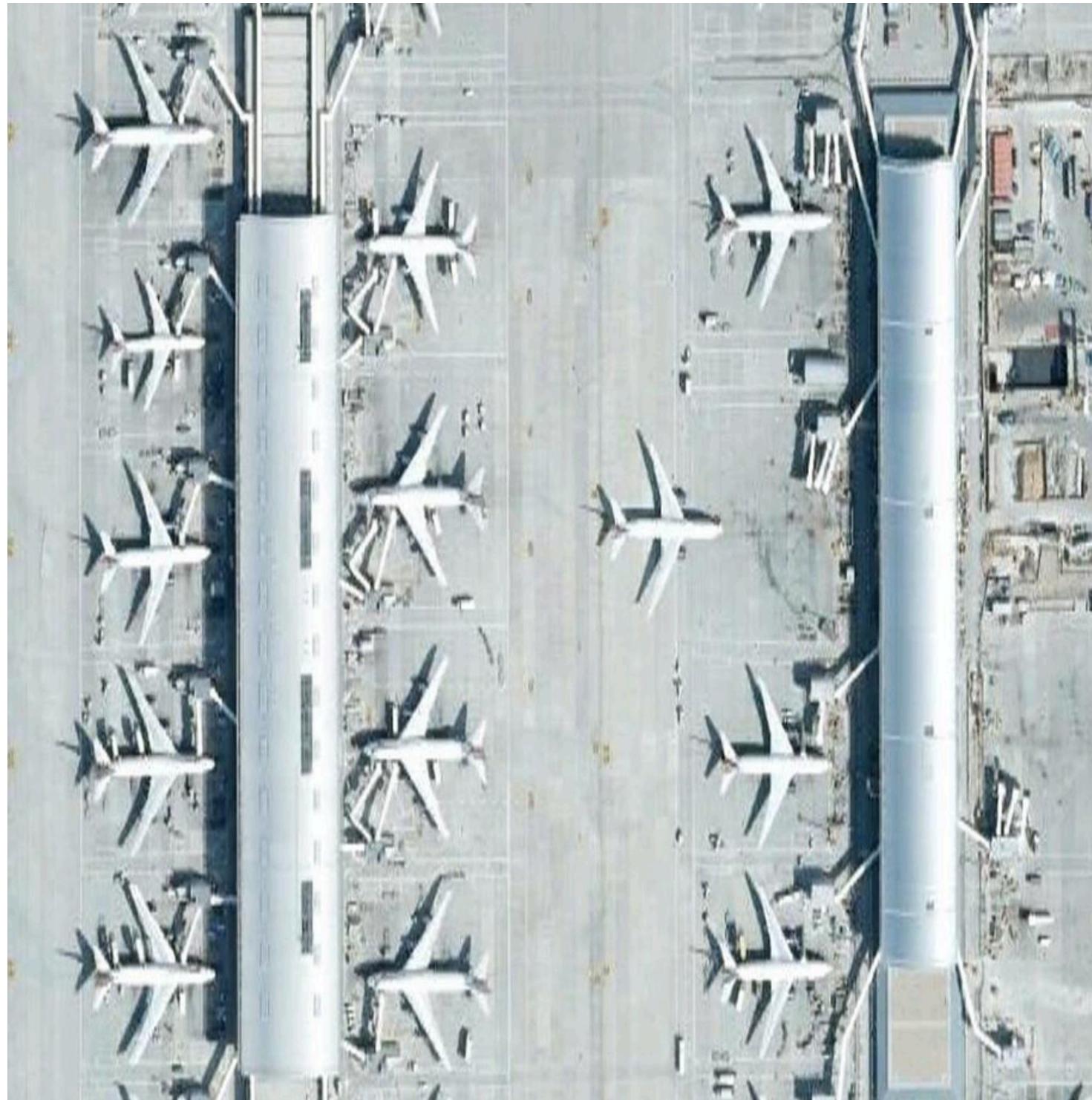


# EVALUATION



Yolo

# EXEMPLE



Yolo

# EXEMPLE



# YOLO V8

**Réutilisation des Couches :** Lors du fine-tuning, les couches initiales (le backbone) sont gelées (on ne les entraîne plus) tandis que les couches supérieures (têtes de détection) sont ajustées aux nouvelles données.

**Re-calibrage des Classes :** Les prédictions de classes doivent être adaptées au nombre exact de classes du nouveau dataset.

## Choix des hyperparamètres :

Lr = 0.01

Epochs = 10

Optimizer = Adam

Batch size = 8

data.yaml

```
train: /content/gdrive/MyDrive/yolov8/custum_dataset/train/images
val: /content/gdrive/MyDrive/yolov8/custum_dataset/validation/images
test: /content/gdrive/MyDrive/yolov8/custum_dataset/test/images

nc: 10
names: ['airplane', 'ship', 'storage_tank', 'baseball_diamond',
'tennis_court', 'basketball_court', 'ground_track_field',
'harbor', 'bridge', 'vehicle']
```

**Ultralytics**

Biblio Python pour la détection d'objets, basée sur PyTorch



# EVALUATION

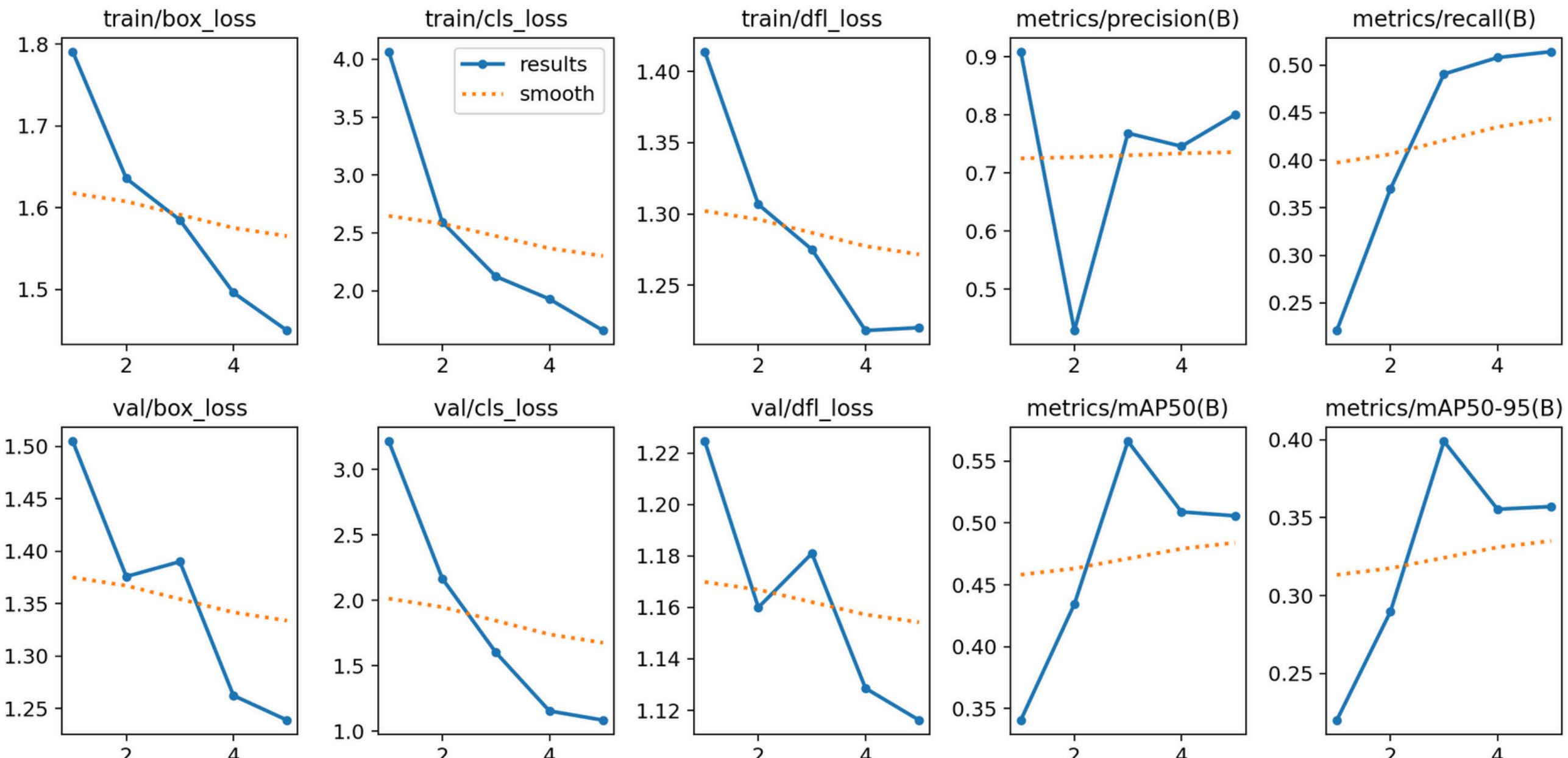
Yolo

ultralytics YOLOv8.2.6 🚀 Python-3.10.12 torch-2.2.1+cu121 CPU (Intel Xeon 2.20GHz)

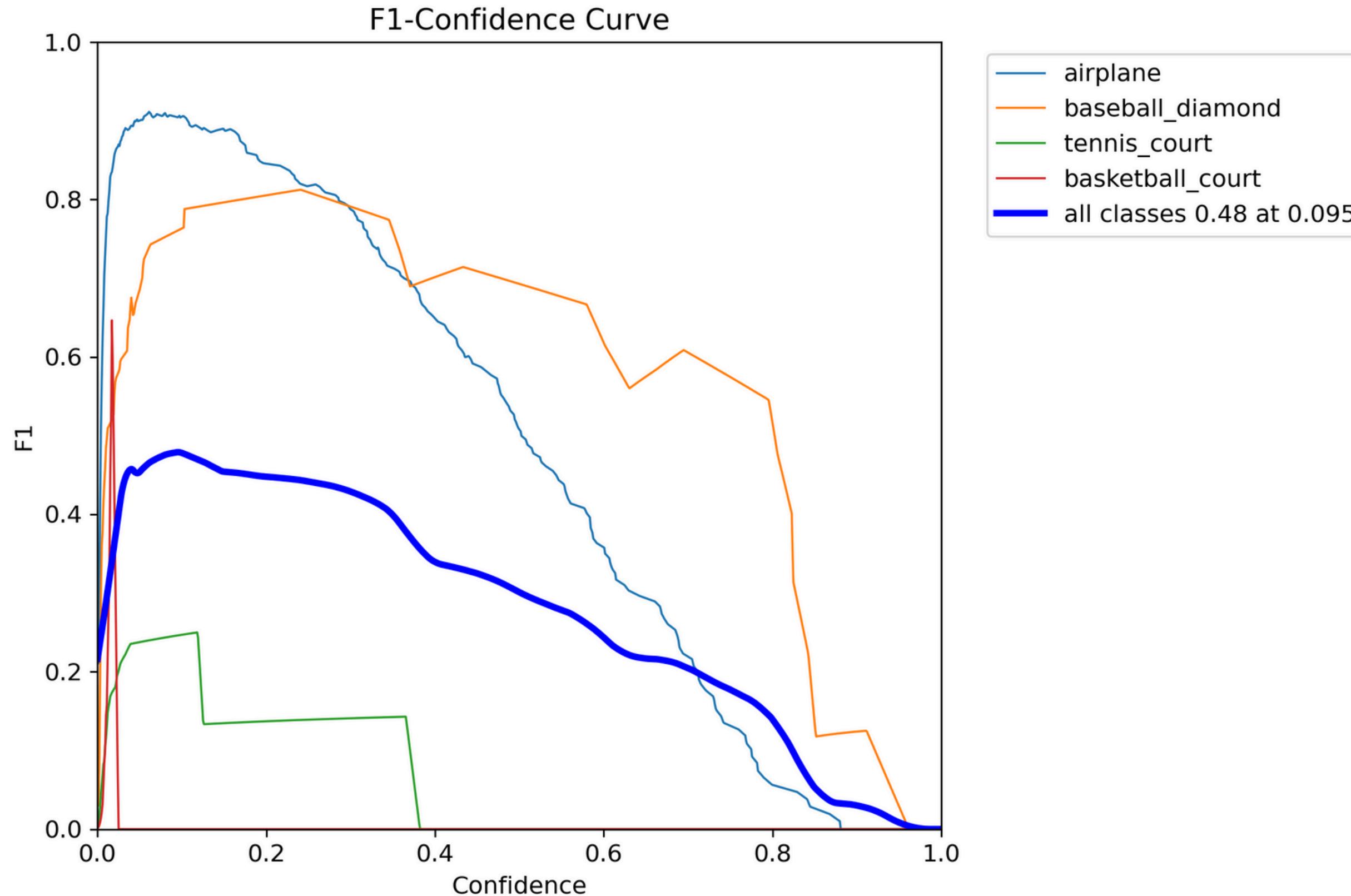
Model summary (fused): 168 layers, 3007598 parameters, 0 gradients, 8.1 GFLOPs

Class	Images	Instances	Box(P)	R	mAP50	mAP50-95	Time
all	36	234	0.769	0.49	0.566	0.399	
airplane	36	207	0.95	0.865	0.953	0.55	
baseball_diamond	36	14	0.648	0.929	0.732	0.544	
tennis_court	36	12	0.477	0.167	0.0832	0.0545	
basketball_court	36	1	1	0	0.497	0.448	

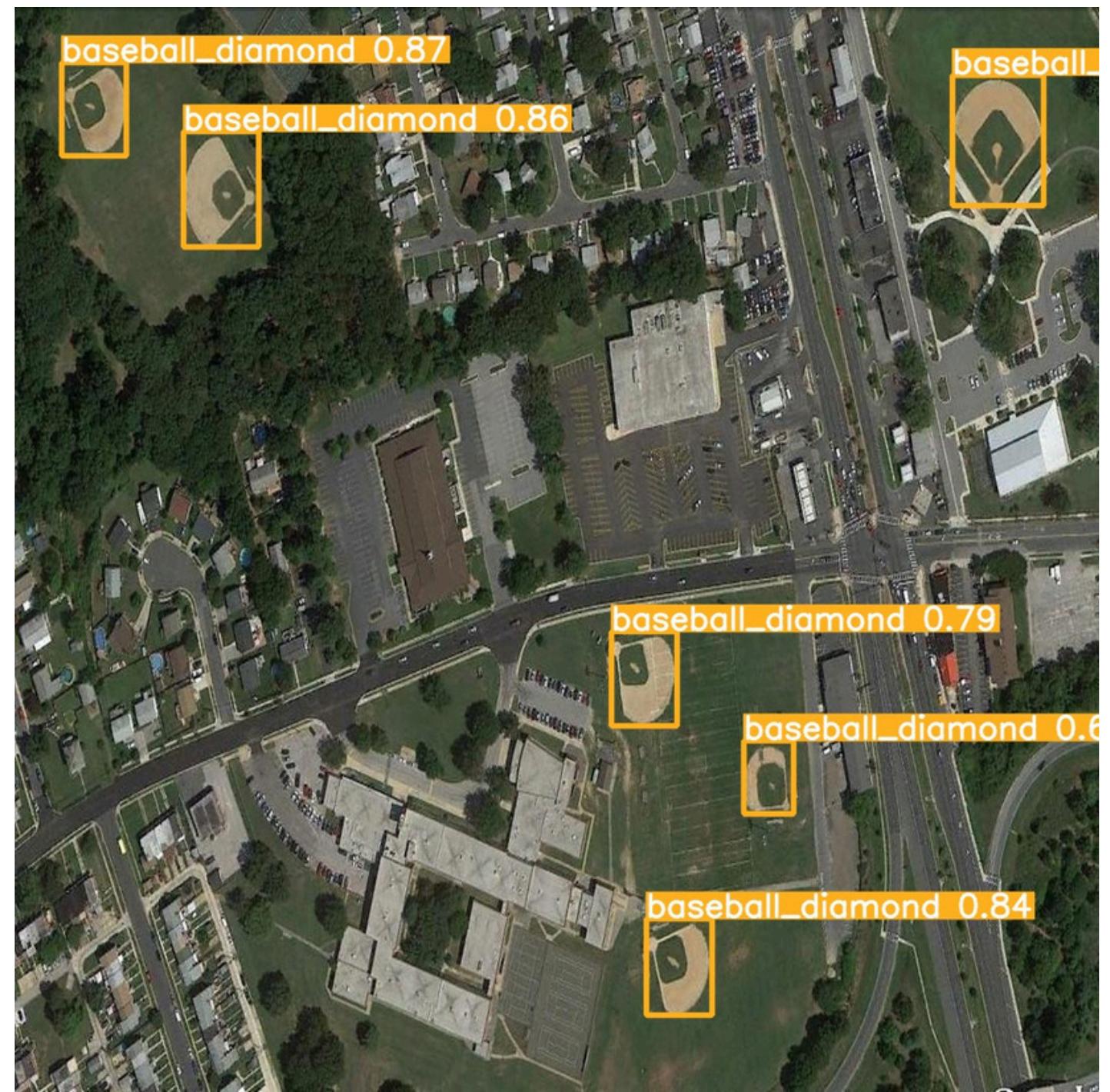
Speed: 14.2ms preprocess, 335.2ms inference, 0.0ms loss, 21.8ms postprocess per image



# EVALUATION



# EXEMPLE

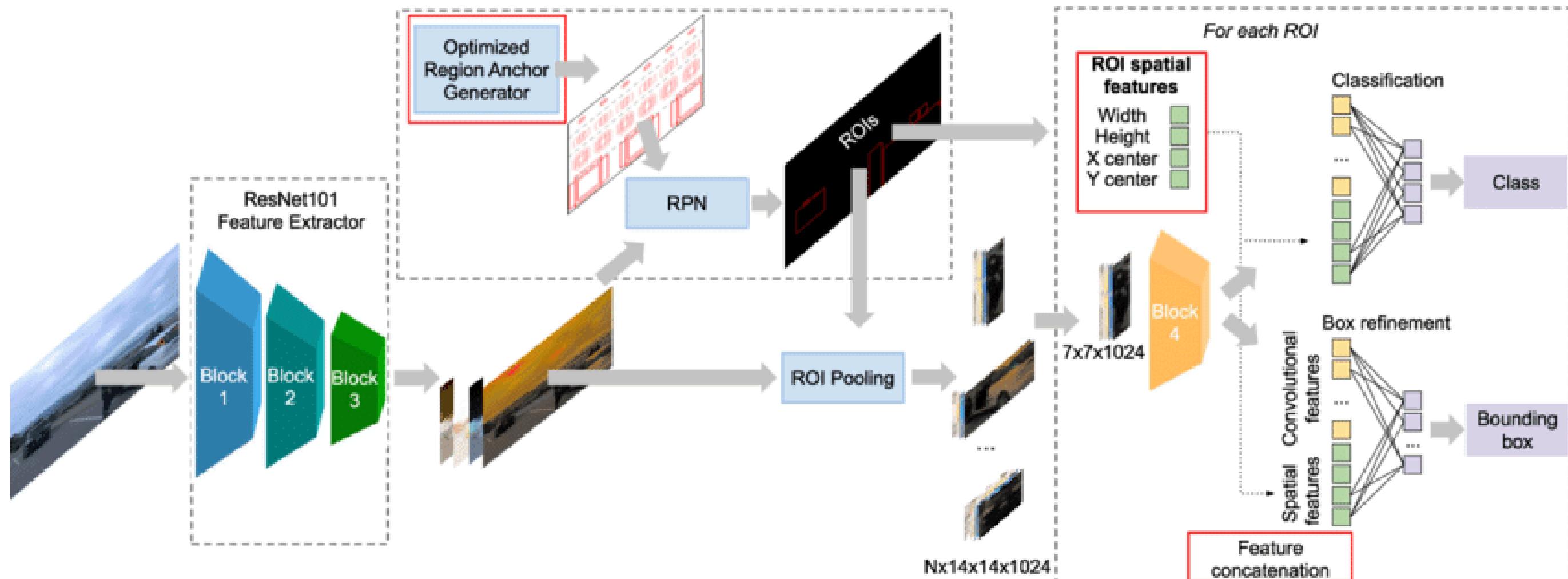


# YOLOv5 VS YOLOv8

Yolo

Critère	YOLOv5	YOLOv8
Architecture	Backbone basé sur CSPDarknet, PANet, Head	Backbone plus optimisé, PANet amélioré, Head affiné
Vitesse	Rapide, optimisé pour l'inférence en temps réel	Encore plus rapide avec des optimisations avancées
Formats pris en charge	ONNX, TensorRT, CoreML	Compatibilité accrue pour les frameworks modernes
Avantages	Modèle stable, bien documenté, rapide	Plus précis, plus modulaire, meilleur équilibre vitesse/précision
Inconvénients	Moins optimisé par rapport aux dernières avancées	Nécessite potentiellement des ajustements pour certains pipelines

# ARCHITECTURE FASTER R-CNN



## Backbone

Utilise un réseau de neurones convolutionnel pré-entraîné, comme VGG16 ou ResNet-50/101, pour extraire les caractéristiques des images d'entrée.

## RPN (Region Proposal Network)

Il génère ces propositions sous forme de boîtes englobantes, souvent appelées "anchors", avec des scores de confiance.

## ROI Pooling

Cette couche ajuste les propositions générées par le RPN à une taille uniforme, permettant ainsi un traitement uniforme par les couches suivantes.

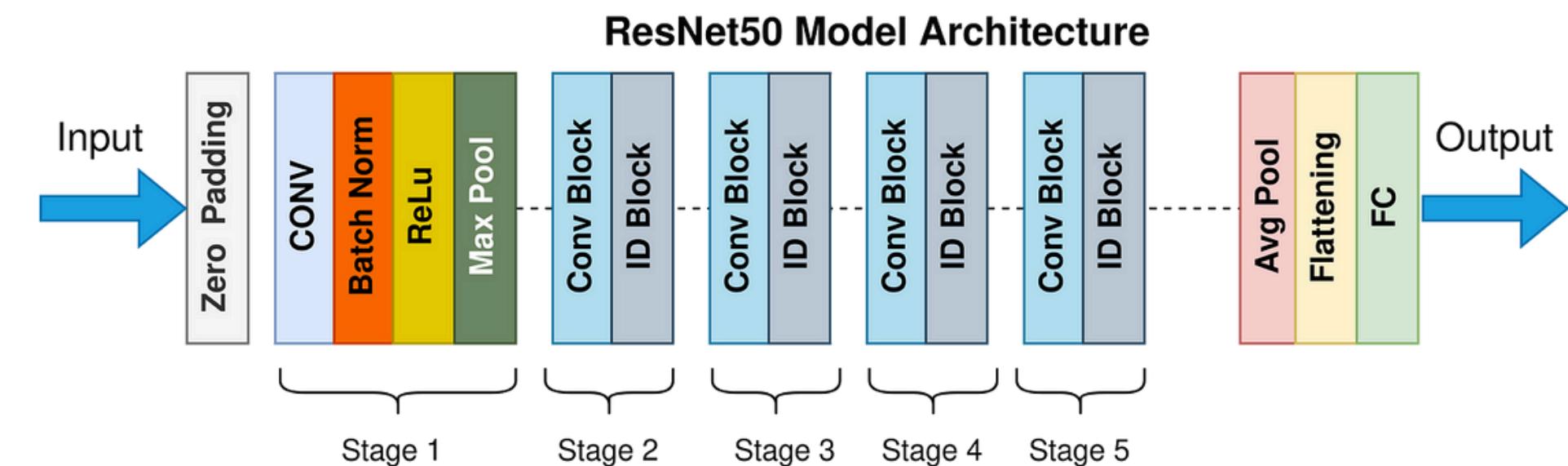
## Détecteur

- Classification : Prédire le type d'objet dans chaque ROI.
- Régression : Affiner les coordonnées des boîtes englobantes pour chaque ROI.

# ARCHITECTURE FASTER R-CNN

## Initialisation du modèle:

- Utilisez la bibliothèque torchvision pour charger un modèle Faster R-CNN préentraîné avec une architecture ResNet-50.
- affiner la couche de prédiction pour correspondre au nombre de classes souhaité.



## Fixation des hyperparamètres:

**Learning rate :** 0.005

**Optimiseur :** L'optimiseur sélectionné est SGD  
(Stochastic Gradient Descent)

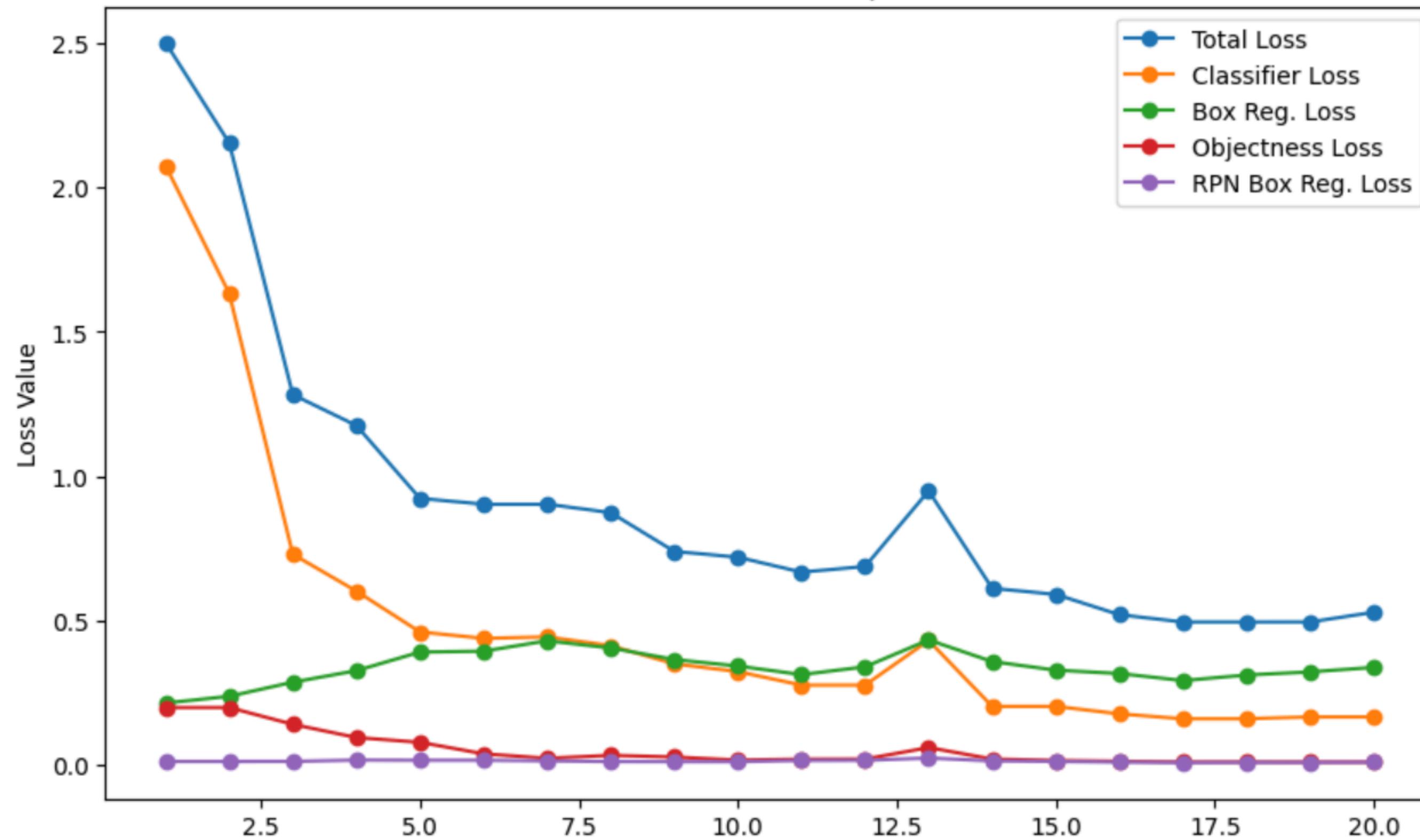
**batch size =** 10

**nombre d époque=** 20

**Facteur de réduction du Learning rate :** (gamma) :  
0.1 par 3 époques.

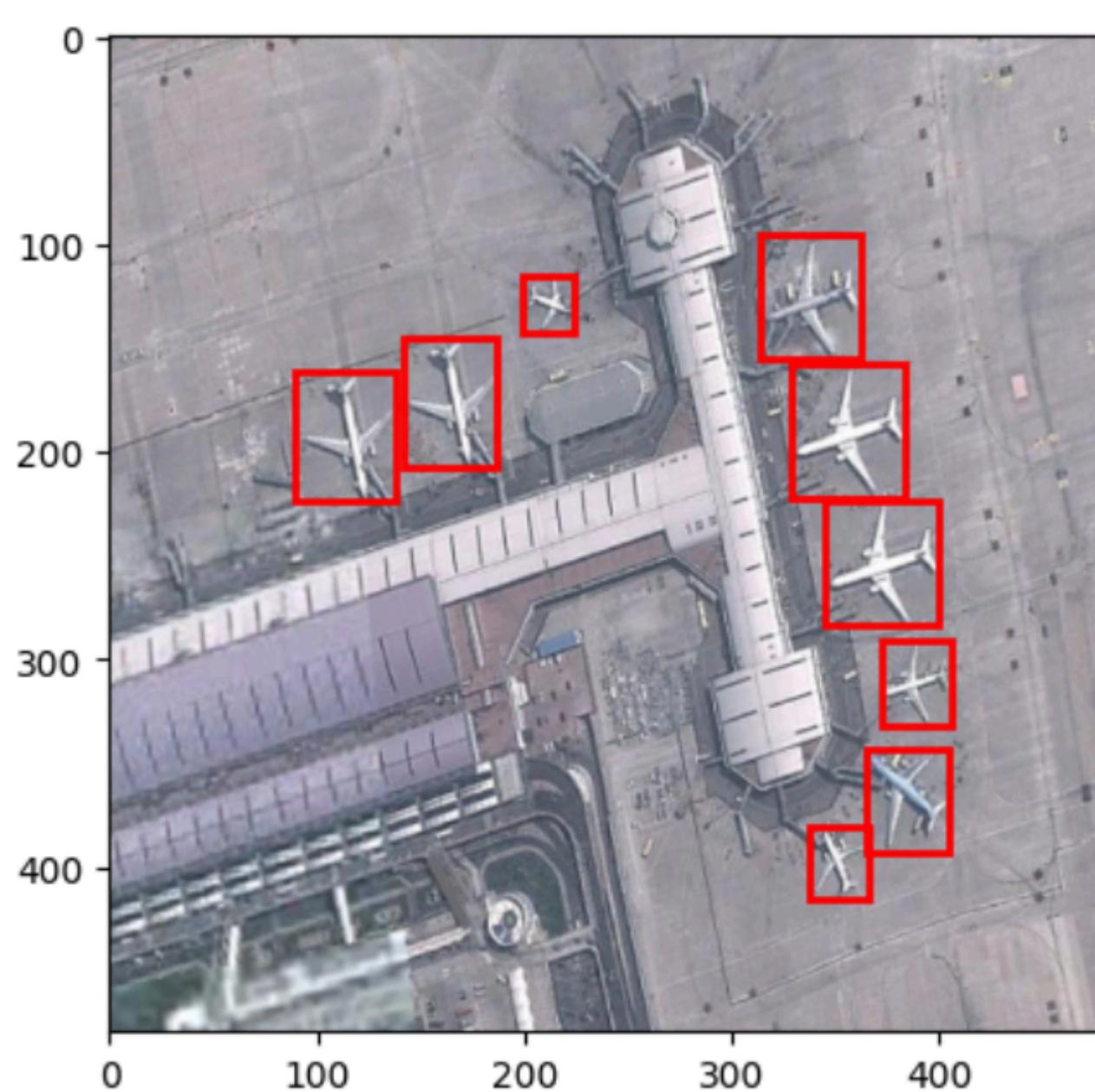
# EVALUATION

Loss Metrics Over Epochs

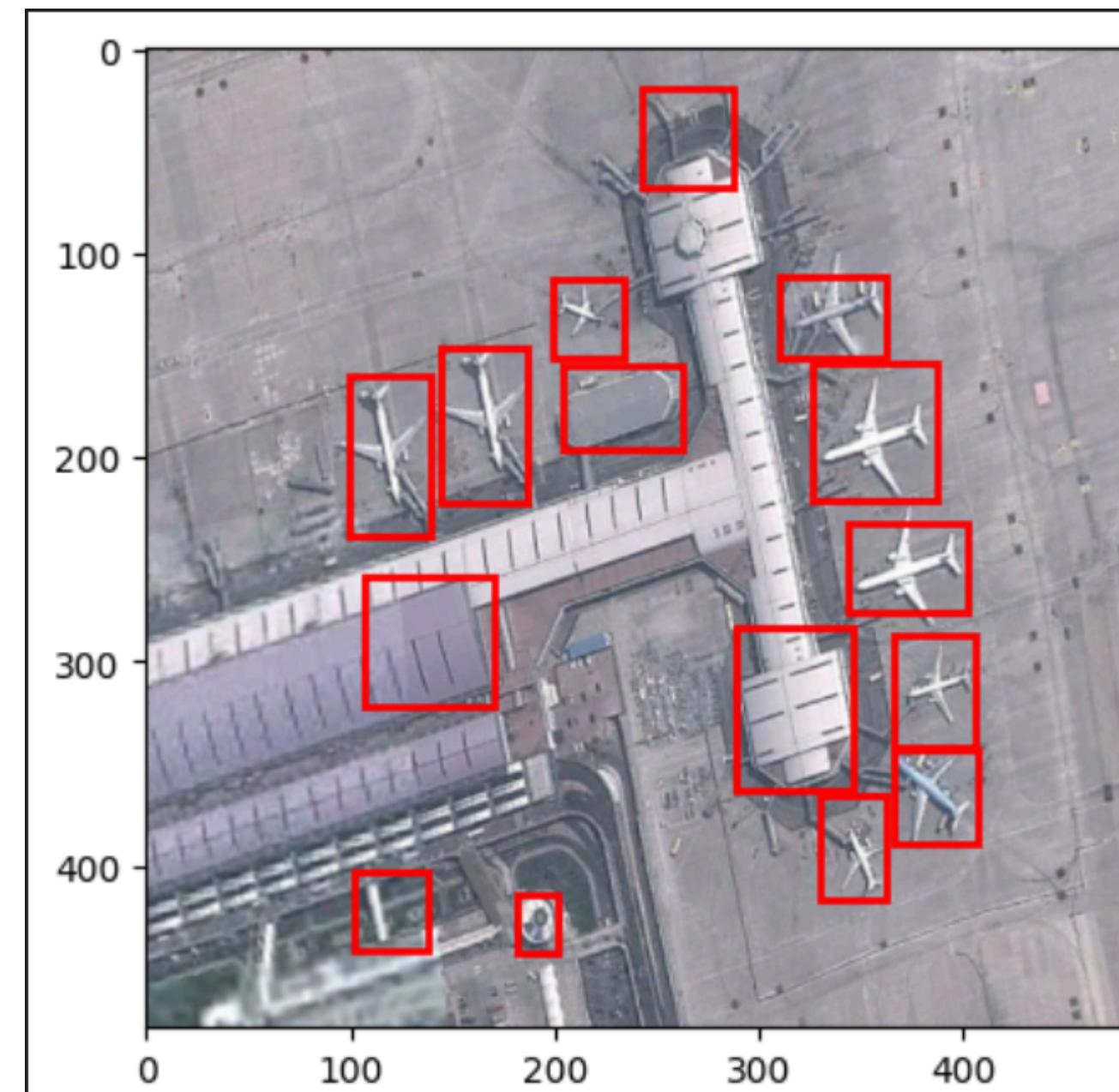


# EXAMPLE

EXPECTED OUTPUT



MODEL OUTPUT



# MASK R-CNN : EXTENSION DE FASTER R-CNN



S'appuie sur la structure de Faster R-CNN en ajoutant une fonctionnalité de segmentation d'instance.



**Détection Avancée** : Tout comme Faster R-CNN, Mask R-CNN peut détecter et classer les objets dans les images, mais il ajoute la capacité de créer des masques binaires précis.



**Génération de Masques** : Fournir des masques binaires pixellisés pour chaque instance d'objet détectée, permettant une segmentation plus précise.



**Détection et Classification** : Identifier et classer les objets dans les images tout en affinant les boîtes de délimitation.

# MASK R-CNN : EXTENSION DE FASTER R-CNN



**Backbone** : Un réseau de neurones pré-entraîné (comme ResNet) utilisé pour extraire les caractéristiques visuelles des images.



**RPN** : Génère les propositions de boîtes candidates pour les objets potentiels.



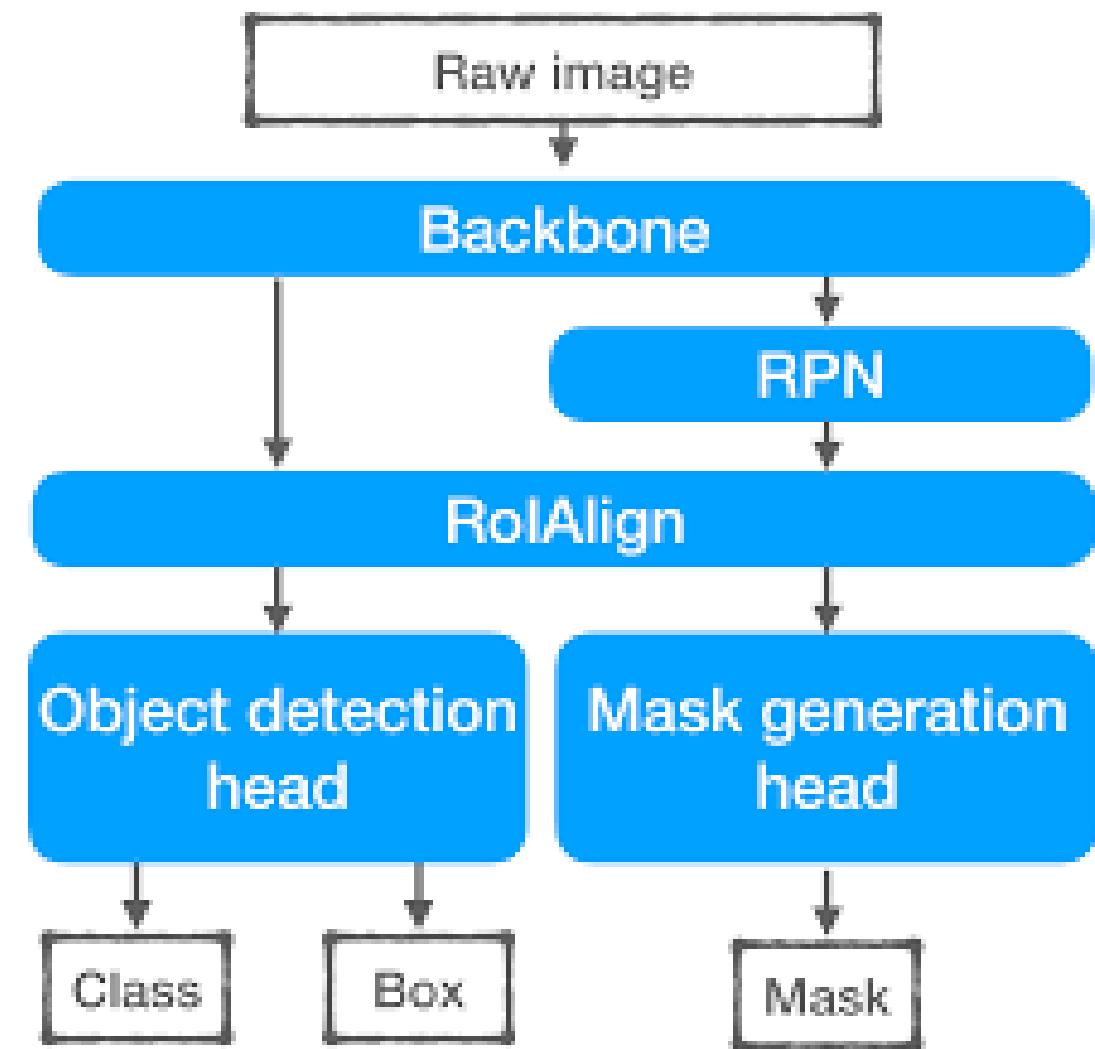
**RoIAlign** : Remplace RoIPooling pour aligner les régions d'intérêt plus précisément avec l'image d'origine, préservant les informations spatiales.



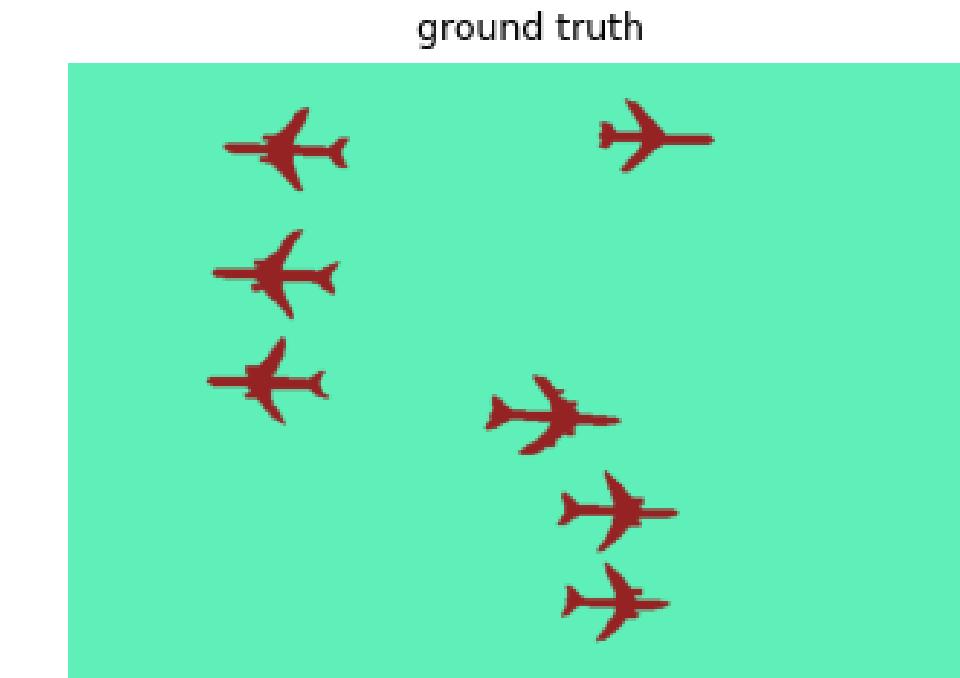
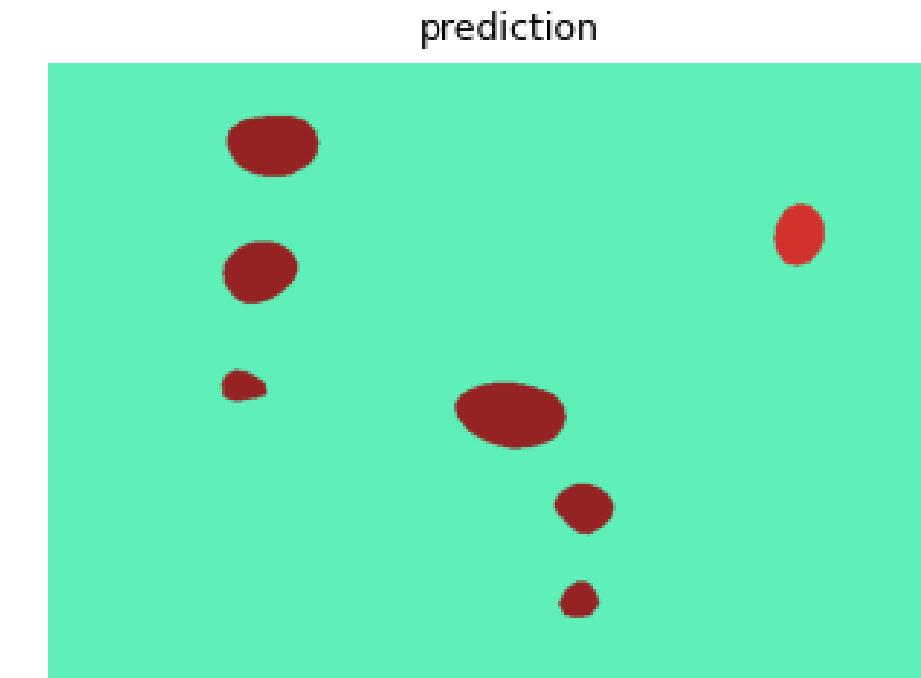
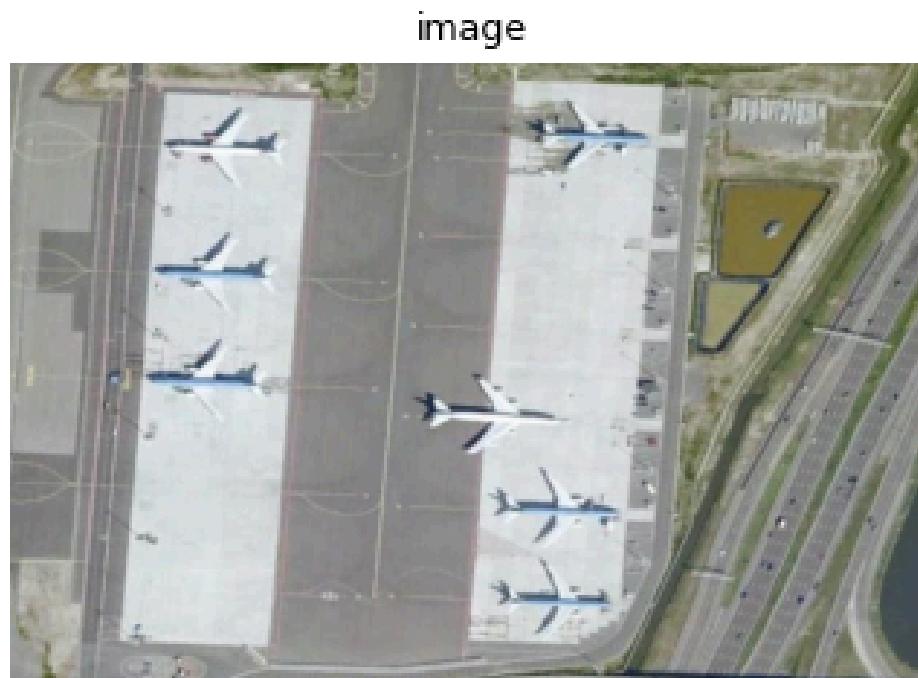
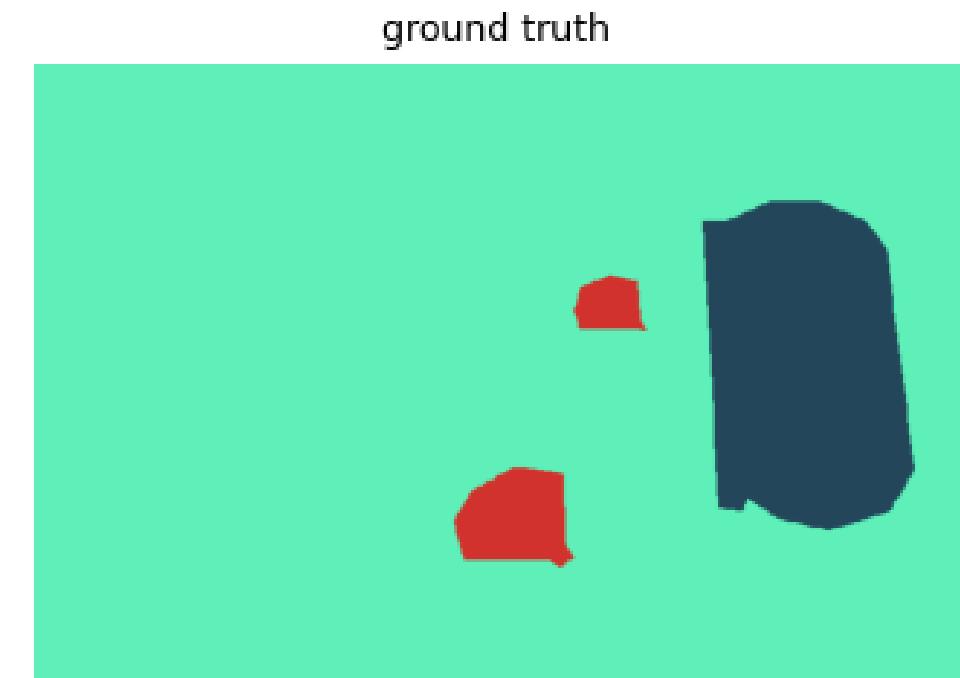
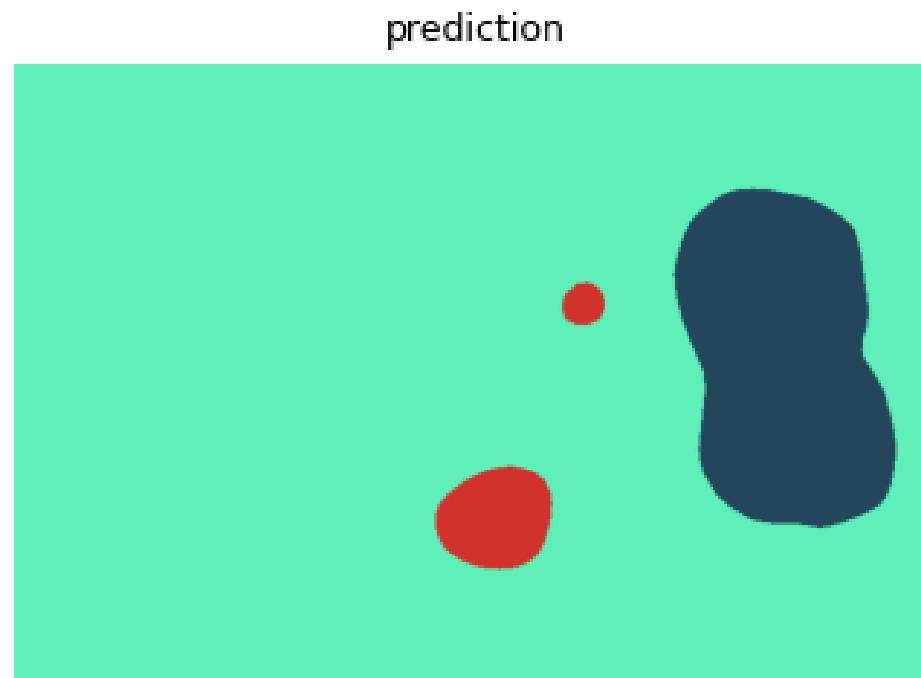
**Tête de Détection** : Affine les boîtes proposées par le RPN et assigne les classes correspondantes.



**Tête de Masque** : Génère des masques binaires pour chaque instance d'objet détectée dans les boîtes.

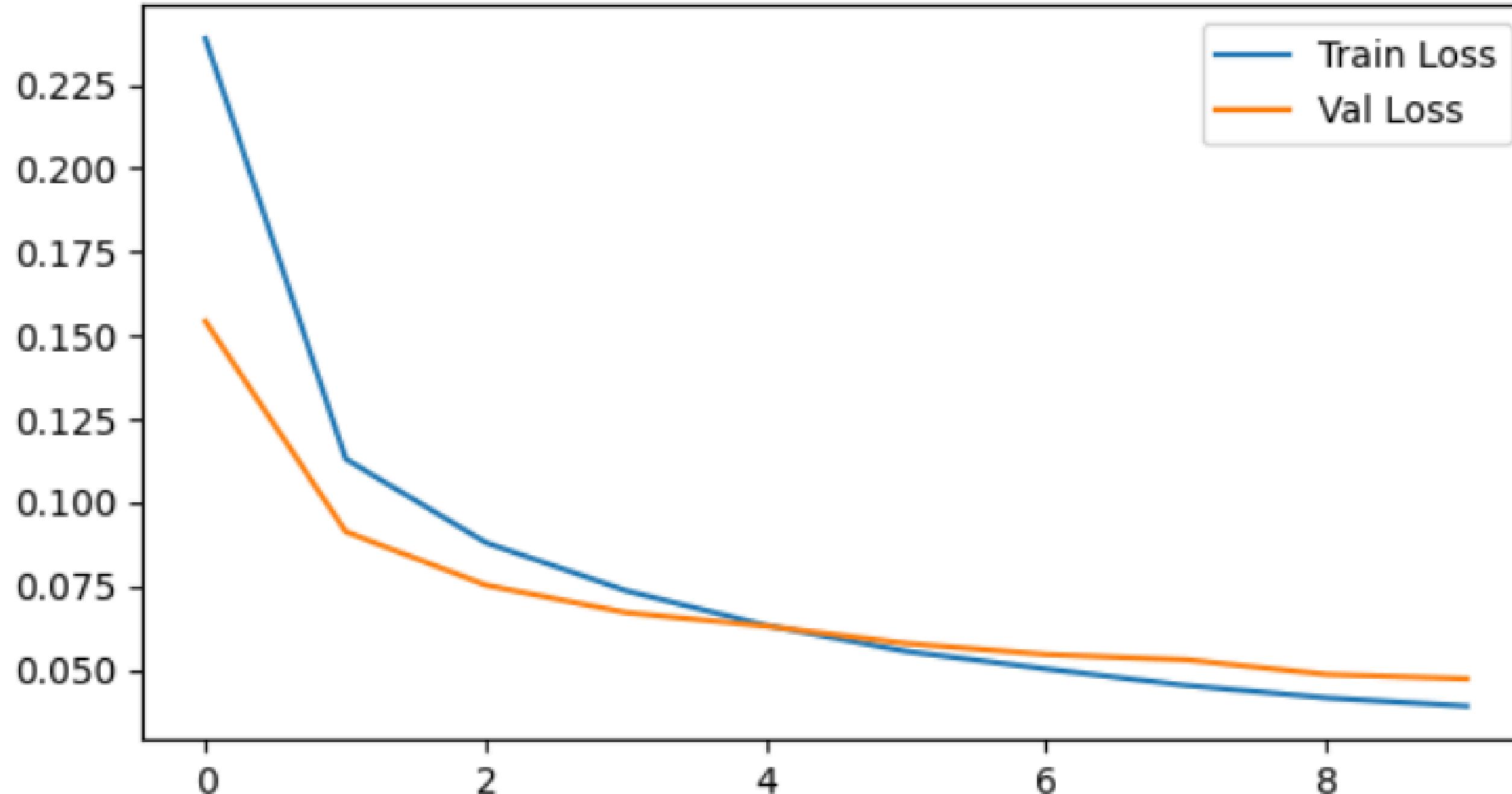


# EXAMPLE

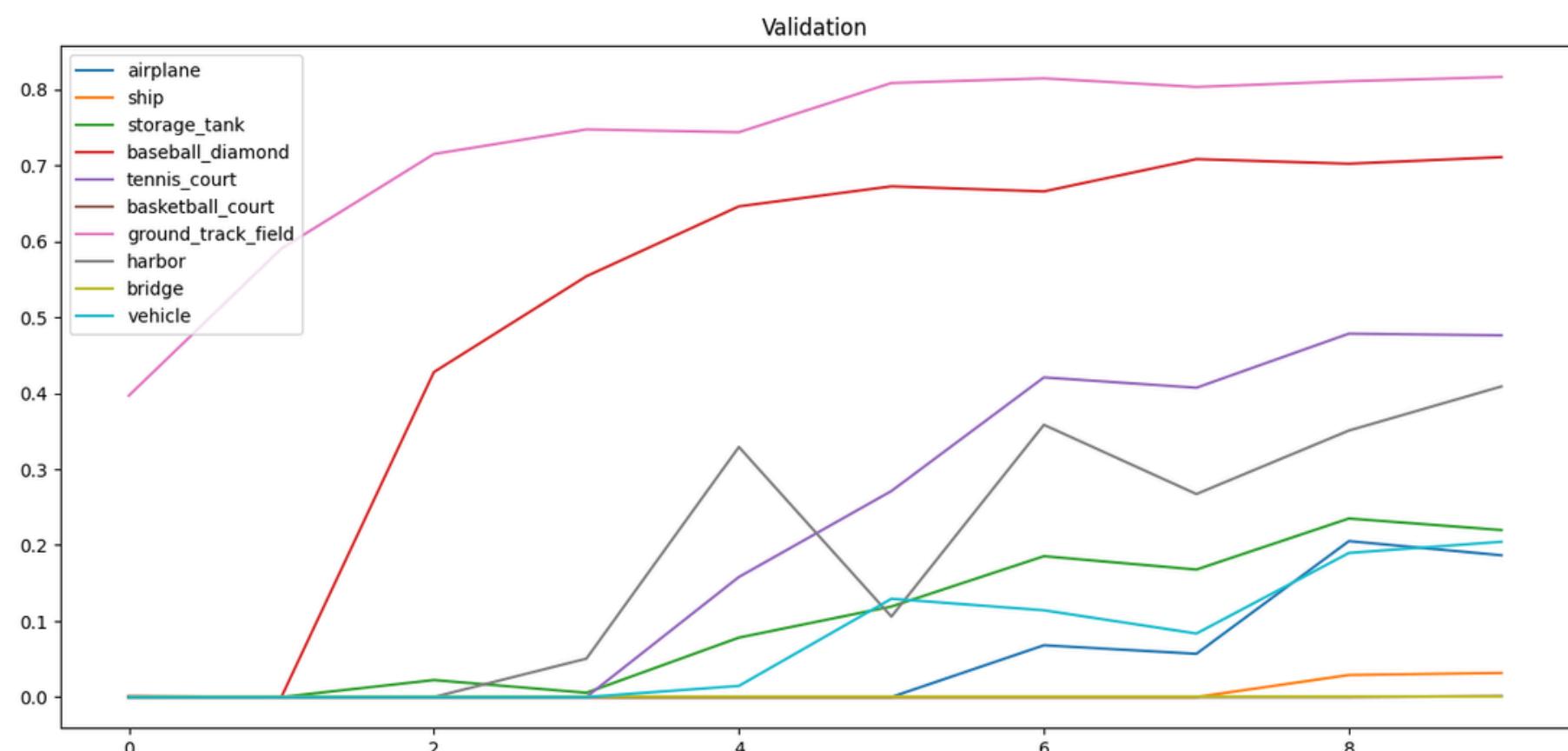
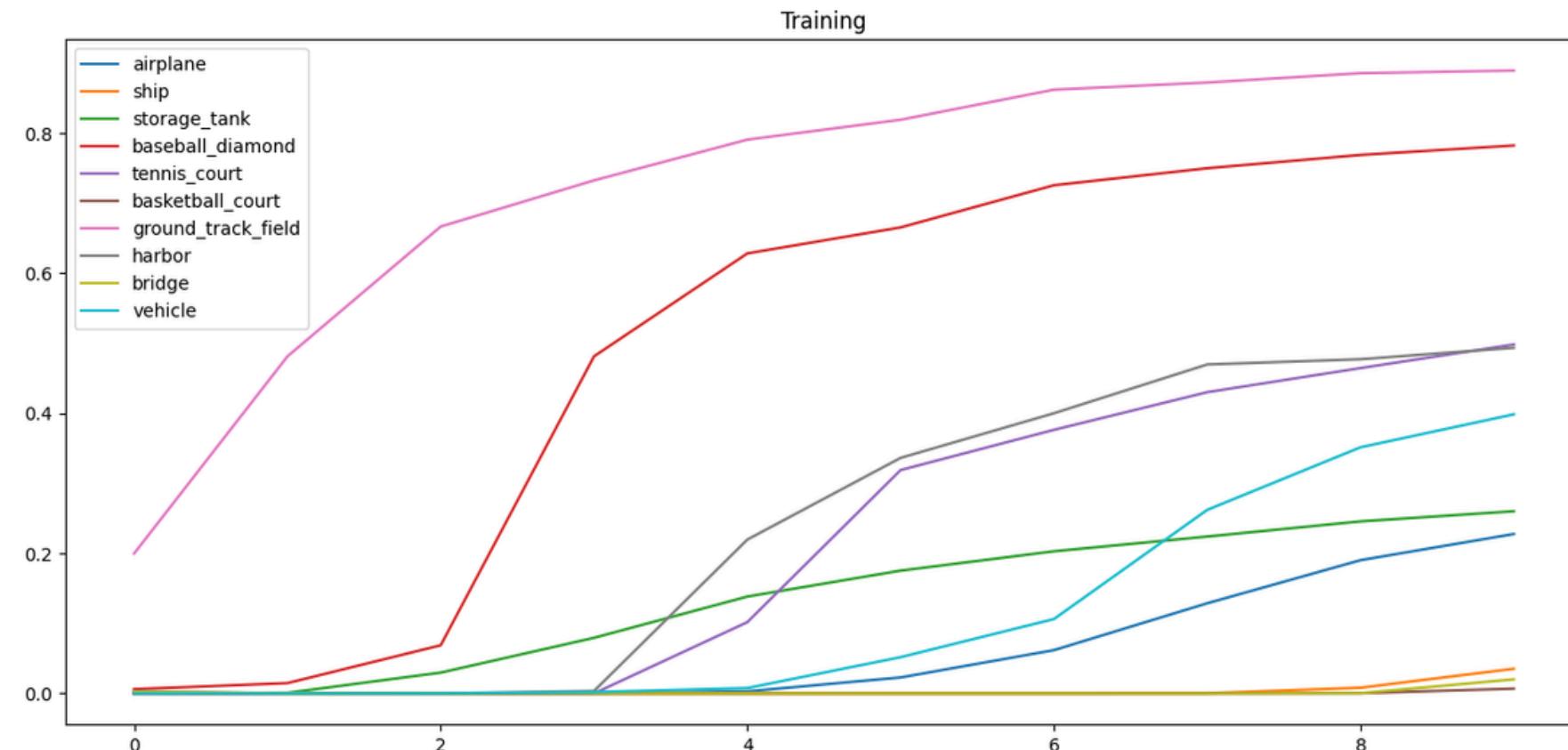


# EVALUATION

Loss



# EVALUATION



# REMERCIEMENT

# DÉPLOIEMENT





www.canva.c... Je vend iPhone 11... > Tous les favoris

Adam Joe  
Admin

Dashboard / Dashboard

YOLO 5

Term & Conditions Privacy & Policy

A screenshot of a web browser showing a user profile for "Adam Joe Admin". The profile includes a small icon, a notification bell, and a message icon. Below the profile is a link to "Dashboard / Dashboard". In the center, there is a pink button with the text "YOLO 5" and a shopping cart icon. At the bottom of the page, there are links for "Term & Conditions" and "Privacy & Policy". The browser interface shows standard icons for back, forward, and search.

# MERCI!