École Spéciale de Mécanique et d'électricité

INGE4 Majeure Intelligence Artificielle

Rapport de stage

Implémentation d'un modèle de classification vidéo par intelligence artificielle

Stagiaire : Jérémie Degorce-Dumas
Tuteur : Clément Macadré

Du 10/06/2024 au 27/09/2024

Chantier Naval Couach
Rue de l'Yser 33470 Gujan-Mestras FRANCE
Tel : +33 (0)5 56.22.35.50
www.couach.com

1 Remerciements

Je tiens à exprimer ma profonde gratitude et mes sincères remerciements à toutes les personnes qui ont contribué à la réussite de mon stage au sein du Chantier Naval Couach. Leur soutien, leur expertise et leurs encouragements ont grandement enrichi mon expérience professionnelle.

Je tiens tout d'abord à remercier chaleureusement mon tuteur de stage, M. Clément MACADRE, pour sa guidance et ses précieux conseils tout au long de mon stage. Sa patience, son professionnalisme et sa disponibilité ont été d'une aide précieuse dans la réalisation de mes missions.

Je tiens également à exprimer ma reconnaissance envers M. Jérémy DUMORA, le responsable du département Technologies de la R&D. Sa confiance et son soutien constant ont été des moteurs essentiels dans la réussite de mon stage.

Mes remerciements vont également à toute l'équipe de la R&D pour leur accueil chaleureux, leur collaboration et leur partage de connaissances. Chaque membre de l'équipe a contribué à ma formation, leur esprit d'équipe et leur bienveillance ont créé un environnement de travail stimulant et convivial.

Je tiens à remercier l'ensemble du personnel du chantier pour son amabilité et sa coopération. Leur accueil chaleureux et leur ouverture d'esprit ont contribué à rendre mon stage agréable et enrichissant.

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

Table des matières

1 Remerciements **1**

2 Introduction **3**
2.1 Chantier Naval Couach 3
2.2 Présentation du projet 4

3 Classification d'objets en mer par IA **5**
3.1 Notions Fondamentales 5
3.1.1 Classification d'objets 5
3.1.2 Détection d'objets 5
3.1.3 Transfert Learning 6
3.1.4 Data augmentation 7
3.2 Datasets 8
3.2.1 Structure et fonctionnement 8
3.2.2 Choix des datasets : ROBOFLOW UNIVERSE 9
3.2.3 Datatsets importants 10
3.3 Méthodes de détection d'objets 12
3.3.1 Détection "Single Shot" : YOLO (You Only Look Once) 12
3.3.2 Detection "Double Shot" : R-CNN (Regional Convolutional Neural Network) 16
3.4 Interpretation des metrics et hyperparametres 17
3.4.1 Interpretation metrics 17
3.4.2 mAP / IoU 18
3.4.3 Hyperparametres 19
3.5 Implémentation du modèle 20

4 Management **21**

5 Développement Durable **22**

6 Conclusion **23**

7 Annexes **24**
7.1 Liens importants 24
7.2 Liens recherches bibliographique 24

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

2 Introduction

2.1 Chantier Naval Couach

Le Chantier Naval Couach est une entreprise emblématique dans la construction de yachts de luxe et de bateaux de haute performance.
Fondée en 1897 par Albert Couach, cette société française s'est imposée comme un leader dans le domaine de la construction navale, avec une expertise qui s'étend sur plus de 120 ans.

Basé à Gujan-Mestras, sur le bassin d'Arcachon, il abrite l'ensemble des locaux qui vont permettre l'étude préliminaire du navire jusqu'à sa fabrication et sa mise à l'eau. Cela comprend : un pôle design, un département recherche et développement, un bureau d'études et d'ingénierie ainsi que tous les différents ateliers de production. Nos Yachts de luxe et navires professionnels sont donc entièrement créés, développés et construits au sein même du chantier.
Le chantier combine tradition et technologies modernes pour garantir la qualité de ses navires. Par exemple, l'utilisation de matériaux composites légers et résistants, comme le kevlar, permet d'améliorer la performance et la durabilité des bateaux tout en réduisant leur poids. En parallèle, les solutions écoresponsables, comme l'intégration de moteurs à faible consommation et de systèmes de propulsion innovant, témoignent de l'attention portée à l'impact environnemental.

Depuis plus de 50 ans, Le chantier naval Couach fournit aux administrations militaires des patrouilleurs et des intercepteurs pour que les opérateurs puissent assurer le maintien de l'ordre dans leurs pays.
En 2018, le chantier met en place une des chaînes industrielles les plus efficaces au monde qui va permettre au chantier la fabrication de 79 intercepteurs de 17 mètres en un temps record dans l'industrie naval.

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

2.2 Présentation du projet

La classification d'images par l'IA consiste à reconnaître des objets spécifiques dans une nouvelle image à l'aide d'un modèle d'IA entraîné sur un ensemble d'images labellisées.
De nombreux modèles de reconnaissance existent actuellement, mais la plupart se contentent d'une classe générale de "bateaux".
Une proposition de projet consisterait à utiliser l'apprentissage par transfert pour obtenir un modèle qui classifie plus finement les catégories de bateaux.
En bonus : détection de nageurs (utile pour le sauvetage) et détection de bouées marines (utile pour se conformer au COLREG).

Le projet couvrira les sujets suivants : recherche bibliographique pour trouver des bases de données annotées (par exemple Singapore Marine Dataset) et choisir un modèle d'IA (par exemple YoloV5) se renseigner sur l'augmentation des données et l'apprentissage par transfert.
Enfin, nous vérifierons que le modèle généralise correctement lors d'essais en mer.

Ce rapport illustrera plus la partie recherche bibliographique/théorique du projet, la partie "pratique" se trouve sur github à l'adresse suivante : https://github.com/JeremDD/Classification_Objets_Marins_IA.git

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

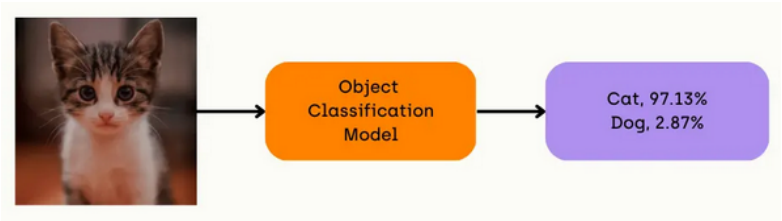
3 Classification d'objets en mer par IA

3.1 Notions Fondamentales

La vision par ordinateur est un domaine qui vise à permettre aux ordinateurs d'interpréter et de comprendre les informations visuelles de la même manière que les humains. Il implique le développement d'algorithmes, de modèles et de systèmes capables d'analyser et de comprendre des images et des vidéos, ainsi que d'en extraire des informations utiles.

3.1.1 Classification d'objets

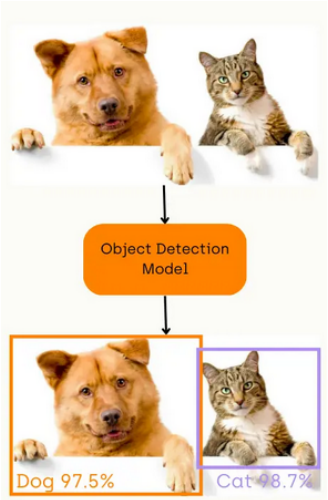
Tâche qui vise à identifier et catégoriser les objets dans une image en classes prédéfinies. Généralement cette classification est effectuée par des techniques d'apprentissage automatique, dans lesquelles un modèle est formé sur un ensemble de d'images étiquetées et leurs étiquettes de classe associées. Après ça le modèle peut être utilisé pour classer de nouvelles images en leur attribuant une étiquette en fonction de leurs caractéristiques apprises.



3.1.2 Détection d'objets

La détection d'objet est une tâche de vision par ordinateur qui traite de l'identification et de la localisation d'objets de certaines classes prédéfinies dans les images d'entrée.

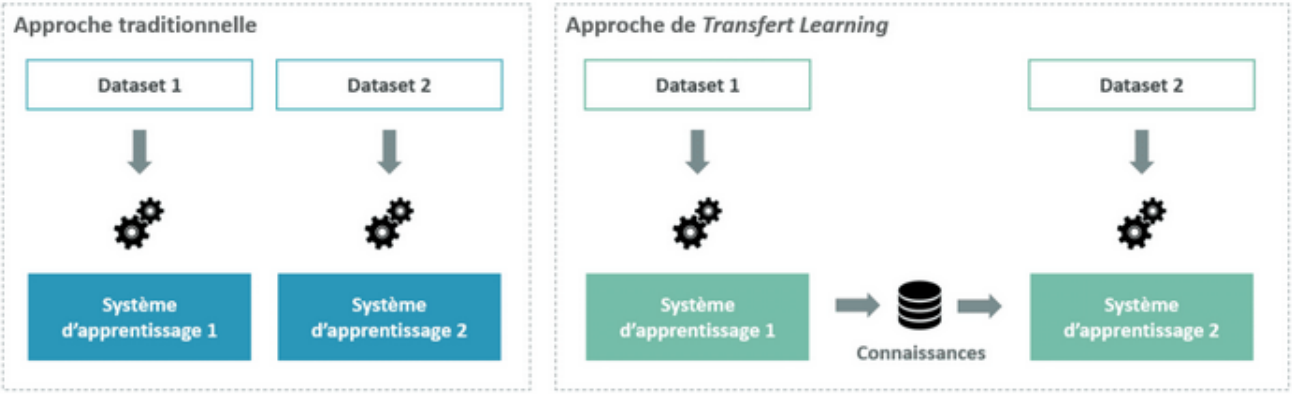
Il existe plusieurs algorithmes et modèles de détection d'objets, tel que YOLO (que l'on va utiliser dans ce projet) mais aussi la gamme R-CNN, SSD ... Ces algorithmes et modèles sont utilisés dans diverses applications, telles que les voitures autonomes, les systèmes de surveillance et le suivi d'objets.



Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

3.1.3 Transfert Learning

Le Transfer Learning désigne l'ensemble des méthodes qui permettent de transférer des connaissances acquises à partir de la résolution d'un problème (domaine source) pour traiter un autre problème (domaine cible).



Le processus consiste à utiliser un modèle pré entraîné, en conservant ses couches déjà optimisées pour une tâche spécifique. Les premières couches, qui extraient des caractéristiques générales, sont figées pour préserver les connaissances acquises. La dernière couche, spécifique à la tâche d'origine, est supprimée pour pouvoir adapter le modèle à une nouvelle tâche. Ensuite, on ajuste les hyperparamètres ou l'architecture du réseau pour améliorer les résultats sur la nouvelle tâche

On dénombre plusieurs types de transfert learning :

- **Inductive :** Les domaines source et cible sont les mêmes (même données), mais les tâches sont différentes. Le modèle pré entraîné connaît déjà les données source et s'entraîne plus rapidement aux nouvelles fonctions.
  Exemple : Utiliser un modèle entraîné pour la détection d'animaux sur des images pour construire un modèle capable d'identifier des chiens.

- **Unsupervised :** Les domaines source est cible sont les mêmes, mais les tâches sont différentes et les données ne sont pas labellisées. Le modèle apprend les caractéristiques communes des données non étiquetées afin de les généraliser avec plus de précision lorsqu'on leur demande d'effectuer une tâche cible.
  Exemple : Identifier différents types de motos sur les images de la circulation. D'abord le modèle est entraîné sur un grand nombre d'images de véhicules non étiquetées ou il détermine indépendamment les similitudes et les caractéristiques entre différents types de véhicules. Ensuite, le modèle est présenté à un petit ensemble spécifique d'images de motos.

- **Transductive :** Les domaines source et cible sont différents (quand même connexe), mais les tâches sont similaires. Les données cibles étant mathématiquement semblables aux données source, le modèle trouve des modèles et fonctionne plus rapidement.
  Exemple : Adapter un modèle d'analyse des sentiments basé sur les critiques de produits pour analyser les critiques de films

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

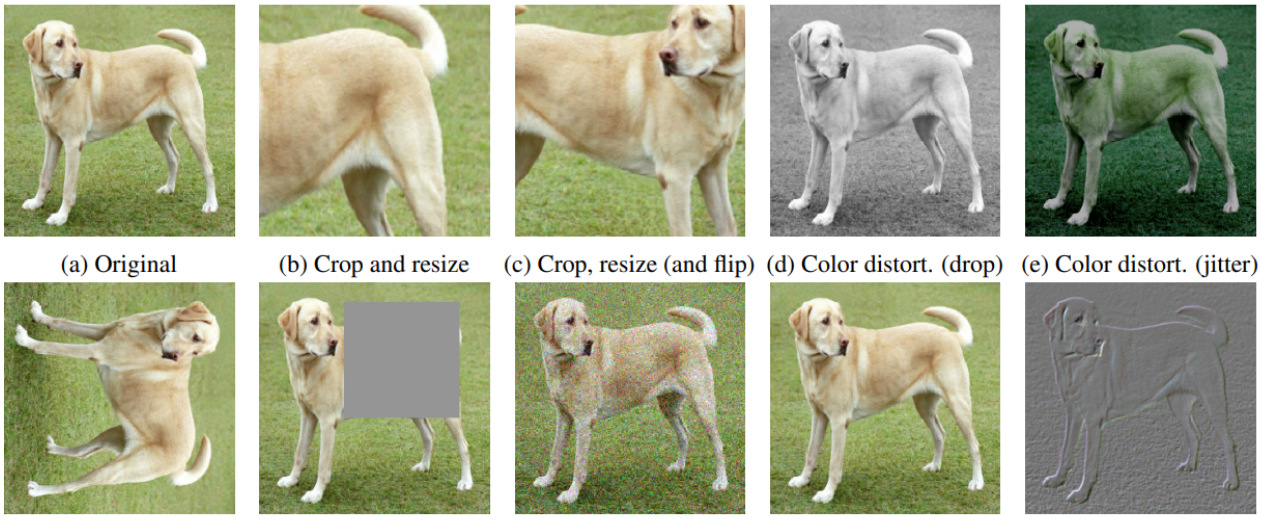
3.1.4 Data augmentation

La data augmentation (augmentation de données) est une technique utilisée pour augmenter la quantité et la diversité des données disponibles pour l'entraînement d'un modèle de machine learning.
Cette technique consiste à créer des variations des données d'entraînement existantes à l'aide de transformations diverses, ce qui aide à améliorer la robustesse et la capacité de généralisation du modèle.

La data augmentation aide à réduire l'overfitting, améliorant ainsi la capacité du modèle à généraliser sur des données nouvelles. Elle permet également d'augmenter artificiellement la taille du dataset, ce qui est particulièrement utile en cas de petit ensemble de données.

Types courants de transformations pour la data augmentation :

- **Transformations géométriques :**
  - Rotation
  - Translation
  - Redimensionnement (Scaling)
  - Miroir (Flip)
  - Découpage (Cropping)
- **Transformations photométriques :**
  - Modification de la luminosité
  - Modification du contraste
  - Modification de la saturation
  - Modification de la teinte
- **Ajout de bruit :**
  - Bruit gaussien : Ajouter du bruit aléatoire à l'image.
  - Distorsions : Appliquer des distorsions ou des filtres spécifiques.
- **Techniques avancées :**
  - Mosaic : Combiner plusieurs images en une seule pour augmenter la diversité des objets et des contextes dans une seule image.
  - MixUp : Mélanger deux images avec leurs annotations pour créer une nouvelle image synthétique.



(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

(f) Rotate $\{90°, 180°, 270°\}$ (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

3.2 Datasets

Un dataset, ou ensemble de données, est une collection structurée de données. Dans le contexte de la vision par ordinateur et de l'apprentissage automatique, un dataset est constitué d'images annotées avec des labels qui décrivent les objets présents dans chaque image. Pour la classification d'objets en mer, le dataset comprend des images de différents types de bateaux, bouées,etc., avec des annotations indiquant la classe de chaque objet.

3.2.1 Structure et fonctionnement

Avant de pouvoir utiliser un dataset pour un entraînement, il est essentiel de le préparer correctement :

- **Annotations :** Chaque image doit être annotée avec les objets qu'elle contient. Les annotations incluent généralement la classe de l'objet et les coordonnées de la bounding box qui l'entoure.
- **Partitionnement :** Le dataset est généralement divisé en sous-ensembles :
  - Formation (train) : utilisé pour entraîner le modèle.
  - Validation (val) : utilisé pour évaluer les performances du modèle.
  - Test (test) : utilisées pour évaluer la performance finale du modèle.

```
dataset/
 train/
    images/
       image_a.jpg
       ...
    labels/
       image_a.txt
       ...
 val/
    images/
       image_b.jpg
       ...
    labels/
       image_b.txt
       ...
 test/
    images/
       image_c.jpg
       ...
    labels/
       image_c.txt
       ...
```

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

L'entraînement d'un modèle de classification d'objets suit plusieurs étapes clés :

- **Chargement du Dataset :** Les images sont pré traitées (redimensionnées, normalisées) et les labels sont convertis en un format que le modèle peut comprendre.
- **Initialisation du Modèle :** On charge un modèle pré-entraîné. C'est ici que l'on utilise le transfert learning pour adapter le modèle pré-entraîné à une nouvelle tâche spécifique.
- **Entraînement :** Le modèle est entraîné en utilisant les images du dossier "train" :
  - Propagation Avant : Les images sont passées à travers le modèle pour obtenir des prédictions.
  - Calcul de la Perte : La différence entre les prédictions et les labels réels est calculée.
  - Propagation Arrière : Les gradients de la perte sont calculés et utilisés pour mettre à jour les poids du modèle.
  - Optimisation : Un optimiseur (comme SGD ou Adam) est utilisé pour ajuster les poids du modèle afin de minimiser la perte.
- **Validation :** Après chaque époque d'entraînement, le modèle est évalué sur le dataset de validation pour mesurer ses performances. Les métriques couramment utilisées incluent la précision (P), le rappel (R) et le Mean Average Precision (mAP)

3.2.2 Choix des datasets : ROBOFLOW UNIVERSE

Pour cette tache j'ai utilisé une plateforme nommée Roboflow. Il s'agit d'un outils dédié au traitement d'images et à la gestion de datasets, particulièrement utile dans le cadre des projets d'apprentissage automatique impliquant la vision par ordinateur.
Voici comment elle aide dans la recherche et la gestion de datasets :

- **Accès à des datasets variés** : Roboflow propose une vaste bibliothèque de datasets publics pour différents types de tâches en vision par ordinateur, comme la détection d'objets, la classification d'images ou la segmentation.

- **Création de datasets personnalisés** : Si vous avez des données propres, Roboflow vous permet de créer un dataset personnalisé en téléchargeant vos images et en les annotant. La plateforme propose des outils d'annotation intégrés, ce qui facilite le marquage des objets ou des zones d'intérêt sur les images.

- **Préparation des données** : Roboflow propose plusieurs fonctionnalités comme la data augmentation ou des outils de prétraitement qui permettent de normaliser, redimensionner ou ajuster les images pour qu'elles s'adaptent aux exigences du modèle. Il prend egalement en charge la conversion des données dans différents formats compatibles avec des frameworks d'apprentissage profond comme TensorFlow, PyTorch ou YOLO.

- **Versioning des datasets** : Chaque modification apportée à votre dataset est tracée, ce qui permet de gérer facilement plusieurs versions d'un dataset. Cela facilite la comparaison des performances des modèles sur différentes versions des données.

- **Intégration facile avec les frameworks** : Roboflow simplifie l'intégration avec les frameworks populaires d'apprentissage automatique. Vous pouvez facilement exporter des datasets pour des modèles de vision par ordinateur comme YOLO, Detectron2, ou encore TensorFlow Object Detection API.

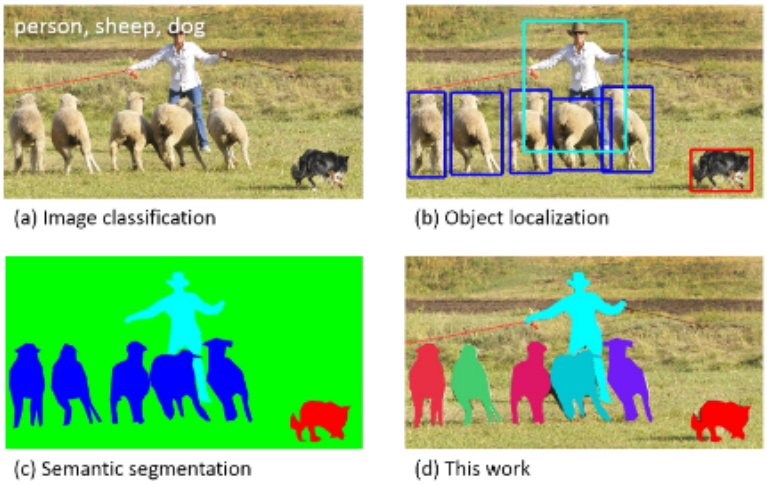
Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

3.2.3 Datatsets importants

COCO (Microsoft Common Objects in COntex)

COCO est conçu pour promouvoir la recherche dans plusieurs domaines de la vision par ordinateur, tels que la détection d'objets, la segmentation d'instances et la compréhension des scènes.

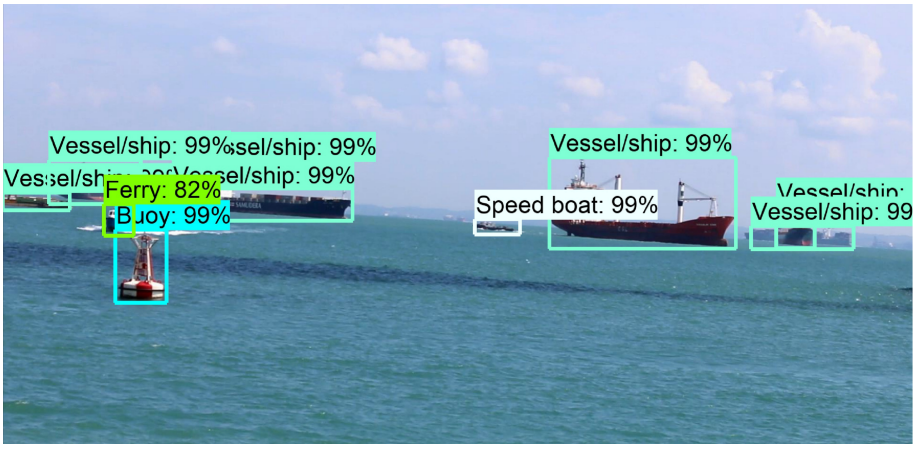
Il contient plus de 330 000 images, avec plus de 2,5 millions d'annotations d'instances. L'ensemble de données comprend 80 catégories d'objets courants, tels que des personnes, des voitures, des animaux, des objets du quotidien, etc.
Il est surtout utilisé pour l'entraînement et l'évaluation de modèles de vision par ordinateur comme YOLO ou R-CNN...



SMD (Singapore Maritime Dataset) :

Le SMD est le dataset le plus utilisé dans le domaine de la classification d'objet en mer, il a été développé par l'Université Nationale de Singapour.
Il est conçu principalement pour les applications de surveillance maritime, en particulier pour la détection et le suivi de navires dans des environnements maritimes.

Il comprend des vidéos enregistrées à partir de caméras placées sur des navires ou des structures maritimes. Les vidéos couvrent diverses conditions environnementales, y compris différentes conditions météorologiques et de luminosité.
Les annotations incluent des informations sur les navires présents dans les vidéos, comme leur localisation, leur taille, et leur type.

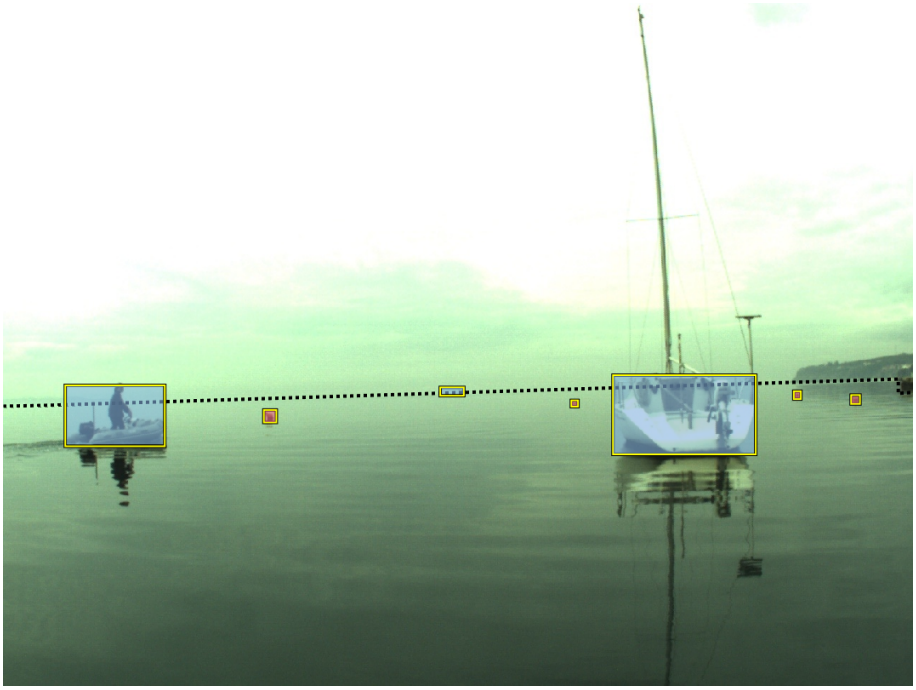


Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

MODD (Maritime Obstacle Detection Dataset) :

Filmé à bord d'un USV (Unmanned Surface Vehicle) sur une période de 15 mois en Pologne. Il a été développé pour les véhicules autonomes de surface (USV), les drones maritimes et à d'autres applications de navigation autonome en mer.
Il segmente chaque image en 3 régions : ciel, rive et mer et concentre sa détection seulement dans la région mer.

Il comprend 28 séquences vidéos capturées sur une période de 15 mois et totalise 11 675 images stéréo avec une résolution de 1 278 x 958 pixels chacune. Les vidéos ont été prises durant diverses conditions météorologiques (ensoleillé, couvert, brumeux) et avec plusieurs petits obstacles.
Il y a aussi 3 cas de situations extrêmes dans le dataset qui affectent les algos de computer vision : mouvement brusque, éblouissement et reflet.



Pour ce projet, ces deux datasets ont servi de base pour orienter les recherches en vue d'identifier le dataset le plus adapté. Bien qu'ils soient intéressants et complets, ils n'ont pas permis au modèle d'atteindre des performances satisfaisantes.
Il est important de noter que, selon les cas, il peut être difficile de trouver un dataset parfaitement adapté aux besoins spécifiques d'un projet. L'idéal, bien que laborieux, reste souvent de créer son propre dataset afin de répondre précisément aux exigences du modèle.
Finalement, les datasets sélectionnés sont des dérivés de SMD et MODD, trouvés sur Roboflow.

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

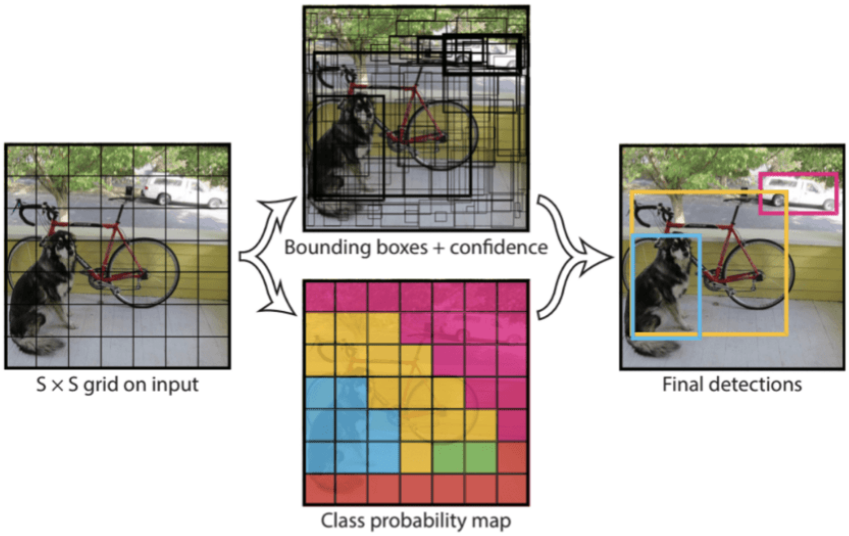
3.3 Méthodes de détection d'objets

3.3.1 Détection "Single Shot" : YOLO (You Only Look Once)

YOLO (You Only Look Once) est un algorithme de détection d'objets en temps réel, initialement développé par Joseph Redmon et Ali Farhadi en 2015. Conçu comme un détecteur d'objets à une seule étape, YOLO utilise un réseau de neurones convolutifs (CNN) pour prédire les boîtes de délimitation et les probabilités de classe des objets présents dans les images d'entrée. Contrairement aux détecteurs d'objets en deux étapes tels que R-CNN, YOLO traite l'image entière en une seule fois, ce qui le rend plus rapide et plus efficace.

L'algorithme YOLO divise l'image d'entrée en une grille de cellules, chaque cellule prédisant la probabilité de présence d'un objet, les coordonnées de la boîte de délimitation et la classe de l'objet. Depuis sa première version, YOLO a évolué avec plusieurs itérations (YOLOv1 à YOLOv9), chaque nouvelle version apportant des améliorations en termes de précision, de vitesse de traitement et de gestion des petits objets.

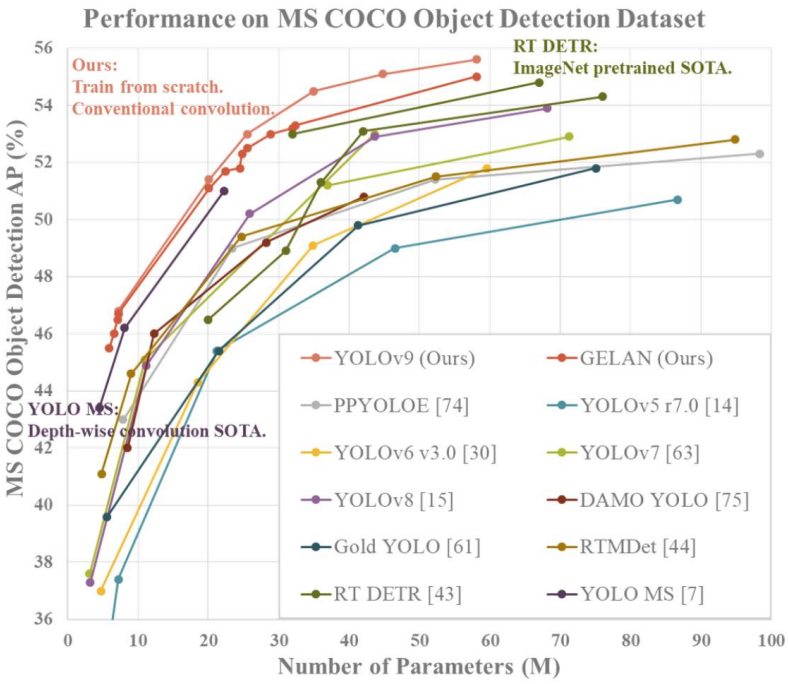
YOLOv9 a démontré des performances supérieures sur des ensembles de données comme MS COCO, surpassant les modèles pré-entraînés existants en termes de précision et de vitesse. Son architecture légère et efficace en fait une solution idéale pour des tâches de détection d'objets en temps réel sur divers dispositifs, des systèmes embarqués aux serveurs puissants.



Yolov9 :

YOLOv9, la dernière version de la série YOLO, se distingue par son approche innovante visant à surmonter les défis inhérents à la perte d'informations dans les réseaux neuronaux profonds. En intégrant l'Information de Gradient Programmable (PGI) et le Réseau d'Agglomération de Couches Efficace Généralisé (GELAN), YOLOv9 améliore non seulement la capacité d'apprentissage du modèle mais assure également la rétention des informations cruciales, garantissant ainsi des performances et une précision exceptionnelles.

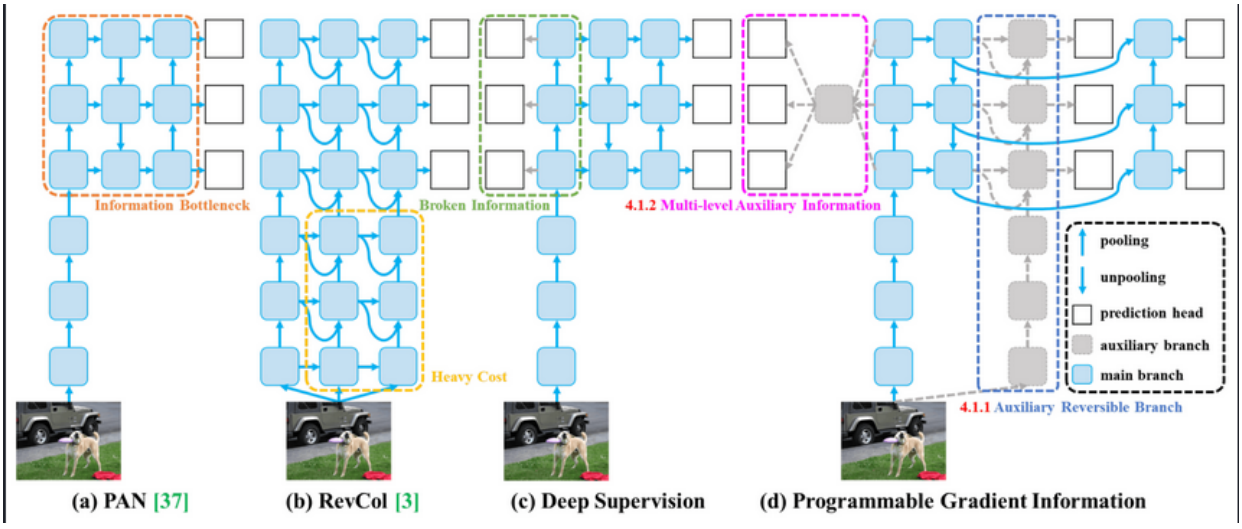
Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.



Information de Gradient Programmable (PGI) :

Le principe du goulot d'étranglement de l'information dans les réseaux profonds présente un défi majeur : la perte d'informations importantes au fur et à mesure que les données traversent les couches successives du réseau.

YOLOv9 aborde ce problème en implémentant PGI, qui aide à préserver les données essentielles et assure une génération de gradient plus fiable. Le gradient indique comment ajuster les paramètres du modèle pour minimiser la fonction de coût. Cette fonction mesure l'écart entre les prédictions du modèle et les valeurs réelles. Minimiser cette fonction de coût permet d'optimiser les performances du modèle en réduisant cet écart. Le processus de minimisation de la fonction de coût utilise des algorithmes comme la descente de gradient, où le gradient indique la direction et l'ampleur des ajustements nécessaires aux poids du modèle.
Grâce à une génération de gradient plus fiable, comme celle assurée par PGI dans YOLOv9, le modèle peut converger plus efficacement vers des solutions optimales, assurant des prédictions plus précises et une meilleure généralisation aux nouvelles données.



Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

Voici les différentes architectures et méthodes de réseaux neuronaux profonds, en particulier celles liées à l'Information de Gradient Programmable (PGI) et les architectures réseau associées.

- **PAN (Path Aggregation Network) :**
  Explications : Dans un réseau de ce type, l'image est transformée couche par couche, comme si elle passait à travers une série de filtres pour identifier des caractéristiques comme les contours, les textures, etc.
  Problème : En passant par ces filtres, l'information sur l'image peut se perdre ou devenir moins précise (c'est ce qu'on appelle un goulot d'étranglement de l'information ou "Information Bottleneck").

- **RevCol (Reversible Columns) :**
  Explications : Ici, chaque transformation ou filtre appliqué à l'image peut être inversé pour revenir à l'état précédent.
  Problème : Bien que cette méthode aide à conserver les détails de l'image, elle demande beaucoup de calculs et de ressources, ce qui peut être coûteux (Heavy Cost).

- **Deep Supervision :**
  Explication : Cette méthode ajoute des contrôles supplémentaires à différentes étapes du processus pour s'assurer que chaque étape conserve les détails importants de l'image.
  Problème : Même avec ces contrôles supplémentaires, des informations peuvent encore être perdues ou mal gérées, surtout si ces contrôles ne sont pas bien coordonnés (Broken Information).

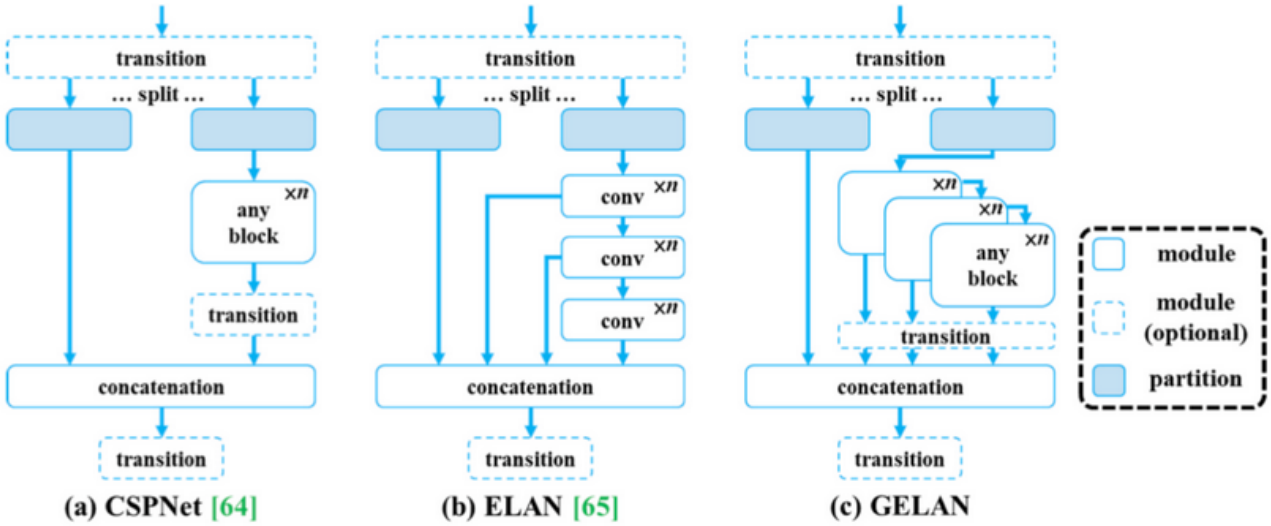
Le PGI est en fait une combinaison des meilleures idées des autres méthodes et les améliore :

- Branche principale (Main Branch) : C'est la voie principale que suit l'image, utilisée pour la reconnaissance (prédiction).
- Branche auxiliaire réversible (Auxiliary Reversible Branch) : Si à une étape des informations importantes sont perdues, il est possible grâce à cette branche de revenir en arrière et corriger l'erreur sans perdre les détails importants de l'image initiale.
- Information auxiliaire à plusieurs niveaux (Multi-level Auxiliary Information) : Cela ajoute plusieurs niveaux d'informations supplémentaires pour s'assurer que chaque partie du réseau apprend de manière planifiée et efficace.

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

Réseau d'Agglomération de Couches Efficace Généralisé (GELAN) :

GELAN permet une utilisation optimale des paramètres et une efficacité de calcul supérieure. Sa conception modulaire permet une intégration flexible de divers blocs de calcul, rendant YOLOv9 adaptable à un large éventail d'applications sans sacrifier la vitesse ou la précision. Applications et Performances de YOLOv9.



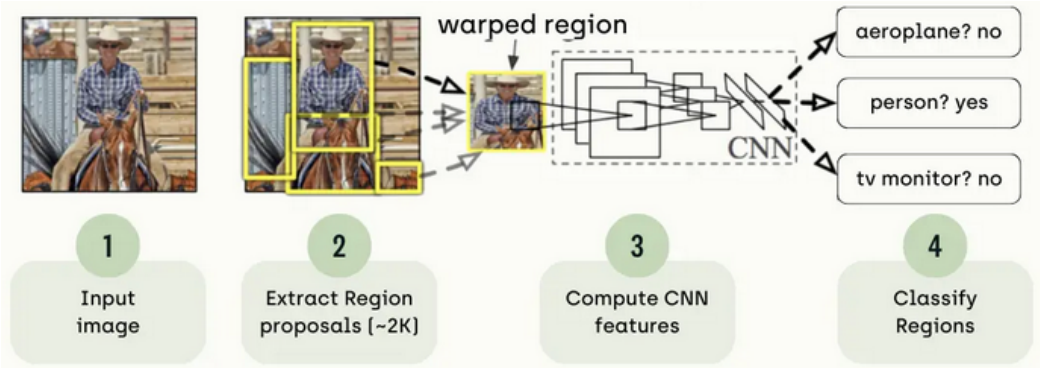
- **CSPNet (Cross Stage Partial Network) :**
  CSPNet vise à réduire les calculs en ne traitant qu'une partie des données, tout en combinant les informations traitées et non traitées pour conserver des détails importants.
  - Transition et Split : L'image est "divisée" en deux parties : Une partie passe par un bloc de traitement (où elle subit des transformations spécifiques), l'autre partie reste inchangée.
  - Concaténation : Après le traitement, les deux parties sont recombinées. Cela permet de conserver certaines informations originales tout en appliquant des transformations sur une partie des données.
- **ELAN (Enhanced Linear Aggregation Network) :**
  ELAN utilise plusieurs convolutions parallèles pour extraire plus d'informations, améliorant ainsi la richesse des caractéristiques extraites.
  - Transition et Split : Comme CSPNet, l'image est d'abord divisée.
  - Blocs de Convolution : La partie divisée passe par plusieurs blocs de convolution en parallèle. Chaque bloc extrait différentes caractéristiques de l'image.
  - Concaténation : Les résultats des différents blocs de convolution sont ensuite combinés ensemble pour former une image enrichie en détails.
- **GELAN (Generalized Efficient Layer Aggregation Network) :**
  GELAN combine les idées de CSPNet et ELAN pour optimiser l'utilisation des paramètres et des ressources de calcul. Il permet une flexibilité dans le choix des blocs de traitement et des convolutions, offrant ainsi une architecture adaptable et efficace.
  - Transition et Split : L'image est divisée en plusieurs parties.
  - Blocs de Convolution et de Traitement : Une partie passe par un ou plusieurs blocs de convolution et des blocs de traitement. Ces blocs peuvent être différents types de transformations qui extraient diverses caractéristiques de l'image.
  - Concaténation : Les différentes parties, après traitement, sont recombinées pour former une image finale plus détaillée.

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

3.3.2 Detection "Double Shot" : R-CNN (Regional Convolutional Neural Network)

Definition :
R-CNN a été introduit par Ross Girshick en 2014. Il s'agit d'une méthode de détection d'objets qui consiste à utiliser un réseau neuronal convolutif (CNN) pour classer les propositions d'objets, ou régions d'intérêt (ROI), au sein d'une image. Il est aussi connu sous le nom "détection d'objet à deux coups".

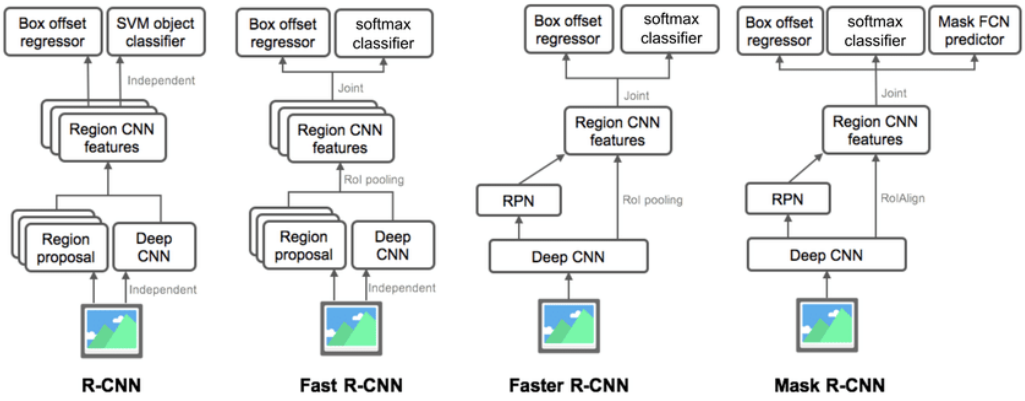
Fonctionnement :
L'idée est de générer d'abord un ensemble de régions candidates dans l'image, puis d'utiliser un CNN pour classer chaque région comme contenant un objet ou non. Par la suite les régions définit comme contenant un objet sont traitées pour affiner l'emplacement de l'objet et par la suite mieux le classer.



Limites :

- Complexité : R-CNN est intensif en calcul. Il s'agit d'extraire des propositions de région, d'appliquer un CNN à chaque proposition, puis d'exécuter les fonctionnalités extraites par l'intermédiaire d'un classificateur.
- Lenteur : En raison de son traitement séquentiel , il est relativement lent pendant l'inférence, ce qui ne convient pas aux applications en temps réel.
- Chevauchement : Il peut générer plusieurs propositions de régions qui se chevauchent de manière significative.

Versions : Le R-CNN et ses variantes (R-CNN rapide, Faster R-CNN et Mask R-CNN), sont des exemples célèbres de CNN à base de région. Ces méthodes utilisent une combinaison de méthodes de recherche sélective pour générer des régions candidates pour classer et affiner les régions qui contiennent des objets.



Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

3.4 Interpretation des metrics et hyperparametres

3.4.1 Interpretation metrics

Après l'entraînement, YOLOv9 génère des graphiques et des métriques pour évaluer les performances du modèle.
On les retrouve dans le répertoire runs/train/expX (indiqué à la fin de l'execution).

Graphiques d'entraînement :

- **train/boxloss :** Est ce que le modèle détecte bien les objets et place bien les box ?
  Grande valeur = grande erreur = mauvaise précision de localisation.
  Petite valeur = petite erreur = bonne précision de localisation.

- **train/clsloss :** Est ce que le modèle classe les objets dans les bonnes catégories ?
  Grande valeur = grande erreur = mauvaise précision de classification.
  Petite valeur = petite erreur = bonne précision de classification.

- **train/dflloss :** Est ce que le modèle place bien les bords des boîtes englobantes ?
  Grande valeur = grande erreur = mauvaise précision.
  Petite valeur = petite erreur = bonne précision.

- **metrics/precision :** Pourcentage de prédictions correctes parmi toutes les prédictions faites.
  Une précision plus élevée signifie que le modèle fait moins de fausses détections.

- **metrics/recall :** Pourcentage d'objets correctement détectés parmi tous les objets présents.
  Un rappel plus élevé signifie que le modèle manque moins d'objets.

Graphiques de validation :

- **val/boxloss :** Identique à train/boxloss mais avec les données de validation.

- **val/clsloss :** Identique à train/clsloss mais avec les données de validation.

- **val/dflloss :** Identique à train/dflloss mais avec les données de validation.

- **metrics/mAP0.5 :** Ce graphique montre la moyenne des précisions (mAP) à un seuil IoU (Intersection over Union) de 0.5 pendant la validation.
  Un mAP plus élevé à ce seuil signifie que le modèle détecte correctement les objets avec une tolérance de localisation de 50%.

- **metrics/mAP0.5 :0.95 :** Ce graphique montre la moyenne des précisions (mAP) sur une gamme de seuils IoU de 0.5 à 0.95 pendant la validation.
  Un mAP plus élevé sur cette gamme indique que le modèle est performant sur une variété de tolérances de localisation, de 50% à 95%.

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

3.4.2 mAP / IoU

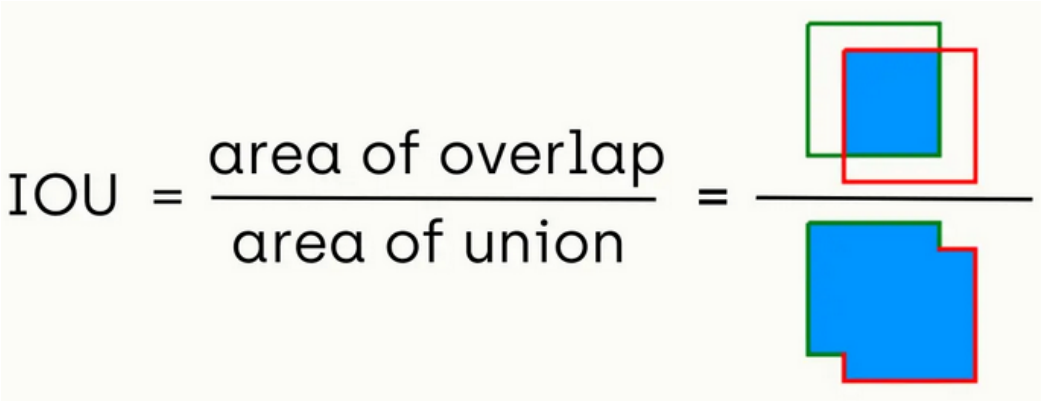
mAP, ou mean Average Precision, est une métrique couramment utilisée pour évaluer la performance des modèles de détection d'objets. Elle prend en compte la précision et le rappel des prédictions à différents seuils de similarité entre les boîtes de détection prédites et les boîtes de détection réelles (appelées Intersection over Union, ou IoU).

L'IoU est une mesure du chevauchement entre la zone de délimitation prédite et la zone de délimitation de la carbure de la vérité. Il est calculé en prenant les zones d'intersection des deux cases et en la divisant par la zone de leur union. Une valeur d'IoU plus élevée indique une meilleure correspondance entre les cases de délimitation prédites et les cases de délimitation de la vérité au sol.

Pour un seuil de IoU donné, l'Average Precision (AP) est calculée comme l'aire sous la courbe Précision-Rappel. Elle représente la précision moyenne obtenue pour différents niveaux de rappel :

- **mAP0.5 :** moyenne des APs pour toutes les classes, avec un seuil IoU fixe de 0.5. Cela signifie que pour qu'une détection soit considérée comme correcte, l'IoU entre la boîte prédite et la boîte réelle doit être au moins 0.5. On calcule l'AP pour chaque classe à IoU=0.5 et on prend la moyenne de ces APs pour obtenir le mAP0.5.

- **mAP0.5 :0.95 :** moyenne des mAPs calculés à différents seuils IoU, allant de 0.5 à 0.95 avec un pas de 0.05. Concrètement, on calcule l'AP pour chaque classe à des seuils IoU de 0.5, 0.55, 0.6, ..., jusqu'à 0.95, puis on prend la moyenne de ces APs. Cela donne une mesure plus robuste de la performance du modèle, car il évalue la précision et le rappel à différents niveaux de chevauchement entre les boîtes prédite et réelle.



Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

3.4.3 Hyperparametres

Les hyperparamètres sont des variables de configuration externes qu'on utilise pour gérer l'entraînement des modèles de machine learning.

Il n'y a pas de méthode unique et universelle, mais plusieurs approches peuvent être utilisées pour trouver des hyperparamètres optimaux :

- **Recherche manuelle :** Méthode la plus simple consiste à ajuster manuellement les hyperparamètres et à observer les résultats.
  Cette méthode peut être chronophage et ne garantit pas de trouver les meilleurs hyperparamètres, mais elle permet de comprendre comment chaque hyperparamètre affecte le modèle.

- **Recherche en grille (Grid Search) :** Ça consiste à définir un ensemble de valeurs pour chaque hyperparamètre et à entraîner le modèle avec toutes les combinaisons possibles de ces valeurs.
  Bien que systématique, la recherche en grille peut être très coûteuse en termes de temps et de ressources, surtout s'il y a beaucoup d'hyperparamètres.

- **Recherche aléatoire (Random Search) :** Au lieu de tester toutes les combinaisons possibles, la recherche aléatoire choisit de manière aléatoire les combinaisons d'hyperparamètres à tester.
  Cette méthode est souvent plus efficace que la recherche en grille car elle explore plus de configurations en moins de temps.

- **Optimisation bayésienne :** L'optimisation bayésienne est une technique plus avancée qui construit un modèle probabiliste de la fonction de perte et choisit les hyperparamètres de manière à maximiser une fonction acquisition.
  Cette méthode est plus efficace que la recherche en grille ou aléatoire, surtout pour les problèmes où l'évaluation de chaque ensemble d'hyperparamètres est coûteuse.

- **Hyperparameter Evolution (Évolution des hyperparamètres) :** YOLOv9 inclut une méthode d'évolution des hyperparamètres qui utilise des algorithmes évolutionnaires pour optimiser les hyperparamètres.
  Cette méthode commence par une population de configurations d'hyperparamètres et itère pour sélectionner et muter les configurations les plus performantes.

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

3.5 Implémentation du modèle

L'intégralité de la partie pratique du projet a été documentée progressivement sur un dépôt Git. Ce dernier contient toutes les explications détaillées, depuis l'installation de YOLOv9 jusqu'au choix des datasets et à l'évolution de la stratégie d'entraînement.
On y retrouve également les différentes métriques issues des entraînements effectués, ainsi que les scripts utilisés pour ces entraînements, offrant ainsi une vue complète et structurée de l'ensemble du processus.
Il a également été réaliser un rendu sur Streamlit affin de pouvoir présenter correctement mon travail.

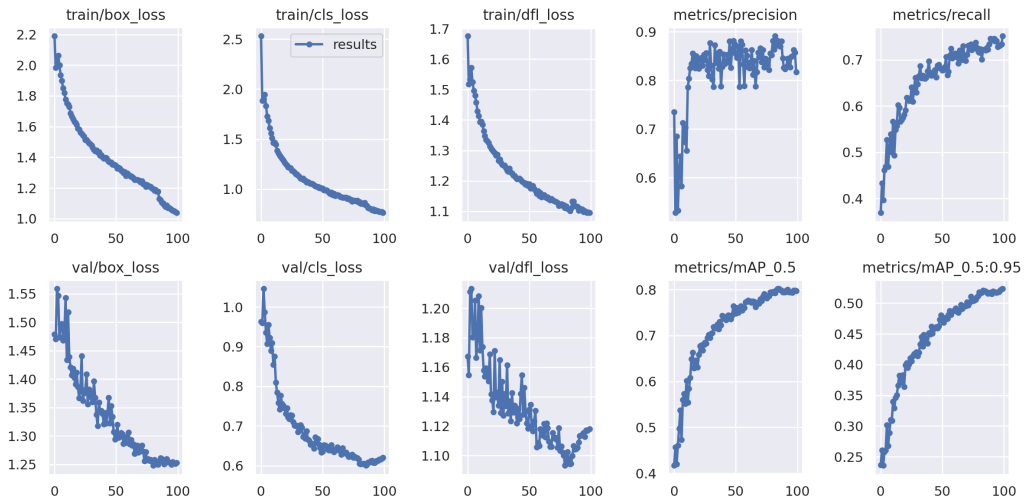


FIGURE 1 – Exemple de résultats d'un entraînement sur 100 epochs

FIGURE 2 – Interface Streamlit du projet

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

4 Management

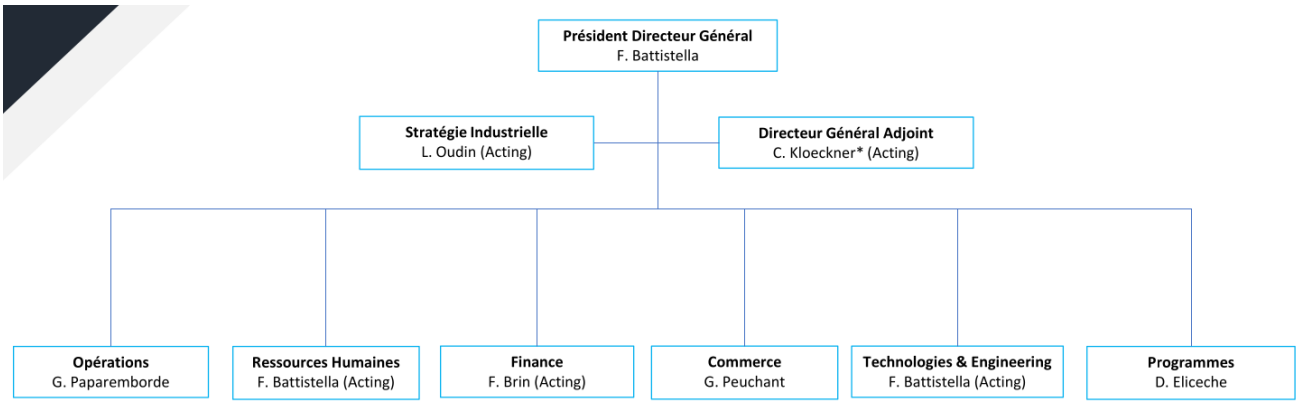


FIGURE 3 – Organigramme général

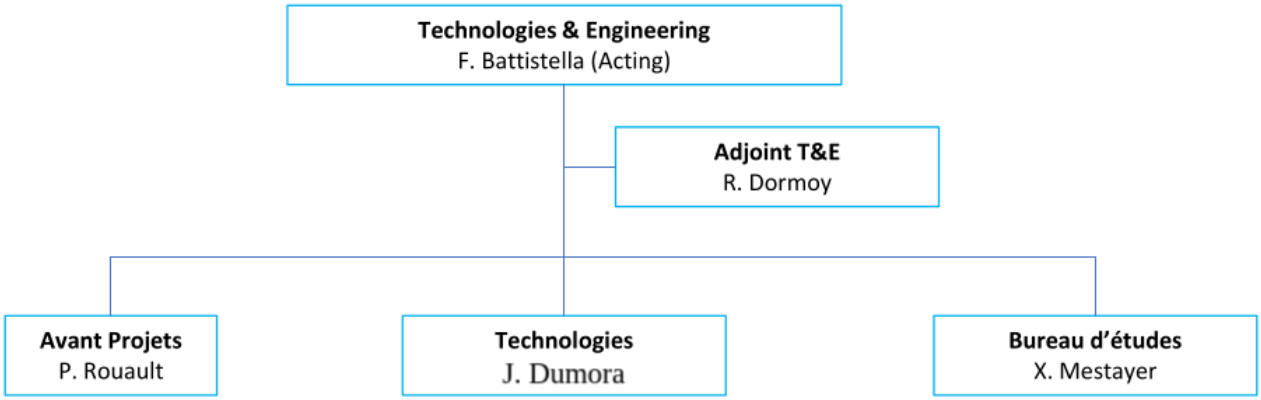


FIGURE 4 – Organigramme direction R&D

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

5 Développement Durable

Le chantier met en place plusieurs actions en lien avec le développement durable et la responsabilité sociétale des entreprises (RSE). Couach se concentre sur l'utilisation de matériaux composites innovants et légers, qui permettent de construire des navires plus efficaces en termes de consommation d'énergie. Ce choix contribue à la réduction des émissions de CO2 et des coûts opérationnels.

Le chantier mise également sur l'artisanat local avec une équipe de plus de 250 artisans spécialisés dans différents métiers liés à la construction navale, comme la menuiserie, la soudure et la mécanique. Cette approche favorise un savoir-faire local et durable tout en assurant des emplois dans la région.

De plus, Couach s'investit dans des projets collaboratifs avec des entreprises technologiques pour développer des solutions innovantes, comme des véhicules de surface non habités, réduisant ainsi l'impact environnemental tout en répondant aux besoins du marché naval moderne.

Ces initiatives démontrent une volonté de Couach de s'inscrire dans une démarche de développement durable, en optimisant ses méthodes de production tout en respectant les normes environnementales et sociales.

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

6 Conclusion

Ce projet a permis de réaliser l'ensemble des objectifs fixés, à savoir l'exploration et l'application des méthodes avancées de détection d'objets en utilisant le modèle YOLOv9 avec des datasets maritimes spécifiques tels que SMD et MODD. L'intégration des techniques de transfert d'apprentissage ainsi que la mise en œuvre d'entraînements personnalisés, avec et sans gel de couches, ont permis de répondre aux exigences de classification fine et de détection précise des objets en mer.

Cette expérience a été particulièrement enrichissante car elle m'a offert l'occasion de travailler sur des technologies de pointe en vision par ordinateur. J'ai pu approfondir mes connaissances en détection d'objets et en traitement de données visuelles tout en participant activement au développement de solutions concrètes pour des applications maritimes.

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.

7 Annexes

7.1 Liens importants

Github du projet : https://github.com/JeremDD/Classification_Objets_Marins_IA.git

Github Yolov9 : https://github.com/WongKinYiu/yolov9

Roboflow Universe : https://universe.roboflow.com/

7.2 Liens recherches bibliographique

SMD (Singapore Maritime Dataset) : https://www.kaggle.com/datasets/mmichelli/singapore-marit
https://www.mdpi.com/2077-1312/10/3/377
Obstacle Detection in a Marine Environment : https://box.vicos.si/borja/viamaro/index.
html
Articles : https://www.researchgate.net/publication/351369117_MODS_--_A_USV-oriented_
object_detection_and_obstacle_segmentation_benchmark
Jaccard index : https://en.wikipedia.org/wiki/Jaccard_index
Semantic seg vs object detection : https://keymakr.com/blog/semantic-segmentation-vs-object-det
Mamba Yolo (better 2024 yolo ?) : https://arxiv.org/pdf/2406.05835
intro + YOLO : https://kili-technology.com/data-labeling/machine-learning/yolo-algorithm-
https://medium.com/nerd-for-tech/from-yolo-to-yolov8-tracing-the-evolution-of-object-de
anchor box : https://datadrivenscience.com/object-detection-with-yolov5-understanding-ancho
Data augmentation : https://aws.amazon.com/fr/what-is/data-augmentation/
implementation yolov9 : https://blog.roboflow.com/train-yolov9-model/#how-to-train-a-yolov9-
metrics tips : https://docs.ultralytics.com/guides/yolo-performance-metrics/
https://github.com/ultralytics/ultralytics/issues/9446

Ce document et les informations qu'il contient ne peuvent être reproduits, communiqués ou utilisés sans son autorisation écrite préalable.