Application de détéction et de tracking d'objet

1st Picard Hugo
Intelligence artificcielle embarquée
ESIEE – 2025
France
hugo.picard@edu.esiee.fr

Résumé - Ce TP consiste à développer un système de détection d'objets basé sur l'apprentissage profond pour détecter une balle de tennis. Le travail repose sur un dataset personnel, constitué à partir d'images que j'ai moi-même acquises. Afin d'augmenter la quantité de données disponibles, ce jeu d'images a été enrichi par des techniques d'augmentation de données avant d'être préparé sur la plateforme Roboflow. Un modèle YOLOv8 a ensuite été entraîné pour détecter et localiser la balle de tennis dans des images et des séquences vidéo. Les paramètres d'apprentissage ont été choisis de manière à obtenir un bon compromis entre performances et temps de calcul. Les performances du modèle ont été évaluées à l'aide de métriques classiques de détection d'objets, puis le modèle entraîné a été appliqué à une vidéo afin de visualiser la détection et la position de la balle dans l'image. Les résultats montrent que le modèle apprend rapidement et permet d'obtenir des détections fiables dans le cadre de ce projet.

I – Objectifs

L'objectif de cette application est de détecter automatiquement un objet dans une image ou une vidéo à l'aide d'un modèle de deep learning (que l'on va présenter par la suite). À partir du flux vidéo, le système doit être capable d'identifier la présence de l'objet et d'en déterminer la position dans l'image.

L'application est conçue pour fonctionner dans des conditions proches du réel, donc avec des variations possibles d'orientation, de luminosité ou de position de l'objet. Le but n'est donc pas uniquement de reconnaître l'objet, mais aussi de le localiser de manière fiable toujours pour une application dans le réel.

En sortie, le modèle fournit une boîte englobante autour de l'objet détecté, ainsi qu'un score de confiance. La position de l'objet est ensuite exploitée pour calculer son décalage par rapport au centre de l'image, ce qui permet d'évaluer sa position relative dans le champ de la caméra.

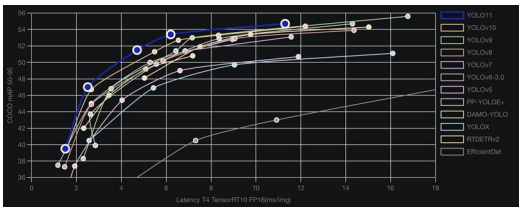
Cette application est une première étape vers des systèmes qu'on souhaite réaliser : Le tracking de la balle de tennis par le robot Turtlebot3 par exemple.

II – Choix du modèle

Pour ce projet, j'ai utilisé le modèle YOLOv8, proposé par Ultralytics. C'est un modèle de détection d'objets qui permet non seulement de reconnaître un objet dans une image, mais aussi de savoir précisément où il se trouve. Concrètement, le modèle renvoie une boîte autour de l'objet détecté ainsi qu'un score de confiance.

YOLOv8 fonctionne en traitant l'image en une seule fois. Cela permet d'obtenir des résultats rapides, ce qui est particulièrement intéressant lorsque l'on travaille sur des images issues d'une vidéo. La documentation Ultralytics met en avant ce compromis entre rapidité et précision, ainsi que la simplicité de mise en œuvre du modèle, notamment pour l'entraînement et l'évaluation des performances.

Un autre point important est que YOLOv8 existe en plusieurs versions, plus ou moins légères. Cela permet d'adapter le modèle aux ressources de calcul disponibles et au niveau de précision recherché. La page officielle d'Ultralytics propose d'ailleurs des comparaisons de performances entre ces différentes versions, ce qui facilite le choix du modèle en fonction du contexte d'utilisation.



[1] https://docs.ultralytics.com/fr/models/yolov8/#can-i-benchmark-yolov8-models-for performance

Ce modèle est bien adapté à ce projet, car l'objectif n'est pas uniquement de détecter un objet, mais aussi d'utiliser sa position dans l'image. En revanche, comme pour tout modèle de détection, les performances dépendent fortement de la qualité des données utilisées pour l'entraînement. Un dataset trop réduit ou mal annoté peut rapidement limiter les résultats (j'y reviendrais dans une autre partie). De plus, certains paramètres comme la taille des images ou le nombre d'époques doivent être ajustés pour obtenir un comportement satisfaisant.

Dans l'ensemble, YOLOv8 constitue un bon compromis pour ce type d'application, en offrant une solution efficace, bien documentée et relativement simple à prendre en main.

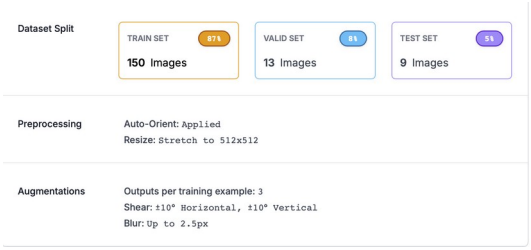
III – Réalisation du Dataset

Le travail a commencé par la constitution du dataset. À l'origine, je disposais de 24 images. Pour augmenter ce nombre, j'ai utilisé un programme externe qui m'a permis de générer de nouvelles images à partir des images initiales, notamment en appliquant des effets miroir et des rotations à 90°. Cette première étape m'a permis d'obtenir un jeu de données de 72 images, qui a ensuite été importé dans Roboflow pour la gestion des annotations et la préparation de l'entraînement.

Une fois le dataset importé dans Roboflow, les données ont été automatiquement réparties entre les différents ensembles. La majorité des images a été utilisée pour l'entraînement, avec environ 87 % du dataset, soit 150 images (après augmentation). Une petite partie a été réservée à la validation, avec 13 images, afin de suivre l'évolution des performances du modèle pendant l'apprentissage. Enfin, 9 images ont été conservées pour le test, ce qui permet d'évaluer le comportement du modèle sur des données qu'il n'a jamais vues.

Avant l'entraînement, un prétraitement a été appliqué aux images. L'orientation automatique a été activée afin de corriger d'éventuelles rotations incorrectes, et toutes les images ont été redimensionnées à une taille de 512 × 512 pixels. Ce redimensionnement permet d'avoir des images homogènes en entrée du modèle et de respecter le format attendu par YOLOv8.

Des augmentations ont également été ajoutées lors de la génération du dataset. Pour chaque image d'entraînement, plusieurs variantes sont créées afin d'augmenter la diversité des données. Ces augmentations sont des cisaillements horizontaux et verticaux ainsi qu'un flou modéré. L'objectif est de rendre le modèle plus robuste face aux variations de position, d'orientation et de qualité d'image, sans dénaturer le contenu des images Originales. Cela va rendre le modèle plus robuste face aux différentes situations qu'il peut être amené à être confronté sur un utilisation dans des conditions réelles.



Vous pouvez vous rendre sur roboflow pour examiner le dataset et essayer le détection d'une balle de tennis avec ce dataset en entrant cet ID.

ID model : dataset2-rtkgf/3

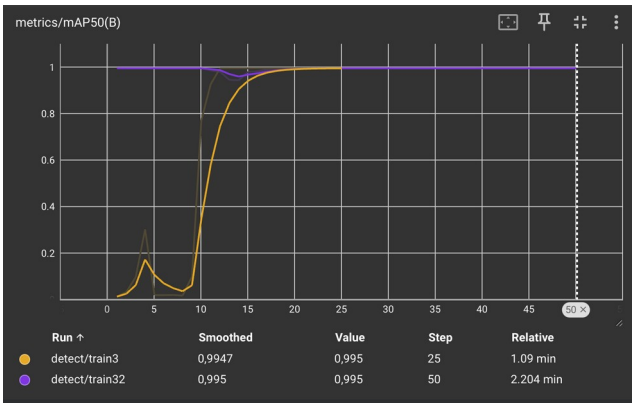
IV – Entrainement du modèle

L'apprentissage du modèle a ensuite été réalisé avec YOLOv8. Le modèle a été entraîné sur 50 époques, ce qui permet de laisser suffisamment de temps au réseau pour apprendre sans risquer un surapprentissage trop important vu la taille du dataset. Le choix de la taille d'image à 224 × 224 pixels correspond à un compromis entre rapidité d'entraînement et conservation de l'information visuelle.

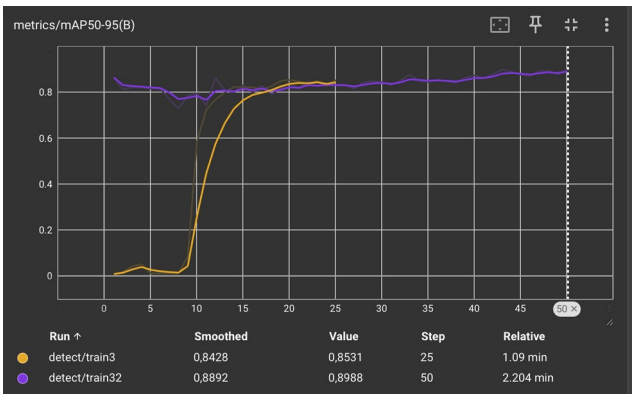
Pour évaluer les performances du modèle, plusieurs métriques ont été utilisées. La loss qui permet de suivre la convergence du modèle au cours de l'apprentissage. Les métriques de précision et de rappel qui donnent une indication sur la qualité des détections, tandis que la mAP permet d'évaluer plus globalement la performance du modèle en tenant compte à la fois de la localisation et de la classification des objets.

Les courbes présentées ci dessous montrent l'évolution des performances du modèle au cours de l'apprentissage, mesurées à l'aide des métriques mAP50 et mAP50-95. La courbe orange correspond à un entraînement arrêté à 25 époques, tandis que la courbe violette correspond à un entraînement mené jusqu'à 50 époques.

On observe d'abord que, pour les deux métriques, les performances augmentent très rapidement durant les premières époques. Cela signifie que le modèle apprend vite à détecter l'objet et à en estimer correctement la position. Autour d'une dizaine d'époques, la progression devient plus lente, ce qui indique que le modèle commence à converger.



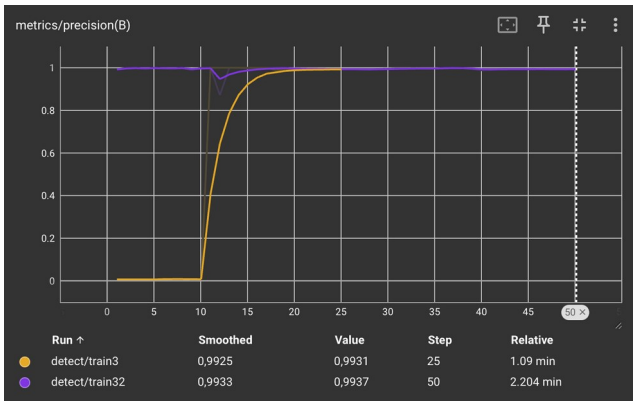
Pour la métrique mAP50, les deux entraînements atteignent une valeur très élevée, proche de 1. À 25 époques, la performance est déjà quasiment maximale, et l'entraînement jusqu'à 50 époques n'apporte qu'un gain très marginal. Cela montre que, pour un seuil de localisation relativement permissif, le modèle est déjà très performant assez tôt.



En revanche, la métrique mAP50-95, qui est plus exigeante car elle évalue la précision de la localisation sur plusieurs seuils, continue de progresser après 25 époques. On observe que la courbe violette atteint une valeur plus élevée que la courbe orange. Cela signifie que l'entraînement plus long permet d'améliorer la précision des boîtes englobantes, même si le gain reste modéré.

Ces résultats montrent qu'un entraînement de 25 époques est suffisant pour obtenir de très bonnes performances globales, mais qu'un entraînement plus long, jusqu'à 50 époques, permet d'affiner la localisation des objets. Le choix du nombre d'époques dépend donc du compromis recherché entre temps d'entraînement et précision finale.

La figure ci dessous montre l'évolution de la précision du modèle au cours de l'apprentissage.



On observe que la précision est très faible au début de l'apprentissage, ce qui est normal puisque le modèle n'a encore rien appris. À partir d'une dizaine d'époques, la précision augmente très rapidement, signe que le modèle commence à détecter correctement l'objet et à réduire fortement le nombre de faux positifs.

Autour de 15 à 20 époques, la précision atteint déjà une valeur très élevée, proche de 1. À partir de ce moment-là, la courbe se stabilise, ce qui indique que le modèle a quasiment atteint son niveau de performance maximal sur cette métrique. L'entraînement jusqu'à 50 époques apporte seulement un gain très léger par rapport à 25 époques, avec une précision finale quasiment identique dans les deux cas.

Ces résultats montrent que le modèle devient très rapidement fiable dans ses détections et que la précision n'est pas le facteur limitant des performances globales. L'amélioration observée lors d'un entraînement plus long est marginale, ce qui confirme que le modèle converge rapidement sur ce dataset.

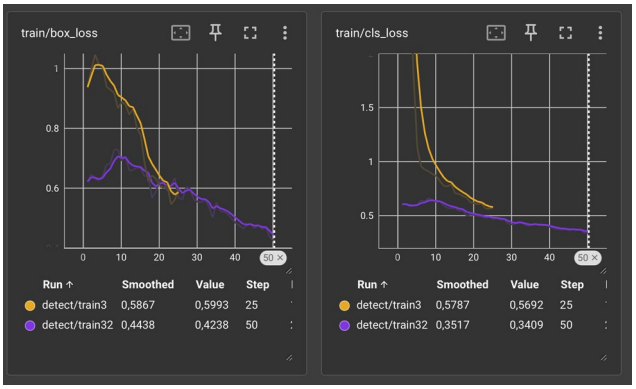
Cette figure montre l'évolution du rappel (recall) du modèle au cours de l'apprentissage.

époques montre que l'entraînement prolongé permet

On observe que la recall est déjà très élevée dès les premières époques, ce qui indique que le modèle détecte rapidement la majorité des objets présents dans les images. Quelques variations apparaissent au début de l'apprentissage, notamment autour d'une dizaine d'époques, mais elles restent limitées et s'expliquent par les ajustements progressifs du modèle lors de la phase de convergence.

À partir d'environ 20 à 25 époques, la recall atteint une valeur proche de 1 et se stabilise complètement. Cela signifie que le modèle parvient à détecter quasiment tous les objets présents dans les images, sans en oublier. L'entraînement jusqu'à 50 époques n'apporte pas d'amélioration significative sur cette métrique, puisque la valeur finale est identique dans les deux cas.

Ces résultats montrent que le rappel n'est pas un facteur limitant dans les performances du modèle. Le réseau apprend très rapidement à ne pas rater les objets, et un nombre d'époques plus élevé permet surtout d'affiner d'autres aspects, comme la précision de la localisation, plutôt que la capacité à détecter les objets eux-mêmes.

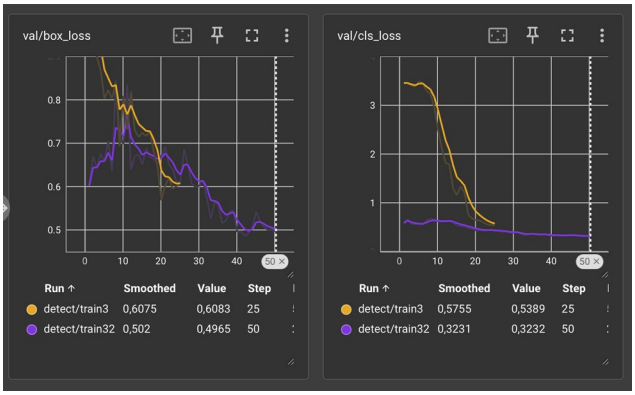


Ces deux figures montrent l'évolution des fonctions de coût utilisées lors de l'entraînement du modèle. La première courbe correspond à la box loss, qui mesure l'erreur sur la localisation des boîtes englobantes, tandis que la seconde correspond à la classification loss, qui évalue la qualité de la prédiction de la classe de l'objet.

Dans les deux cas, on observe une diminution progressive de la loss au fil des époques, ce qui indique que le modèle apprend correctement. Pour l'entraînement arrêté à 25 époques, la loss diminue rapidement au début, puis tend à se stabiliser, signe que le modèle a déjà acquis l'essentiel de l'information. Lorsque l'entraînement est poursuivi jusqu'à 50 époques, la loss continue de diminuer, mais de manière plus lente.

La baisse plus marquée de la box loss entre 25 et 50 surtout d'améliorer la précision de la localisation des objets. De la même manière, la diminution de la classification loss indique que le modèle affine progressivement sa capacité à attribuer la bonne classe, même si les gains deviennent plus limités avec le temps. Ces résultats sont cohérents avec les métriques précédemment observées. Les performances globales du modèle sont déjà élevées après 25 époques, mais un entraînement plus long permet de réduire davantage les erreurs, en particulier sur la précision des boîtes englobantes. Cela confirme que le modèle converge correctement et que l'amélioration apportée par un nombre d'époques plus élevé reste progressive.

Ces figures présentent l'évolution des fonctions de coût calculées sur le jeu de validation, pour la localisation des boîtes (val/box_loss) et pour la classification (val/cls_loss). La courbe orange correspond à un entraînement arrêté à 25 époques, tandis que la courbe violette correspond à un entraînement poursuivi jusqu'à 50 époques.



On observe que, dans les deux cas, les losss de validation diminuent progressivement au fil des époques. Cela indique que le modèle ne se contente pas de bien apprendre sur les données d'entraînement, mais qu'il généralise correctement sur des images qu'il n'a pas vues pendant l'apprentissage. Il n'y a pas de divergence entre les courbes d'entraînement et de validation, ce qui suggère l'absence de surapprentissage marqué.

Pour la box loss de validation, l'entraînement prolongé jusqu'à 50 époques permet d'obtenir une valeur plus faible que l'entraînement à 25 époques. Cela montre que le modèle continue à améliorer la précision de la localisation des boîtes englobantes sur des données nouvelles. De la même manière, la classification loss de validation diminue davantage lorsque l'entraînement est poursuivi, ce qui traduit une meilleure stabilité des prédictions de classe.

Ces résultats confirment que l'entraînement sur 50 époques apporte un léger gain en termes de

généralisation, même si les performances globales sont déjà élevées après 25 époques.

V – Conclusion

Ce TP m'a permis de mettre en pratique l'ensemble des étapes nécessaires à la réalisation d'un système de détection d'objets basé sur l'apprentissage profond. À partir d'un dataset personnel composé d'images que j'ai moi-même acquises, j'ai pu construire, enrichir et préparer les données avant d'entraîner un modèle YOLOv8 pour détecter une balle de tennis.

L'entraînement du modèle a montré une convergence rapide et des performances satisfaisantes, malgré un nombre limité d'images. L'analyse des métriques de précision, de rappel et de mAP a permis de mieux comprendre le comportement du modèle et l'impact des paramètres choisis. L'application du modèle sur une vidéo a également permis de valider concrètement le fonctionnement de la détection et de la localisation de l'objet.

Ce TP m'a permis de mieux comprendre le fonctionnement global d'un modèle de détection d'objets, depuis la constitution du dataset jusqu'à l'exploitation des résultats, et constitue une base pour des applications de vision plus avancées comme celle qui nous attend dans le projet.
