الجمهورية الجزائرية الديمقراطية الشعبية
People's Democratic Republic of Algeria
وزارة التعليم العالي و البحث العلمي
Ministry of Higher Education And Scientific Research
جامعة عبد الحميد بن باديس مستغانم
University Abdelhamid Ibn Badis Mostaganem

كلية العلوم والتكنولوجيا
Faculty of Sciences and Technology
قسم الهندسة المدنية
Civil Engineering Department

N° d'ordre : M2 ….. /GC/2025

MEMOIRE DE FIN D'ETUDES DE MASTER ACADEMIQUE

Filière : Travaux publics

Option : Voie et Ouvrage d'Art (VOA)

Thème

Détection automatique des dégradations de chaussée souple par Deep Learning : Application de YOLOv8 sur le tronçon routier de 9Km SIDI ALI – Sidi Lakhdar

Présenté par :

- HALIMA SALEM Kawthar

Soutenu le 24 /06/2025 devant le jury composé de :

Pr. MEBROUKI Abdelkader	—Président du jury	— Université de Mostaganem
Pr. MERAH Mustapha	— Examinateur	—Université de Mostaganem
Dr. El Mascri Setti	— Examinatrice	— Université de Mostaganem
Dr. ZAOUI Mohamed	— Encadrant	— Université de Mostaganem

Année Universitaire 2024/2025

Résumé

L'évaluation de l'état des chaussées souples constitue un enjeu majeur pour la gestion durable des infrastructures routières. Les détériorations telles que les fissures, le faïençage et l'arrachement sont des indicateurs révélateurs de la dégradation structurelle. Actuellement, leur identification s'effectue par des moyens rudimentaires fondés sur inspection visuelle des chaussées. Ce traitement manuel est complexe, long et surtout subjectif. Dans ce travail de fin d'études, nous proposons une approche automatisée de détection des dégradations basée sur le Deep Learning, en utilisant le modèle YOLOv8 et ses cinq variantes. L'étude a été appliquée sur un tronçon routier de 9 km, avec des données collectées par deux moyens : un drone et un smartphone embarqué sur véhicule. Toutefois, seules les données issues du drone ont été exploitées, en raison de la contrainte temporelle du projet. Les résultats expérimentaux ont montré que la variante YOLOv8s a offert les meilleures performances en matière de détection et de classification. Cette étude met en évidence la pertinence des techniques d'apprentissage profond dans le cadre de l'inspection automatique des routes et ouvre des perspectives pour l'intégration de données multi sources dans de futurs travaux.

Mots clés : Deep Learning, YOLOv8, détection automatique, fissures, faïençage, arrachement, chaussées souples.

Abstract

The evaluation of flexible pavement conditions is a key aspect of sustainable road infrastructure management. Distresses such as cracks, alligator cracking, and raveling are critical indicators of structural deterioration. Traditionally, these are detected through manual visual inspections, which are time-consuming and subjective. This final year project proposes an automated approach based on Deep Learning to detect such road surface defects, using the YOLOv8 model and its five variants. The methodology was applied to a 9 km road section, with data collected through two methods: drone imagery and a smartphone mounted on a moving vehicle. Due to time constraints, only drone-acquired data were used in this study. Experimental results revealed that the YOLOv8s variant provided the most accurate detection and classification outcomes. This research demonstrates the potential of deep learning models in automating road condition assessments and suggests future studies incorporating multi-source data for enhanced reliability.

Keywords: Deep Learning, YOLOv8, automatic detection, cracks, alligator cracking, raveling, flexible pavement.

الملخص

يُعدّ تقييم حالة الأرصفة المرنة من التحديات الأساسية في سبيل إدارة مستدامة للبنية التحتية الطرقية. وتُعتبر التدهورات مثل الشقوق، التّشققات التمساحية (الفَيْنَصَة)، والانفصال السطحي (الاقتلاع) من المؤشرات الدالة على تدهور البنية الهيكلية للطريق. وتُعتمد حالياً طرق تقليدية للكشف عنها تعتمد أساساً على المعاينة البصرية، وهي وسائل يدوية تتسم بالتعقيد، وتستغرق وقتاً طويلاً، كما أنها تخضع للذاتية بشكل كبير.

في هذا العمل البحثي، نقترح منهجاً آلياً للكشف عن تدهورات سطح الطريق بالاعتماد على تقنيات التعلم العميق، من خلال استخدام نموذج YOLOv8بجميع نسخه الخمس. وقد تم تطبيق هذه المنهجية على مقطع طولي من طريق يبلغ طوله 9 كيلومترات، باستخدام بيانات تم جمعها بطريقتين: بواسطة طائرة مسيرة (درون)، وبواسطة هاتف ذكي مُثبّت على مركبة. غير أنّه، ونظراً لضيق الوقت المخصص للمشروع، تم اعتماد البيانات المأخوذة عن طريق الدرون فقط.

أظهرت النتائج التجريبية أن النسخة YOLOv8s هي الأكثر كفاءة من حيث دقة الكشف والتصنيف. وتؤكد هذه الدراسة مدى فاعلية تقنيات التعلم العميق في عمليات التفتيش الآلي للطرق، كما تفتح آفاقاً مستقبلية واعدة، لاسيما فيما يتعلق بدمج مصادر بيانات متعددة في الدراسات القادمة.

الكلمات المفتاحية :التعلم العميق، YOLOv8، الكشف الآلي، الشقوق، التشققات التمساحية، الاقتلاع، الأرصفة المرنة.

Remercîment

À Dieu,

Lui qui m'a appris que la lumière naît toujours des ténèbres,
que le soulagement surgit du fond de la patience, et que la réussite ne se donne pas : elle se conquiert, avec détermination et prière.

À Lui, toute ma gratitude — une gratitude qui embrasse mes faiblesses et me revêt de force,
Lui qui, à chaque instant où je vacillais, m'a entourée de Sa miséricorde.

À mon encadrant M. ZAOUI Mohamed,

Vous qui avez cru en moi, même lorsque moi-même je doutais.Vous qui avez perçu en moi l'étincelle d'une chercheuse, et l'avez nourrie par vos conseils exigeants et bienveillants.

À mon père,

Cet homme digne et silencieux, dont les sacrifices n'ont jamais eu besoin de mots.
Chaque regard, chaque pas en retrait pour me laisser avancer... je te les dois.Tu es mon pilier, ma force tranquille, ma fierté.

À ma mère,

Source de prières, de tendresse et de lumière.Si j'ai marché aussi loin, c'est porté(e) par tes invocations, ton amour inépuisable et ta force silencieuse.Tu es le cœur de ma réussite.

À mes grands-parents,

Vous êtes mes racines, mes bénédictions vivantes.Votre affection et vos prières ont été mon armure contre toutes les tempêtes.

À ma sœur Selma, mon miroir et ma confidente,

À mes frères Ilyas et Imad Eddine, mes soutiens fidèles et discrets,

Mes nièce Aridj er Rahma et mes neveux Omar, Mohamed, Yacine, Idris et Soheib.

À mes professeurs,

Merci d'avoir éveillé en moi le goût du savoir,
Merci pour chaque mot transmis, chaque conseil offert avec exigence et foi.

À mes amis,

Asma, Lina, Imane, Malika

Vous avez été les étoiles de mes nuits de doute. Votre rire a apaisé bien plus que mille discours.
Merci d'avoir été un foyer lorsque tout semblait froid.

À madame Akermi kheira d'être notre deuxième maman

À Meriem, Malika et Mensouria

À ma professeur de primaire SALEM Halima

Enfin...

Ce mémoire n'est pas une simple étape. Il est le fruit d'un chemin parcouru avec lenteur, parfois douleur, mais toujours illuminé par la foi, l'amour et l'espérance.

Dédicace

À mes parents, pour leur amour inconditionnel et leurs sacrifices silencieux.
À mes grands-parents, sources de bénédictions et racines de mon être.
À ma sœur Selma, mon reflet tendre et fidèle,
À mon beau-frère Sofiane
À mes frères Ilyes et Imad Eddine, protecteurs discrets de mes pas,
À mes belles-sœurs Kheira et Fatima
À mes nièces Aridj et Rahma et neveux Omar, Mohamed, Yacine, Idris et Soheib, éclats de vie et promesses d'avenir.
À mes amis Lina, Iman, Asmaa, Malika
Et à moi-même…
Pour avoir tenu bon, quand tout semblait vaciller.
Pour n'avoir jamais cessé de croire, même dans le doute.
Pour avoir avancé, avec foi, patience et dignité.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale :

Dans un pays aussi vaste que l'Algérie, dont le réseau routier s'étend sur plus de 141 500 km et comprend plus de 4 800 ouvrages d'art [1], le transport routier joue un rôle crucial dans la mobilité des biens et des personnes. Néanmoins, ce réseau est exposé à diverses dégradations telles que les fissures, les nids-de-poule, les affaissements ou encore les ornières [2], [3]. Ces détériorations, s'ils ne sont pas détectés et traités à terme, peuvent compromettre la sécurité des usagers et accentuer les coûts d'entretien [4], [5]. En Algérie, l'inspection des chaussées reste très rudimentaire, se limitant souvent à des méthodes manuelles et subjectives [6]. Ni l'inspection semi-automatique, ni les technologies avancées de détection ne sont mises en œuvre à l'échelle nationale, malgré les risques encourus et les pertes économiques associées à la dégradation des routes [7], [8]. Dans ce contexte, et face à l'absence de solutions locales automatisées, la détection intelligente des détériorations de chaussée devient une nécessité.

Avec l'essor de l'intelligence artificielle (IA), et notamment des réseaux de neurones convolutionnels profonds (DCNN), de nombreux travaux de recherche ont permis des avancées majeures dans la détection d'objets, y compris les défauts de surface des routes [9], [10], [11]. Les algorithmes d'IA utilisés pour la détection automatique des défauts peuvent être classés en deux grandes catégories : - Les méthodes en une étape, comme SSD (Single Shot Multibox Detector) [12] et YOLO (You Only Look Once) [13], qui sont rapides et adaptées aux environnements contraints en ressources ; - Les méthodes en deux étapes, comme R-CNN [14], Fast R-CNN [15], Faster R-CNN [16] et Mask R-CNN [17], qui offrent souvent une meilleure précision mais nécessitent plus de puissance de calcul. Les méthodes en une étape sont aujourd'hui privilégiées pour les applications embarquées, notamment sur smartphone ou drone, en raison de leur rapidité et de leur légèreté [18], [19].

C'est dans ce cadre que s'inscrit le présent projet de fin d'étude, qui s'articule autour de cinq chapitres complémentaires : - Le chapitre I est consacré à une étude détaillée des différents types de dégradations des chaussées souples observés en Algérie, illustrée par des cas concrets et des classifications pertinentes. - Le chapitre II présente les principes fondamentaux de l'intelligence artificielle, avec un accent particulier sur l'apprentissage profond et les réseaux de neurones convolutionnels. - Le chapitre III explore les modèles d'IA appliqués à la détection d'objets, en détaillant les architectures en une étape et en deux étapes, et en justifiant le choix des modèles retenus pour notre cas d'étude. - Le chapitre IV décrit la mise en œuvre pratique de la solution proposée, appliquée à un tronçon routier réel reliant Sidi Lakhdar à Sidi Ali, inspecté à l'aide d'un drone et d'un smartphone embarqué sur véhicule. - Le chapitre V est consacré à l'analyse des résultats et à leur discussion, afin d'évaluer l'efficacité du système proposé.

Enfin, une conclusion générale viendra clore ce travail en mettant en lumière les contributions principales et en proposant des perspectives d'amélioration et de recherche future, notamment dans le contexte du développement durable et de la gestion proactive des infrastructures routières en Algérie.

Chapitre I. : Les dégradations de la chaussée souple

I.1. Introduction :

Au fils du temps, malgré la qualité des études et de réalisation des projets routiers, les chaussées subissent diverses dégradations qui peuvent affecter leur performance et leur durée de vie.

La compréhension des causes et des types des endommagements est donc primordiale pour assurer un entretien efficace et garantir la sécurité des usagers.

Ce chapitre vise à définir la chaussée souple, en expliquant ses principales caractéristiques et sa structure. Il abordera ensuite les différentes causes pouvant entraîner sa dégradation, qu'elles soient d'origine mécanique, climatique ou liée à un défaut de conception. Enfin, une classification des principaux types de dégradations observés sera présentée, afin de poser les bases nécessaires à l'application du Deep Learning pour leur détection .

I.2. Structure et Comportement des Chaussées Souples :

I.2.1. Définition

Les chaussées souples sont des chaussées dont le corps est réalisé avec des matériaux non liés ou traités avec un liant hydrocarboné. Elles sont caractérisées par leur flexibilité, permettant une meilleure adaptation aux déformations du sol support et aux charges appliquées par le trafic[20] .

I.2.2. Structure de chaussée souple :

Une chaussée souple est composée de plusieurs couches ayant chacune un rôle spécifique :

- **Revêtement en béton bitumineux** : assure l'adhérence des véhicules et protège les couches inférieures.
- **Couches de base et de fondation** : répartissent les charges et garantissent la stabilité de la chaussée.
- **Couche de forme**: Lorsque le sol support présente une faible portance, une couche de forme est ajoutée afin d'améliorer la capacité portante de la plateforme et de garantir une base plus stable pour les couches supérieures de la chaussée.
- **Sol support** : élément fondamental influençant la performance de la structure.

Pour illustrer ces propos, on se réfère à la figure I.1 suivante :

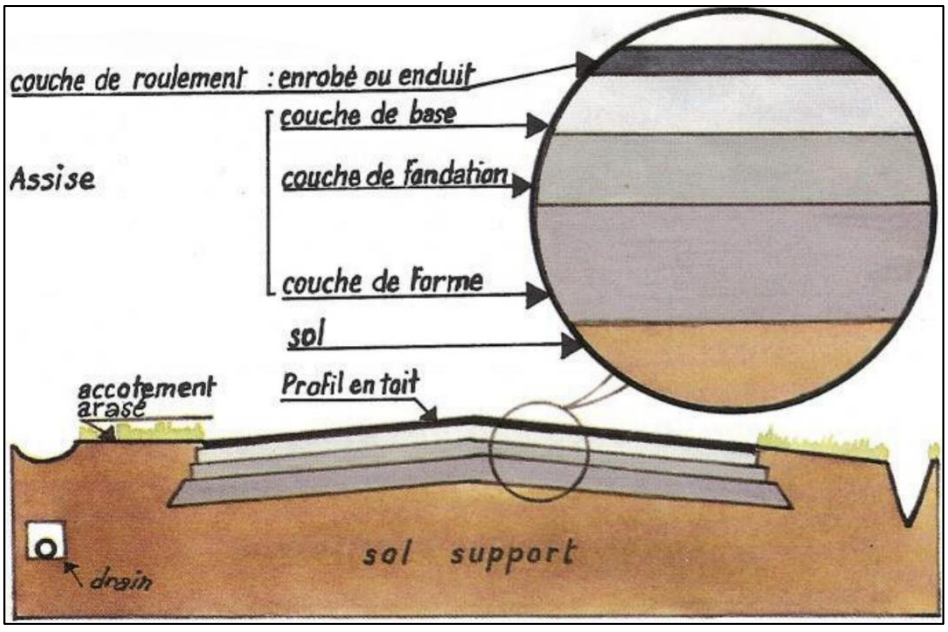


Figure I.1: la structure d'une chaussée souple.

I.2.3. Transmission des Efforts au Sol Support:

Dans une chaussée souple, les charges appliquées par les véhicules sont transmises progressivement au sol support à travers les différentes couches. La répartition des efforts dépend de la qualité et de l'épaisseur de ces couches, ainsi que des caractéristiques mécaniques du sol sous-jacent, comme le montre la figure suivante :

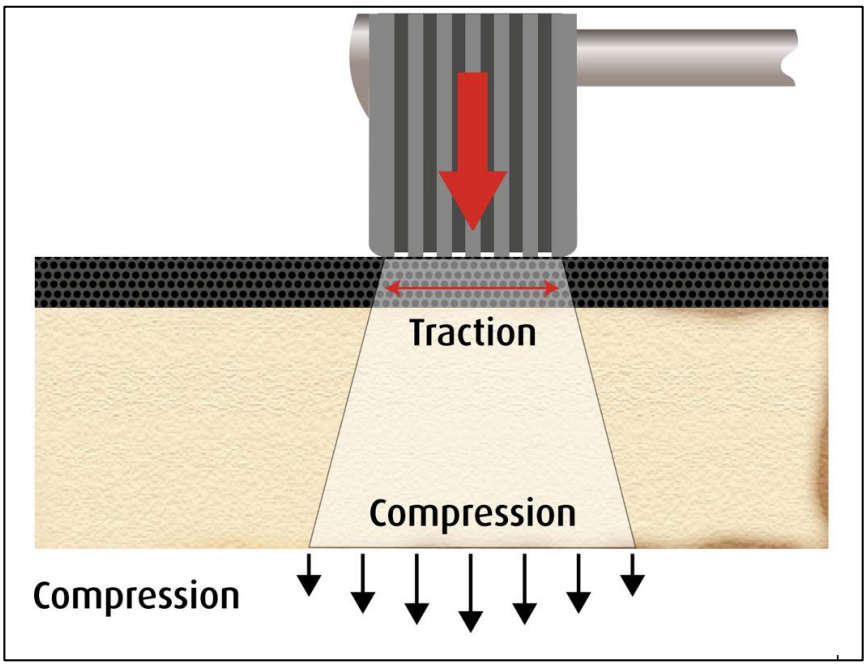


Figure I.2: transmission des charges.

I.3. Les dégradations :

I.3.1. Définition:

Une dégradation de chaussée est une altération locale ou étendue du revêtement routier ou de ses couches structurelles, causée par des contraintes mécaniques (trafic), des facteurs environnementaux (eau, température), ou un défaut de conception ou de mise en œuvre. Elle affecte les performances fonctionnelles et/ou mécaniques de la chaussée [21].

D'une manière générale, les dégradations observées dans les chaussées souples peuvent être répertoriées en quatre principales familles qui sont :

- La famille des fissurations
- La famille des déformations
- La famille des arrachements
- La famille des remontes de matériaux

I.3.2. Les types des dégradations :

I.3.2.1. La famille des fissurations

I.3.2.1.1. Les fissurations longitudinales :

Les fissurations longitudinales est une fissure qui apparaît dans le sens de la route, généralement alignée avec la direction de la circulation. Elle peut traverser partiellement ou entièrement la chaussée, sur une longue distance [22].

Comme le montre la figure I.3 suivante:

Figure I.3: fissure longitudinale.

I.3.2.1.2. Fissures Transversales :

Les fissures transversales sont des fissures qui apparaissent perpendiculairement à l'axe de la chaussée. Elles affectent principalement la couche de roulement et peuvent évoluer en pathologies plus graves si elles ne sont pas traitées à temps [22].

Comme le montre la figure I.4 suivante :

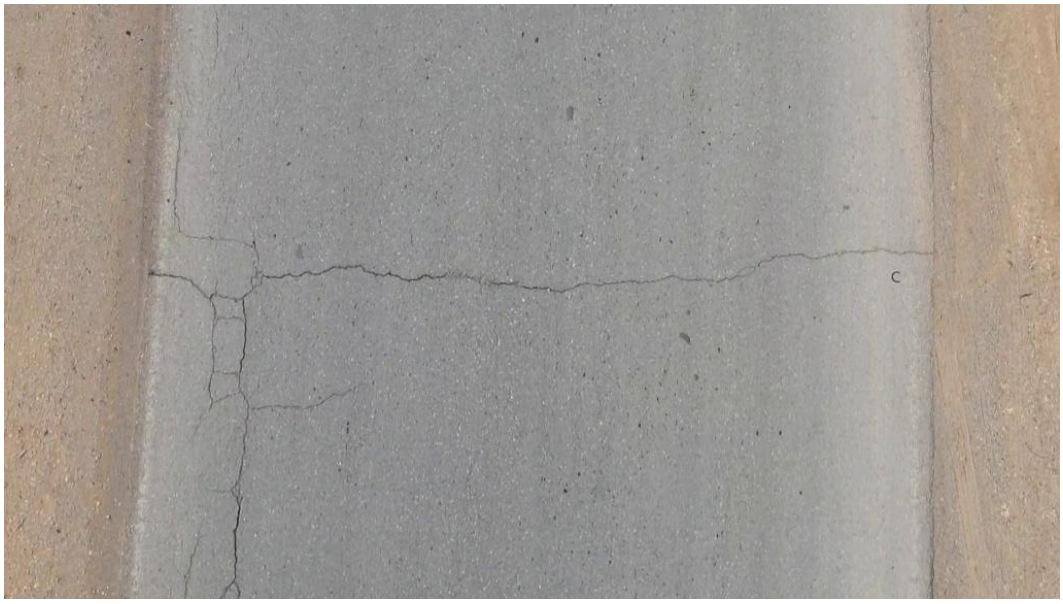


Figure I.4: fissure transversale.

I.3.2.1.3. Le faïençage :

Le faïençage est par définition un ensemble de fissures plus ou moins rapprochées formant des mailles. Elles sont dites à mailles fines ou peaux de crocodiles [22].

Comme le montre la figure I.5 suivante :

Figure I.5:faiençage.

I.3.2.2. La famille des déformations:

I.3.2.2.1. Affaissement :

L'affaissement est une pathologie courante qui affecte les bords des chaussées souples. Il se manifeste par un affaissement progressif du revêtement en bordure de la chaussée, pouvant entraîner des déformations importantes et compromettre la sécurité des usagers [22].

Comme le montre la figure I.6 suivante :

Figure I.6: affaissement.

I.3.2.2.2. Flache :

La flache est une déformation localisée de la chaussée, se manifestant sous la forme d'un tassement ovale. Il est souvent le signe d'une faiblesse structurelle et peut évoluer vers des pathologies plus graves si aucune intervention n'est réalisée.

Comme le montre la figure I.7 suivante :

Figure I.7: flache.

I.3.2.3. Famille des arrachements :

I.3.2.3.1. Nid de poule :

une cavité plus ou moins profonde , souvent irrégulière, causée par l'arrachement du revêtement routier (bitume, enrobé) et parfois des couches situées en dessous [23].

Comme le montre la figure I.8 suivante :

Figure I.8: Nid de poule.

I.3.2.3.2. Pelade :

Arrachement par plaque plus ou moins grande de l'enrobé de la couche de roulement.

Comme le montre la figure I.9 suivante :

Figure I.9: Pelade .

I.3.3. Le mode d'endommagement :

Dans une chaussée souple, par exemple, les couches inférieures (sol, matériaux granulaires) peuvent se déformer peu à peu sous les charges répétées des véhicules. Ces déformations deviennent permanentes avec le temps et apparaissent en surface sous forme d'ornières, de bosses ou de creux.

En même temps, les couches bitumineuses situées en surface subissent des flexions dues au passage des roues. À force, cela provoque des fissures qui s'étendent et finissent par créer un réseau de petites cassures : c'est ce qu'on appelle le faïençage.

L'eau joue aussi un rôle important dans le mode d'endommagement. Elle peut s'infiltrer par les fissures ou venir du sol lui-même. Une fois à l'intérieur de la chaussée, elle affaiblit les couches internes et accélère les dégâts : épaufrures, détachement de matériaux, et apparition de nids-de-poule [24].

I.4. Conclusion :

Ce chapitre a permis de caractériser les principales formes de dégradations affectant les chaussées souples ainsi que leurs mécanismes d'apparition. Ces pathologies constituent des indicateurs importants de l'état structurel et fonctionnel de la chaussée. Une bonne compréhension de ces phénomènes est indispensable pour développer des outils de surveillance intelligents. Dans ce contexte, le recours à des approches basées sur la vision par ordinateur, telles que le modèle YOLOv8, s'inscrit comme une solution prometteuse pour la détection rapide, fiable et automatisée des dégradations.

Chapitre II.: L'intelligence Artificielle

II.1. Introduction

L'intelligence artificielle (IA) est aujourd'hui au cœur de nombreuses innovations technologiques, transformant des secteurs aussi variés que la santé, les transports, le droit ou encore l'analyse textuelle. Dans le cadre de notre projet de fin d'études intitulé "**Détection automatique des dégradations de chaussée par Deep Learning : Application de YOLOv8 sur le tronçon routier Sidi Lakhdar – Sidi Ali**" , il est essentiel de bien comprendre les fondements théoriques de ces technologies. Ce chapitre propose une introduction structurée à l'intelligence artificielle , en distinguant clairement ses branches principales :

- L'apprentissage automatique (Machine Learning - ML)
- L'apprentissage profond (Deep Learning - DL)

Nous aborderons leurs définitions , leurs fonctionnements , leurs architectures typiques , ainsi que les différences entre eux.

II.2. Définitions :

II.2.1. Intelligence Artificielle (IA)

L'intelligence artificielle est une discipline scientifique visant à doter les machines de capacités cognitives proches de celles de l'homme, telles que la perception, le raisonnement, l'apprentissage et la prise de décision [8].]. La figure II.1 illustre les domaines et les disciplines de l'IA.

Elle englobe plusieurs domaines :

- Vision par ordinateur
- Traitement du langage naturel
- Robotique
- Apprentissage automatique

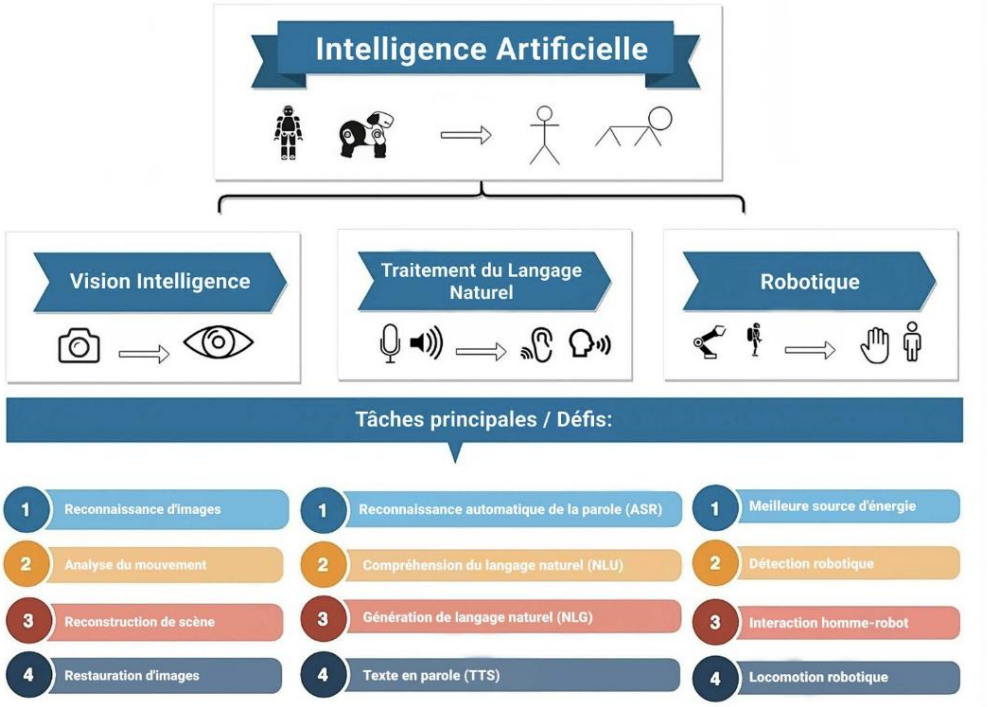


Figure II.1: Domaines et disciplines de l'IA [25].

II.2.2. Apprentissage Automatique (Machine Learning - ML) :

Le Machine Learning constitue une branche essentielle de l'intelligence artificielle. Il repose sur la conception d'algorithmes capables d'apprendre à partir de données préalablement collectées.

L'expression « Machine Learning » a été introduite en 1959 par Arthur Samuel , qui la définit comme la capacité donnée à une machine d'apprendre automatiquement à partir d'expériences passées, c'est-à-dire à partir de données, afin d'optimiser ses performances au fil du temps. Autrement dit, il s'agit de méthodes permettant aux systèmes informatiques de tirer des conclusions et de produire des résultats sans avoir été programmés explicitement pour chaque tâche spécifique. [26].

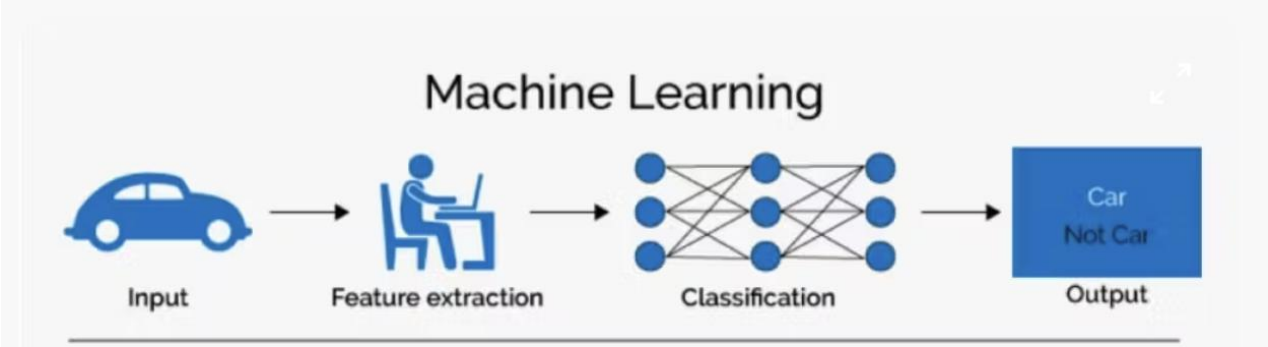


Figure II.2 : Processus du Machine Learning [27]

II.2.2.1. Apprentissage supervisé :

L'apprentissage supervisé englobe l'ensemble des techniques qui permettent à un modèle de tirer des enseignements à partir de données étiquetées. Autrement dit, il s'agit d'une approche où chaque exemple utilisé lors de l'entraînement est associé à une réponse correcte (étiquette), ce qui permet au système d'apprendre les relations entre les entrées et les sorties attendues. Une fois cet apprentissage réalisé, le modèle est capable de généraliser ces connaissances pour traiter et classer de nouvelles données inconnues.

Parmi les algorithmes supervisés les plus courants, on peut citer :

- Méthodes de régression :Régression linéaire, régression logistique, méthodes d'ensemble (comme Bagging et Boosting), arbres de décision, forêts aléatoires, régression par processus gaussien (GPR).
- Algorithmes de classification :Machine à vecteurs de support (SVM), Naïve Bayes, k-plus proches voisins (k-NN).
- Modèles basés sur les réseaux de neurones :Architectures telles que le perceptron multi-couches, utilisées dans divers domaines pour modéliser des relations complexes à partir de données brutes[26].

Dans ce type d'apprentissage, l'algorithme est entraîné à partir de données étiquetées, c'est-à-dire que chaque exemple d'entrée est associé à une sortie attendue. Cette approche permet au modèle d'apprendre à prédire des résultats sur de nouvelles données en se basant sur les relations observées durant l'entraînement. Les principales tâches associées sont :

- **La classification**, qui a pour objectif de prédire une catégorie ou une classe (par exemple : identifier le type de détérioration d'une chaussée, comme une fissure longitudinale, un nid-de-poule ou un affaissement) ;
- **La régression**, qui consiste à estimer une valeur numérique continue (par exemple : prédire la profondeur ou la largeur d'une fissure en fonction des caractéristiques de la chaussée et des conditions climatiques).

II.2.2.2. L'apprentissage non supervisé

L'apprentissage non supervisé se réfère à une catégorie de méthodes d'apprentissage automatique dans lesquelles aucune donnée étiquetée n'est utilisée. Contrairement à l'apprentissage supervisé, ces algorithmes fonctionnent de manière autonome, sans intervention humaine explicite, en cherchant à explorer et à découvrir des structures sous-jacentes dans les données.

L'objectif ici est d'identifier des motifs, des regroupements ou des relations au sein des données, sans connaître à l'avance leurs classes ou catégories d'appartenance.

Parmi les techniques non supervisées les plus courantes, on retrouve :

- **Clustering (regroupement)** :Méthodes visant à partitionner les données en groupes homogènes, tels que :
  - L'algorithme K-means
  - L'analyse hiérarchique de clustering
  - L'Analyse en Composantes Principales (ACP)
  - L'ACP à noyau (Kernel PCA)
  - Les réseaux de neurones auto-organisés (SOM)
- **Analyse d'association (approche descriptive)** :Utilisée principalement pour identifier des relations entre des variables catégorielles dans de grands ensembles de données [26].

II.2.2.3. Apprentissage par renforcement:

Dans une application basée sur un drone de surveillance, l'apprentissage par renforcement repose sur l'interaction continue entre le drone (l'agent) et son environnement (la chaussée à inspecter). Le drone prend des décisions séquentielles, comme ajuster son altitude, modifier sa trajectoire ou adapter la fréquence de prise d'images, dans le but d'optimiser la détection des dégradations routières. L'objectif est de maximiser une récompense globale, par exemple liée au taux de détection correcte des fissures, nids-de-poule ou affaissements.

À chaque action effectuée (changer d'angle de vue, ralentir au-dessus d'une zone suspecte, etc.), l'environnement renvoie une rétroaction sous forme de récompense (en cas de détection correcte) ou de pénalité (si des défauts sont manqués ou mal identifiés).

Deux grandes approches peuvent être utilisées pour entraîner ce type de système :

- **L'algorithme Monte Carlo**, où l'apprentissage repose sur les récompenses calculées à la fin d'une mission complète (par exemple, un vol d'inspection d'une portion de route).
- **L'apprentissage par différence temporelle (TD)**, qui met à jour les estimations à chaque étape du vol, en prenant en compte les récompenses immédiates (images détectant bien les défauts) ainsi que les prédictions futures sur la qualité de la détection dans les zones suivantes.

II.2.3. L'apprentissage profond (Deep learning):

L'apprentissage profond (Deep Learning) constitue une branche avancée et stratégique de l'apprentissage automatique (Machine Learning), centrée sur la modélisation de représentations hiérarchiques à partir de données brutes. Contrairement aux méthodes classiques où les caractéristiques pertinentes doivent souvent être extraites manuellement, le deep learning permet au modèle d'apprendre de manière autonome les structures sous-jacentes nécessaires pour accomplir des tâches comme la classification, la détection ou la reconnaissance.

Ce type d'apprentissage repose sur des réseaux de neurones artificiels profonds , organisés en plusieurs couches interconnectées. Ces réseaux imitent, dans une certaine mesure, le fonctionnement du cerveau humain : chaque unité (ou « neurone ») effectue des opérations simples, mais leur agencement en couches successives permet de capturer des relations complexes. Par exemple, lors de l'analyse visuelle, les premières couches détectent des éléments élémentaires comme des contours ou des angles, tandis que les couches supérieures apprennent progressivement à reconnaître des formes, des objets ou même des contextes visuels.

Le Deep Learning se distingue particulièrement par sa capacité à traiter directement des données non structurées telles que les images, le texte ou encore l'audio, sans prétraitement intensif. Il s'agit donc d'un outil puissant pour résoudre des problèmes complexes dans divers domaines, notamment la vision par ordinateur, le traitement du langage naturel ou encore le diagnostic médical assisté par ordinateur.

Les performances élevées atteintes par ces modèles sont étroitement liées à deux facteurs essentiels :

- Une grande quantité de données étiquetées , souvent issues de bases massives (big data),
- Des architectures réseau comportant de nombreuses couches, rendant possible l'extraction progressive de caractéristiques de haut niveau.

Grâce à l'amélioration constante des capacités de calcul et à la disponibilité croissante de données, les modèles d'apprentissage profond ont atteint, dans certains cas, une précision égale voire supérieure à celle de l'humain, notamment dans des tâches comme la reconnaissance faciale, la détection de maladies à partir d'imagerie médicale, ou la compréhension du langage parlé.

Ainsi, le Deep Learning ne se limite pas à une simple extension technique du Machine Learning ; il représente une évolution majeure vers des systèmes capables d'apprendre de manière autonome, en capturant des patterns invisibles ou difficiles à modéliser par des approches traditionnelles[8][9].

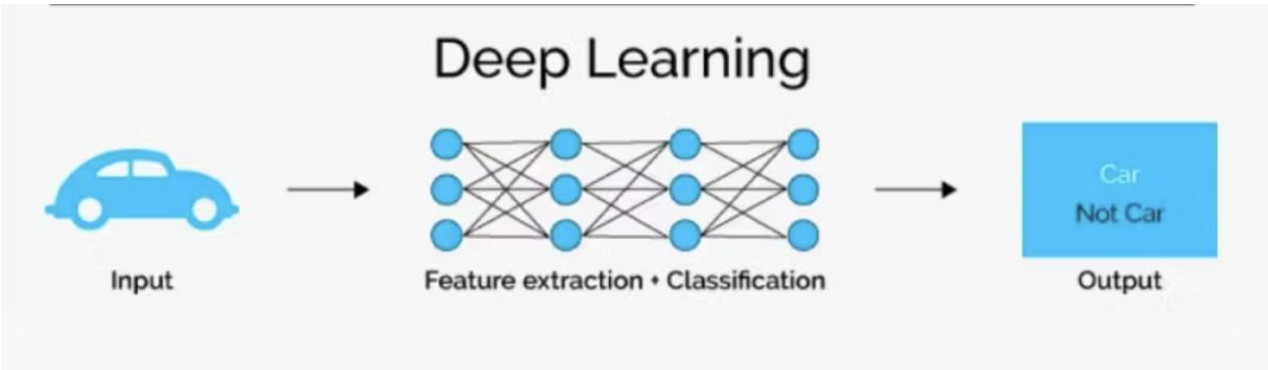


Figure II.3 : Processus du Deep Learning [27]

II.3. Concepts fondamentaux du Deep Learning

Le Deep Learning repose sur des architectures avancées de modèles prédictifs basés sur la descente de gradient. Ce domaine se distingue principalement par sa capacité à apprendre et modéliser des relations complexes grâce à l'utilisation de réseaux de neurones profonds .

Un réseau de neurones est constitué d'un ensemble de nœuds (ou neurones) interconnectés, organisés en couches. Il prend en entrée un ensemble de données brutes, les transforme à travers ces

couches successives, et produit en sortie une prédiction ou une classification. Chaque couche extrait progressivement des caractéristiques de plus en plus abstraites à partir des données fournies.

Parmi les principales architectures utilisées dans le cadre du DL, on distingue notamment :

II.3.1. Réseaux de Neurones Profonds (DNN) :

Les DNN (Deep Neural Networks) sont des réseaux composés de nombreuses couches cachées entre l'entrée et la sortie. Ces couches permettent de capturer des représentations hiérarchiques des données.

Chaque neurone d'une couche est connecté à tous les neurones de la couche suivante via des poids ajustables , ce qui constitue une architecture dite entièrement connectée .

Au fil des couches, les données subissent successivement une transformation linéaire suivie d'une fonction d'activation non linéaire, permettant au modèle de s'adapter à des problèmes de grande complexité [28].

II.3.2. Réseaux de Neurones Convolutionnels (CNN) :

Contrairement aux DNN classiques, les réseaux de neurones convolutionnels (CNN) exploitent des couches spécifiques appelées couches de convolution , capables d'extraire automatiquement des motifs spatiaux pertinents.

Ces réseaux sont particulièrement adaptés au traitement des images, grâce à leur capacité à identifier localement des structures visuelles comme des bords, textures ou objets.

Ils reposent sur l'utilisation de filtres (kernels) qui balayent l'image pour détecter des caractéristiques pertinentes, tout en préservant les relations spatiales entre pixels. Cette propriété les rend très efficaces pour des tâches telles que la classification d'images , la détection d'objets ou la segmentation [28].

Dans le cadre de cette étude, les CNN occupent une place centrale, notamment pour leur puissance dans l'extraction de motifs visuels à partir des images routières analysées.

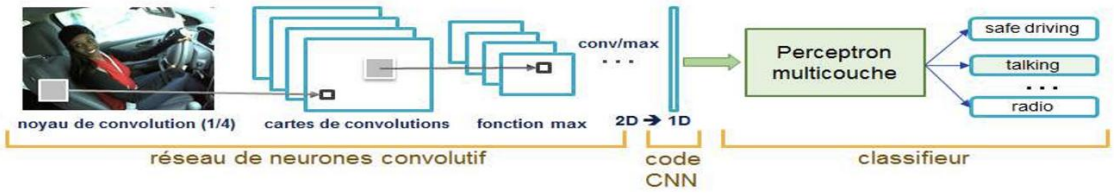


Figure II.4: Réseaux de Neurones Convolutionnels [29].

II.3.3. Les RNN (Recurrent Neural Networks) :

sont conçus pour traiter des données structurées sous forme de séquences temporelles ou contextuelles.

Ils intègrent une mémoire implicite , leur permettant de prendre en compte l'information précédente lors du traitement d'un nouvel élément de la séquence.

Cette architecture est particulièrement utile pour des applications comme la compréhension du langage naturel ou l'analyse de séries temporelles.

Afin de pallier certaines limites des RNN classiques, notamment le problème du gradient évanescent , des variantes plus performantes ont été développées :

- Les LSTM (Long Short-Term Memory)
- Les GRU (Gated Recurrent Units)

Ces dernières disposent de mécanismes internes sophistiqués permettant de mieux gérer les dépendances à long terme, ce qui améliore grandement leur performance sur les données séquentielles.

II.4. Différences entre Machine Learning et Deep Learning :

Dans le cadre de la détection de détériorations routières à l'aide de drones, il est essentiel de distinguer les approches basées sur le Machine Learning (ML) de celles reposant sur le Deep Learning (DL). Le ML repose généralement sur des algorithmes classiques, tels que les arbres de décision ou les machines à vecteurs de support (SVM), qui nécessitent une extraction manuelle des caractéristiques à partir des images (texture, contraste, formes géométriques, etc.). Ces méthodes sont moins exigeantes en termes de ressources informatiques et de données, mais leur précision peut être limitée face à la variabilité des conditions réelles (éclairage, type de revêtement, débris, etc.).

À l'inverse, le Deep Learning, sous-catégorie du ML, utilise des réseaux de neurones profonds comme les CNN (Convolutional Neural Networks), capables d'apprendre automatiquement des représentations visuelles complexes directement à partir des images capturées par le drone. Cette approche nécessite de grandes quantités de données annotées et des ressources de calcul importantes (souvent des GPU), mais elle offre une meilleure précision et une plus grande capacité de généralisation, notamment pour détecter des défauts subtils ou non structurés. Ainsi, le DL s'avère particulièrement adapté aux environnements routiers variés et dynamiques, où les détériorations peuvent être irrégulières ou difficiles à identifier par des méthodes classiques. Le tableau ci-dessous met en évidence les principales différences entre le Machine Learning (ML) et le Deep Learning (DL). Bien que le DL soit une sous-catégorie du ML, il se distingue par des exigences techniques, des performances et une complexité accrue. [26]

Tableau II.1:Comparaison entre le Deep Learning (DL) et le Machine Learning (ML)

Caractéristique	**Deep Learning (DL)**	**Machine Learning (ML)**
Relation hiérarchique	Sous-catégorie du Machine Learning	Branche de l'intelligence artificielle
Complexité	Plus complexe	Moins complexe
Architecture de base	Utilise des réseaux de neurones profonds, adaptés à des données complexes	Utilise divers algorithmes classiques (SVM, arbres de décision, k-NN, etc)
Volume de données requis	Nécessité de très grandes quantités de données pour être performant	Peut fonctionner avec des volumes de données moyens à grands
Ressources nécessaires	Requiert des **GPU spécialisés** pour l'entraînement intensif	Fonctionne généralement avec des **CPU standards**
Type d'apprentissage	L'apprentissage est largement automatisé, sans besoin d'extraction manuelle	Nécessite souvent une intervention humaine pour l'extraction de caractéristiques
Durée d'entraînement	Plus longue, en raison de la complexité des modèles	Généralement plus courte
Précision	Tendance à offrir une **plus grande précision**, surtout sur des tâches complexes	Précision **moindre** dans certains cas, notamment avec peu de données

Enfin, bien que les réseaux de neurones ne représentent qu'une approche parmi d'autres en ML (au même titre que les SVM, les forêts aléatoires ou les méthodes statistiques), ils constituent la base exclusive du DL . Ce dernier peut donc être vu comme une évolution avancée du ML, où l'apprentissage est entièrement fondé sur ces architectures neuronales profondes [9]. la figure II.5 expose la différence entre IA, ML et DL..

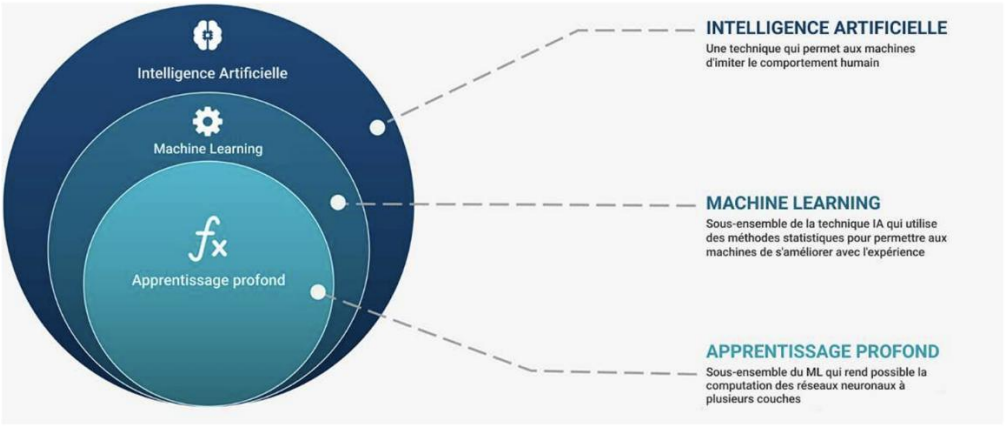


Figure II.5: Différences entre IA, Machine Learning et Deep Learning [25].

II.5. II.4 Les Réseaux de Neurones

II.5.1. Définition

Un réseau de neurones, aussi appelé réseau neuronal artificiel (ou artificial neural network, ANN, en anglais), est un modèle informatique s'inspirant du fonctionnement du cerveau humain. Il est constitué de nœuds appelés neurones ou unités, organisés en plusieurs couches, chacune jouant un rôle essentiel dans le traitement des données et dans le processus d'apprentissage. [30]

II.5.2. Les principales couches d'un réseau de neurones

Dans un réseau de neurones, il existe trois types principaux de couches : [30]

II.5.2.1. Structure d'un Réseau de Neurones

Un réseau de neurones est généralement structuré en trois types de couches principales :

II.5.2.1.1. Couche d'entrée (input layer)

La couche d'entrée est la première couche du réseau. Elle reçoit les données brutes à traiter. Chaque neurone de cette couche correspond à une caractéristique ou une variable des données d'entrée (par exemple, les pixels d'une image ou les valeurs numériques d'un tableau de données).

II.5.2.1.2. Couches cachées (hidden layers) :

Situées entre la couche d'entrée et la couche de sortie, les couches cachées réalisent les traitements internes complexes du réseau. Chaque neurone d'une couche cachée reçoit des signaux en provenance des neurones de la couche précédente, leur applique des poids, utilise une fonction d'activation pour introduire de la non-linéarité, puis transmet le signal transformé à la couche suivante. Les réseaux comprenant plusieurs couches cachées sont appelés réseaux profonds (deep neural networks) et constituent la base de l'apprentissage profond (deep learning).

a) **Couche de sortie (output layer)** :

La couche de sortie génère le résultat final du réseau, en fonction de la tâche à accomplir (classification, régression, etc.). Le nombre de neurones dans cette couche dépend du type de problème : un neurone unique pour une tâche de régression, plusieurs neurones pour une classification multi-classes, par exemple. Chaque neurone est connecté aux neurones des couches adjacentes par des connexions pondérées. Au cours de l'apprentissage, ces poids sont ajustés de manière à minimiser l'erreur de prédiction. Cette capacité d'apprentissage permet aux réseaux de neurones d'identifier des motifs complexes dans les données, ce qui les rend particulièrement efficaces pour des tâches variées telles que la reconnaissance d'images, la reconnaissance vocale, la traduction automatique de langues, ainsi que la détection de la dégradation des chaussées.

II.5.3. Perceptron:

II.5.3.1. Définition :

Le perceptron, introduit par Frank Rosenblatt en 1958, est l'un des premiers modèles de réseau de neurones artificiels. Il constitue un modèle simple et fondamental basé sur une seule couche de neurones, dite couche de sortie, directement connectée aux entrées du modèle. Chaque connexion entre une entrée et un neurone est associée à un poids, qui est ajusté lors de la phase d'apprentissage supervisé (comme illustré à la figure III.5). Le perceptron est particulièrement adapté aux tâches de classification linéaire, c'est-à-dire aux situations dans lesquelles les classes peuvent être séparées par une droite (en 2D) ou un hyperplan (en dimensions supérieures). Il permet donc de modéliser efficacement des relations simples dans les données.

Cependant, le perceptron présente des limites importantes : il est incapable de résoudre des problèmes non linéaires, comme la fonction logique XOR, car ces types de problèmes ne peuvent pas être séparés par un simple hyperplan. Pour dépasser ces limitations, des architectures plus avancées ont été développées, notamment les réseaux de neurones multicouches (MLP - Multilayer Perceptrons), capables de modéliser des relations non linéaires complexes grâce à l'empilement de plusieurs couches et à l'utilisation de fonctions d'activation non linéaires. [31]

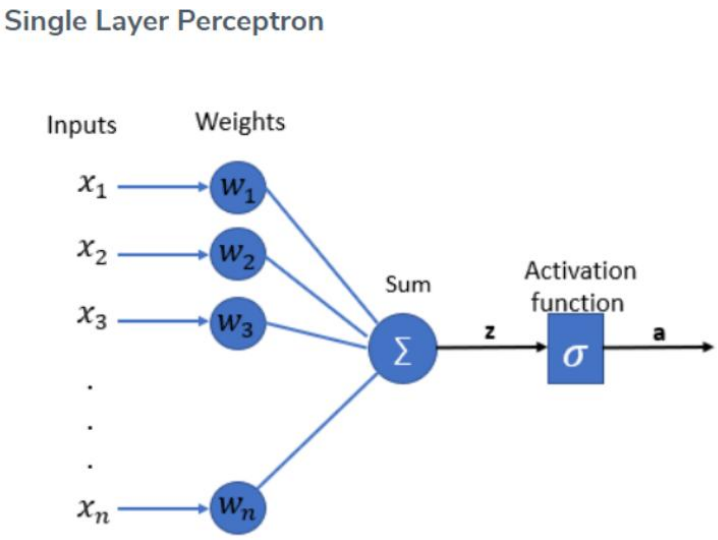


Figure II.6: perceptron monocouche [31]

II.5.3.2. Perceptron multicouche (MLP):

Le perceptron multicouche (MLP) constitue une évolution du perceptron simple, dont il surmonte les limites en matière de classification linéaire. Il s'agit d'un réseau de neurones artificiels composé d'au moins une couche cachée, permettant de réaliser des transformations non linéaires des données via des fonctions d'activation spécifiques (voir figure II.6). Grâce à cette structure, le MLP est capable de modéliser des relations complexes au sein des données, ce qui le rend particulièrement adapté à des tâches d'apprentissage supervisé plus sophistiquées. En raison de sa capacité à extraire des représentations abstraites, il est largement utilisé comme base dans de nombreuses architectures avancées, telles que les réseaux de neurones convolutifs (CNN) [32].

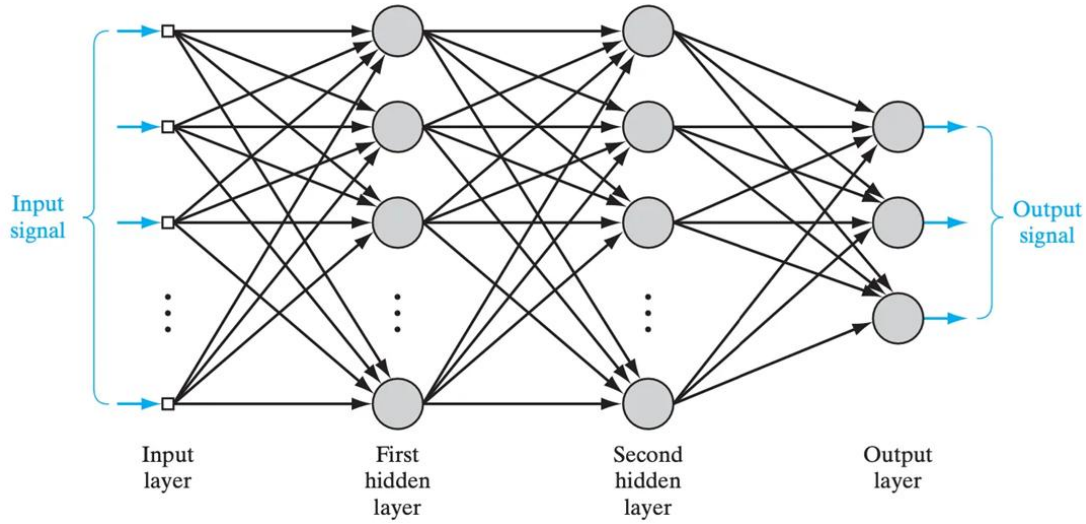


Figure II.7 : perceptron multicouche [32]

II.5.3.3. Composants de base d'un perceptron

Le perceptron se compose de plusieurs éléments fondamentaux qui définissent son fonctionnement (voir figure II.8). Ses principales caractéristiques incluent :

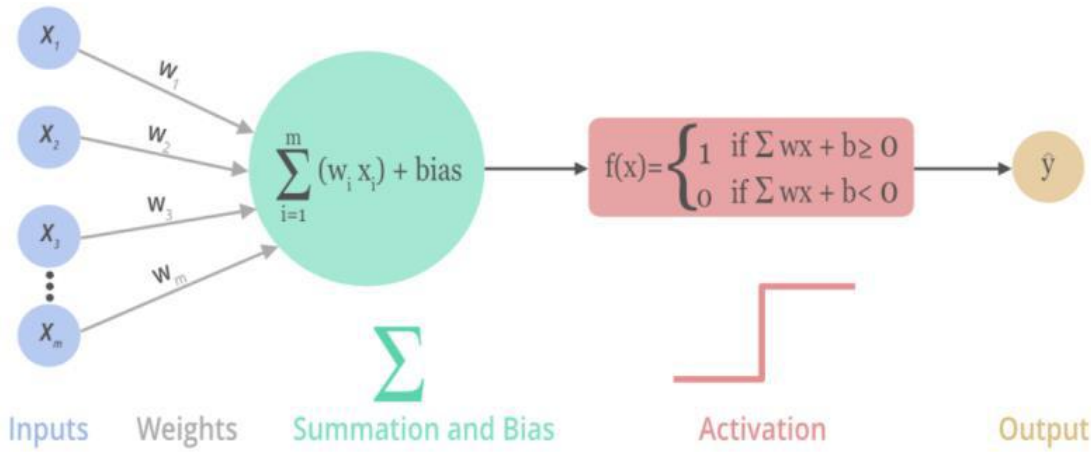


Figure II.8: Composants de base d'un Preceptron [32]

II.5.3.3.1. Les données d'entrée :

Également appelées features en anglais, les caractéristiques d'entrée représentent les données initiales fournies au perceptron pour le traitement de l'information. Elles constituent les éléments fondamentaux sur lesquels le modèle s'appuie pour apprendre et effectuer des prédictions. Chaque caractéristique correspond à une valeur spécifique, telle que l'intensité d'un pixel dans une image ou une mesure particulière dans un ensemble de données tabulaires.

II.5.3.3.2. Poids

Les poids constituent des paramètres essentiels dans un perceptron, car ils modulent l'importance accordée à chaque caractéristique d'entrée xi dans le calcul de la sortie finale. Chaque entrée est multipliée par un poids associé Wi, lequel est ajusté de manière itérative au cours de l'entraînement afin d'optimiser la précision du modèle et de réduire l'écart entre la prédiction et la valeur réelle. La valeur des poids détermine ainsi la contribution relative de chaque caractéristique : un poids élevé reflète une influence marquée, tandis qu'un poids faible indique une contribution moindre. En ce sens, les poids jouent un rôle central dans l'apprentissage automatique, permettant au perceptron de s'adapter aux spécificités des données et d'améliorer ses performances prédictives.

II.5.3.3.3. Biais :

Le biais (noté b) est un paramètre supplémentaire dans un perceptron qui permet d'ajuster la fonction d'activation. Il agit comme une constante ajoutée à la somme pondérée des entrées, proposant ainsi plus de flexibilité au modèle. Grâce au biais, le perceptron peut s'adapter même lorsque toutes les entrées sont nulles, ce qui améliore sa capacité à modéliser des données complexes. Sans biais, le perceptron serait limité à des fonctions linéaires passant forcément par l'origine (le point zéro), ce qui limite sa capacité à capturer certaines relations entre les données.

II.5.3.3.4. La somme pondérée :

Les produits des caractéristiques d'entrée et des poids sont additionnés plus le biais, comme montré sur la fonction suivante :

$$z = \sum_{i=1}^{n}(x_i * w_i) + b \qquad \text{II.1}$$

Ou en notation vectorielle

$$z = w * x + b \qquad \text{II.2}$$

Ensuite, ce z est passé dans une fonction d'activation (comme fonction seuil, sigmoïde, Relu, etc.) pour produire la sortie du neurone.

$$Sortie = \emptyset(z) = \emptyset(w * x + b) \qquad \text{II.3}$$

Avec

x= (x1, x2, ….,xn) : les caractéristiques d'entrée (features)

w = (w1, w2,….,wn) : les poids associés à chaque caractéristique

b : le biais, un paramètre libre,

$\emptyset$: la fonction d'activation.

II.5.3.3.5. La phase d'activation dans un perceptron

La phase d'activation est une étape cruciale du fonctionnement d'un perceptron. Elle intervient après le calcul de la somme pondérée des entrées, auquel on ajoute un biais. Cette valeur intermédiaire est ensuite transmise à une fonction d'activation, dont le rôle est de déterminer la sortie finale du neurone.

Dans le perceptron classique, la fonction d'activation est généralement une fonction seuil (y), aussi appelée fonction échelon de Heaviside. Celle-ci produit une sortie binaire :

- 1 si la somme pondérée dépasse un certain seuil (souvent 0),
- 0 sinon.

Ce mécanisme permet au perceptron de prendre des décisions simples, comme la classification linéaire binaire. Dans les architectures plus complexes (réseaux de neurones profonds), d'autres fonctions d'activation comme ReLU, sigmoid ou tanh sont souvent utilisées pour introduire de la non-linéarité, ce qui améliore les capacités d'apprentissage du modèle.

$$y = \begin{cases} 0 & si\ z < 0 \\ 1 & si\ z \geq 0 \end{cases} \qquad \text{II.4}$$

II.5.3.3.6. Les fonctions d'activation

Dans un réseau de neurones, la fonction d'activation joue un rôle fondamental : elle détermine la sortie d'un neurone à partir de la somme pondérée de ses entrées. Son objectif principal est d'introduire une non-linéarité dans le réseau. Cette non-linéarité est indispensable, car sans elle, même un réseau profond se comporterait comme un simple modèle linéaire, incapable de modéliser des relations complexes entre les données. Il existe plusieurs types de fonctions d'activation, chacune ayant des caractéristiques particulières. Le choix judicieux de la fonction dépend du problème à résoudre et peut significativement influencer les performances du modèle. Voici quelques-unes des fonctions d'activation les plus courantes illustrées dans la figure II.8 **:**

- **La fonction *Sigmoïde* :**

Cette fonction transforme toute valeur réelle en un résultat compris entre (0 et 1). Elle est souvent utilisée pour des problèmes de classification binaire, notamment en sortie de réseau. Son principal inconvénient est le risque de saturation (les gradients deviennent très faibles), ce qui ralentit l'apprentissage. Elle est exprimée par l'équation suivante II.5 suivante :

$$S(x) = \frac{1}{1+e^{-ax}} \qquad \text{II.5}$$

- **La fonction ReLU (Rectified Linear Unit) :**

C'est la fonction la plus utilisée dans les réseaux de neurones profonds. Elle est simple, rapide à calculer et permet de résoudre en partie le problème du gradient qui disparaît. Cependant, elle peut souffrir du phénomène de « neurones morts » (lorsque la sortie reste constamment nulle). Elle est formulée par l'équation II.6.

$$\mathrm{R(x)} = \max(0, \mathrm{x}) \qquad \text{II.6}$$

- **TanH (Tangente hyperbolique) :**

Similaire à la sigmoïde, mais centrée sur zéro (ses valeurs sont comprises entre -1 et 1). Elle permet souvent une convergence plus rapide que la sigmoïde, mais reste sujette au même problème de

gradients faibles pour des valeurs extrêmes. [33] Pour plus de détails techniques sur chaque fonction, voir référence [34]. Elle exprimée par l'équation II.7 suivante :

$$T(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} = \frac{1-e^{-2x}}{1+e^{-2x}} = 2 * \frac{1}{1+e^{-2x}} - 1 = 2 * logistic(2x) - 1 \qquad \text{II.7}$$

- **La fonction Softmax**

Est principalement utilisée dans la couche de sortie des réseaux de neurones pour les tâches de classification multi-classe. Elle permet de transformer un vecteur de scores (logits) en une distribution de probabilités, rendant ainsi les résultats interprétables.

Mathématiquement, pour un vecteur d'entrée z=(z1,z2,…,zk), la sortie Softmax pour la classe est définie par l'éqaution II.8 :

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_i}} \text{ pour i=1,2….N} \qquad \text{II.8}$$

Les caractéristiques principales de cette fonction sont :

- Toutes les sorties sont **comprises entre 0 et 1** ;
- La somme des sorties est égale à 1, ce qui permet de les interpréter comme des probabilités d'appartenance à chaque classe ;
- Elle est particulièrement adaptée aux situations où une seule classe doit être choisie parmi plusieurs.

Dans le cadre d'un système de détection automatique de détérioration de chaussée à l'aide d'un réseau de neurones convolutifs (CNN), la fonction Softmax est utilisée en couche de sortie pour classifier les types de défauts détectés dans les images routières. Par exemple, les classes peuvent être :

- Fissure longitudinale
- Fissure transversale
- Nids-de-poule

À partir des caractéristiques extraites de l'image (features), le réseau attribue une probabilité à chaque type de détérioration. La classe avec la probabilité la plus élevée est alors sélectionnée comme résultat de la classification. Cette approche permet d'automatiser efficacement la surveillance de l'état des routes à grande échelle, notamment à l'aide de caméras embarquées sur des véhicules ou de drones .

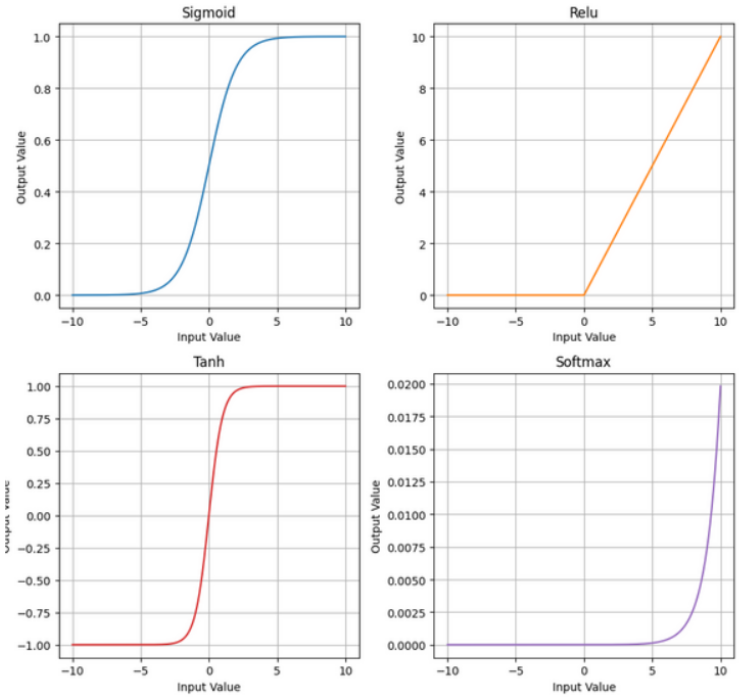


Figure II.9: Graphe des fonctions d'activation [32]

- **Sortie :**

La couche de sortie d'un perceptron multicouche (MLP) produit les prédictions finales en appliquant une fonction d'activation aux données issues de la dernière couche cachée. Le réseau ajuste ensuite les poids afin de minimiser l'erreur entre les prédictions et les cibles, ce qui lui permet d'apprendre des motifs complexes et d'effectuer des prédictions précises sur de nouvelles données [33].

II.6. Conclusion :

Dans ce chapitre, nous avons présenté les principes fondamentaux de l'intelligence artificielle, en mettant particulièrement l'accent sur l'apprentissage automatique (Machine Learning) et l'apprentissage profond (Deep Learning). Cette exploration a permis de comprendre les différences structurelles et fonctionnelles entre ces deux approches, ainsi que le rôle central des réseaux de neurones dans le traitement automatique des données complexes.

Les réseaux profonds, notamment les CNN et RNN, se sont révélés particulièrement adaptés à des tâches de reconnaissance visuelle et d'analyse séquentielle, ouvrant la voie à de nombreuses applications concrètes, dont l'analyse d'images routières.

Cette base théorique nous permet désormais d'aborder plus concrètement les outils de Deep Learning utilisés dans la pratique. Le chapitre suivant sera ainsi consacré aux modèles préentraînés de détection d'objets, qui constituent aujourd'hui des solutions puissantes, rapides et efficaces pour traiter de grands volumes d'images sans nécessiter un apprentissage complet depuis zéro.

Chapitre III. : Les modèles pré-entraînés de DL pour la détection des objets .

III.1. Introduction :

La détection d'objets est aujourd'hui un enjeu majeur dans de nombreux domaines de l'intelligence artificielle, tels que la sécurité, la surveillance, l'agriculture de précision ou encore l'inspection des infrastructures routières. Grâce aux progrès rapides du Deep Learning, il est désormais possible de localiser et d'identifier automatiquement des objets dans une image, avec une précision élevée et en un temps d'exécution réduit.

Dans ce contexte, les modèles pré-entraînés jouent un rôle crucial. En étant initialement entraînés sur de vastes bases de données comme ImageNet, COCO ou Pascal VOC, ces modèles exploitent des connaissances déjà acquises, ce qui permet d'améliorer significativement leurs performances tout en réduisant le temps et les ressources nécessaires à un nouvel entraînement.

Ce chapitre présente les principaux modèles pré-entraînés utilisés pour la détection d'objets, en décrivant leurs caractéristiques, leurs différences architecturales ainsi que leurs domaines d'application. L'objectif est de mettre en lumière les avantages qu'ils offrent dans le cadre de projets de détection automatique, notamment pour l'analyse d'images captées par drone.

III.2. La détection des objet :

La détection d'objets s'est imposée comme l'une des tâches clés en vision par ordinateur et en intelligence artificielle. Son but, apparemment simple, est de permettre à un système informatique de repérer et d'identifier automatiquement les objets présents dans une image ou une vidéo. Cependant, cette tâche présente de nombreux défis techniques, liés notamment à la grande diversité des formes, tailles et contextes visuels des objets détectés [35].

Les chercheurs développent aujourd'hui des modèles capables non seulement de reconnaître les objets avec précision, mais aussi de comprendre leur contexte, leur position et leurs interactions au sein de scènes complexes. L'objectif ultime est d'atteindre, voire de dépasser dans certaines situations, les performances humaines. Le Deep Learning joue un rôle central dans cette évolution, grâce à des architectures puissantes telles que les réseaux de neurones convolutifs (CNN) et les modèles pré-entraînés sur de larges bases de données [36].

Ce chapitre a pour vocation de présenter les modèles les plus répandus pour la détection d'objets, d'en expliquer le fonctionnement, et de mettre en évidence leurs points forts. Nous analyserons également pourquoi ces modèles sont aujourd'hui indispensables dans les projets d'intelligence artificielle modernes, en illustrant leur mise en œuvre à travers des cas concrets, notamment dans le domaine de l'analyse d'images routières.

III.3. Les déférents modèles pré-entraînés :

La vision par ordinateur a profondément transformé notre capacité à interpréter les images numériques, en particulier dans le domaine de la détection d'objets. Depuis l'apparition des premiers modèles tels que R-CNN, les approches ont considérablement évolué, donnant naissance à des algorithmes plus performants et plus rapides, tels que Fast R-CNN, SSD et YOLO (You Only Look Once).

Dans ce chapitre, nous présenterons les principes fondamentaux de la détection d'objets par vision par ordinateur, en étudiant les modèles avancés les plus courants : R-CNN, Fast R-CNN, SSD et YOLO. Un focus particulier sera accordé à YOLOv8, qui s'impose aujourd'hui comme l'un des algorithmes les plus efficaces pour les systèmes embarqués et les applications en temps réel.

La vision par ordinateur a profondément transformé notre capacité à interpréter les images numériques, en particulier dans le domaine de la détection d'objets. Depuis l'apparition des premiers modèles tels que R-CNN, les approches ont considérablement évolué, donnant naissance à des algorithmes plus performants et plus rapides, tels que Fast R-CNN, SSD et YOLO (You Only Look Once).

Dans ce chapitre, nous présenterons les principes fondamentaux de la détection d'objets par vision par ordinateur, en étudiant les modèles avancés les plus courants : R-CNN, Fast R-CNN, SSD et YOLO. Un focus particulier sera accordé à YOLOv8, qui s'impose aujourd'hui comme l'un des algorithmes les plus efficaces pour les systèmes embarqués et les applications en temps réel.

Chaque modèle fera l'objet d'une analyse détaillée de son architecture, de son mode de fonctionnement, ainsi que d'une évaluation critique de ses avantages et limites selon les contextes d'application.

III.4. La détection d'objets en vision par ordinateur

III.4.1. Définition

La détection d'objets est une branche de la vision par ordinateur qui regroupe trois tâches complémentaires : l'identification, la classification et la localisation d'objets dans des images ou des vidéos. Elle consiste à dessiner des cadres de délimitation (bounding boxes) autour des objets détectés et à leur attribuer une étiquette correspondant à leur catégorie.

Les méthodes de détection basées sur les réseaux de neurones convolutifs (CNN) se divisent principalement en deux grandes catégories, illustrées à la figure III.1 : [15]

- **Méthodes en deux étapes (Two-Stage)** : Ces approches proposent d'abord des régions candidates susceptibles de contenir un objet, puis procèdent à leur classification et à la localisation précise.
- **Méthodes en une étape (One-Stage)** : Ces techniques réalisent simultanément la détection et la classification, permettant une exécution plus rapide, adaptée aux applications temps réel.

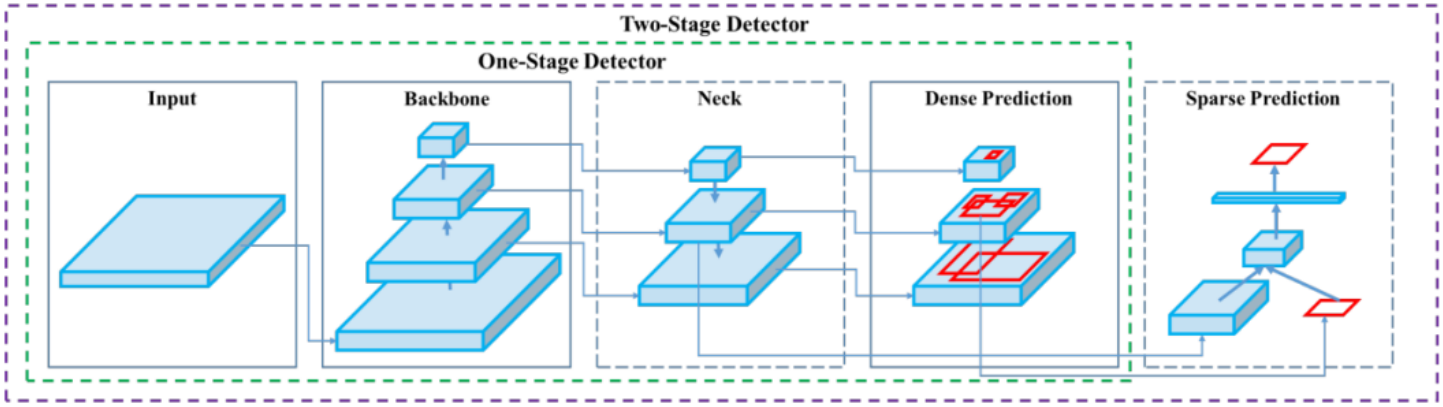


Figure III.1:Architecture générale des détecteurs d'objets basés sur des réseaux de neurones convolutifs a une étape et a deux étapes [16]

III.4.2. Les bases de la détection d'objets

Un réseau de détection d'objets a deux objectifs principaux : repérer les objets présents dans une image et les classer en fonction de leur catégorie. Cette technologie repose sur des concepts et des méthodes clés qui permettent aux systèmes de vision par ordinateur de localiser avec exactitude et

d'identifier des objets spécifiques au sein d'images ou de vidéos. Voici les principes fondamentaux de la détection d'objets :

III.4.2.1. Traitement d'image

Le traitement d'image constitue une phase fondamentale en vision artificielle. Il permet non seulement d'améliorer la qualité visuelle des images, mais aussi de faciliter l'extraction d'informations utiles pour les étapes ultérieures d'analyse. Cette amélioration repose sur diverses opérations telles que le filtrage (réduction du bruit, accentuation des contours), la segmentation (isolement des régions d'intérêt) et la normalisation (uniformisation de l'échelle des données). Ces techniques jouent un rôle clé dans la préparation des images pour une détection ou une classification plus efficace des objets.

III.4.2.1.1. Le filtrage :

Le filtrage en traitement d'images vise à modifier les valeurs des pixels afin d'améliorer la qualité visuelle de l'image ou d'en extraire des caractéristiques particulières (voir figure 3.3). Il englobe plusieurs techniques, telles que le lissage pour atténuer le bruit, la correction du contraste pour ajuster la luminosité, l'accentuation des contours pour renforcer la netteté, ainsi que la détection de bords, étape clé pour préparer l'image à des analyses plus poussées [17].

III.4.2.1.2. La normalisation :

En deep learning appliqué à l'analyse d'images, la normalisation consiste à standardiser les valeurs des pixels en les rapportant à leur valeur maximale possible (255 pour les images en 8 bits, 4095 pour les 12 bits, et 65 535 pour les 16 bits). Ce processus permet d'harmoniser les données en entrée, ce qui accélère la convergence de l'apprentissage, améliore la stabilité des modèles, et facilite la comparaison entre différentes images [18].

III.4.2.1.3. La segmentation :

La segmentation est une étape clé du traitement d'image qui consiste à diviser une image en régions homogènes, chacune correspondant à une zone présentant des caractéristiques visuelles similaires (couleur, intensité, texture, etc.). Cette opération permet d'isoler les objets d'intérêt du reste de l'image, facilitant ainsi leur analyse ou leur détection par les algorithmes d'intelligence artificielle. En vision par ordinateur, la segmentation est souvent utilisée comme prétraitement pour des tâches plus complexes comme la reconnaissance d'objets ou la classification d'images.

III.4.2.2. Techniques d'ancrage et de sélection de régions

III.4.2.2.1. Les régions d'intérêt (RoI) :

Les régions d'intérêt (RoI) désignent des zones spécifiques de l'image susceptibles de contenir des objets. Elles sont générées à l'aide de techniques telles que la recherche sélective ou la segmentation, afin d'identifier des candidats potentiels à la détection. En focalisant l'analyse sur ces régions pertinentes, les Rois permettent de réduire considérablement l'espace de recherche, ce qui améliore l'efficacité des algorithmes de détection. Dans les approches classiques comme Faster R-CNN, les Rois jouent un rôle central : elles sont utilisées pour extraire des caractéristiques locales précises à partir des cartes de convolution. À l'inverse, des modèles plus récents comme YOLO (You Only Look Once) adoptent une stratégie différente : au lieu d'extraire explicitement des RoI, YOLO prédit directement les boîtes englobantes et les classes pour chaque cellule de la grille spatiale de l'image. Cette approche unifiée permet d'obtenir une détection rapide tout en maintenant un compromis efficace entre précision et vitesse.

III.4.2.2.2. Anchor Boxes :

Les boîtes d'ancrage, également appelées *anchor boxes* ou boîtes par défaut, sont des cadres prédéfinis ayant des tailles, des rapports d'aspect et des positions spécifiques, utilisés comme références dans les algorithmes de détection d'objets (voir figure 3.4). Réparties sur une grille couvrant l'image, elles permettent de capturer efficacement des objets de différentes tailles et proportions. Ces boîtes jouent un rôle central dans les processus d'apprentissage et d'inférence : au lieu de prédire directement des boîtes englobantes à partir de zéro, le modèle ajuste les coordonnées des ancres pour les adapter aux objets détectés. Le nombre et les dimensions des boîtes d'ancrage sont généralement définis à partir de l'analyse statistique des objets présents dans le jeu de données d'entraînement. Leur bon paramétrage influe fortement sur la qualité de la détection [19].

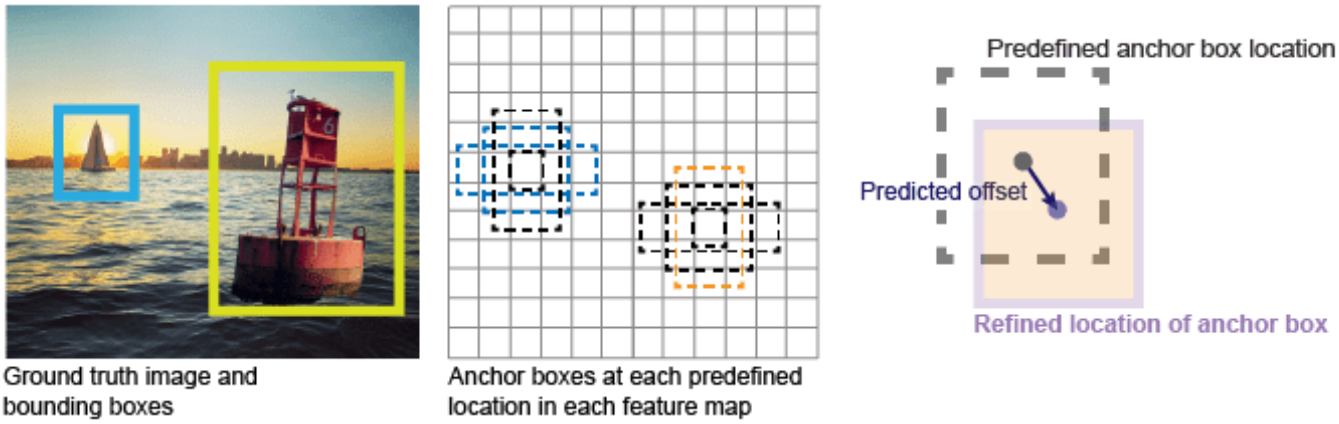


Figure III.2:Anchor boxes [19]

III.4.2.2.3. Boîtes de délimitation (Bounding Boxes)

Dans le cadre de la détection automatique des dégradations de chaussée par Deep Learning, les boîtes de délimitation (*bounding boxes*) jouent un rôle central dans l'annotation et l'évaluation des performances des modèles. Elles consistent en des cadres rectangulaires tracés autour des dégradations visibles sur les images, telles que les fissures, l'arrachement ou le faïençage. Chaque boîte est définie par les coordonnées du coin supérieur gauche (xmin,ymin)(x_{\text{min}}, y_{\text{min}})(xmin,ymin) et du coin inférieur droit (xmax,ymax)(x_{\text{max}}, y_{\text{max}})(xmax,ymax), permettant ainsi une localisation précise des défauts. Ces annotations, appelées aussi *ground truth*, servent de référence pendant l'entraînement du modèle YOLOv8 ainsi que lors de la phase de validation, afin de comparer les prédictions du réseau avec la réalité annotée. La figure III.3 illustre un exemple de boîtes de délimitation appliquées à une image du tronçon routier Sidi Lakhdar – Sidi Ali.

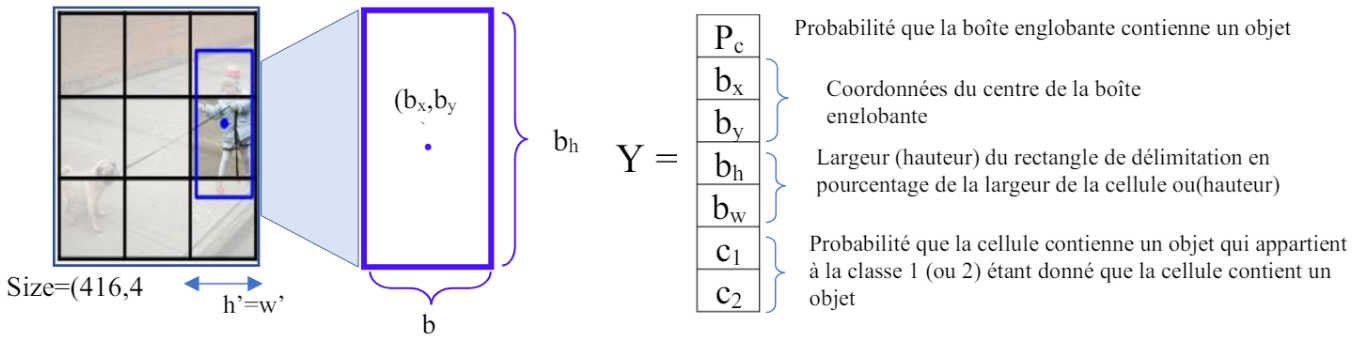


Figure III.3:Le vecteur prédit dans le cas d'une seule boite

III.4.2.2.4. Différence entre Anchor Boxes et Bounding Boxes

Les anchor boxes (*boîtes d'ancrage*) et les bounding boxes (*boîtes englobantes*) sont deux concepts liés mais distincts dans la détection d'objets.

- Anchor boxes : ce sont des cadres prédéfinis appliqués à chaque cellule de la grille d'une image. Elles servent de références initiales pendant l'apprentissage et l'inférence. Elles ne sont pas apprises, mais paramétrées ou calculées à partir du jeu de données.
- Bounding boxes : ce sont les boîtes finales prédites par le modèle. Elles représentent les cadres les plus adaptés aux objets détectés dans l'image. Le modèle prédit des ajustements (décalage, redimensionnement) par rapport aux anchor boxes pour produire les bounding boxes.

Autrement dit, les anchor boxes sont les points de départ de la prédiction, tandis que les bounding boxes sont les résultats finaux après ajustement.

III.4.2.2.5. Suppression non maximale (Non-Maximum Suppression – NMS)

La suppression non maximale (NMS) constitue une étape essentielle dans les algorithmes de détection d'objets. Elle vise à conserver, pour chaque objet détecté, la boîte englobante la plus pertinente, tout en éliminant les boîtes superflues ou redondantes (voir figure III.4). Le principe de fonctionnement de NMS repose sur une procédure itérative : l'algorithme commence par sélectionner la boîte ayant le score de confiance le plus élevé, puis il calcule les chevauchements (généralement via l'IoU – Intersection over Union) entre cette boîte et toutes les autres. Les boîtes présentant un chevauchement supérieur à un seuil prédéfini sont alors supprimées. Ce processus est répété jusqu'à ce qu'il ne reste que les détections les plus significatives, garantissant ainsi une meilleure précision et une clarté dans la localisation des objets détectés dans l'image [20].

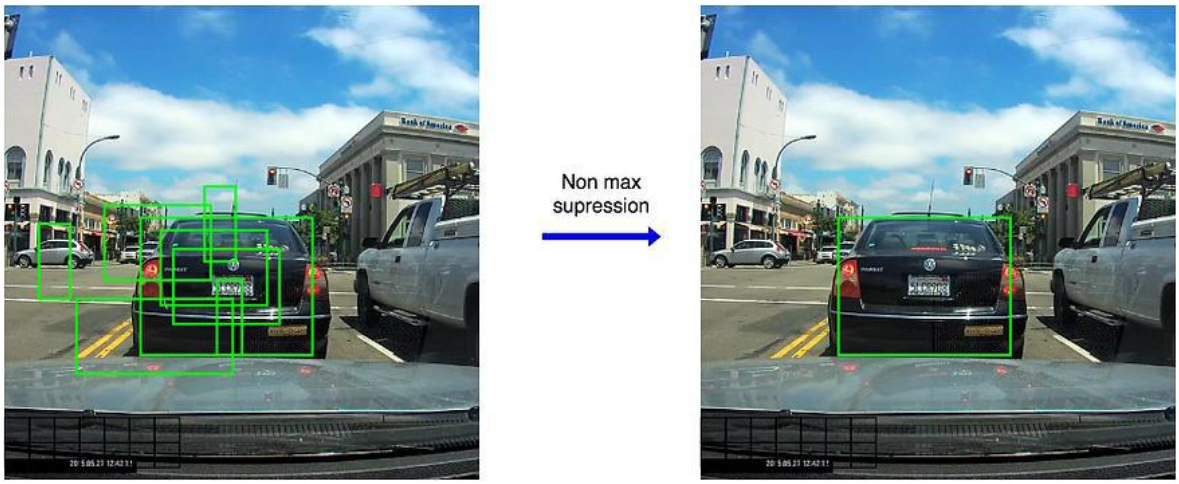


Figure III.4:Application de la suppression non maximale (Non Max Suppression) pour la détection d'objet

III.4.2.2.6. Intersection over Union (IoU)

L'Intersection over Union (IoU), également connue sous le nom d'indice de Jaccard, constitue une mesure standard pour évaluer la qualité de la localisation des objets dans les systèmes de détection. Elle se calcule comme le rapport entre l'aire de l'intersection et l'aire de l'union des deux boîtes de délimitation : celle prédite par le modèle et celle annotée comme référence (vérité terrain), comme illustré dans la figure ci-dessous [37].

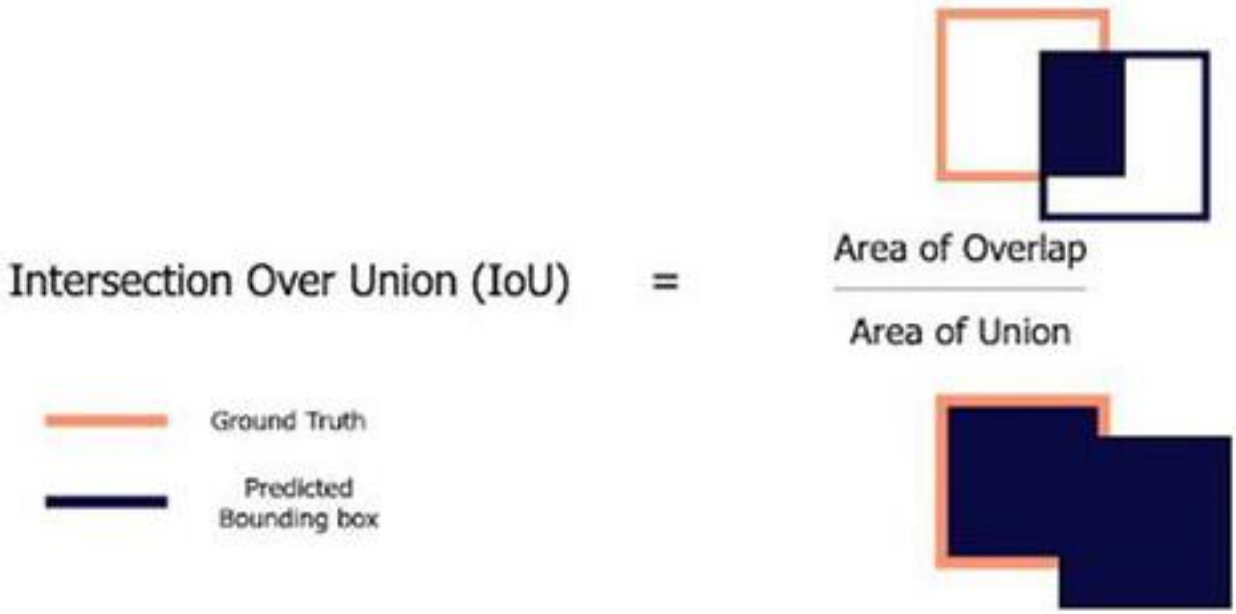


Figure III.5:IoU

La formule mathématique de l'IoU est donnée par l'équation III.1 :

$$IoU = \frac{A_{intersection}}{A_{union}} = \frac{|B_{pred} \cap B_{gt}|}{|B_{pred} \cup B_{gt}|} \quad III.1$$

où :

- Boîte_pred : la boîte prédite par le modèle
- Boîte_gt : la boîte de vérité terrain (ground truth)
- ∩ représente la zone d'intersection
- ∪ représente la zone d'union

Une valeur d'IoU proche de 1 traduit une forte superposition entre les deux boîtes, indiquant ainsi une détection très précise. Inversement, une faible valeur signale une mauvaise correspondance. Cette métrique est largement utilisée pour évaluer la performance des algorithmes de détection d'objets, notamment dans le calcul d'indicateurs tels que la précision moyenne (mAP) [37].

a. **Zone d'intersection** : correspond à la région commune occupée simultanément par la boîte prédite et la boîte de vérité terrain. Elle représente la partie où les deux détections se superposent.

b. **Aire d'union** : désigne la surface totale englobant à la fois la boîte prédite et la boîte de vérité terrain, sans compter deux fois la zone d'intersection. Elle reflète l'étendue combinée des deux boîtes.

La formule mathématique pour l'Intersection over Union (IoU) est donnée par l'équation III.2:

$$IoU = \frac{TP}{(TP+FP+FN)} \quad \text{III.2}$$

Cette formule produit des valeurs d'IoU allant de 0 à 1, où 0 signifie aucun chevauchement et 1 signifie une correspondance parfaite entre la boîte prédite et la boîte de référence (voir figure 3.8).

La formule mathématique permettant de calculer l'Intersection over Union (IoU) est présentée par l'équation (III.2) :

$$IoU = \frac{TP}{(TP+FP+FN)} \quad \text{III.2}$$

où :

- TP (*True Positives*) : nombre de détections correctes,
- FP (*False Positives*) : nombre de fausses détections,
- FN (*False Negatives*) : nombre de cibles non détectées.

Cette équation permet d'obtenir des valeurs comprises entre 0 et 1 :

- une valeur IoU = 0 indique aucun chevauchement entre la boîte prédite et la boîte de vérité terrain,
- une valeur IoU = 1 correspond à une superposition parfaite entre les deux boîtes.

Une illustration de ce principe est fournie dans la figure 3.8.

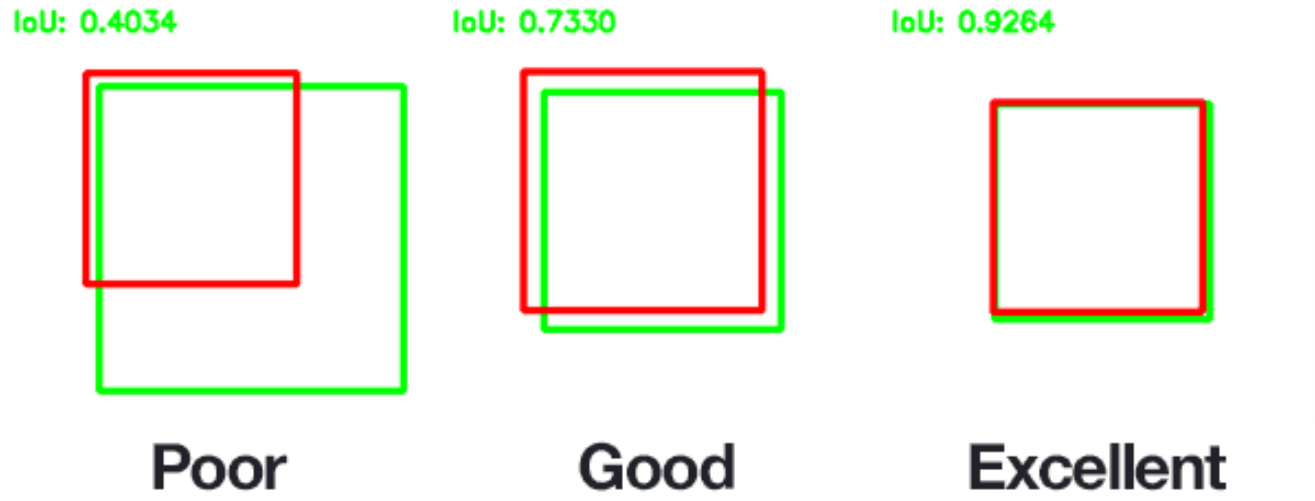


Figure III.6:Un exemple de calcul d'intersection sur des unions pour différentes boîtes englobantes.

III.5. Panorama des algorithmes de détection d'objets : des modèles R-CNN à YOLOv8

III.5.1. R-CNN (Region-based Convolutional Neural Network)

Proposé en 2014, R-CNN a marqué une étape importante dans la détection d'objets. Il repose sur la génération de plusieurs régions candidates (RoI) à partir d'une image, que l'on soumet ensuite individuellement à un réseau de neurones convolutifs pour en extraire des caractéristiques. Ces

caractéristiques sont ensuite classées par un SVM. Bien que précis, ce modèle est très lourd et lent en raison du traitement séparé de chaque région.

III.5.1.1. Architecture

Le modèle R-CNN, introduit par Ross Girshick en 2014 [32], se compose de trois modules principaux :

- Génération de propositions régionales : Utilisation de méthodes non supervisées comme Selective Search.
- Extraction de caractéristiques : Un réseau CNN extrait un vecteur de caractéristiques à partir de chaque région candidate.
- Classification et localisation : Des SVM linéaires effectuent la classification, tandis qu'un classifieur multiclasse réalise la localisation précise via la régression spatiale.
- Précision élevée grâce au traitement fin des régions candidates.
- Bonne performance sur les bases de données bien annotées comme VOC ou COCO.

III.5.1.2. Inconvénients

- Très lent : extraction indépendante pour chaque région.
- Coût mémoire élevé.
- Complexité du pipeline.

III.5.2. Fast R-CNN (2015)

Introduit peu après R-CNN, Fast R-CNN améliore considérablement l'efficacité du processus. L'image entière est d'abord traitée par un CNN unique pour produire une carte de caractéristiques, puis les RoI sont projetées sur cette carte. Chaque RoI est ensuite convertie en une taille fixe par RoI Pooling, et un classifieur (avec régression de boîte) est appliqué. Fast R-CNN est plus rapide et plus précis que R-CNN, car il évite la redondance du traitement.

III.5.2.1. Présentation

Fast R-CNN, proposé par Ross Girshick en 2015, améliore considérablement les performances de R-CNN en termes de vitesse et de précision. Plutôt que de traiter chaque région candidate séparément, Fast R-CNN effectue une seule propagation de l'image entière à travers un réseau convolutionnel, ce qui rend l'architecture beaucoup plus efficace.

III.5.2.2. Architecture

Le modèle se compose des modules suivants :

- Extraction de caractéristiques partagées : l'image entière passe une seule fois dans un CNN (souvent VGG16).
- RoI Pooling : les régions proposées sont mappées sur la carte de caractéristiques pour obtenir des sous-régions fixes en taille.
- Classification et régression : deux branches en tête : une pour classer les objets, l'autre pour prédire les coordonnées des boîtes englobantes.

III.5.2.3. Fonctionnement

- L'image est d'abord traitée par un CNN pour obtenir une carte de caractéristiques.
- Les régions candidates (RoIs) générées par Selective Search sont projetées sur cette carte.

- Chaque RoI passe par une couche de RoI Pooling, suivie d'un classificateur softmax et d'un régresseur de boîtes.

III.5.2.4. Avantages

- Gain de vitesse important par rapport à R-CNN.
- Entraînement de bout en bout dans un seul réseau.
- Réduction du stockage : plus besoin de sauvegarder les vecteurs de caractéristiques.

III.5.2.5. Inconvénients

- Dépend encore de Selective Search, une méthode externe lente pour générer les RoIs.
- Moins adapté au traitement en temps réel.

III.5.3. Faster R-CNN

Faster R-CNN pousse encore plus loin l'optimisation en intégrant un RPN (Region Proposal Network) qui génère automatiquement les régions d'intérêt. Cela élimine le besoin d'algorithmes externes pour la génération de propositions (comme Selective Search). Le réseau est entièrement entraînable de bout en bout et permet d'atteindre des performances élevées, bien que le temps de traitement reste élevé pour les applications en temps réel.

III.5.3.1. Présentation

Faster R-CNN, proposé également en 2015 par Shaoqing Ren et al., représente une avancée majeure en introduisant un module de proposition de régions intégré au réseau : le Region Proposal Network (RPN). Ce modèle est entièrement convolutionnel et ne dépend plus d'algorithmes externes pour générer les RoIs.

III.5.3.2. Architecture

Elle repose sur trois grandes parties :

- Extraction de caractéristiques : à l'aide d'un CNN partagé (ex. ResNet, VGG).
- Region Proposal Network (RPN) : propose des boîtes candidates à partir des cartes de caractéristiques.
- RoI Pooling + tête de détection : classification + régression de la position des objets.

III.5.3.3. Fonctionnement

- Une seule propagation de l'image dans le CNN produit la carte de caractéristiques.
- Le RPN génère automatiquement les RoIs avec un mécanisme d'ancres (anchors).
- Chaque RoI passe par le RoI Pooling, puis est classé et localisé.

III.5.3.4. Avantages

- Suppression de Selective Search → vitesse fortement augmentée.
- Architecture entièrement end-to-end, efficace et précise.
- Très bons résultats sur VOC, COCO.

III.5.3.5. Inconvénients

- Complexité de l'entraînement : nécessite un entraînement alterné ou multitâche.
- Temps de traitement encore élevé pour les applications en temps réel.

III.5.4. SSD (Single Shot Multibox Detector)

SSD propose une approche radicalement différente. Au lieu de traiter l'image en deux étapes (proposition + classification), il effectue la détection en une seule passe sur l'image complète. Il divise l'image en grilles et prédit directement les classes et les coordonnées des boîtes englobantes. Cette structure rend SSD beaucoup plus rapide, tout en conservant une précision satisfaisante pour de nombreuses applications, notamment mobiles et embarquées.

III.5.4.1. Présentation

Le modèle SSD, proposé par Liu et al. en 2016, adopte une approche radicalement différente : il ne génère pas de propositions de régions, mais prévoit directement les classes et les coordonnées des objets dans un processus unique et rapide. Il est conçu pour la détection en temps réel.

III.5.4.2. Architecture : SSD repose sur :

- Un réseau de base (VGG16 ou MobileNet) pour extraire les caractéristiques.
- Plusieurs couches de prédiction à différentes échelles.
- Filtres convolutifs appliqués directement sur les cartes de caractéristiques pour produire des prédictions de classes et de boîtes.

III.5.4.3. Fonctionnement

- L'image passe dans le CNN pour extraire les caractéristiques.
- Chaque couche prédictive applique des petits filtres (ex. 3x3) pour produire plusieurs détections par cellule.
- Chaque détection est comparée à des ancres (priors) à différentes tailles et ratios.
- Le réseau sort un grand nombre de prédictions, filtrées par Non-Maximum Suppression (NMS).

III.5.4.4. Avantages

- Très rapide : conçu pour des applications temps réel (jusqu'à 59 FPS avec MobileNet).
- Architecture simple, sans modules externes.
- Bonne détection des objets de taille moyenne à grande.

III.5.4.5. Inconvénients

- Moins performant pour les petits objets, notamment sur COCO.
- Moins précis que Faster R-CNN sur les tâches complexes.

III.5.5. Mask R-CNN : un modèle de référence pour la segmentation d'objets

Proposé par Kaiming He et al. en 2017 [38], Mask R-CNN est une extension du modèle Faster R-CNN, spécialement conçue pour réaliser à la fois la détection et la segmentation sémantique d'objets au niveau du pixel. Il introduit une troisième branche dans l'architecture de Faster R-CNN, dédiée à la génération de masques binaires pour chaque objet détecté. L'architecture de Mask R-CNN repose généralement sur un backbone de type ResNet couplé à un FPN (Feature Pyramid Network) pour extraire des caractéristiques multi-échelles. Elle comprend :

- Un RPN (Region Proposal Network) pour proposer des régions candidates ;
- Un module de classification et de régression pour localiser et identifier les objets ;

- Une branche supplémentaire de segmentation, constituée d'un réseau de type Fully Convolutional Network (FCN), permettant de produire un masque de segmentation pour chaque objet identifié.

III.5.5.1. Fonctionnement :

Le fonctionnement global du modèle se déroule en trois étapes :

1. Proposer des régions d'intérêt via le RPN ;
2. Classifier et localiser les objets présents ;
3. Générer un masque de segmentation précis pour chaque instance détectée.

III.5.5.2. Avantage :

Mask R-CNN offre plusieurs avantages notables :

- Il permet une segmentation fine au niveau du pixel ;
- Il affiche une précision spatiale élevée, idéale pour des applications complexes comme l'analyse de fissures, l'imagerie médicale ou la reconnaissance faciale.

III.5.5.3. Inconvénient :

Cependant, ces performances s'accompagnent de certains inconvénients :

- Le modèle est relativement lent, notamment en phase d'inférence, en raison de sa nature à deux étapes ;
- Il requiert des annotations pixel-level, difficiles à obtenir à grande échelle ;
- Il est également gourmand en ressources GPU, ce qui peut limiter son déploiement sur des systèmes embarqués ou à faibles capacités.

Bien que Mask R-CNN représente une solution puissante pour la segmentation d'objets avec une précision élevée au niveau du pixel, ses exigences en termes de calcul et de temps de traitement en limitent l'usage dans des contextes pratiques à grande échelle ou en temps réel, comme l'inspection automatisée des chaussées. Face à ces contraintes, des modèles dits *one-stage* tels que la famille des YOLO (You Only Look Once) se sont imposés comme des alternatives robustes, alliant rapidité, simplicité d'implémentation et bonnes performances de détection. Parmi ces modèles, YOLOv8 représente l'une des versions les plus récentes et les plus performantes, intégrant des capacités avancées de détection et de segmentation tout en maintenant une grande vitesse d'exécution, y compris sur des machines à ressources limitées.

III.5.6. YOLO

III.5.6.1. Définition :

YOLO (*You Only Look Once*) est un algorithme de vision par ordinateur largement reconnu pour ses performances en détection d'objets et en segmentation d'image en temps réel. Développé en 2016 par Joseph Redmon et Ali Farhadi, il a été initialement implémenté dans le framework Darknet. Ce modèle repose sur une architecture à étape unique utilisant un réseau de neurones convolutionnel (CNN) capable de prédire simultanément les boîtes englobantes et les probabilités d'appartenance aux différentes classes d'objets présents dans une image. Cette approche unifiée confère à YOLO une grande rapidité d'exécution ainsi qu'une précision appréciable, ce qui le rend particulièrement adapté aux applications nécessitant une analyse instantanée. Depuis sa création, plusieurs versions et variantes ont été développées, notamment YOLOv1 à YOLOv9, ainsi que YOLO-NAS, YOLO-World, PP-YOLO, et plus récemment YOLOv10. Chacune de ces itérations a

permis d'améliorer les performances en termes de précision, de réduction du temps de traitement et de capacité à détecter des objets de petite taille [39].

III.5.6.2. Architecture Yolo

Dans le cadre de ce travail, le modèle YOLOv8 a été adopté pour la détection automatique des détériorations de chaussée. Ce modèle de type one-stage repose sur un réseau de neurones convolutifs (CNN) profond, conçu pour effectuer simultanément la localisation et la classification des objets dans une seule passe, ce qui le rend à la fois rapide et précis. L'architecture de YOLOv8 est composée d'un backbone profond intégrant environ 24 couches de convolution, permettant une extraction hiérarchique des caractéristiques visuelles à partir des images d'entrée (comme illustré dans la figure III.7). À la différence des architectures classiques, la réduction de la résolution spatiale n'est pas réalisée par des opérations de max-pooling, mais via des convolutions avec stride égal à 2, garantissant une meilleure préservation des informations. Pour optimiser le traitement, le modèle utilise des convolutions 1×1 suivies de convolutions 3×3, ce qui permet de réduire efficacement le nombre de canaux tout en maintenant la richesse des descripteurs.

L'architecture est ensuite prolongée par un neck de type FPN/PAN, qui assure la fusion des caractéristiques à différentes échelles, indispensable pour détecter à la fois les fissures fines et les dégradations étendues. La fonction d'activation Leaky ReLU est appliquée dans toutes les couches intermédiaires pour éviter la désactivation des neurones en cas de valeurs négatives, tandis qu'une fonction linéaire est utilisée en sortie pour prédire les coordonnées des boîtes englobantes. Enfin, bien que le dropout ne soit pas explicitement intégré, YOLOv8 applique diverses stratégies de régularisation (augmentation des données, early stopping, normalisation) pour améliorer la généralisation et éviter le surapprentissage.

Cette architecture moderne et optimisée permet à YOLOv8 de répondre efficacement aux exigences du contexte routier, en détectant avec fiabilité diverses formes de détériorations telles que les fissures, le faïençage, l'arrachement de la couche de roulement ou encore les zones saines. [40]

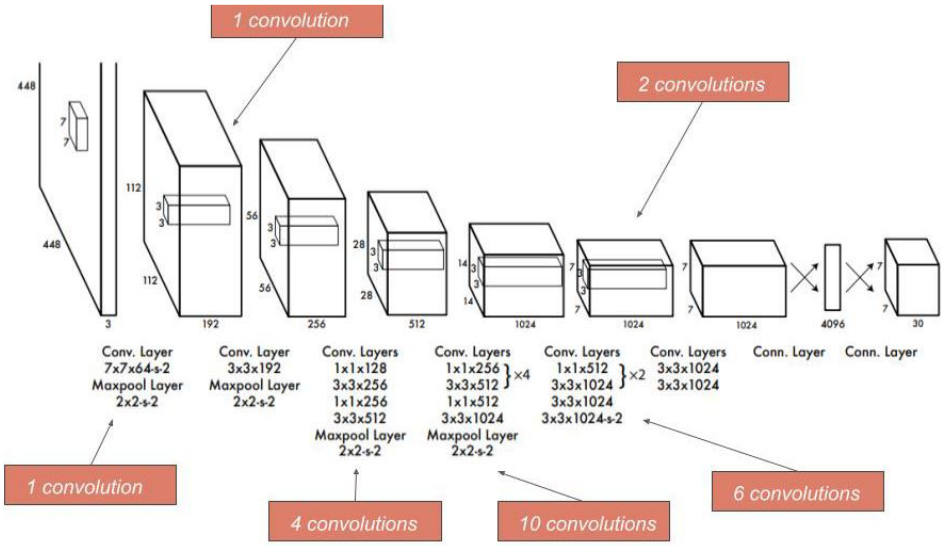


Figure III.7:Architecture YOLO

Dans le cadre des architectures de détection d'objets, une approche couramment adoptée consiste à diviser une image d'entrée de dimensions S×SS \times S en une grille régulière de N×NN \times N

cellules (voir figure III.8). Chacune de ces cellules est chargée d'analyser une petite portion de l'image et de générer des prédictions, comprenant plusieurs boîtes englobantes et un ensemble de probabilités de classes, chacune associée à un score de confiance.

Chaque boîte prédite est définie par cinq paramètres : les coordonnées (x,y)(x, y) du centre de la boîte, exprimées relativement aux limites de la cellule, les dimensions (w,h)(w, h) représentant sa largeur et sa hauteur, et un score de confiance. Ce dernier indique à la fois la probabilité qu'un objet soit effectivement présent dans la boîte et la qualité de sa localisation.

En complément, chaque cellule prédit un unique vecteur de probabilités de classes C, partagé par toutes les boîtes qu'elle génère, quel que soit leur nombre B. Les résultats de l'ensemble de la grille sont regroupés dans un tenseur tridimensionnel de taille N×N×(5B+C) contenant à la fois les paramètres géométriques et les informations de classification.

Pour aboutir à des prédictions cohérentes et sans redondance, une opération de suppression non maximale (Non-Maximum Suppression, NMS) est appliquée. Elle permet d'éliminer les détections multiples d'un même objet, ne conservant que les prédictions les plus fiables.

Ce fonctionnement général, caractéristique des modèles comme YOLO (You Only Look Once), servira de base à la phase suivante de ce travail, consacrée à la conception et à l'implémentation d'un système de détection adapté à notre problématique spécifique.[41]

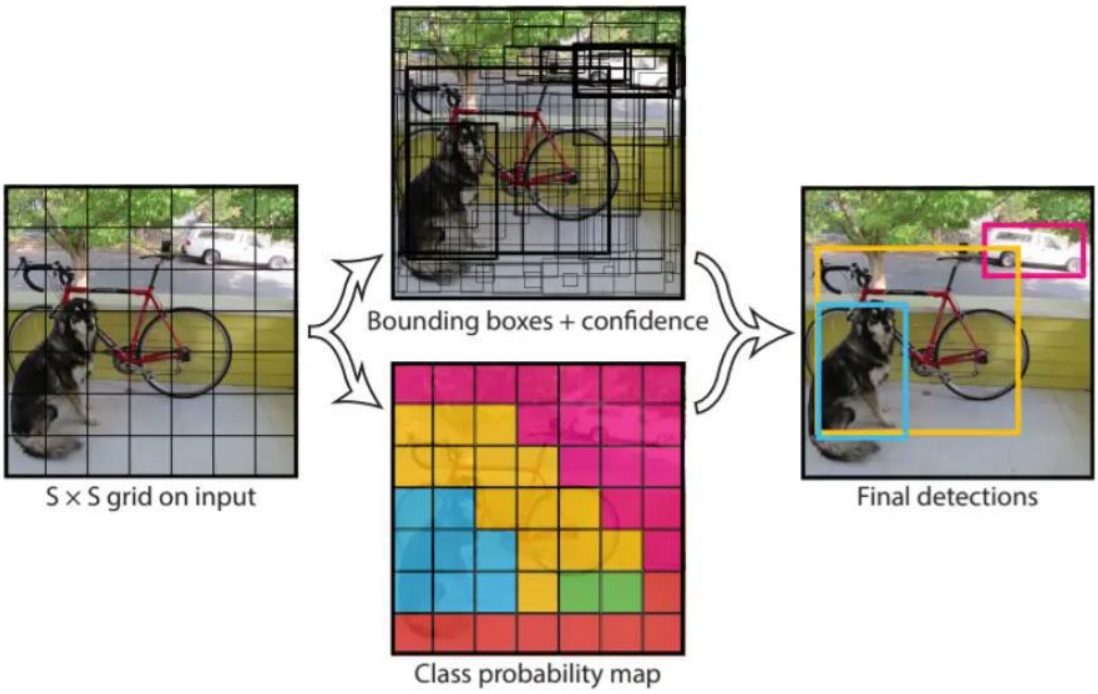


Figure III.8:processus d'identification d'objet avec Yolo [41]

III.5.6.3. Les indicateurs de performance YOLO

L'évaluation des performances des modèles de détection d'objets, tels que ceux basés sur l'architecture YOLO, repose sur un ensemble de métriques standardisées, largement reconnues dans la littérature. Ces indicateurs permettent de quantifier non seulement la précision des prédictions, mais aussi la qualité de la localisation des objets détectés. Ils constituent un outil essentiel pour comparer différents modèles ou configurations et guider les choix d'optimisation au cours du

développement. Parmi les plus couramment utilisés, on retrouve la précision, le rappel (recall), la F1-score, l'Intersection over Union (IoU) ainsi que la moyenne de précision moyenne (mAP), qui sera détaillée dans les sections suivantes [71].

III.5.6.3.1. Matrice de confusion et mesures Précision/Rappel

Dans le cadre de la détection d'objets, **la matrice de confusion** constitue un outil fondamental pour évaluer les performances d'un modèle. Elle permet de quantifier les différents types de prédictions produites par le modèle, en les comparant aux annotations réelles. On distingue quatre cas principaux :

- **Vrai Positif (TP)** : Nombre d'objets correctement détectés, c'est-à-dire ceux pour lesquels une prédiction est faite et qui correspondent à un objet réel (avec une correspondance satisfaisante en position et en classe).
- **Faux Positif (FP)** : Nombre de détections incorrectes, c'est-à-dire des prédictions où le modèle détecte un objet alors qu'il n'en existe pas (ou attribue une mauvaise classe).
- **Faux Négatif (FN)** : Nombre d'objets présents mais non détectés par le modèle.
- **Vrai Négatif (TN)** : Nombre de cas où l'absence d'objet a été correctement identifiée. Dans le contexte de la détection d'objets, cette valeur est peu informative et rarement utilisée, car le fond de l'image contient une infinité de régions sans objet.

À partir de ces mesures, on peut dériver plusieurs indicateurs de performance essentiels :

- **Précision (Precision)** : la formule de la précision est présentée par l'équation III.3 :

$$Précision = \frac{TP}{TP+FP} \quad \text{III.3}$$

Cette métrique mesure la **fiabilité des détections positives**. Elle exprime la proportion de prédictions positives (objets détectés) qui sont réellement correctes. Une précision élevée indique que le modèle fait peu d'erreurs de détection (peu de faux positifs).

- **Rappel (Recall)** exprimé par l'équation III.4 suivante :

$$Rappel = \frac{TP}{TP+FN} \quad \text{III.4}$$

Le rappel mesure la capacité du modèle à **retrouver l'ensemble des objets présents**. Un rappel élevé signifie que peu d'objets ont été oubliés (peu de faux négatifs).

Ces deux métriques sont souvent complémentaires. Dans les systèmes de détection, un compromis entre précision et rappel est recherché, souvent résumé par la **mesure F1**.

- **Mesure F1 (F1-score)** illustré par l'équation III.5

$$F_1 - score = 2 * \frac{Précision * Rappel}{Précision + Rappel} \quad \text{III.5}$$

La mesure F1 est une moyenne harmonique entre la précision et le rappel. Elle permet de trouver un compromis équilibré entre ces deux indicateurs. Cette métrique est particulièrement utile lorsque les données sont déséquilibrées (par exemple, lorsqu'il y a beaucoup plus d'images sans défaut que d'images avec fissures).

III.5.6.3.2. Précision moyenne (Average Precision – AP)

La précision moyenne (AP, *Average Precision*) est une mesure synthétique utilisée pour évaluer les performances d'un modèle de détection d'objets en tenant compte de l'évolution simultanée de la précision et du rappel à différents seuils de confiance. Contrairement à une évaluation à un seuil fixe, l'AP permet de mesurer la qualité globale du modèle sur toute la gamme des prédictions.

a. Courbe Précision–Rappel

Pour construire cette courbe :

- On fait varier le seuil de confiance du modèle (par exemple, de 0 à 1).
- À chaque seuil, on détermine les ensembles de vrais positifs (TP), faux positifs (FP) et faux négatifs (FN).
- À partir de ces valeurs, on calcule la précision et le rappel correspondants.
- En traçant ces couples (rappel, précision), on obtient une courbe Précision–Rappel.

Cette courbe permet de visualiser le compromis entre la capacité du modèle à retrouver les objets (rappel) et sa fiabilité dans les détections (précision).

b. Calcul de l'AP

La précision moyenne (AP) correspond à l'aire sous la courbe Précision–Rappel (*Area Under Curve*, ou AUC) :

$$Ap = \int_0^1 Précision(r) d_r \qquad \text{III.6}$$

Elle peut être interprétée comme la moyenne pondérée des précisions obtenues pour chaque niveau de rappel. Plus la courbe est élevée et « plate », plus l'AP est grande, ce qui traduit un bon équilibre entre précision et rappel.

En pratique, l'AP est souvent calculée de manière approximative à l'aide de méthodes de discrétisation (par interpolation ou par moyennage sur des points de rappel prédéfinis).

▪ Lien avec la matrice de confusion

Chaque point de la courbe Précision–Rappel correspond à une matrice de confusion particulière, calculée pour un seuil donné. Ainsi :

- Une variation du seuil modifie les TP, FP et FN.
- Ces valeurs sont utilisées pour recalculer la précision et le rappel.
- L'ensemble des points obtenus constitue la base du calcul de l'AP.

III.5.6.3.3. Précision moyenne moyenne (mAP)

La précision moyenne moyenne (*mean Average Precision*, ou mAP) est une mesure globale de performance des modèles de détection d'objets. Elle étend la notion d'Average Precision (AP) en l'appliquant à plusieurs classes d'objets. Cette métrique est largement utilisée dans les compétitions et les benchmarks comme PASCAL VOC ou COCO pour évaluer la qualité d'un détecteur.

a. AP pour chaque classe

Pour chaque classe d'objet (par exemple : fissure, faïençage, arrachement, chaussée saine), on effectue les étapes suivantes :

1. Le modèle effectue ses prédictions avec des scores de confiance.
2. Pour chaque classe, on **varie le seuil de confiance** afin d'obtenir plusieurs points de la **courbe précision–rappel**.
3. On **calcule l'AP** comme l'aire sous cette courbe pour la classe considérée.

Ainsi, on obtient une **valeur d'AP pour chaque classe**.

b Calcul de la mAP

Une fois les AP obtenues pour toutes les classes, la **mAP** est calculée comme la moyenne de ces AP, montrée par l'équation III.7 :

$$mAp = \frac{1}{N}\sum_{i=1}^{N} A * P_i \qquad \text{III.7}$$

Où :

N est le nombre total de classes,

Api est la précision moyenne de la classe i.

Plus la mAP est élevée, plus le modèle est performant sur l'ensemble des classes.

Voici une version un peu plus fluide et complète pour ta sous-section sur YOLOv8. J'ai ajouté un peu de contexte et précisé certains points pour rendre la lecture plus claire et professionnelle :

III.6. YOLOv8

III.6.1. Définition

YOLOv8, développé par l'équipe Ultralytics et publié en janvier 2023, est un algorithme de détection d'objets de dernière génération. Contrairement aux méthodes traditionnelles qui analysent une image en plusieurs étapes, YOLOv8 effectue la détection en un seul passage, ce qui lui permet d'identifier et de localiser rapidement les objets présents dans des images ou des vidéos.

Grâce à une architecture optimisée, YOLOv8 offre un excellent compromis entre rapidité d'exécution et précision de détection, ce qui le rend particulièrement adapté aux applications en temps réel (voir figure III.9). Cette version améliore également la robustesse du modèle face aux différentes tailles d'objets et aux conditions variées d'éclairage ou de fond, par rapport à ses prédécesseurs.

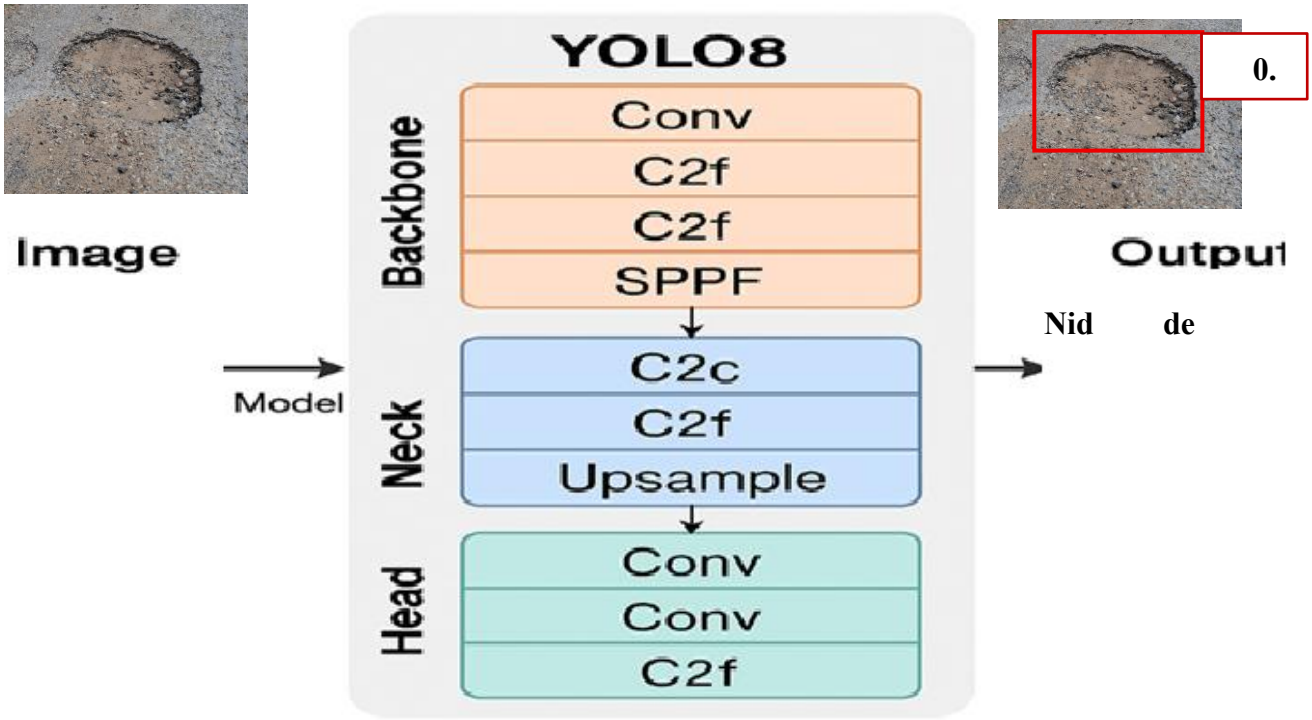


Figure III.9:Détection de détérioration sur image

III.6.2. Les variantes de YOLOv8

Le modèle YOLOv8 est décliné en plusieurs variantes afin de répondre à des exigences diverses en matière de précision et de rapidité. Ces variantes sont désignées par les suffixes *s* (small), *m*

(medium), *l* (large) et *x* (extra large), chacun correspondant à une complexité et une capacité de traitement croissantes. Le choix de la version appropriée dépend du compromis souhaité entre la vitesse d'inférence et la précision de détection. Le tableau III.1 synthétise les performances comparatives de ces différentes variantes, notamment en termes de taille de modèle, de nombre de paramètres, de précision (mAP) et de vitesse d'exécution. En outre, YOLOv8 se distingue par sa grande facilité d'utilisation grâce à son package Python bien documenté, ainsi qu'à son interface en ligne de commande (CLI) intégrée, qui permet une exécution rapide des tâches d'entraînement, de validation et d'inférence. Ces outils rendent YOLOv8 particulièrement accessible aux chercheurs comme aux développeurs. La figure III.10 illustre l'architecture du modèle Yolov8 [42]

Tableau III.1:Performances comparées des variantes YOLOv8

Variante	**Nombre de paramètres (M)**	**Taille du modèle (Mo)**	**mAP@0.5 (%)**	**Vitesse d'inférence (ms/image)**
YOLOv8n (nano)	3.2	5.1	37.3	1.0
YOLOv8s (small)	11.2	22.5	44.9	1.4
YOLOv8m (medium)	25.9	48.2	50.2	2.4
YOLOv8l (large)	43.7	79.3	52.9	3.8
YOLOv8x (xlarge)	68.2	127.5	53.9	6.2

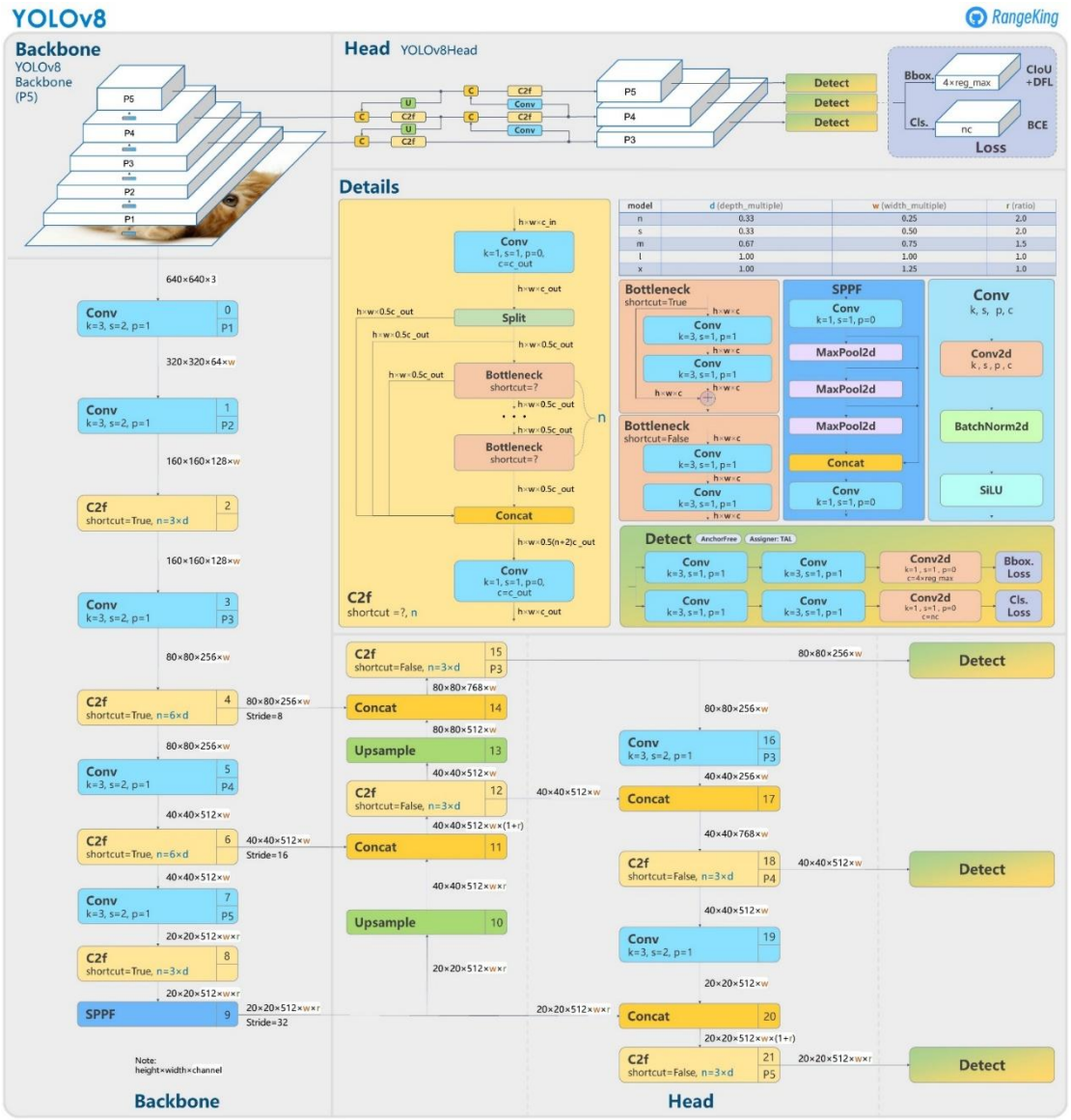


Figure III.10:Architecture du modèle Yolov8

https://github.com/ultralytics/ultralytics/issues/189

III.6.3. Les différents modes de fonctionnement de YOLOv8

Le modèle YOLOv8 (You Only Look Once, version 8), développé par Ultralytics, est un framework de vision par ordinateur polyvalent qui propose plusieurs modes de fonctionnement adaptés à différentes tâches d'analyse d'images. Chacun de ces modes exploite l'architecture de base de YOLO tout en s'ajustant aux spécificités de la tâche, les différents modes sont illustrés dans la figure III.11 ci-dessous :

III.6.3.1. Détection d'objets (Détection) :

Il s'agit du mode le plus couramment utilisé, où le modèle localise et identifie les objets présents dans une image au moyen de boîtes englobantes et de classes associées.

III.6.3.1.1. Classification d'images (Classification) :

Dans ce mode, YOLOv8 attribue une seule étiquette de classe à une image entière sans localisation d'objets. Il est utile pour catégoriser globalement des images (ex : route endommagée vs. route saine).

III.6.3.1.2. Segmentation sémantique (Segmentation) :

Ce mode permet d'obtenir une masque pixel-par-pixel pour chaque objet détecté, offrant une précision bien supérieure à la simple boîte englobante, notamment utile pour l'analyse fine des dégradations.

III.6.3.1.3. Suivi d'objets (Tracking) :

Il combine la détection avec le suivi multi-objets dans des séquences vidéo. Chaque objet détecté est suivi dans le temps grâce à un identifiant unique.

III.6.3.1.4. Estimation de pose (Pose estimation) :

Ce mode permet de détecter des points clés (keypoints) sur les objets, notamment utile pour les applications biométriques ou l'analyse de mouvements.

Chacun de ces modes utilise une architecture optimisée et des têtes spécifiques pour produire des sorties adaptées, tout en s'appuyant sur un même socle entraîné avec des données diverses.

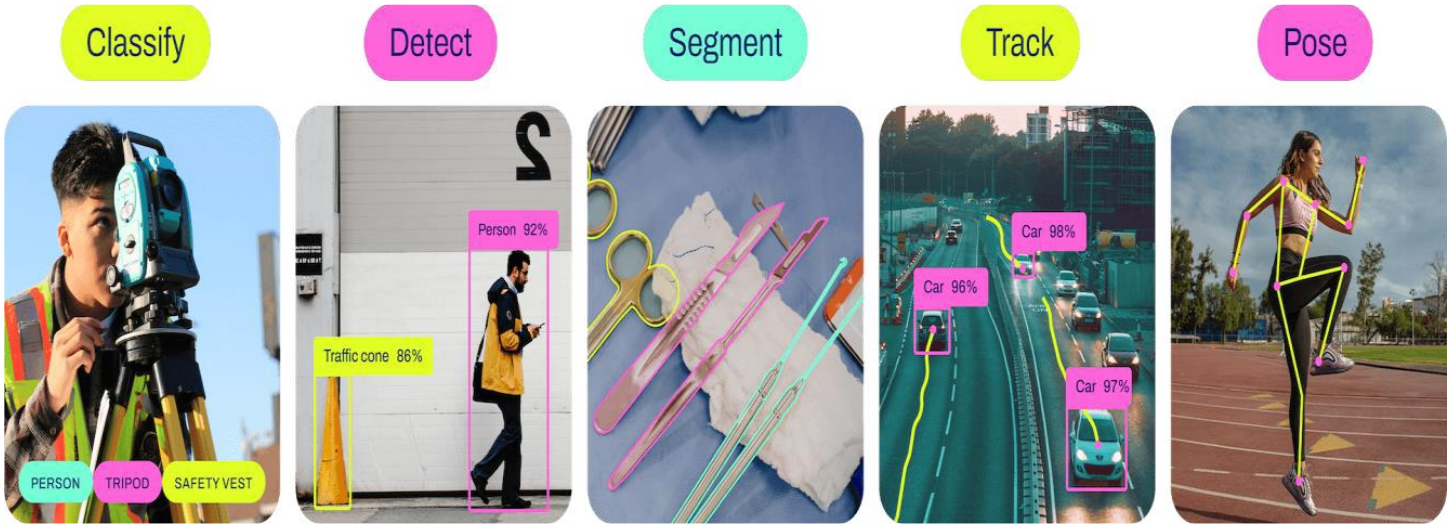


Figure III.11: Les différents modes de fonctionnement de YOLOv8

III.7. L'architecture de YOLOv8

L'architecture de YOLOv8, issue de l'évolution des versions précédentes de l'algorithme YOLO (You Only Look Once) basées sur les réseaux de neurones convolutifs (CNN), repose sur une structure modulaire composée de trois blocs principaux : le **Backbone**, le **Neck** et le **Head** (comme illustré dans la figure III.12) [43].

- Le **Backbone :** est chargé d'extraire les caractéristiques visuelles essentielles à partir des images d'entrée à l'aide de couches convolutives profondes.
- Le **Neck :** joue un rôle de fusion multi-échelle des caractéristiques extraites, permettant ainsi une détection d'objets plus efficace, notamment pour les objets de différentes tailles.
- Enfin, le **Head :** est responsable de la prédiction finale, c'est-à-dire la localisation (boîtes englobantes), la classification des objets détectés, et l'attribution d'un score de confiance.

Cette architecture optimisée permet à YOLOv8 d'atteindre des performances élevées en termes de vitesse et de précision, ce qui en fait un modèle particulièrement adapté aux tâches de détection d'objets en temps réel.

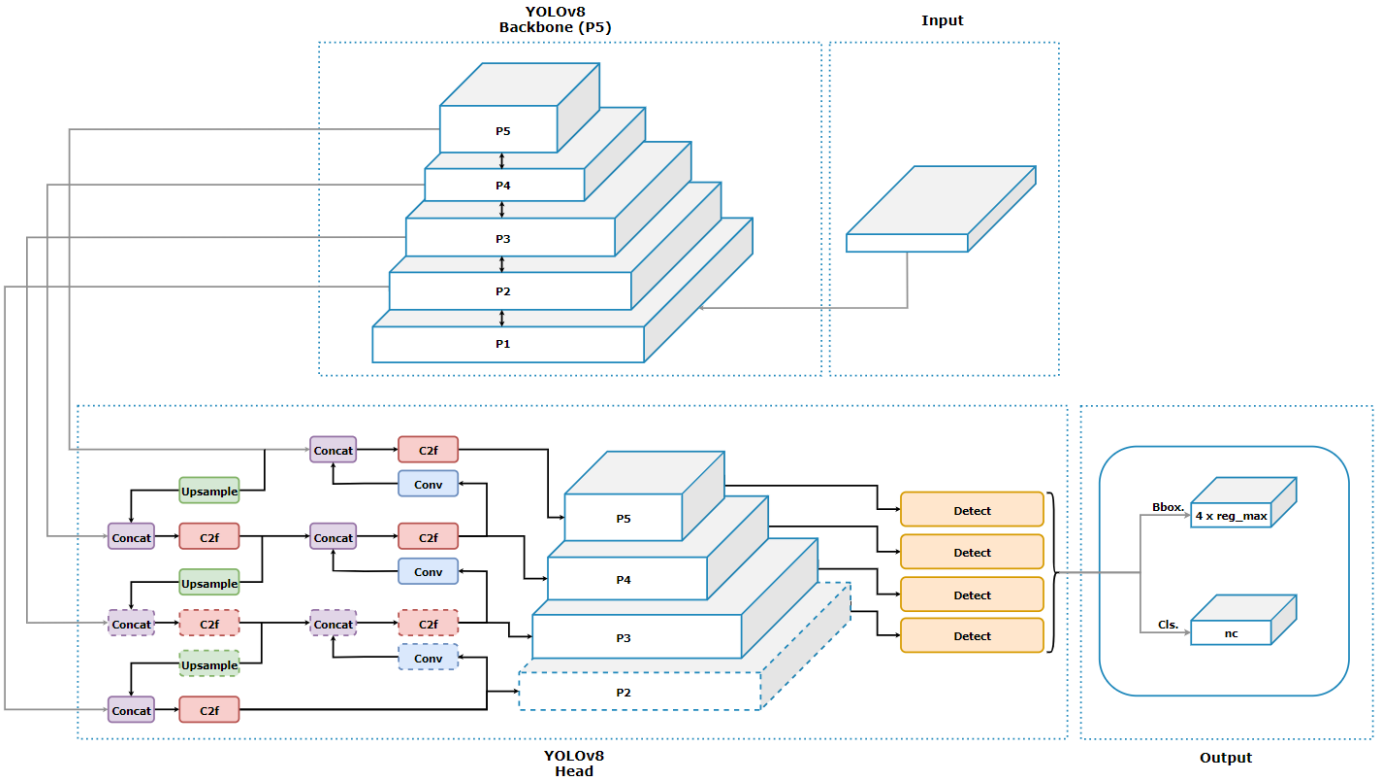


Figure III.12: L'architecture réseau YOLOv8 améliorée comprend un module supplémentaire pour la tête représentée dans le rectangle avec un contour en pointillé [43]

III.8. Comparaison des performances et améliorations entre les versions de YOLO:

Au fil des versions, l'algorithme YOLO (You Only Look Once) a connu de nombreuses améliorations, tant sur le plan de la précision que de la rapidité d'exécution. Chaque version a introduit des innovations architecturales qui visent à optimiser le compromis entre vitesse d'inférence, précision de détection et efficacité computationnelle.

Le tableau III.2 ci-dessous résume les principales évolutions techniques apportées par les différentes versions de YOLO, de YOLOv1 à YOLOv8. Ces améliorations concernent notamment :

- l'introduction de nouvelles couches de traitement (comme CSPNet, PANet ou encore le module SPP) ;
- l'adoption de nouvelles fonctions d'activation (comme Mish dans YOLOv4) ;
- l'optimisation du backbone (Darknet, EfficientNet, etc.) ;
- et plus récemment, l'intégration de techniques d'apprentissage automatique plus avancées (comme les transformers dans YOLOv8).

Ces progrès successifs ont permis d'augmenter de manière significative la mAP (mean Average Precision) tout en réduisant la latence, ce qui rend YOLO toujours plus adapté à des applications en temps réel, notamment dans le domaine de la détection de détériorations routières.

Le modèle YOLOv8 se distingue nettement par rapport à ses prédécesseurs, notamment YOLOv5 et YOLOv7, en offrant de meilleures performances en matière de classification des objets. Cette supériorité se manifeste aussi bien en termes de précision moyenne (mAP) qu'en capacité de généralisation sur des ensembles de données complexes. La figure III.13 ci-dessous illustre cette amélioration en comparant les résultats obtenus par ces différentes versions.

Tableau III.2:Comparaison des versions de YOLO : innovations et ameliorations

Version	**Année**	**Backbone**	**Innovations principales**	**Améliorations en termes de performance**
YOLOv1	2016	Custom CNN	Détection en une seule étape, très rapide	Faible précision, surtout sur petits objets
YOLOv2	2017	Darknet-19	Batch Normalization, Anchor Boxes, résolution d'entrée adaptable	Précision améliorée, mais encore limitée
YOLOv3	2018	Darknet-53	Résidus (skip connections), détection multi-échelle	Meilleure précision sur petits objets
YOLOv4	2020	CSPDarknet-53	CSPNet, Mish, SPP, PANet, data augmentation (Mosaic, DropBlock)	Excellent compromis vitesse/précision
YOLOv5	2020	CSPDarknet (PyTorch)	Implémentation en PyTorch, auto-learning bounding boxes	Facilité d'entraînement, très léger
YOLOv6	2022	EfficientRep	Optimisé pour les edge devices, RepOptimizer	Très rapide avec précision acceptable
YOLOv7	2022	E-ELAN	Trainable Bag-of-Freebies (BoF), CoT attention, nouvelle tête	Précision SOTA sur benchmarks COCO
YOLOv8	2023	C2f + Transformer	Backbone C2f, tête découpée, prise en charge du format ONNX	Meilleure flexibilité, intégration facile, meilleure mAP

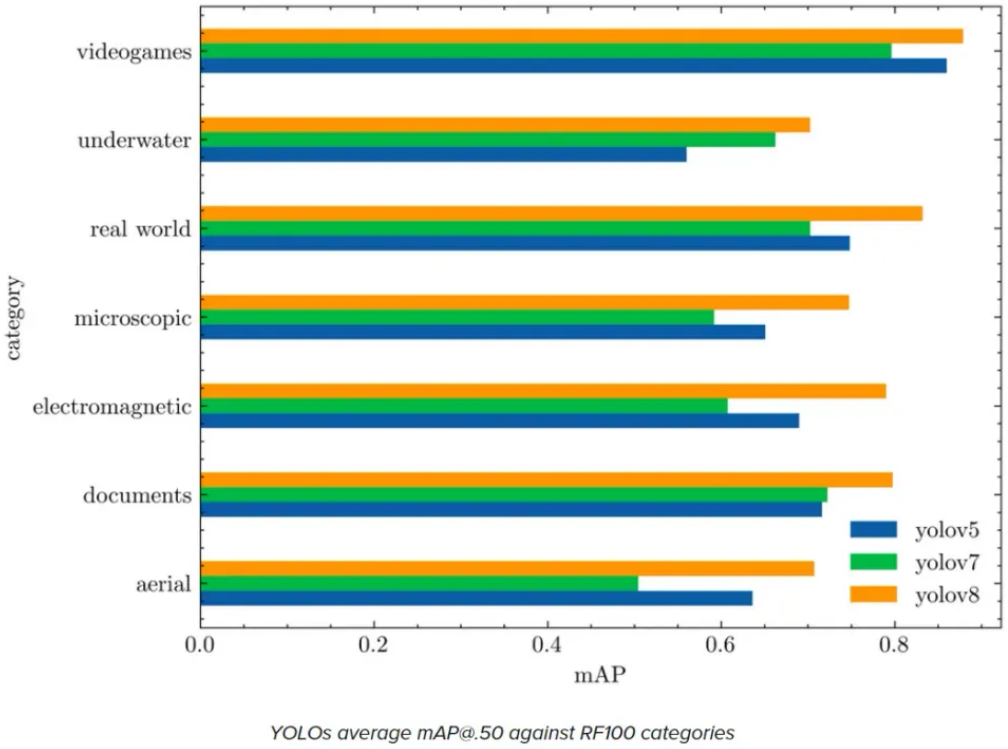


Figure III.13:Moyenne de mAP de YOLO par catégoriesRF100[44]

Le modèle YOLOv8 se distingue non seulement par sa polyvalence, mais également par une série d'innovations techniques qui renforcent considérablement ses performances dans les domaines de la détection d'objets et de la segmentation d'images. Il intègre notamment un nouveau backbone optimisé, une tête de détection sans ancrage (anchor-free), ainsi qu'une fonction de perte améliorée, contribuant à une meilleure précision et stabilité lors de l'apprentissage. De plus, YOLOv8 se caractérise par une grande efficacité computationnelle, capable de s'exécuter aussi bien sur des unités centrales (CPU) que sur des processeurs graphiques (GPU), ce qui le rend adapté à une variété de configurations matérielles, même les plus modestes.

La figure III.14 ci-dessous illustre l'évolution de la précision moyenne (mAP) au fil des versions successives de YOLO développées par Ultralytics, notamment YOLOv5, YOLOv6, YOLOv7 et YOLOv8. Cette représentation met en évidence une amélioration progressive des performances,

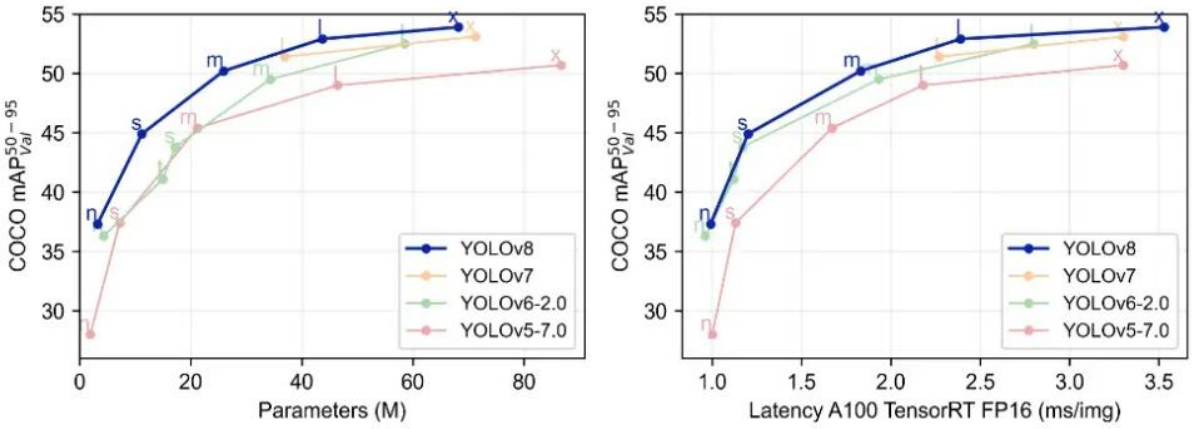


Figure III.14: l'évolution de la précision moyenne (mAP) au fil des versions successives de YOLO

culminant avec YOLOv8, qui atteint la meilleure précision globale parmi toutes les versions testées. Ces résultats confirment les gains constants en termes d'optimisation algorithmique et d'efficacité de détection apportés par chaque itération du modèle. [44]

Figure III.14 : La précision moyenne (mAP) des différentes versions de YOLO [44]

Comme le montre la figure III.14, YOLOv8 surclasse systématiquement ses prédécesseurs, quel que soit le nombre de paramètres ou la latence d'inférence. Il offre ainsi le meilleur équilibre entre précision (mAP-50:95) et performance temps réel, grâce à ses optimisations architecturales. Ces résultats confirment sa pertinence pour des applications exigeantes telles que la détection automatique de défauts routiers.

Le tableau III.3 présente une comparaison des performances entre les premières générations des modèles Ultralytics (tels que YOLOv5 et ses dérivés) et la version plus récente et optimisée YOLOv8, en se basant sur une taille d'image fixée à 640 pixels. Cette comparaison met en évidence les gains en précision et en robustesse apportés par les améliorations architecturales de YOLOv8.

III.9. Analyse comparative des modèles YOLOv5 et YOLOv8

Le tableau III.3 présente une comparaison des performances entre les premières générations des modèles Ultralytics (tels que YOLOv5 et ses dérivés) et la version plus récente et optimisée YOLOv8, en se basant sur une taille d'image fixée à 640 pixels. Cette comparaison met en évidence les gains en précision et en robustesse apportés par les améliorations architecturales de YOLOv8.

Tableau III.3:Comparaison entre la détection d'objet avec YOLOv8 vs YOLOv5 [44]

Taille du modèle	YOLOv5	YOLOv8	Difference
Nano	28	37,3	+33,2%
Small	37,4	44,9	+20,05%
Medium	45,4	50,2	+10,57%
Large	49	52,9	+7,96%
Extra Large	50,7	53,9	+6,31%

Le tableau III.4 met en évidence les performances relatives des différentes tailles de modèles YOLOv8 par rapport à leurs homologues YOLOv5, en termes de **score de détection** (ex. : mAP@0.5 ou une métrique similaire).

On observe une **amélioration systématique** des résultats avec YOLOv8 pour chaque taille de modèle. Les gains les plus significatifs apparaissent dans les versions les plus légères :

- Le modèle **Nano** affiche une amélioration remarquable de **+33,2 %**, traduisant l'optimisation notable de l'architecture YOLOv8 même dans des configurations fortement contraintes.
- Le modèle **Small** présente un gain de **+20,05 %**, ce qui confirme la robustesse de YOLOv8 pour des applications en temps réel avec des ressources limitées.
- À mesure que la taille des modèles augmente (Medium, Large, Extra Large), les gains deviennent plus modérés (**+10,57 %**, **+7,96 %**, puis **+6,31 %**), ce qui s'explique par le fait que YOLOv5 atteint déjà des performances proches du plafond pour les architectures volumineuses.

Globalement, cette comparaison illustre **l'efficacité structurelle de YOLOv8**, notamment pour les modèles compacts, ce qui en fait un choix privilégié pour les systèmes embarqués, les applications mobiles ou les projets où les ressources matérielles sont limitées.

Le tableau III.5 illustre une étude comparative des modèles YOLOv5 et YOLOv8 pour une taille d'image de 640 pixels .

Tableau III.4:Comparaison des performances des modèles YOLOv5 et YOLOv8

Modèle	mAP@0.5	FPS	Taille (Mo)	Paramètres (M)
YOLOv5s	0.375	140	14.5	7.2
YOLOv5m	0.456	100	40.2	21.2
YOLOv5l	0.472	70	89.0	46.5
YOLOv8s	0.510	140	22.5	11.2
YOLOv8m	0.560	100	58.0	25.9

V.10 Les avantages de YOLOv8

YOLOv8 se distingue dans plusieurs domaines majeurs, ce qui en fait un choix privilégié pour la détection d'objets et d'autres tâches de vision par ordinateur :

- Vitesse en temps réel et haute précision : YOLOv8 combine des vitesses d'inférence très rapides avec une excellente précision, ce qui le rend particulièrement adapté aux applications exigeantes telles que les véhicules autonomes, la robotique et la surveillance vidéo.
- Polyvalence et compatibilité matérielle : Il supporte plusieurs types de tâches, notamment la classification, la détection d'objets, la segmentation d'instance et l'estimation de la pose. De plus, il fonctionne efficacement sur une large gamme de matériels, des processeurs classiques aux GPU hautes performances.
- Modèles pré-entraînés accessibles : YOLOv8 propose une large gamme de modèles pré-entraînés, facilitant le transfert d'apprentissage sur divers ensembles de données, ce qui accélère le développement et l'adaptation aux besoins spécifiques.
- Légèreté et efficacité dans l'utilisation des ressources : Conçu pour être compact et optimisé, YOLOv8 peut être déployé sur des dispositifs aux capacités limitées tout en maintenant des performances élevées.
- Projet open source avec une communauté active : Bénéficiant du soutien d'une communauté dynamique, YOLOv8 évolue constamment grâce à un développement collaboratif et l'intégration rapide de nouvelles fonctionnalités.
- Optimisation avancée de l'architecture : Son architecture et ses algorithmes d'entraînement intègrent des techniques avancées d'optimisation, garantissant des performances maximales avec une consommation minimale des ressources.

V.11 Conclusion

Dans ce chapitre, nous avons exploré plusieurs architectures majeures de réseaux de neurones convolutifs dédiés à la détection d'objets, depuis les premiers modèles R-CNN, Fast R-CNN et Faster R-CNN jusqu'aux architectures plus récentes et efficaces telles que SSD, Mask R-CNN, et enfin YOLOv8. Chaque modèle présente ses avantages et limites, en termes de précision, vitesse d'inférence, complexité et capacité à gérer des tâches spécifiques comme la segmentation d'instance.

L'étude approfondie de ces modèles a permis de comprendre leur évolution progressive, marquée par une amélioration constante des performances et une meilleure adaptation aux contraintes pratiques, notamment pour la détection rapide et précise dans des environnements réels. YOLOv8, en particulier, illustre cette convergence vers un compromis optimal entre rapidité, légèreté et précision, ce qui le rend particulièrement adapté à la détection des détériorations sur chaussées dans notre contexte d'étude.

Cette revue détaillée des techniques et modèles de détection jette les bases nécessaires pour la prochaine étape de notre travail : la conception et l'implémentation d'une solution personnalisée visant à détecter efficacement les défauts des chaussées.

Chapitre IV. : Conception et implémentation

IV.1. Introduction :

Dans le cadre de ce projet visant à détecter automatiquement les dégradations des chaussées par des techniques d'intelligence artificielle, la constitution d'une base de données fiable et représentative apparaît comme une étape essentielle. En effet, l'efficacité des modèles de Deep Learning repose sur la qualité, la variété et la précision des données utilisées pour leur entraînement.

Ce chapitre présente de manière détaillée l'ensemble du processus mis en œuvre pour constituer cette base de données. Il couvre :

- La sélection du tronçon routier ,
- La prise de vue aérienne via drone ,
- Le traitement vidéo et l'extraction des images ,
- L'annotation fine des pathologies ,
- Et enfin la structuration finale de la base de données au format compatible avec YOLOv8.

IV.2. L'élaboration de la base de données

IV.2.1. Choix du tronçon routier :

IV.2.1.1. Localisation du tronçon :

Le tronçon sélectionné s'étend sur une distance environ 8 kilomètres , CW42 entre Sidi Ali et Sidi Lakhdar , dans la wilaya de Mostaganem . Ce segment a été choisi après inspection terrain, en raison de son état général très dégradé, offrant une grande diversité de pathologies de surface.

On peut visualiser la localisation dans la figure ci-après:

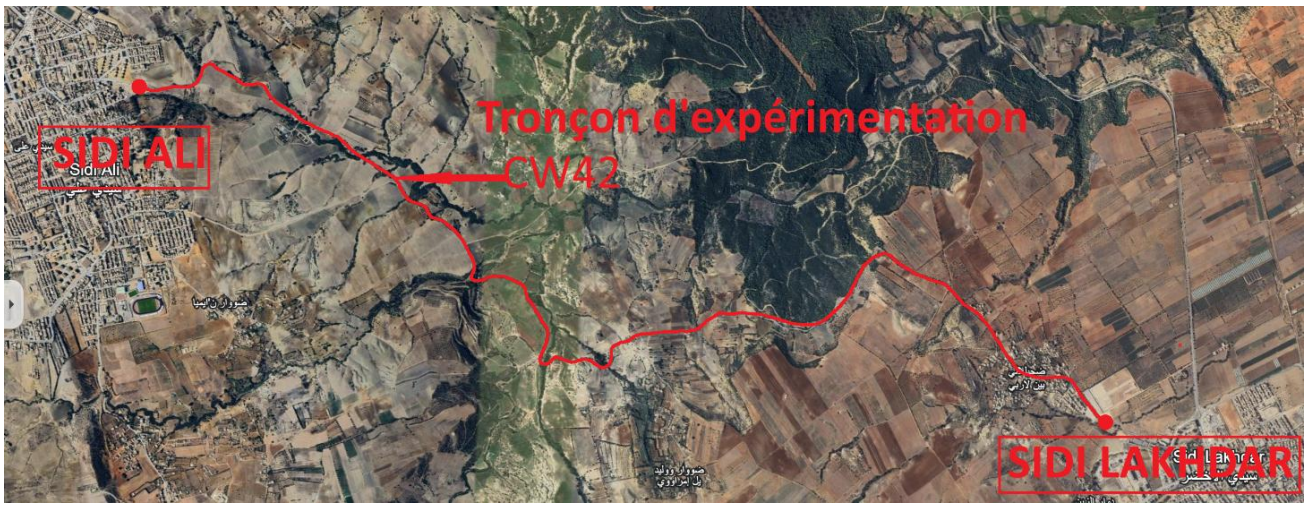


Figure IV.1: localisation du tronçon Sidi Ali - Sidi Lakhdar (Google Earth)

IV.2.1.2. Critères de sélection :

Plusieurs raisons ont motivé ce choix :

- Présence de différents types de dégradations: fissures, faïençage, arrachement.

- Chaussée presque continuellement endommagée, ce qui permet de capturer un grand nombre d'exemples utiles.

- Faible densité de trafic en dehors des heures de pointe, facilitant les prises de vue sans perturbation.

IV.2.2. Acquisition des données:

IV.2.2.1. Conditions de prise de vue :

La campagne de prise de vue a eu lieu le 25 février 2025 à 14h35 , moment choisi pour éviter les périodes de trafic intense. Deux vidéo ont été capturées l'un est à l'aide d'un drone DJI Mavic Air 2, volant à une altitude de 10 à 15 mètres et à une vitesse moyenne de 35 km/h, et l'autre avec un smartphone embarqué sur une véhicule. Ces paramètres ont été retenus pour garantir une bonne couverture visuelle du tronçon tout en maintenant une qualité d'image suffisante pour une exploitation ultérieure. la figure IV.2 suivante montre une photos de drone le jour de l'expérimentation:

Figure IV.2: image de drone et de smartphone embarqué prise le jour d'expérimentation (par l'auteur)

IV.2.2.2. Caractéristiques du drone utilisé :

Nous avons utilisé un drone **DJI Mavic Air 2**, équipé d'un capteur CMOS 1/2 pouce capable de filmer en 4K à 60 IPS, avec une ouverture f/2.8 et un champ de vision de 84°. Cela permet une capture claire et détaillée de la surface de la route.

a) Caractéristiques de l'aéronef :

Le drone utilisé dans cette étude est le modèle DJI Mavic Air 2 , un quadricoptère léger et compact conçu pour des applications variées allant de la photographie aérienne aux missions de surveillance. Ce drone pèse seulement 570 grammes , ce qui facilite son transport et son utilisation sans nécessiter d'autorisation particulière dans plusieurs pays. Ses dimensions pliées sont de 180 × 97 × 84 mm et atteignent 183 × 253 × 77 mm lorsqu'il est déployé, avec une distance diagonale entre les hélices de 302 mm .La figure IV.3 est une photos de près de drone utilisé

Figure IV.3: drone Mavic air 2 (par l'auteur)

En termes de performance, le Mavic Air 2 offre une autonomie maximale de 34 minutes en vol (sans vent) et peut parcourir une distance maximale de 18,5 km . Il peut voler à une vitesse horizontale maximale de 19 m/s en mode Sport (S), tandis qu'en mode Normal (N), sa vitesse est limitée à 12 m/s . En descente rapide, il peut atteindre 5 m/s , mais cette valeur diminue à 3 m/s au-delà de 4500 mètres d'altitude . Le drone est capable d'évoluer jusqu'à une altitude maximale de 5000 mètres au-dessus du niveau de la mer.

Grâce à ses capteurs GPS et GLONASS, associés à une boussole simple et un IMU (unité de mesure inertielle), le drone assure une précision de stationnement verticale de ±0,1 m avec le positionnement visuel et ±0,5 m avec le GPS, tandis que la précision horizontale est respectivement de ±0,1 m et ±1,5 m. Il résiste à des vents allant jusqu'à 10,5 m/s (Force du vent : Niveau 5) et dispose d'un système de transmission en double bande de fréquence (2.4 GHz et 5.8 GHz) avec une puissance variant selon les réglementations locales. Le drone intègre également un stockage interne de 8 Go et accepte les cartes microSD jusqu'à 256 Go, formatées en FAT32 ou exFAT[7] .

b) Caractéristiques de la caméra

La caméra embarquée du Mavic Air 2 est équipée d'un capteur CMOS 1/2 pouce permettant de capturer des photos en résolution 12 mégapixels (MP) et 48 MP , avec différentes options comme la rafale, l'exposition bracketing automatique (AEB), la prise de vue timée et le panorama HDR. L'objectif présente un champ de vision (FOV) de 84° , une focale équivalente de 24 mm , une ouverture fixe de f/2.8 et une mise au point possible à partir d'une distance minimale de 1 mètre .

La sensibilité ISO varie de 100 à 6400 selon le mode vidéo ou photo sélectionné. Les vidéos peuvent être enregistrées en 4K Ultra HD à 60 images par seconde (IPS) , en 2.7K ou en Full HD

(FHD) avec des fréquences allant jusqu'à 240 IPS , garantissant une grande qualité d'image même en conditions mouvementées. Le drone supporte les formats vidéo MP4 et MOV , encodés en H.264 ou H.265 , avec un débit binaire maximal de 120 Mbps . Enfin, les profils couleur disponibles incluent D-Cinelike et Normal , offrant une grande flexibilité post-traitement [7].

Figure IV.4: camera de drone (par l'auteur)

IV.2.3. Conversion de la vidéo en images successives :

Dans le cadre de la création d'une base de données pour l'entraînement d'un modèle de détection automatique des dégradations routières, la vidéo enregistrée par le drone a été convertie en une série d'images fixes. Cette étape est essentielle, car les modèles d'apprentissage profond tels que YOLOv8 nécessitent des images individuelles annotées comme données d'entrée, et non des vidéos.

La vidéo utilisée pour cette conversion a une durée totale de 27 minutes et 16 secondes. Pour extraire des images de manière régulière et exploitable, un **s**cript Python utilisant la bibliothèque OpenCV **(**une bibliothèque open-source largement utilisée dans le domaine du traitement d'images et de vidéos une bibliothèque open-source largement utilisée dans le domaine du traitement d'images et de vidéos) a été développé. Le but de ce traitement est d'extraire automatiquement une image toutes les secondes à partir de la vidéo d'origine. Ce paramètre, appelé IPS (Image Par Seconde), a été fixé à **1**, ce qui permet d'obtenir un ensemble d'images suffisamment espacées dans le temps pour éviter la redondance, tout en conservant une bonne représentativité des scènes filmées.

IV.2.3.1. Méthodologie de la conversion:

La vidéo est lue image par image à l'aide de la fonction cv2.VideoCapture(). Le nombre d'images par seconde (FPS) de la vidéo est d'abord récupéré, ce qui permet de déterminer à quel intervalle

extraire les images. L'algorithme parcourt ensuite la vidéo et enregistre chaque image correspondant à un multiple du FPS, garantissant ainsi l'extraction d'une image toutes les secondes.

La figure IV.5 Représente un schéma de la conversion

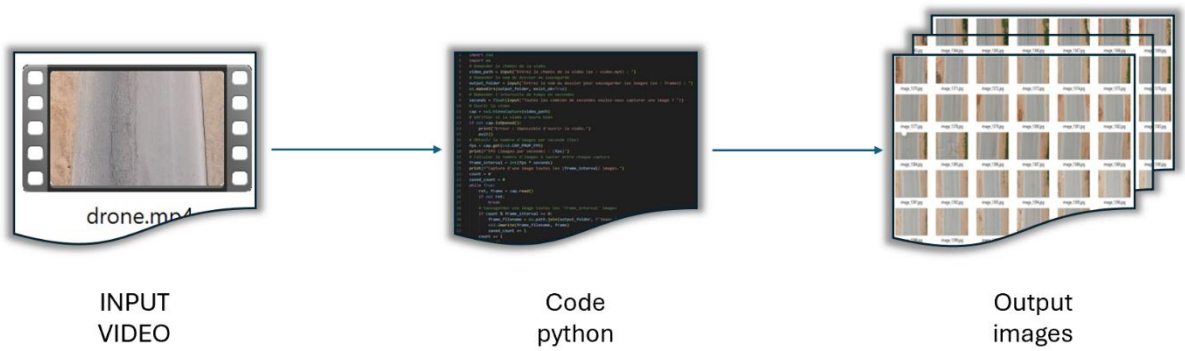


Figure IV.5: schéma de la conversion de la vidéo (par l'auteur)

IV.2.3.2. Résultat de la conversion:

Grâce à cette procédure, environ 1 636 images ont été extraites automatiquement de la vidéo. Ces images constituent un jeu de données varié et équilibré, couvrant l'ensemble des dégradations. Elles seront ensuite utilisées pour l'annotation manuelle.

Après l'obtention des images, il faut les visualiser et les classifier une par une, en éliminant celles qui sont floues ou contiennent trop de bruit, puis les diviser en deux dossiers :

- Dossier1 : train71% et valid 19%
- Dossier2 : test 10%

Comme représenter dans la figure IV.6 ci-desous :

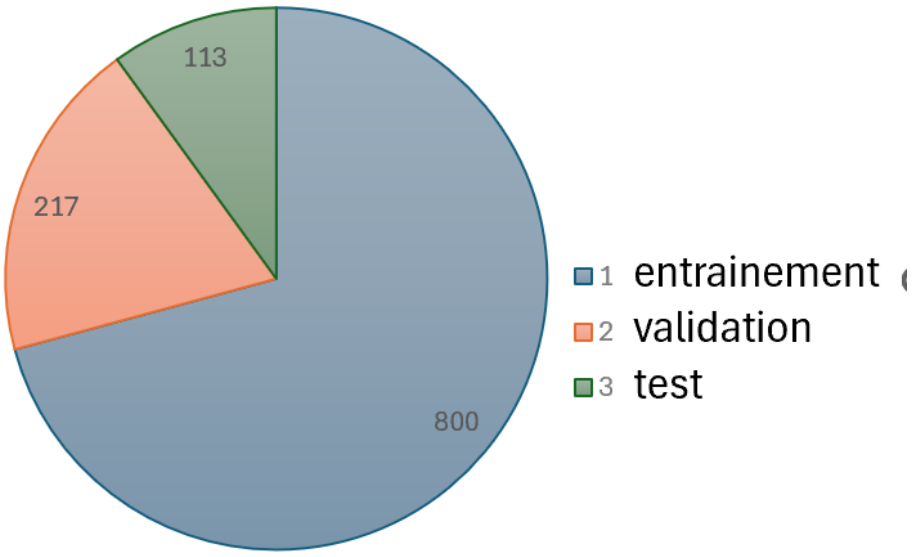


Figure IV.6: diagramme circulaire représente la dévision des images

IV.2.4. Outils d'annotation :

IV.2.4.1. Roboflow:

Roboflow est une plateforme tout-en-un conçue pour faciliter le développement d'applications de vision par ordinateur, quels que soient le niveau d'expérience ou les compétences techniques du développeur.
Elle permet de gérer l'ensemble du cycle de vie d'un projet en vision par ordinateur, de l'annotation des données jusqu'à l'entraînement et au déploiement de modèles d'intelligence artificielle.

Lancée en janvier 2020, Roboflow est née d'un constat pratique : entraîner et déployer un modèle de vision par ordinateur est souvent un processus complexe, répétitif et peu collaboratif. Les développeurs devaient souvent écrire du code redondant pour formater les données, ce qui freinait leur productivité.

Face à ces difficultés, Roboflow a pour mission de supprimer les barrières techniques à l'adoption de la vision par ordinateur. Elle propose un environnement collaboratif, intuitif et performant qui permet aux développeurs d'annoter, préparer, exporter, entraîner et tester leurs modèles avec une grande simplicité.

IV.2.4.2. Makesense.ai :

makesense.ai est un outil en ligne gratuit pour étiqueter les photos. Grâce à l'utilisation d'un navigateur, il ne nécessite aucune installation compliquée : il suffit de visiter le site Web et vous êtes prêt à partir.

Voici l'interface du makesense.ai illustré an la figure IV.6 suivante:

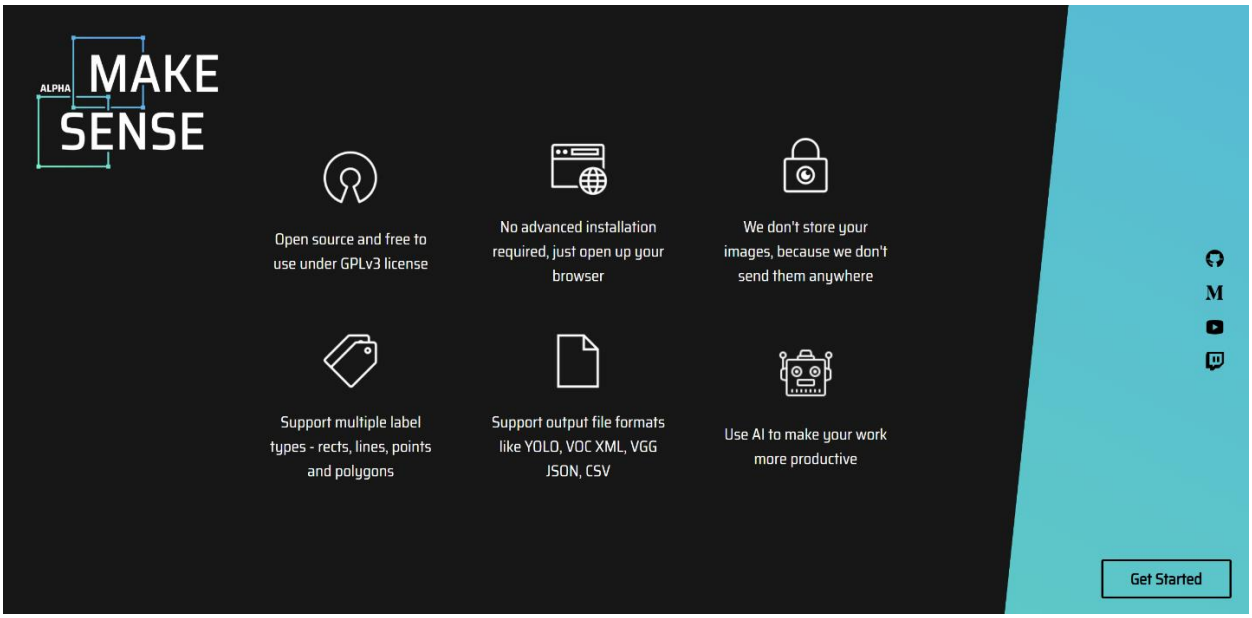


Figure IV.7: l'interface du makesense.ai

IV.2.5. annotation des images :

Dans le cadre de ce travail, un nouveau projet sous le nom "chaussesoupledegradations"

Missing project description a été créé sur la plateforme Roboflow, au sein duquel un ensemble de 1017 images a été importé.

Une phase d'annotation manuelle a ensuite été entreprise, image par image, en se basant sur trois classes de dégradations de chaussée :

- Arrachement,
- Faïençage,
- Fissure.

comme illustré dans la figure suivante:

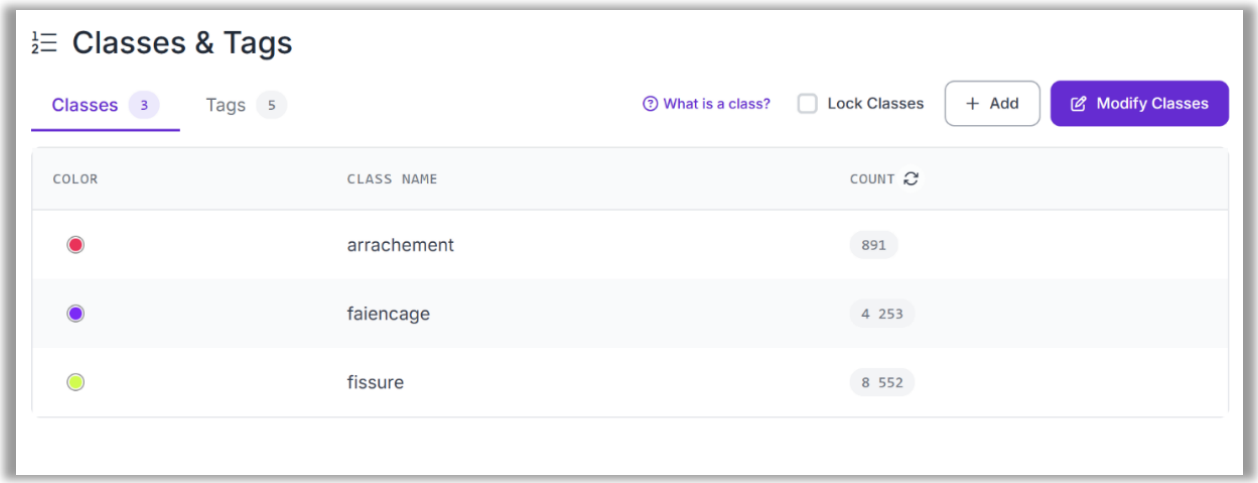


Figure IV.8:les classes de dégradations annotés (par l'auteur)

Pour garantir une annotation précise et fidèle à la réalité des formes dégradées, l'outil polygone a été utilisé, Cela est montré dans la figure IV.10 suivante. Cet outil offre la possibilité de tracer des polygones multi-points, s'ajustant avec finesse aux contours des dégradations observées.

Il est important de souligner que certaines images peuvent contenir simultanément plusieurs types de dégradations, nécessitant une annotation rigoureuse et différenciée pour chacune.

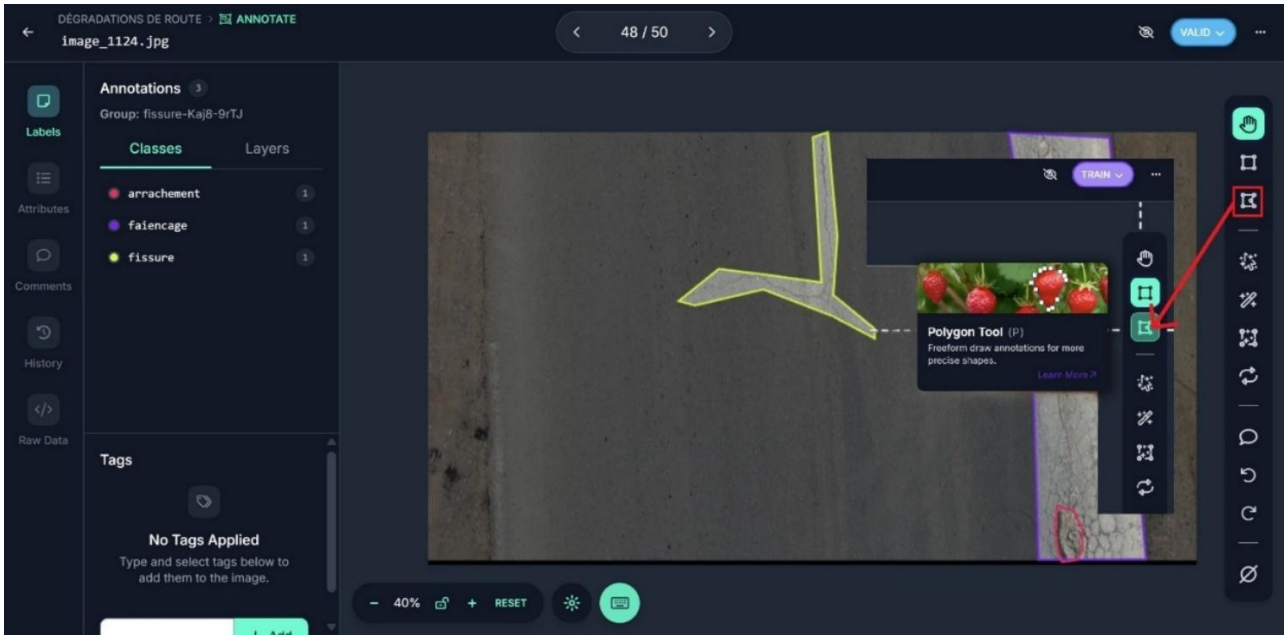


Figure IV.9 : l'outil polygone sur Roboflow (par l'auteur)

IV.2.6. La base de données :

Après l'annotation de tous les images j'ai générer la base de donnes sur roboflow pour être une base de données public avec un ID " degradations-de-route" qui contient 1017 images annotées ensuite j'ai la télécharger format d'annotations txt et YAML utiliser avec YOLOv8.

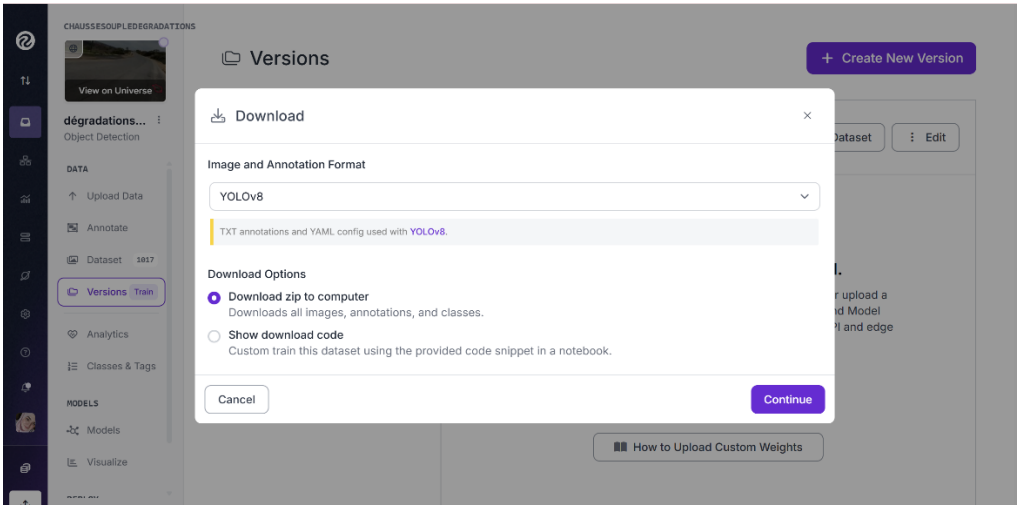


Figure IV.10: format d'annotation téléchargée (par l'auteur)

IV.2.6.1. Le fichier YAML :

Le fichier YAML (Yet Another Markup Language) est un fichier de configuration au format texte utilisé pour décrire la structure du jeu de données dans YOLOv8. Il permet au modèle de savoir où trouver les images, combien de classes sont présentes, et quelles sont leurs étiquettes.

Voici notre fichier illustré dans la figure IV.12 suivante :

```yaml
train: ../train/images
val: ../valid/images
test: ../test/images

nc: 3
names: ['arrachement', 'faiencage', 'fissure']

roboflow:
  workspace: chaussesoupledegradations
  project: degradations-de-route
  version: 1
  license: CC BY 4.0
  url: https://universe.roboflow.com/chaussesoupledegradations/degradations-de-route/dataset/1
```

Figure IV.11: le fichier YAML

IV.3. Application de yolov8:

IV.3.1. Installation de YOLOv8 et préparation de l'environnement:

Afin de mettre en œuvre le modèle de détection d'objets YOLOv8, il a été nécessaire de configurer un environnement de travail compatible avec les bibliothèques d'apprentissage profond. L'environnement a été installé localement sur un ordinateur ASUS ROG ZEPHYRUS équipé de Windows11 avec Python 3.8.

Le modèle YOLOv8 a été téléchargé depuis le dépôt officiel de la bibliothèque Ultralytics, qui fournit une implémentation moderne, optimisée et maintenue de YOLO.

L'installation a été réalisée sur l'invite de commande CMD à l'aide de la commande suivante :

- pip install ultralytics

Cette commande installe automatiquement toutes les bibliothèques nécessaires au bon fonctionnement de YOLOv8, telles que :

- **PyTorch** : bibliothèque de référence pour le Deep Learning, utilisée pour construire et entraîner le modèle.
- **OpenCV** : utilisée pour la manipulation et l'affichage des images.
- **Matplotlib et Seaborn** : pour la visualisation des métriques d'évaluation et des courbes d'apprentissage.
- **tqdm** : pour l'affichage de la progression lors de l'entraînement.

Une fois installée, la bibliothèque Ultralytics met à disposition une interface en ligne de commande (yolo) ainsi qu'une API Python, permettant à l'utilisateur de lancer facilement des tâches telles que l'entraînement (train), la prédiction (predict) ou encore la validation (val) d'un modèle, tout en gardant la possibilité de personnaliser les paramètres (taille des images, nombre d'époques, batch size, etc.).

IV.3.2. L'entrainement:

Dans le cadre de cette étude, une comparaison a été réalisée entre les différentes variantes du modèle YOLOv8, à savoir : YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l et YOLOv8x. Chaque version se distingue par des caractéristiques propres, notamment en termes de taille du modèle, de vitesse de traitement (latence) et de précision (mAP).

IV.3.2.1. YOLOv8n (nano) :

est la version la plus légère de la famille YOLOv8. Elle est optimisée pour une vitesse d'exécution maximale et une faible consommation de ressources, ce qui la rend idéale pour les applications embarquées et les appareils à capacité limitée (ex. : drones, Raspberry Pi). Cependant, elle offre une précision plus faible, ce qui la rend moins adaptée aux tâches nécessitant une détection fine.

Pour lancer l'entrainement on lance la commande suivante :

- yolo task=detect mode=train model=yolov8n.pt data=chemin/vers/dataset.yaml epochs=100 imgsz=640

Cette commande peut être décomposée comme suit :

- yolo : il s'agit de l'appel à la bibliothèque Ultralytics, permettant d'exécuter les différentes tâches liées au modèle YOLO.
- task=detect : indique que la tâche concernée est la détection d'objets.
- mode=train : signifie que l'on souhaite entraîner un nouveau modèle à partir des données fournies.
- model=yolov8n.pt : spécifie le modèle pré-entraîné utilisé comme point de départ. Ici, la lettre n fait référence à la version nano.
- data=.../dataset.yaml : désigne le fichier YAML contenant les informations sur la base de données (chemins vers les images d'entraînement et de validation, nombre de classes, noms des classes).

- epochs=100 : indique que le modèle sera entraîné sur 100 époques, ce qui correspond à 100 passages complets sur l'ensemble d'apprentissage.
- imgsz=640 : précise que la taille des images utilisées pendant l'entraînement est fixée à 640 x 640 pixels, une dimension standard et compatible avec l'architecture YOLO.

IV.3.2.2. YOLOv8s (smal) :

constitue un bon compromis entre rapidité et précision. Moins léger que YOLOv8n mais plus performant, il est souvent utilisé dans les systèmes en temps réel où l'on cherche un équilibre entre qualité de détection et temps d'inférence réduit.

Pour lancer l'entrainement on lance la commande suivante :

- yolo task=detect mode=train model=yolov8s.pt data=chemin/vers/dataset.yaml epochs=100 imgsz=640

Cette commande peut être décomposée comme suit :

- yolo : il s'agit de l'appel à la bibliothèque Ultralytics, permettant d'exécuter les différentes tâches liées au modèle YOLO.
- task=detect : indique que la tâche concernée est la détection d'objets.
- mode=train : signifie que l'on souhaite entraîner un nouveau modèle à partir des données fournies.
- model=yolov8s.pt : spécifie le modèle pré-entraîné utilisé comme point de départ. Ici, la lettre s fait référence à la version smal.
- data=.../dataset.yaml : désigne le fichier YAML contenant les informations sur la base de données (chemins vers les images d'entraînement et de validation, nombre de classes, noms des classes) le même pour tous les versions.
- epochs=100 : indique que le modèle sera entraîné sur 100 époques, ce qui correspond à 100 passages complets sur l'ensemble d'apprentissage le même pour tous les versions.
- imgsz=640 : précise que la taille des images utilisées pendant l'entraînement est fixée à 640 x 640 pixels, une dimension standard et compatible avec l'architecture YOLO la même pour tous les versions.

IV.3.2.3. YOLOv8m (medium) :

est une version intermédiaire, offrant une meilleure capacité d'apprentissage et une précision plus élevée que les versions nano et small. Elle nécessite davantage de ressources matérielles, mais reste adaptée à une utilisation sur des stations de travail ou des plateformes cloud disposant de GPU.

Pour lancer l'entrainement on lance la commande suivante :

- yolo task=detect mode=train model=yolov8m.pt data=chemin/vers/dataset.yaml epochs=100 imgsz=640

Cette commande peut être décomposée comme suit :

- yolo : il s'agit de l'appel à la bibliothèque Ultralytics, permettant d'exécuter les différentes tâches liées au modèle YOLO.
- task=detect : indique que la tâche concernée est la détection d'objets.
- mode=train : signifie que l'on souhaite entraîner un nouveau modèle à partir des données fournies.

- model=yolov8m.pt : spécifie le modèle pré-entraîné utilisé comme point de départ. Ici, la lettre m fait référence à la version medium.
- data=.../dataset.yaml : désigne le fichier YAML contenant les informations sur la base de données (chemins vers les images d'entraînement et de validation, nombre de classes, noms des classes) le même pour tous les versions.
- epochs=100 : indique que le modèle sera entraîné sur 100 époques, ce qui correspond à 100 passages complets sur l'ensemble d'apprentissage le même pour tous les versions.

imgsz=640 : précise que la taille des images utilisées pendant l'entraînement est fixée à 640 x 640 pixels, une dimension standard et compatible avec l'architecture YOLO la même pour tous les versions.

IV.3.2.4. YOLOv8l (large) :

est conçue pour les applications exigeant une détection précise. Elle dispose d'un nombre de paramètres élevé, permettant une meilleure représentation des objets complexes. Toutefois, elle engendre un temps de traitement plus long et requiert des capacités matérielles importantes.

Pour lancer l'entrainement on lance la commande suivante :

- yolo task=detect mode=train model=yolov8l.pt data=chemin/vers/dataset.yaml epochs=100 imgsz=640

Cette commande peut être décomposée comme suit :

- yolo : il s'agit de l'appel à la bibliothèque Ultralytics, permettant d'exécuter les différentes tâches liées au modèle YOLO.
- task=detect : indique que la tâche concernée est la détection d'objets.
- mode=train : signifie que l'on souhaite entraîner un nouveau modèle à partir des données fournies.
- model=yolov8l.pt : spécifie le modèle pré-entraîné utilisé comme point de départ. Ici, la lettre l fait référence à la version large.
- data=.../dataset.yaml : désigne le fichier YAML contenant les informations sur la base de données (chemins vers les images d'entraînement et de validation, nombre de classes, noms des classes) le même pour tous les versions.
- epochs=100 : indique que le modèle sera entraîné sur 100 époques, ce qui correspond à 100 passages complets sur l'ensemble d'apprentissage le même pour tous les versions.

imgsz=640 : précise que la taille des images utilisées pendant l'entraînement est fixée à 640 x 640 pixels, une dimension standard et compatible avec l'architecture YOLO la même pour tous les versions.

IV.3.2.5. YOLOv8x (x-large) :

est la version la plus puissante et la plus précise de la série. Elle est dotée de la plus grande capacité de traitement, avec le plus grand nombre de couches et de paramètres. Cette version est réservée aux systèmes disposant de GPU performants, et elle est utilisée dans les projets où la précision est prioritaire sur la vitesse.

Pour lancer l'entrainement on lance la commande suivante :

- yolo task=detect mode=train model=yolov8x.pt data=chemin/vers/dataset.yaml epochs=100 imgsz=640

Cette commande peut être décomposée comme suit :

- yolo : il s'agit de l'appel à la bibliothèque Ultralytics, permettant d'exécuter les différentes tâches liées au modèle YOLO.
- task=detect : indique que la tâche concernée est la détection d'objets.
- mode=train : signifie que l'on souhaite entraîner un nouveau modèle à partir des données fournies.
- model=yolov8x.pt : spécifie le modèle pré-entraîné utilisé comme point de départ. Ici, la lettre x fait référence à la version x-large.
- data=.../dataset.yaml : désigne le fichier YAML contenant les informations sur la base de données (chemins vers les images d'entraînement et de validation, nombre de classes, noms des classes) le même pour tous les versions.
- epochs=100 : indique que le modèle sera entraîné sur 100 époques, ce qui correspond à 100 passages complets sur l'ensemble d'apprentissage le même pour tous les versions.

imgsz=640 : précise que la taille des images utilisées pendant l'entraînement est fixée à 640 x 640 pixels, une dimension standard et compatible avec l'architecture YOLO la même pour tous les versions.

IV.4. Conclusion :

Dans ce chapitre, nous avons présenté de manière détaillée les différentes étapes méthodologiques mises en œuvre, depuis l'acquisition des images, jusqu'à l'entraînement des modèles. Nous avons décrit le processus de collecte et d'annotation manuelle des images, la génération de la base de données, ainsi que la configuration de l'environnement d'entraînement.

Dans ce chapitre, nous avons présenté de manière détaillée les différentes étapes méthodologiques mises en œuvre, depuis l'acquisition des images, seule les images acquise par drone sont annoter et entrainé, jusqu'à l'entraînement des modèles. Nous avons décrit le processus de collecte et d'annotation manuelle des images, la génération de la base de données, ainsi que la configuration de l'environnement d'entraînement.

Par la suite, les cinq variantes de YOLOv8 (n, s, m, l, x) ont été entraînées séparément, en utilisant le même jeu de données et dans des conditions identiques (nombre d'époques, taille des images, etc.), afin de garantir une comparaison équitable et objective des performances obtenues pour chaque version.

Chapitre V.: Analyse et interprétation des résultats

V.1. Introduction :

Ce chapitre constitue l'aboutissement du travail entrepris dans le cadre de ce projet de fin d'études. Il vise à analyser en profondeur les résultats obtenus par les différents modèles YOLOv8 appliqués à la détection automatique des dégradations de chaussée. À travers une lecture critique des courbes de performance (Precision-Recall, F1-score, Recall, etc.) et des matrices de confusion, nous évaluons la qualité des prédictions et mettons en évidence les points forts ainsi que les limites de chaque variante du modèle.

V.2. Indicateurs de performance utilisés

L'évaluation d'un modèle de détection ne se limite pas à une simple mesure de précision globale. Elle s'appuie sur une panoplie d'indicateurs graphiques permettant une lecture multidimensionnelle de la performance :

V.2.1. Matrice de confusion :

permet d'identifier les erreurs de classification par type (faux positifs, faux négatifs).

V.2.2. Matrice de confusion normalisèe :

permet d'avoir une vue proportionnelle des performances d'un modèle de classification, comme YOLOv8, en exprimant les résultats en pourcentages plutôt qu'en valeurs absolues.

V.2.3. F1-Confidence Curve :

fournit une mesure globale de performance en combinant précision et rappel.

V.2.4. P-Curve (Précision) :

permet de visualiser la fiabilité des prédictions du modèle YOLOv8 selon le niveau de confiance, et d'adapter le seuil de confiance pour obtenir des résultats à la fois précis et utiles. C'est un outil fondamental pour évaluer et calibrer un modèle de détection dans une application pratique comme la surveillance de l'état des routes.

V.2.5. PR-Curve (Precision-Recall) :

illustre le compromis entre la précision et le rappel selon différents seuils.

V.2.6. R-Curve (Rappel) :

montre la capacité du modèle à détecter l'ensemble des objets réels.

V.2.7. mAP@0.5 :

moyenne des précisions pour des IoU supérieures à 0.5 ; indicateur standard de référence.

V.3. Yolov8n:

V.3.1. Matrice de confusion

La figure V.1 illustre la matrice de confusion obtenue lors de l'évaluation du modèle YOLOv8n. Cette matrice permet d'analyser les performances du modèle en termes de classification des différentes classes de dégradations.

- Les **lignes** de la matrice correspondent aux **classes prédites** par le modèle.
- Les **colonnes** représentent les **classes réelles**, c'est-à-dire celles effectivement présentes dans les données de test.
- Chaque **cellule** indique le nombre d'exemples d'une classe réelle spécifique (colonne) que le modèle a classés dans une autre classe (ligne).

- Les **valeurs situées sur la diagonale principale** (de haut en bas) correspondent aux **prédictions correctes** (ou vrais positifs), tandis que les **valeurs en dehors de cette diagonale** traduisent des **erreurs de classification** (confusions entre classes).

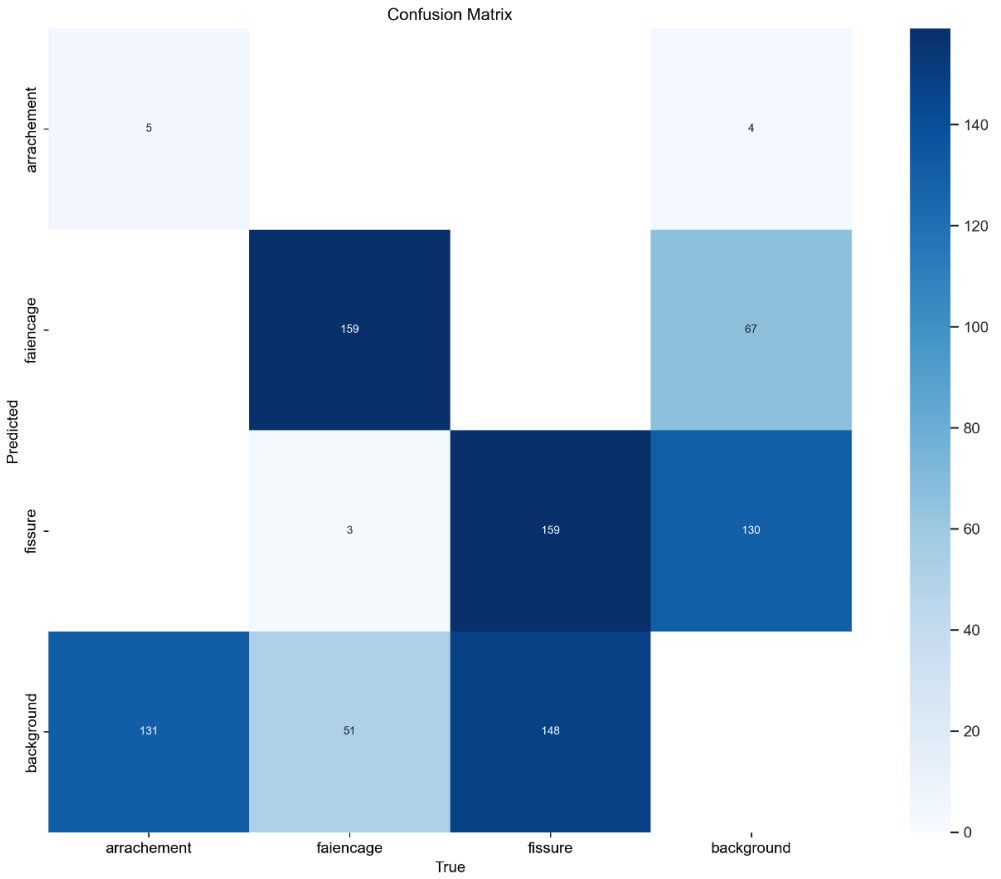


Figure V.1: matrice de confusion de Yolov8n

V.3.1.1. Classes analysées :

Le modèle YOLOv8n a été entraîné pour classifier quatre catégories de données issues d'images aériennes : Arrachement, faïençage, fissure et background (zones sans défaut).

V.3.1.2. Analyse détaillée par classe:

V.3.1.2.1. Classe "arrachement"

a. **Prédictions correctes** : 5 véritables arrachements ont été correctement identifiés par le modèle, ce qui indique une capacité de détection très limitée pour cette classe.

b. **Erreurs de classification** : 131 instances ont été classées à tort comme background, ce qui constitue l'erreur dominante et suggère une forte confusion entre arrachement et fond d'image, peut-être en raison d'une faible présence de cette classe dans les données d'entraînement.

V.3.1.2.2. Classe "faïençage"

a. **Prédictions correctes** : 159 cas ont été correctement identifiés, témoignant d'une très bonne performance pour cette classe.

b. Erreurs de classification :

- 3 instances ont été confondues avec fissure, ce qui reste négligeable.
- 51 instances ont été prédites comme background, ce qui représente une perte partielle de sensibilité.

V.3.1.2.3. Classe "fissure"

a. **Prédictions correctes** : 159 fissures ont été correctement détectées, ce qui montre une bonne précision globale.

b. Erreurs de classification :

- 67 instances ont été confondues avec faïençage, ce qui constitue une erreur notable, révélatrice d'une proximité visuelle entre ces deux types de dégradations.
- 130 instances ont été mal classées comme background, signalant une confusion importante avec l'arrière-plan.

V.3.1.2.4. Classe "background"

a. **Prédictions correctes** : Aucune instance de background n'a été correctement identifiée. Ce résultat souligne une faiblesse critique du modèle, déjà observée dans les autres versions : YOLOv8n ne parvient pas à reconnaître les zones sans dégradation.

b. **Erreurs de classification** :

- 4 instances de background ont été prédites comme arrachement
- 67 comme faïençage
- 148 comme fissure

V.3.2. Matrice de confusion normalisée:

La matrice de confusion normalisée comme illustré dans la figure V.2 représente les performances du modèle YOLOv8n en termes proportionnels plutôt qu'en valeurs absolues. Chaque cellule de la matrice indique la part des instances d'une classe réelle donnée ayant été classées dans une catégorie prédite.

- La valeur (i, j) correspond à la proportion d'instances réellement de la classe j qui ont été prédites comme étant de la classe i.
- Ainsi, la somme des valeurs de chaque colonne est égale à 1 (ou 100 %), car elle traduit la distribution des prédictions effectuées pour une classe réelle spécifique. Les valeurs situées sur la diagonale principale (du haut gauche vers le bas droit) représentent
- le rappel (ou sensibilité) pour chaque classe, c'est-à-dire la capacité du modèle à détecter correctement les instances appartenant à cette classe.
- Les valeurs hors diagonale traduisent les erreurs de classification, c'est-à-dire les cas où le modèle a confondu une classe avec une autre.

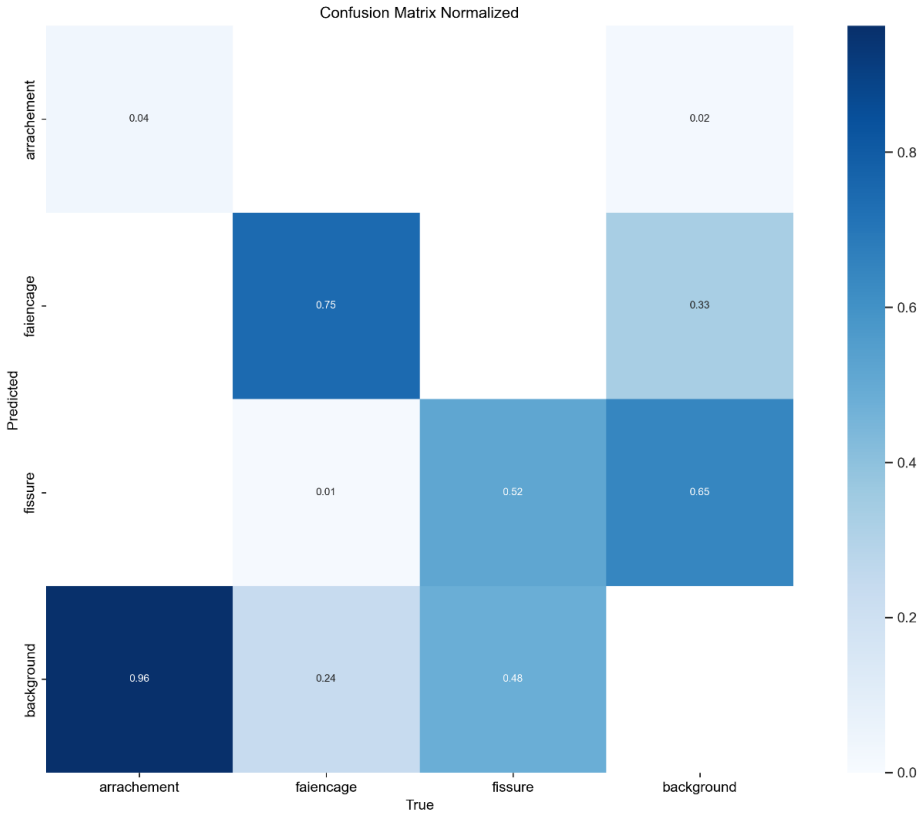


Figure V.2 : Matrice de confusion normalisée Yolov8n

V.3.2.1. Classes évaluées

Le modèle YOLOv8n a été entrainé pour classifier les quatre catégories suivantes : (arrachement, faïençage, fissure et background (zone sans défaut).

V.3.2.2. Analyse des performances (rappel/sensibilité)

V.3.2.2.1. Classe « Arrachement »

a. **Rappel** : 0,04 (soit 4 %) Le modèle ne détecte que 4 % des véritables instances d'arrachement, ce qui constitue un rappel extrêmement faible.

b. **Erreurs de classification** : 96 % des arrachements réels sont classés à tort comme background, ce qui indique que le modèle ne parvient presque jamais à reconnaître cette classe et la confond massivement avec l'arrière-plan.

V.3.2.2.2. Classe "faïençage"

a. **Rappel** : 0,75 (soit 75 %) Le modèle montre ici une bonne capacité de détection, faisant de cette classe la mieux reconnue parmi les défauts.

b. **Erreurs de classification** :

- 1 % des faïençages sont confondus avec fissure (erreur négligeable).
- 24 % sont prédits comme background, ce qui demeure une source significative de perte de rappel.

V.3.2.2.3. Classe "fissure"

a. **Rappel** : 0,52 (soit 52 %) Le modèle parvient à détecter un peu plus de la moitié des véritables fissures, ce qui constitue une performance modérée mais insuffisante pour une détection fiable.

b. **Erreurs de classification** :

- 1 % des fissures sont prédites comme faïençage.
- 48 % sont classées comme background, ce qui traduit une confusion importante avec l'arrière-plan.

V.3.2.2.4. Classe "background"

a. **Rappel** : 0,00 (soit 0 %) Ce résultat souligne une incapacité totale du modèle à reconnaître correctement le fond de l'image.

b. **Erreurs de classification** : 2 % des vrais backgrounds sont prédites comme arrachement, 33 % comme faïençage et 65 % comme fissure. Ces chiffres montrent que le modèle confond massivement les zones neutres avec des défauts, en particulier des fissures, ce qui engendre un grand nombre de faux positifs.

V.3.3. courbe F1-Confiance :

La courbe F1-Confidence illustrée par la figure V.3 ci-dessous, permet d'évaluer la performance du modèle YOLOv8n en fonction du seuil de confiance appliqué aux prédictions. Elle constitue un outil essentiel pour analyser l'équilibre entre précision et rappel à différents niveaux de certitude dans la détection.

L'Axe des abscisses (Confidence) : représente le seuil de confiance appliqué aux prédictions du modèle, allant de 0.0 à 1.0. Plus le seuil est élevé, plus le modèle doit être sûr de sa prédiction pour la valider.

L'Axe des ordonnées (F1-score) : indique le score F1, qui combine la précision (précision des détections) et le rappel (taux de détection des objets réels).

Un F1-score élevé traduit un bon compromis entre éviter les faux positifs et minimiser les oublis.

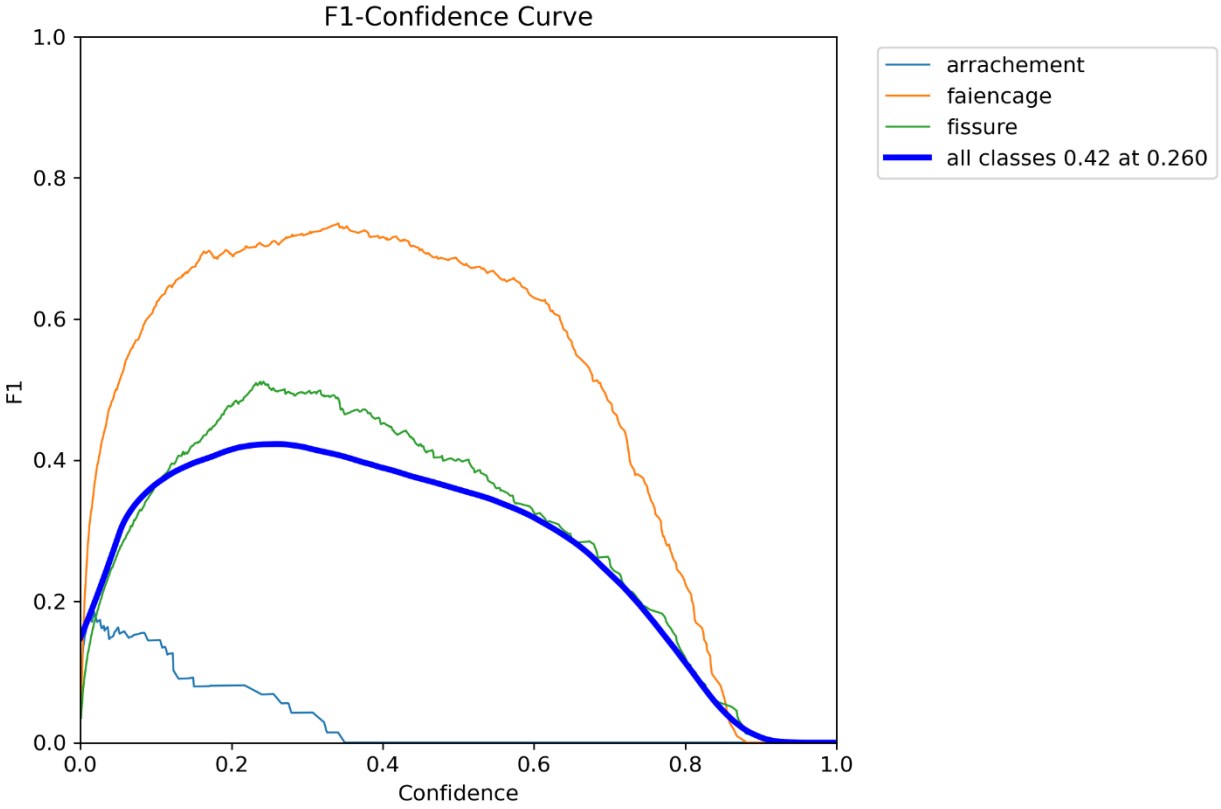


Figure V.3: Courbe F1-Confiance

V.3.3.1. Analyse des courbes par classe

V.3.3.1.1. Classe faïençage (courbe orange)

Cette courbe présente la meilleure performance parmi toutes les classes. Le pic du score F1 atteint environ 0,75 à 0,76, pour un seuil de confiance situé entre 0,2 et 0,4. La stabilité du score F1 sur une large plage de seuils (jusqu'à ~0,6-0,7) démontre que le modèle est fiable, robuste et confiant pour la détection de faïençage.

V.3.3.1.2. Classe fissure (courbe verte)

Elle se classe en deuxième position en termes de performance. Le score F1 maximal est d'environ 0,50 à 0,52, pour un seuil de confiance situé entre 0,2 et 0,4. Bien que moins performant que pour faïençage, ce résultat demeure suffisant pour certaines applications où une détection partielle est tolérable.

V.3.3.1.3. Classe arrachement (courbe bleu clair)

Il s'agit de la classe la moins bien détectée. Le F1-score reste nettement inférieur à 0,2, atteignant un maximum autour de 0,15 à 0,18. La courbe chute rapidement avec l'augmentation du seuil de confiance, ce qui indique que le modèle manque de fiabilité pour cette catégorie, même à faible seuil.

V.3.3.1.4. Moyenne toutes classes (All classes, courbe bleue foncée épaisse)

Cette courbe reflète la performance globale du modèle sur l'ensemble des classes. Le point optimal est atteint à un seuil de confiance de 0,260, avec un score F1 de 0,42. Ce point représente le meilleur compromis entre précision et rappel global sur ce jeu de données.

V.3.4. Courbe de Precision-Confidence (P-curve):

La figure V.4 ci-dessous montre la courbe Précision-Confidence (ou P-curve), qui permet d'évaluer la performance du modèle YOLOv8 en fonction du niveau de confiance associé à ses prédictions. Cette courbe est nécessaire pour comprendre le compromis entre précision et seuil de confiance, et pour déterminer un seuil optimal à partir duquel les prédictions sont fiables.

L'Axe des abscisses (Confidence) : représente le seuil de confiance minimal requis pour qu'une détection soit considérée comme valide. Les valeurs s'échelonnent de 0.0 à 1.0.

L'Axe des ordonnées (Precision) : indique la précision au sens classique :

Une précision élevée reflète une capacité du modèle à limiter les fausses détections.

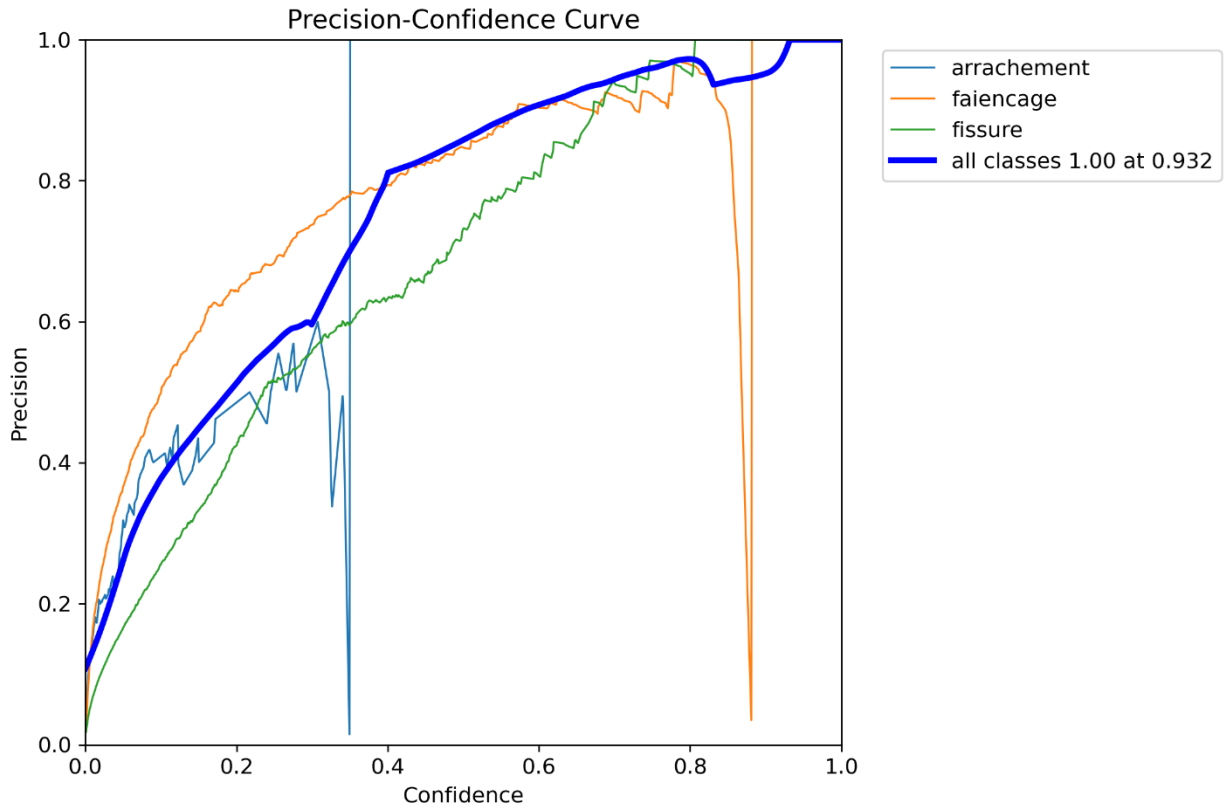


Figure V.4 : Courbe Précision-Confiance (P- curve)Yolov8n

V.3.4.1. Analyse des courbes par classe

V.3.4.1.1. Classe faïençage

Cette courbe affiche la meilleure performance de précision parmi toutes les classes. La précision initiale est très élevée, avoisinant 1.0 pour des seuils de confiance élevés. Elle demeure supérieure à 0.8 sur une grande plage de seuils (jusqu'à environ 0.9), ce qui indique que lorsque le modèle prédit un faïençage, il a très peu de chances de se tromper. Ce comportement est cohérent avec les résultats précédents sur les courbes F1 et les matrices de confusion.

V.3.4.1.2. Classe fissure

La courbe présente également une bonne précision, généralement comprise entre 0.7 et 0.9 sur une large plage de seuils. Cela montre que les prédictions de fissures sont fiables, bien que légèrement moins précises que pour faïençage.

V.3.4.1.3. Classe arrachement

La courbe est très irrégulière et montre des valeurs généralement faibles, qui ne dépassent jamais 0,6. La précision diminue fortement dès que la confiance dépasse 0,35, ce qui indique que le modèle détecte mal cette classe. Les valeurs élevées à de faibles seuils de confiance, ils sont sûrement dus à un petit nombre de détections.

V.3.4.1.4. Courbe globale- toutes classes (épaisse, bleu foncé)

Elle synthétise la précision moyenne du modèle sur l'ensemble des classes. La valeur maximale atteint 1.0 à un seuil de confiance de 0.932. Cela signifie qu'à ce niveau, le modèle ne commet quasiment aucune fausse détection, mais il est alors extrêmement restrictif. Cette situation illustre le compromis classique entre précision et rappel : une précision parfaite s'accompagne souvent d'un taux de détection très faible.

Dans ce cas d'étude, la classe « faïençage » se distingue par une excellente performance, tandis que les classes "fissure" et surtout "arrachage" présentent plus de variabilité. Le seuil optimal situé à 0.932 garantit une précision maximale, critère fondamental dans une application de détection automatique sur infrastructures routières.

V.3.5. La courbe Recall-Confidence :

La courbe Recall–Confidence présentée par la figure V.5 ci-dessus, illustre la capacité du modèle YOLOv8n à détecter correctement les objets pertinents en fonction du seuil de confiance appliqué aux prédictions. Elle permet de quantifier, pour chaque classe, la proportion des instances réelles qui ont été correctement identifiées (rappel) en fonction de la rigueur avec laquelle le modèle accepte ou rejette une détection.

L'Axe des abscisses (Confidence) : représente le seuil minimal requis pour valider une prédiction.

- Un seuil bas permet de retenir davantage de détections, tandis qu'un seuil élevé filtre les prédictions les moins sûres.
- Un rappel élevé signifie que le modèle détecte la majorité des objets réellement présents, même si cela se fait parfois au détriment de la précision.

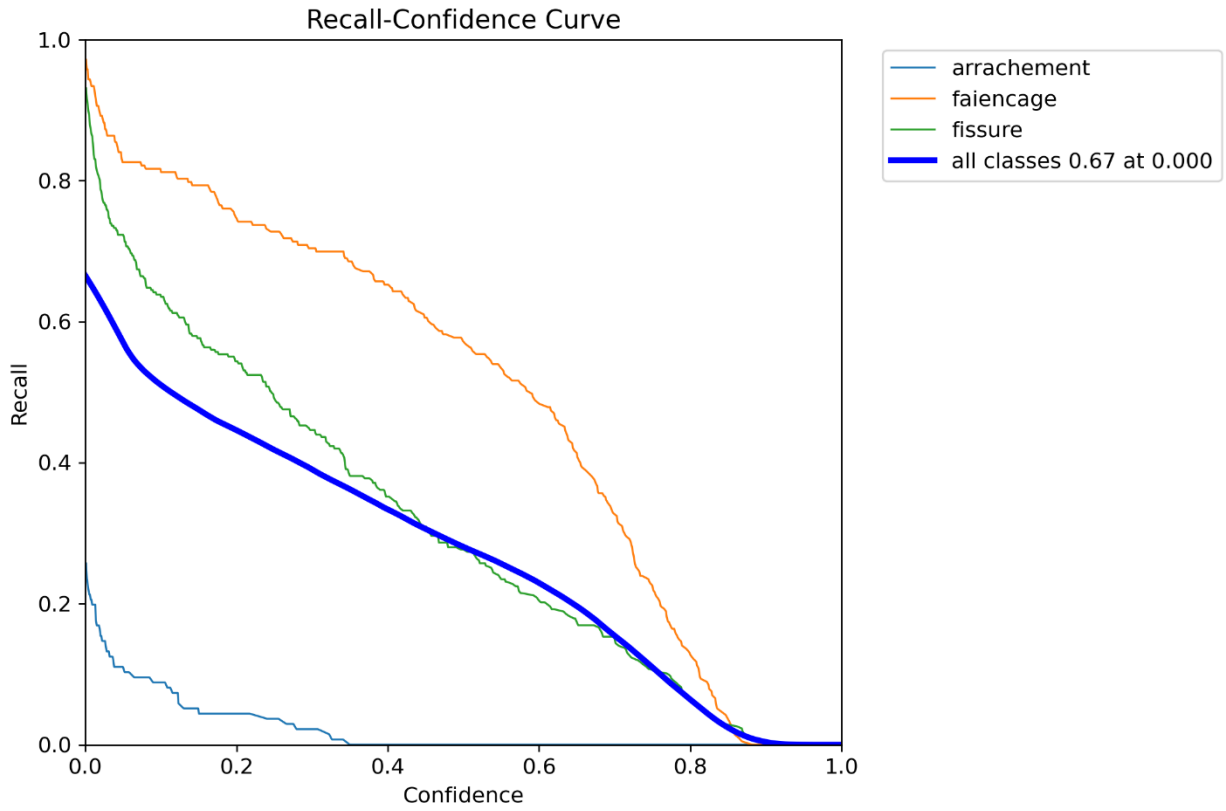


Figure V.5: Courbe Recall-Confidence

V.3.5.1. Analyse des courbes par classe

V.3.5.1.1. Classe faïençage :

La courbe de rappel pour cette classe est la plus performante. A un seuil de confiance nul (0.0), le rappel est proche de 1.0, ce qui indique que le modèle détecte pratiquement toutes les instances de faïençage. Même en augmentant le seuil jusqu'à 0.6, le rappel reste supérieur à 0.6, traduisant

une grande robustesse de détection, notamment à faible ou moyenne confiance. Cela confirme la fiabilité du modèle pour cette classe, déjà observée dans les courbes F1 et les matrices de confusion.

V.3.5.1.2. Classe fissure :

Le rappel initial est également élevé (≈ 1.0) pour des seuils proches de zéro. La courbe décroît plus rapidement que celle de faïençage, mais le rappel reste au-dessus de 0.4 jusqu'à un seuil d'environ 0.4, ce qui témoigne d'une capacité modérée à bonne à détecter les fissures, sous réserve d'un seuil de confiance adapté.

V.3.5.1.3. V.6.1.3 Classe arrachement :

Il s'agit de la classe la plus problématique. Le rappel débute à un niveau très bas (≈ 0.25), même à un seuil nul, et chute brutalement à des seuils modérés (dès 0.2 à 0.3, il devient quasi nul). Cela indique une incapacité du modèle à identifier correctement les arrachements, quelles que soient les conditions. Ce constat est en totale cohérence avec les matrices de confusion normalisées, où la quasi-totalité des arrachements (jusqu'à 96 %) sont confondus avec le background.

V.3.5.2. V.6.1.4 Courbe globale (All classes) – Moyenne pondérée

À un seuil nul, le modèle atteint un rappel global de 0.67. Autrement dit, en acceptant toutes les détections sans filtrage, YOLOv8n est capable de retrouver 67 % des objets réellement présents. Toutefois, cette approche s'accompagne d'une baisse significative de la précision, rendant ce niveau de rappel difficilement exploitable en pratique sans calibration.

V.3.6. La courbe Precision-Recall :

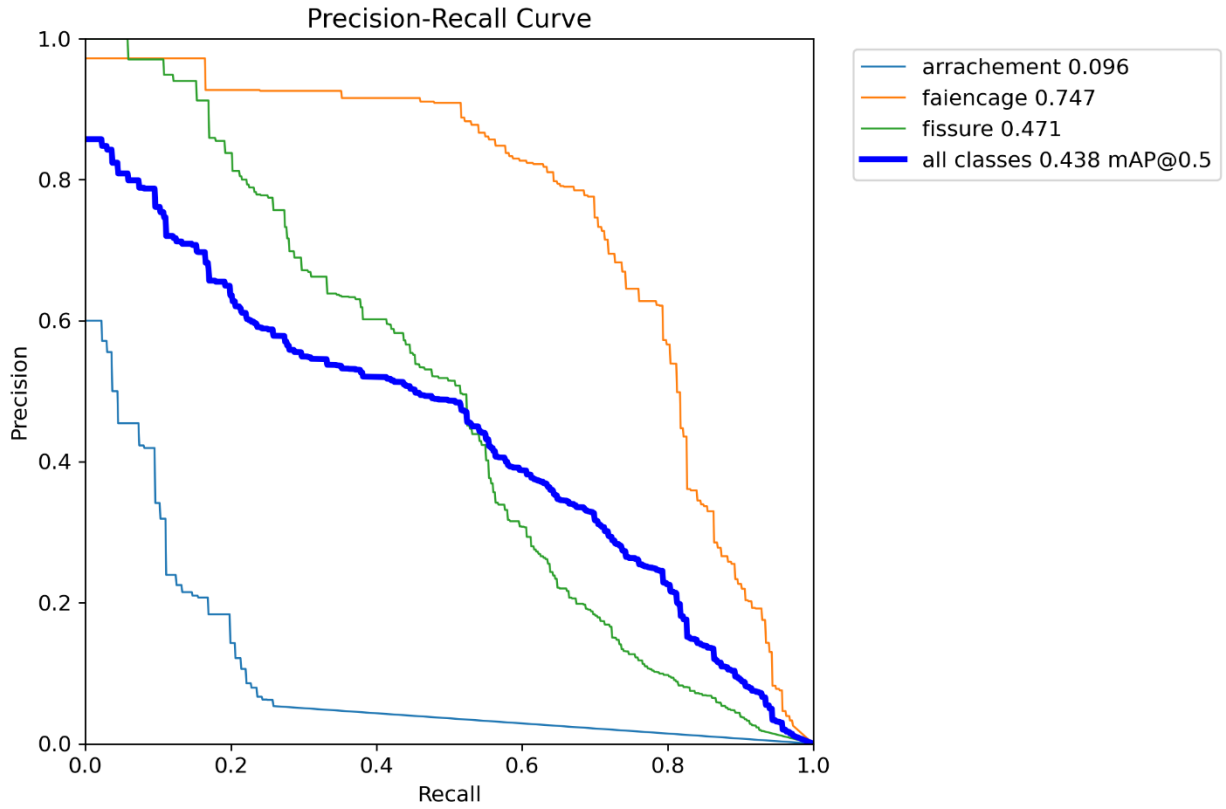


Figure V.6 : courbe Précision-Recall

La courbe Precision–Recall (PR) illustrée par la figure V.6 ci-dessus, permet de visualiser le compromis entre la précision (exactitude des prédictions positives) et le rappel (capacité à détecter toutes les instances positives) pour chaque classe. Elle est particulièrement utile dans le contexte des ensembles de données déséquilibrés, où une simple mesure de précision ou d'exactitude peut être trompeuse.

L'Axe des abscisses (Recall) : représente le taux de détection des objets réellement présents. Plus on avance vers la droite, plus le modèle détecte d'instances correctes.

L'Axe des ordonnées (Precision) : mesure la fiabilité des détections positives. Une valeur élevée indique un faible taux de faux positifs.

Une courbe idéale conserve une précision élevée sur une large gamme de rappels. L'aire sous la courbe (Average Precision, AP) est utilisée pour résumer la performance de chaque classe.

V.3.6.1. Analyse par classe

V.3.6.1.1. V.7.1.1 Classe faïençage :

AP (Average Precision) **: 0.747,** La courbe est la plus performante de toutes, témoignant d'un excellent équilibre entre précision et rappel. La précision reste proche de 0.9 même pour des rappels élevés (0.7 à 0.8), ce qui indique que le modèle détecte efficacement la majorité des faïençages tout en limitant les erreurs.

V.3.6.1.2. Classe fissure :

AP : 0.471, la performance est modérée, mais significative. La précision décroît progressivement à mesure que le rappel augmente. Pour un rappel de 0.6, la précision avoisine 0.6, indiquant une capacité correcte à identifier les fissures, avec un compromis acceptable.

V.3.6.1.3. Classe arrachement :

AP : 0.096, Il s'agit de la classe la moins bien détectée par le modèle. La courbe chute brutalement : la précision est faible à tous les niveaux de rappel, et devient quasiment nulle dès les premiers seuils. Ce résultat traduit une incapacité manifeste à détecter la classe arrachement, corroborée par les faibles rappels (entre 1 % et 9 %) observés dans les matrices de confusion, et par la forte confusion avec la classe background (jusqu'à 96 %).

V.3.6.1.4. Courbe moyenne (all classes) – Performance globale :

mAP@0.5 : 0.438, Cette métrique représente la moyenne des AP de toutes les classes (mean Average Precision). Une valeur de 0.438 indique une performance globale modérée, principalement pénalisée par la très faible performance du modèle sur la classe arrachement.

V.3.7. Train/val :

Le modèle YOLOv8n a été entraîné sur 100 époques. L'analyse des courbes générées (losses et métriques) permet d'évaluer l'évolution de son comportement en termes d'apprentissage, de généralisation, et de détection par classe. Cette section synthétise les principales observations issues des figures générées dans results.jpg. La figure V.7 ci-après montre l'évolution des pertes et des métriques

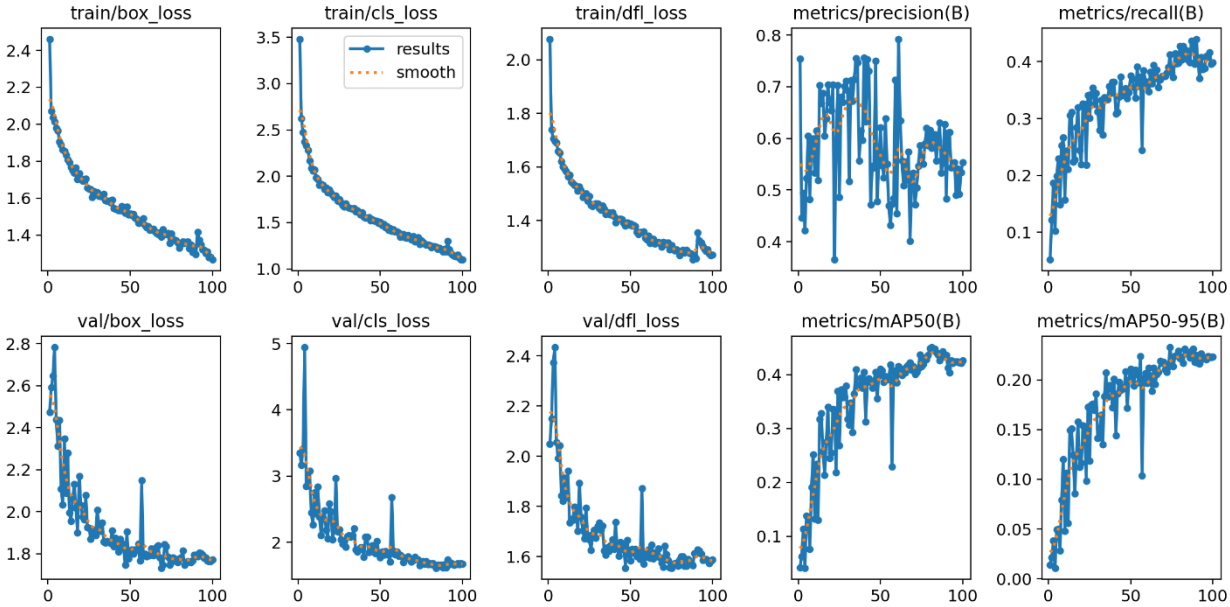


Figure V.7: Courbes losses et métriques de Yolov8n

V.3.7.1. Perte de localisation (box_loss)

V.3.7.1.1. Entraînement (train/box_loss) :

la perte diminue progressivement de 2.5 à environ 1.1, traduisant une amélioration continue dans la précision de localisation des boîtes englobantes sur les données d'apprentissage.

V.3.7.1.2. Validation (val/box_loss) :

la courbe suit une tendance similaire, bien que plus bruitée, se stabilisant autour de 1.75 à 1.8 après 40 époques. Ce qui confirme que la convergence parallèle des pertes d'entraînement et de validation suggère une bonne capacité de généralisation du modèle, sans signe évident de surapprentissage.

V.3.7.2. Perte de classification (cls_loss) :

V.3.7.2.1. Entraînement :

la perte de classification chute de plus de 3.5 à environ 1.0, montrant que le modèle apprend à mieux attribuer la bonne classe aux objets détectés.

V.3.7.2.2. Validation :

initialement très élevée (>2.4), elle diminue rapidement pour se stabiliser vers 1.6–1.7.

Ce qui exprime que la forte décroissance initiale est typique d'une phase d'adaptation rapide. L'absence de remontée en validation exclut un phénomène d'overfitting. Toutefois, la variabilité restante traduit une possible confusion inter-classes.

V.3.7.3. Perte DFL (Distribution Focal Loss)

V.3.7.3.1. Entraînement (train/dfl_loss) :

décroissance continue de 2.2 à environ 1.2.

V.3.7.3.2. Validation (val/dfl_loss) :

valeur initiale extrêmement haute (>2.40), suivie d'une chute rapide et d'une stabilisation vers 1.6.

Le pic au début montre une instabilité temporaire du modèle au démarrage de l'apprentissage, rapidement corrigée. Ensuite, la courbe suit une tendance similaire aux autres pertes, ce qui reflète une bonne stabilité générale de l'apprentissage.

V.3.7.4. Evolution de la précision et du rappel

V.3.7.4.1. Précision (metrics/precision(B)) :

la courbe évolue de 0.0 vers un plateau autour de 0.60–0.65, mais avec une certaine instabilité (effet de "dents de scie"), signe d'une sensibilité aux fluctuations dans les classes ou les échantillons de validation.

V.3.7.4.2. Rappel (metrics/recall(B)) :

progression régulière de 0.1 à environ 0.45, indiquant une amélioration stable de la capacité du modèle à détecter les objets réels.

Analyse : la croissance conjointe des deux métriques est encourageante. Toutefois, la variabilité de la précision souligne des limites dans la capacité du modèle à distinguer correctement certaines classes, notamment dans les cas de forte similarité visuelle ou de déséquilibre de données.

V.3.7.5. Evaluation par mAP

V.3.7.5.1. mAP@0.5 :

augmente progressivement jusqu'à 0.43–0.45, ce qui signifie que près de la moitié des prédictions atteignent un chevauchement suffisant (IoU ≥ 0.5) avec les objets réels.

V.3.7.5.2. mAP@0.5:0.95 :

atteint environ 0.22–0.25, ce qui est une valeur attendue pour un modèle léger de type "nano".

Il en résulte que les scores obtenus restent cohérents avec les attentes pour un modèle de faible capacité comme YOLOv8n. Bien que le mAP global soit modeste, il traduit une certaine efficacité sur les classes majoritaires.

V.4. V.9 YOLOv8s :

V.4.1. V.9.1.1 Matrice de confusion :

La figure V.8 illustre la matrice de confusion obtenue lors de l'évaluation du modèle YOLOv8s. Cette matrice permet d'analyser les performances du modèle en termes de classification des différentes classes de dégradations.

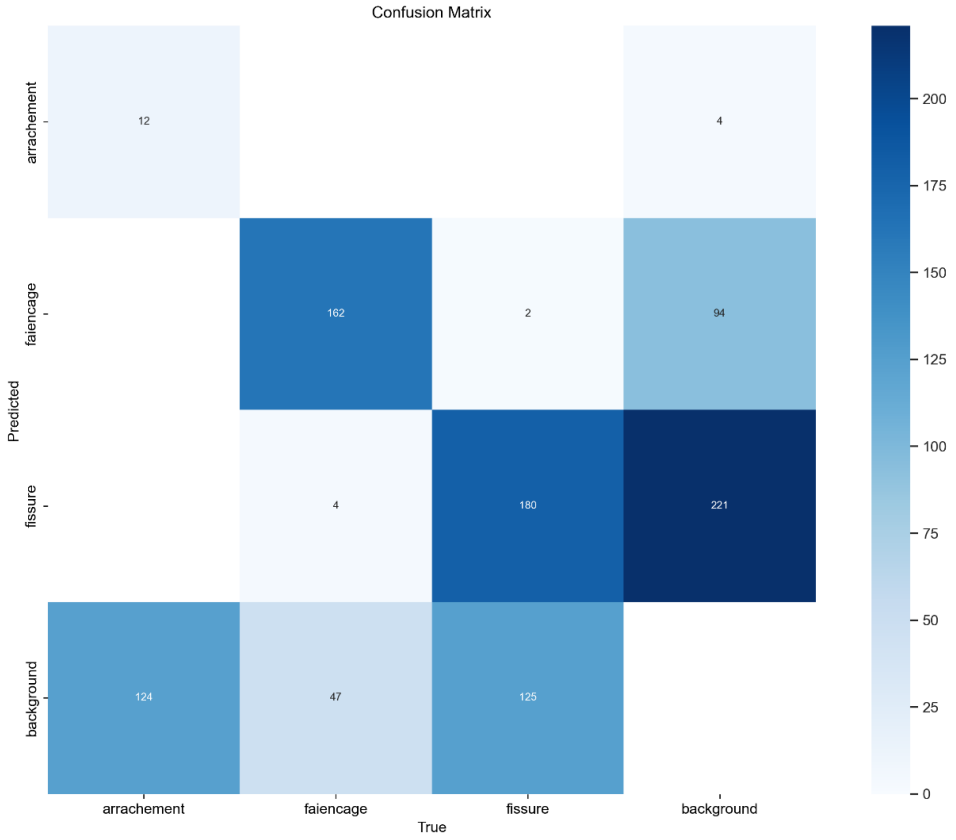


Figure V.8: Matrice de confusion YOLOv8s.

- Les **lignes** de la matrice correspondent aux **classes prédites** par le modèle.
- Les **colonnes** représentent les **classes réelles**, c'est-à-dire celles effectivement présentes dans les données de test.
- Chaque **cellule** indique le nombre d'exemples d'une classe réelle spécifique (colonne) que le modèle a classés dans une autre classe (ligne).
- Les **valeurs situées sur la diagonale principale** (de haut en bas) correspondent aux **prédictions correctes** (ou vrais positifs), tandis que les **valeurs en dehors de cette diagonale** traduisent des **erreurs de classification** (confusions entre classes).

V.4.1.1.

V.4.1.2. Classes analysées

Le modèle YOLOv8s a été entraîné pour classifier quatre catégories de données issues d'images aériennes : Arrachement, faïençage, fissure et background (zones sans défaut).

V.4.1.3. Analyse détaillée par classe

V.4.1.3.1. Classe "arrachement"

- **Prédictions correctes** : 12 véritables arrachements ont été identifiés correctement. Il s'agit de la meilleure performance obtenue pour cette classe parmi l'ensemble des modèles testés. Toutefois, ce nombre reste faible en valeur absolue, traduisant une sensibilité limitée du modèle à ce type de défaut.
- **Erreurs de classification** :

124 instances ont été erronément classées comme background, ce qui représente une erreur dominante et compromet la fiabilité du modèle pour cette catégorie.

V.4.1.3.2. Classe "faïençage"

- **Prédictions correctes** : 162 instances ont été correctement classées, ce qui démontre une très bonne performance de détection pour cette classe.
- **Erreurs de classification** :

  4 instances ont été confondues avec fissure (erreur minime).

  47 instances ont été mal classées comme background, ce qui, bien que non négligeable, reste inférieur aux erreurs observées pour la classe "fissure".

V.4.1.3.3. Classe "fissure"

- **Prédictions correctes** : 180 cas ont été correctement prédits, ce qui reflète également une très bonne capacité du modèle à identifier cette classe.
- **Erreurs de classification** :
  - 2 cas ont été classés comme faïençage (erreur très faible).
  - 125 instances ont été confondues avec le background, ce qui constitue une erreur de classification importante, affectant négativement le rappel pour cette catégorie.

V.4.1.3.4. Classe "background"

- **Prédictions correctes** : Aucune instance de background n'a été correctement identifiée. Ce résultat est particulièrement préoccupant, puisqu'il indique que le modèle échoue systématiquement à reconnaître les zones sans défaut.
- **Erreurs de classification** :
  - 4 cas ont été classés comme arrachement,
  - 94 comme faïençage,
  - 221 comme fissure.

Cela confirme une tendance forte à la surdétection : le modèle interprète à tort des zones saines comme des défauts, ce qui peut entraîner une génération excessive de faux positifs.

V.4.2. Matrice de confusion normalisée:

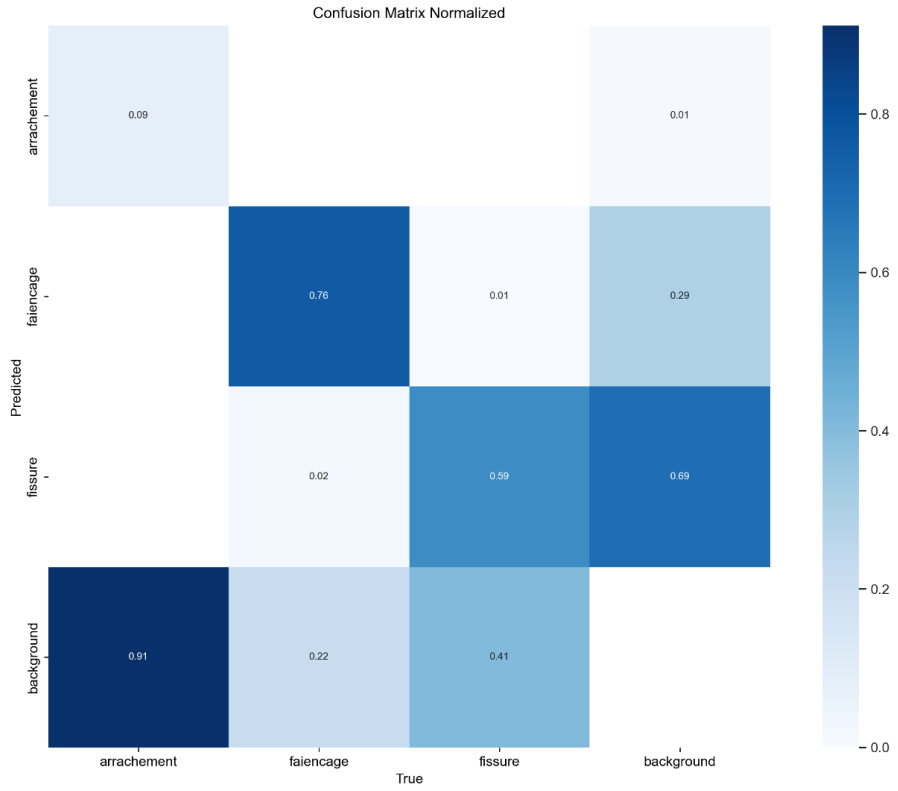


Figure V.9 : Matrice de cnfusion yolov8s

La matrice de confusion normalisée illustré dant la figure V.9 représente les performances du modèle YOLOv8s en termes proportionnels plutôt qu'en valeurs absolues. Chaque cellule de la matrice indique la part des instances d'une classe réelle donnée ayant été classées dans une catégorie prédite.

- La valeur (i, j) correspond à la proportion d'instances réellement de la classe j qui ont été prédites comme étant de la classe i.
- Ainsi, la somme des valeurs de chaque colonne est égale à 1 (ou 100 %), car elle traduit la distribution des prédictions effectuées pour une classe réelle spécifique. Les valeurs situées sur la diagonale principale (du haut gauche vers le bas droit) représentent
- le rappel (ou sensibilité) pour chaque classe, c'est-à-dire la capacité du modèle à détecter correctement les instances appartenant à cette classe.
- Les valeurs hors diagonale traduisent les erreurs de classification, c'est-à-dire les cas où le modèle a confondu une classe avec une autre.

V.4.2.1. Classes évaluées

Le modèle a été entrainé pour classifier les quatre catégories suivantes : (arrachement, faïençage, fissure et background (zone sans défaut). Analyse détaillée des performances (rappel/sensibilité)

V.4.2.1.1. Classe "arrachement"

- **Rappel** : 0,09 (soit 9 %) Ce score indique que le modèle parvient à détecter moins de 10 % des véritables arrachements, ce qui est très insuffisant.
- **Erreurs principales** : 91 % des arrachements réels sont classés à tort comme background, révélant une forte confusion entre les zones dégradées et l'arrière-plan.

V.4.2.1.2. Classe "faïençage"

- **Rappel** : 0,76 (soit 76 %) Le modèle démontre une bonne capacité à détecter cette classe, qui figure parmi les mieux identifiées.
- **Erreurs de classification** :

  1 % des faïençages sont confondus avec fissure (erreur négligeable).

  22 % sont mal classés comme background, ce qui représente une perte de performance significative.

V.4.2.1.3. Classe "fissure"

- **Rappel** : 0,59 (soit 59 %) Le modèle identifie un peu plus de la moitié des véritables fissures, ce qui constitue une performance modérée.
- **Erreurs de classification** :

  1 % sont prédites comme faïençage.

  41 % sont classées comme background, un taux élevé qui réduit fortement la sensibilité globale de la classe.

V.4.2.1.4. Classe "background"

- **Rappel** : 0,00 (soit 0 %) Le modèle échoue complètement à détecter les zones d'arrière-plan. Aucune instance réelle de background n'a été reconnue comme telle.
- **Erreurs principales** :
  - 1 % des backgrounds sont prédits comme arrachement,
  - 29 % comme faïençage,
  - 69 % comme fissure. Cette forte confusion du fond avec les classes de défauts entraîne une génération massive de faux positifs.

V.4.3. La courbe F1-confiance:

La courbe F1-Confidence illustrée par la figure V.10 ci-dessous, permet d'évaluer la performance du modèle YOLOv8s en fonction du seuil de confiance appliqué aux prédictions. Elle constitue un outil essentiel pour analyser l'équilibre entre précision et rappel à différents niveaux de certitude dans la détection.

L'Axe des abscisses (Confidence) : représente le seuil de confiance appliqué aux prédictions du modèle, allant de 0.0 à 1.0. Plus le seuil est élevé, plus le modèle doit être sûr de sa prédiction pour la valider.

L'Axe des ordonnées (F1-score) : indique le score F1, qui combine la précision (précision des détections) et le rappel (taux de détection des objets réels).

Un F1-score élevé traduit un bon compromis entre éviter les faux positifs et minimiser les oublis.

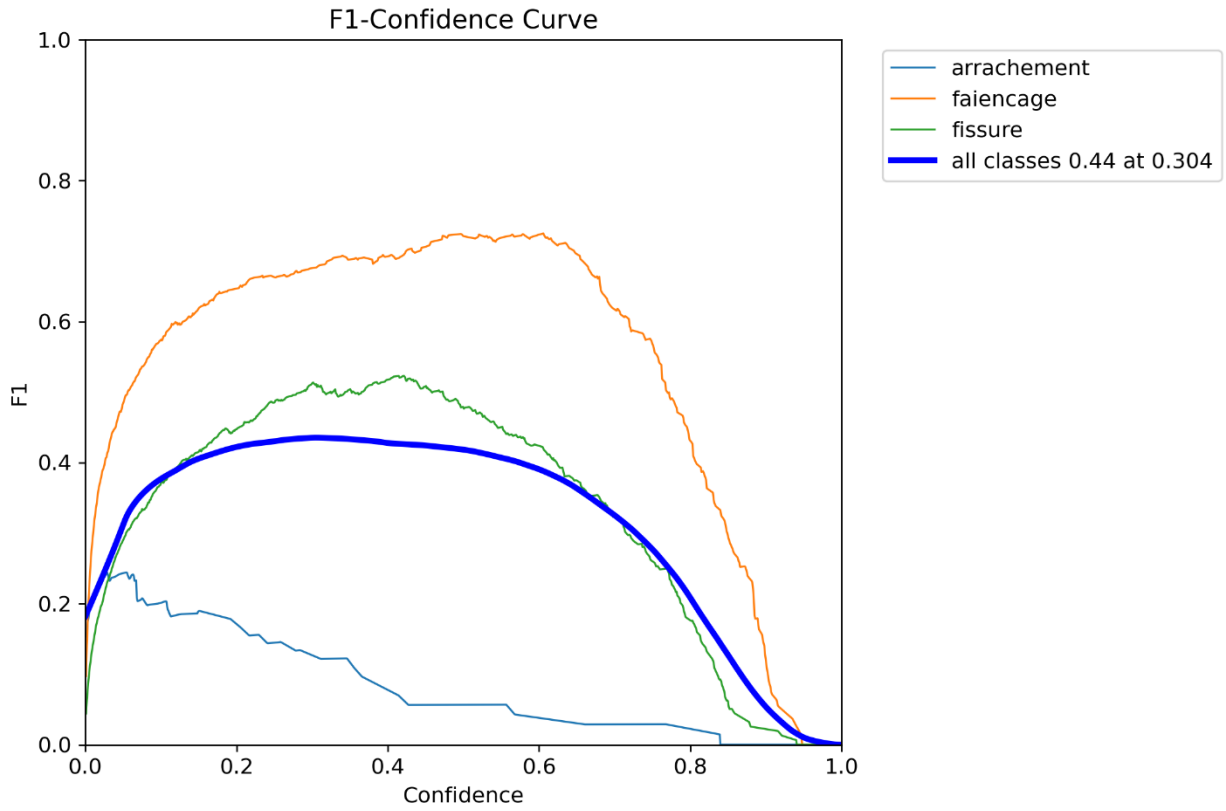


Figure V.10 :courbe F1-Confiance yolov8s

V.4.3.1. Analyse par classe

V.4.3.1.1. Classe Faïençage (orange):

Cette courbe présente la meilleure performance parmi toutes les classes. Le pic du score F1 atteint environ 0,72 à 0,73, pour un seuil de confiance situé entre 0,6 et 0,8. La stabilité du score F1 sur une large plage de seuils (de 0,2 à 0,8) Cela traduit une capacité remarquable du modèle à détecter cette classe, y compris à des seuils de confiance faibles ou modérés. La régularité de la courbe témoigne d'une stabilité et d'une fiabilité élevées dans la prédiction de faïençage.

V.4.3.1.2. Classe : Fissure (vert)

Elle se classe en deuxième position en termes de performance. Le score F1 maximal est d'environ 0,50 à 0,52, pour un seuil de confiance situé entre 0,4 et 0,6. Ce niveau reste acceptable, bien que moins performant que le faïençage. Il suggère que le modèle atteint un compromis raisonnable entre rappel et précision, mais reste affecté par des erreurs de classification, en particulier des confusions avec le fond ou d'autres classes proches visuellement.

V.4.3.1.3. Classe : Arrachement (bleu clair)

Il s'agit de la classe la moins bien détectée. Le F1-score maximal est d'environ 0,24 à0,25, pour un seuil de confiance situé entre 0 et 0,1 reste nettement inférieur à 0,3, en déclin rapide dès que le seuil de confiance dépasse 0.2.

V.4.3.1.4. Courbe moyenne (toutes classes – bleu foncé)

Le score F1 moyen atteint un pic à 0.44 lorsque le seuil de confiance est fixé à 0.304.

Cela indique que le meilleur compromis entre précision et rappel est obtenu lorsque seules les prédictions dont la confiance dépasse 30 % sont retenues. Ce point constitue donc un seuil optimal dans une logique de déploiement opérationnel, permettant de limiter les faux positifs tout en maintenant une détection satisfaisante.

V.4.4. Courbe de précision :

La figure V.11 ci-dessous montre la courbe Précision-Confidence (ou P-curve), qui permet d'évaluer la performance du modèle YOLOv8 en fonction du niveau de confiance associé à ses prédictions. Cette courbe est nécessaire pour comprendre le compromis entre précision et seuil de confiance, et pour déterminer un seuil optimal à partir duquel les prédictions sont fiables.

L'Axe des abscisses (Confidence) : représente le seuil de confiance minimal requis pour qu'une détection soit considérée comme valide. Les valeurs s'échelonnent de 0.0 à 1.0.

L'Axe des ordonnées (Precision) : indique la précision au sens classique :

Une précision élevée reflète une capacité du modèle à limiter les fausses détections.

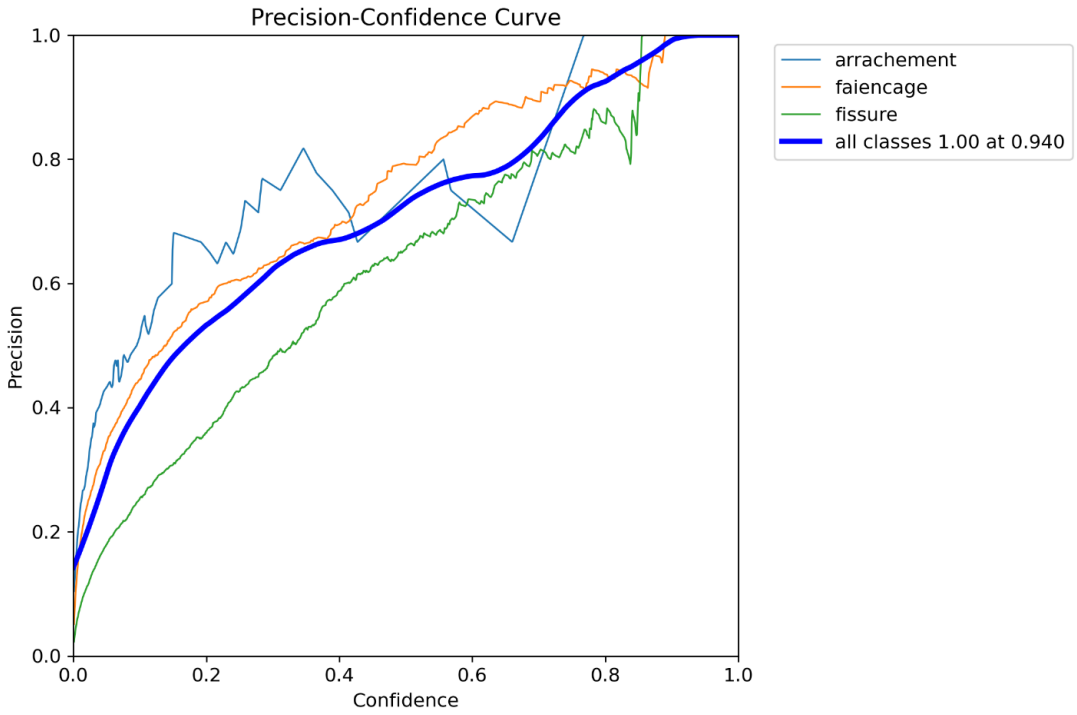


Figure V.11 : courbe de Précision-Confidence de Yolov8s

V.4.4.1. Classes analysées :

Le modèle YOLOv8s a été entraîné pour classifier quatre catégories de données issues d'images aériennes : Arrachement, faïençage, fissure et background (zones sans défaut).

V.4.4.2. Analyse par classe

V.4.4.2.1. Classe Faïençage (orange):

La courbe correspondant à la classe faïençage se distingue par sa grande stabilité sur une large plage de seuils de confiance (de 0.2 à 0.9).

Cette régularité témoigne d'une excellente capacité du modèle à identifier correctement le faïençage, même lorsque le niveau de certitude est relativement modeste. Cette robustesse en fait l'une des classes les mieux apprises et les plus fiables du système.

V.4.4.2.2. Classe Fissure (vert):

La courbe montre une croissance moins rapide que celles des autres classes, surtout à des seuils de confiance faibles. Cependant, à des seuils élevés (> 0.8), la précision converge également vers 1.00, montrant que cette classe peut aussi fournir des prédictions très fiables si on impose un seuil strict.

V.4.4.2.3. Classe Arrachement (bleu clair):

La classe arrachement affiche une courbe irrégulière, marquée par des variations imprévisibles de la précision.

Ces oscillations traduisent une incertitude du modèle vis-à-vis de cette classe, et une sensibilité marquée aux fluctuations du seuil de confiance. Ce comportement indique un manque de cohérence dans la détection des arrachements.

V.4.4.2.4. Courbe moyenne – toutes classes (bleu foncé)

La courbe moyenne révèle une progression régulière de la précision avec l'augmentation du seuil de confiance. Elle atteint un score parfait de 1.00 à partir d'un seuil de 0.94, ce qui signifie que lorsque le modèle conserve uniquement les prédictions dont la confiance est supérieure ou égale à 94 %, il ne commet pratiquement aucune erreur.

Ce comportement est hautement rassurant sur la fiabilité du modèle en contexte de décision stricte : plus la confiance est élevée, plus la précision est excellente.

V.4.5. La courbe Recall-Confidence:

La courbe Recall–Confidence présentée par la figure V.12 ci-dessus, illustre la capacité du modèle YOLOv8n à détecter correctement les objets pertinents en fonction du seuil de confiance appliqué aux prédictions. Elle permet de quantifier, pour chaque classe, la proportion des instances réelles qui ont été correctement identifiées (rappel) en fonction de la rigueur avec laquelle le modèle accepte ou rejette une détection.

L'Axe des abscisses (Confidence) : représente le seuil minimal requis pour valider une prédiction.

- Un seuil bas permet de retenir davantage de détections, tandis qu'un seuil élevé filtre les prédictions les moins sûres.
- Un rappel élevé signifie que le modèle détecte la majorité des objets réellement présents, même si cela se fait parfois au détriment de la précision.

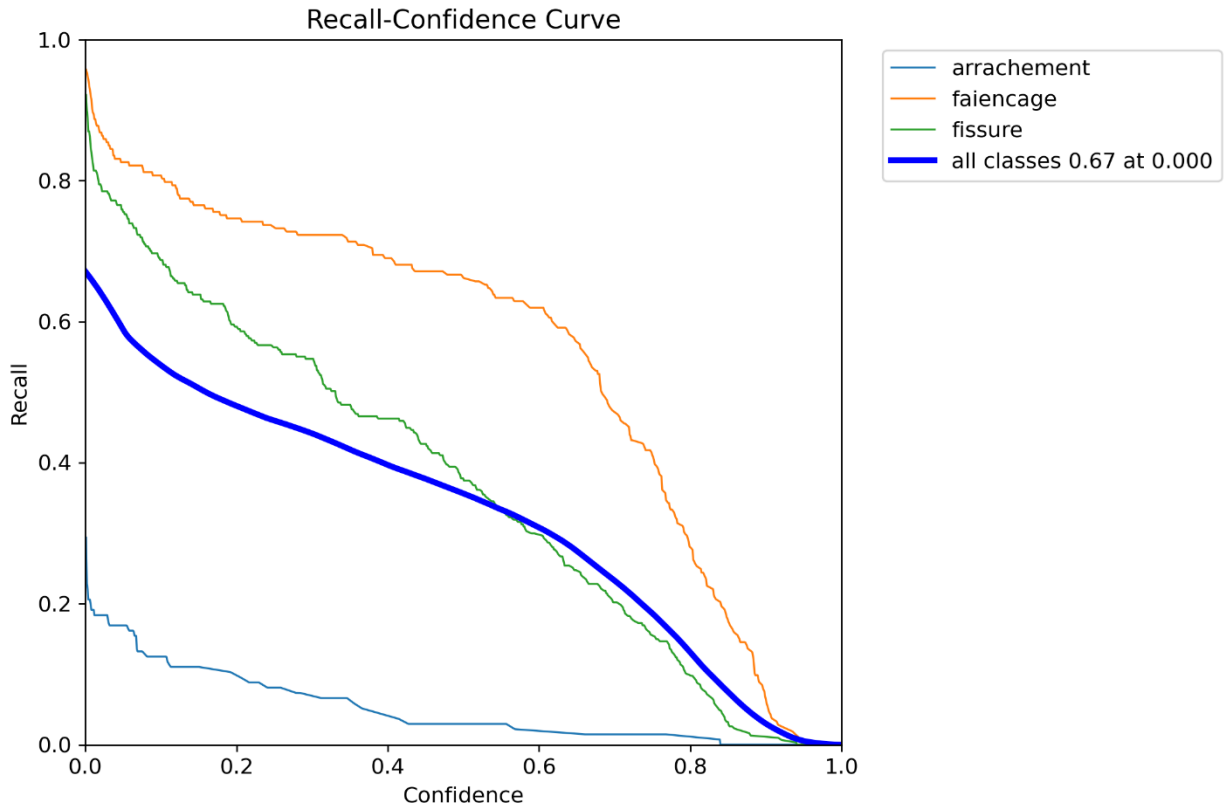


Figure V.12 :courbe Recall-Confiance de Yoolov8s

V.4.5.1. Analyse par classe

La courbe de rappel pour cette classe est la plus performante. A un seuil de confiance nul (0.0), le rappel est proche de 1.0, ce qui indique que le modèle détecte pratiquement toutes les instances de faïençage. Même en augmentant le seuil jusqu'à 0.7, le rappel reste supérieur à 0.6, traduisant une grande robustesse de détection, notamment à faible ou moyenne confiance. Cela confirme la fiabilité du modèle pour cette classe, déjà observée dans les courbes F1 et les matrices de confusion.

V.4.5.1.1. Classe : Faïençage (orange)

La classe *faïençage* présente une excellente capacité de détection, avec un rappel supérieur à 0.80 pour des niveaux de confiance bas à modérés.

Le modèle détecte efficacement cette classe, même lorsqu'il est peu strict sur la validation des prédictions.

Cette performance peut s'expliquer par :

- Une bonne représentativité de la classe dans l'ensemble d'apprentissage.
- Des caractéristiques visuelles claires, qui facilitent la différenciation et l'apprentissage automatique.

V.4.5.1.2. Classe : Fissure (vert)

La courbe de la classe fissure au début est très élevé de 0,95, avec un rappel avoisinant 0.60 pour des seuils faibles.

Le modèle parvient à détecter une partie significative des fissures, mais avec moins de régularité que pour le faïençage.

V.4.5.1.3. Classe : Arrachement (bleu clair)

Il s'agit de la classe la plus problématique. Le rappel débute à un niveau très bas (≈ 0. 3), même à un seuil nul, et chute brutalement à des seuils modérés (dès 0.2 à 0.85, il devient quasi nul). Cela indique une capacité très faible du modèle à identifier correctement les arrachements.

V.4.6. La courbe Precision-Recall :

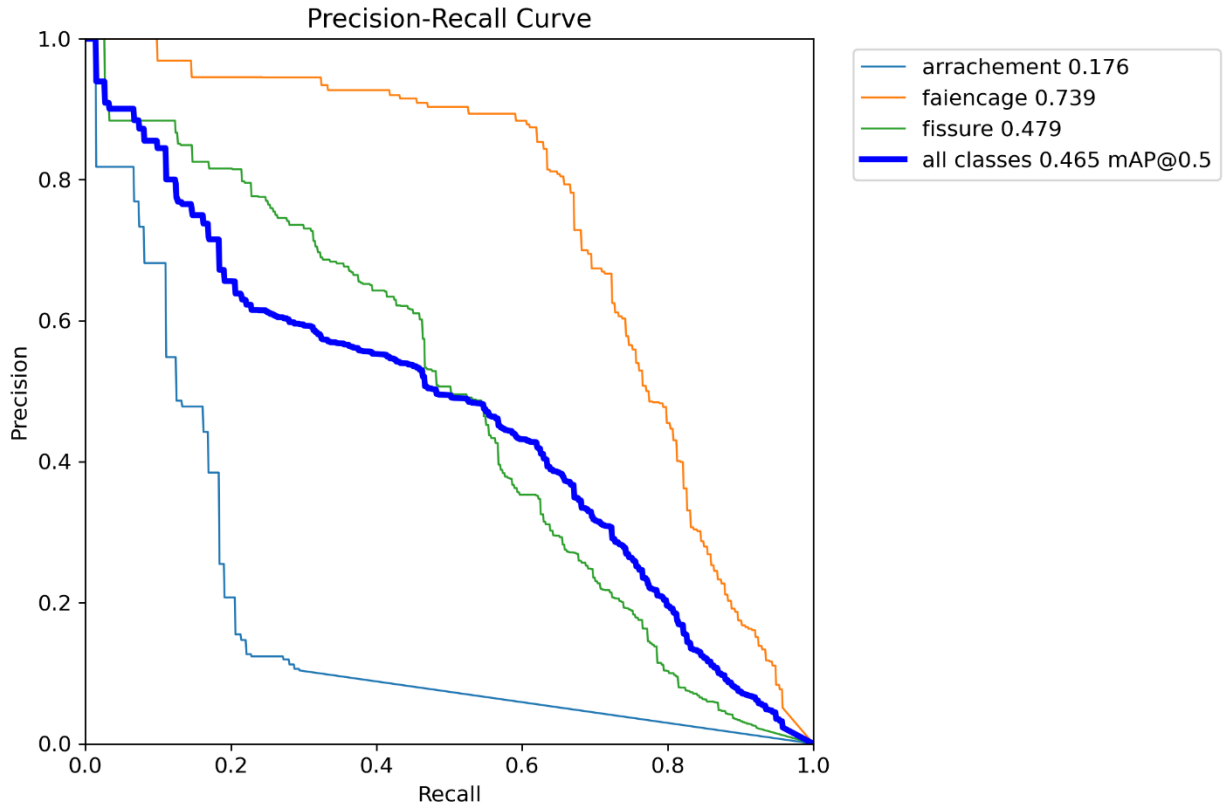


Figure V.13:Courbe précision- Rappel de Yolov8s

La courbe Precision–Recall (PR) illustrée par la figure V.13 ci-dessus, permet de visualiser le compromis entre la précision (exactitude des prédictions positives) et le rappel (capacité à détecter toutes les instances positives) pour chaque classe. Elle est particulièrement utile dans le contexte des ensembles de données déséquilibrés, où une simple mesure de précision ou d'exactitude peut être trompeuse.

L'Axe des abscisses (Recall) : représente le taux de détection des objets réellement présents. Plus on avance vers la droite, plus le modèle détecte d'instances correctes.

L'Axe des ordonnées (Precision) : mesure la fiabilité des détections positives. Une valeur élevée indique un faible taux de faux positifs.

Une courbe idéale conserve une précision élevée sur une large gamme de rappels. L'aire sous la courbe (Average Precision, AP) est utilisée pour résumer la performance de chaque classe.

V.4.6.1. Analyse par classe

V.4.6.1.1. Classe faïençage :

AP (Average Precision) **: 0.739,** La courbe est la plus performante de toutes, témoignant d'un excellent équilibre entre précision et rappel. La précision reste égale à 1 avec le rappel de 0,0 à 1,3 , et reste supérieure à 0,9 même pour des rappels élevés (0.6 à 0.7), ce qui indique que le modèle détecte efficacement la majorité des faïençages tout en limitant les erreurs.

V.4.6.1.2. Classe fissure :

AP : 0.479, la performance est modérée, mais significative. La précision décroît progressivement à mesure que le rappel augmente. Pour un rappel de 0.5, la précision avoisine 0.5, indiquant une capacité correcte à identifier les fissures, avec un compromis acceptable.

V.4.6.1.3. Classe arrachement :

AP : 0.176, Il y a une amélioration au début mais reste la moins détectée par le modèle. La courbe chute brutalement : la précision est faible à tous les niveaux de rappel, et devient quasiment nulle dès les premiers seuils. Ce résultat traduit une incapacité manifeste à détecter la classe arrachement, corroborée par les faibles rappels (entre 1 % et 9 %) observés dans les matrices de confusion, et par la forte confusion avec la classe background (jusqu'à 96 %).

V.4.6.1.4. Courbe moyenne (all classes) – Performance globale :

mAP@0.5 : 0.465, Cette métrique représente la moyenne des AP de toutes les classes (mean Average Precision). Une valeur de 0.465 est considéré comme correct dans le contexte d'un modèle de taille réduite (*YOLOv8s*), principalement pénalisée par la très faible performance du modèle sur la classe arrachement.

V.4.7. Train/val :

Le modèle YOLOv8n a été entraîné sur 100 époques. L'analyse des courbes générées (losses et métriques) permet d'évaluer l'évolution de son comportement en termes d'apprentissage, de généralisation, et de détection par classe. Cette section synthétise les principales observations issues des figures générées dans results.jpg. La figure V.14 ci-après montre l'évolution des pertes et des métriques

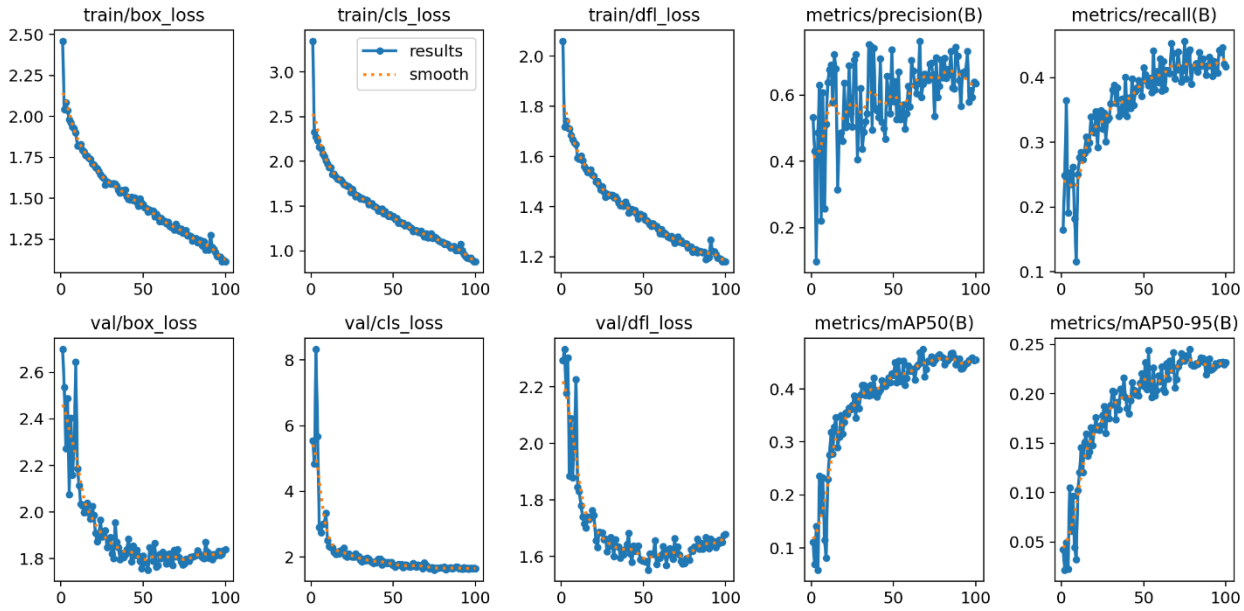


Figure V.14: les courbes losses et métrique de Yolov8s

V.4.7.1. Perte de localisation (box_loss)

V.4.7.1.1. Entraînement (train/box_loss) :

la perte diminue progressivement de 2.5 à environ 1.1, traduisant une amélioration continue dans la précision de localisation des boîtes englobantes sur les données d'apprentissage.

V.4.7.1.2. Validation (val/box_loss) :

la courbe suit une tendance similaire, bien que plus bruitée, se stabilisant autour de 1.8 après 50 époques. Ce qui confirme que la convergence parallèle des pertes d'entraînement et de validation suggère une bonne capacité de généralisation du modèle, sans signe évident de surapprentissage.

V.4.7.2. Perte de classification (cls_loss) :

V.4.7.2.1. Entraînement :

la perte de classification chute de plus de 3.5 à environ 1.0, montrant que le modèle apprend à mieux attribuer la bonne classe aux objets détectés.

V.4.7.2.2. Validation :

initialement très élevée 8, elle diminue rapidement pour se stabiliser vers 1.5–1.6.

Ce qui exprime que la forte décroissance initiale est typique d'une phase d'adaptation rapide. L'absence de remontée en validation exclut un phénomène d'overfitting. Toutefois, la variabilité restante traduit une possible confusion inter-classes.

V.4.7.3. Perte DFL (Distribution Focal Loss)

V.4.7.3.1. Entraînement (train/dfl_loss) :

décroissance continue de 2.2 à environ 1.2.

V.4.7.3.2. Validation (val/dfl_loss) :

valeur initiale extrêmement haute 2 ,3, suivie d'une chute rapide et d'une stabilisation vers 1.6.

Le pic au début montre une instabilité temporaire du modèle au démarrage de l'apprentissage, rapidement corrigée. Ensuite, la courbe suit une tendance similaire aux autres pertes, ce qui reflète une bonne stabilité générale de l'apprentissage.

V.4.7.4. Evolution de la précision et du rappel

V.4.7.4.1. Précision (metrics/precision(B)) :

la courbe évolue de 0.0 vers un plateau autour de 0.65–0.7, mais avec une certaine instabilité , signe d'une sensibilité aux fluctuations dans les classes ou les échantillons de validation.

V.4.7.4.2. Rappel (metrics/recall(B)) :

progression régulière de 0.1 à environ 0.5, indiquant une amélioration stable de la capacité du modèle à détecter les objets réels.

Analyse : la croissance conjointe des deux métriques est encourageante. Toutefois, la variabilité de la précision souligne des limites dans la capacité du modèle à distinguer correctement certaines classes, notamment dans les cas de forte similarité visuelle ou de déséquilibre de données.

V.4.7.5. Evaluation par mAP

V.4.7.5.1. mAP@0.5 :

augmente progressivement jusqu'à 0.43–0.45, ce qui signifie que près de la moitié des prédictions atteignent un chevauchement suffisant avec les objets réels.

V.4.7.5.2. mAP@0.5:0.95 :

atteint environ 0.22–0.23, ce qui est une valeur attendue pour un modèle léger de type "small".

Il en résulte que les scores obtenus restent cohérents avec les attentes pour un modèle de capacité un peut faible comme YOLOv8. Bien que le mAP global soit modeste, il traduit une certaine efficacité sur les classes majoritaires.

V.5. Yolov8m :

V.5.1. Matrice de confusion :

La figure V.15 illustre la matrice de confusion obtenue lors de l'évaluation du modèle YOLOv8n. Cette matrice permet d'analyser les performances du modèle en termes de classification des différentes classes de dégradations.

- Les **lignes** de la matrice correspondent aux **classes prédites** par le modèle.
- Les **colonnes** représentent les **classes réelles**, c'est-à-dire celles effectivement présentes dans les données de test.
- Chaque **cellule** indique le nombre d'exemples d'une classe réelle spécifique (colonne) que le modèle a classés dans une autre classe (ligne).
- Les **valeurs situées sur la diagonale principale** (de haut en bas) correspondent aux **prédictions correctes** (ou vrais positifs), tandis que les **valeurs en dehors de cette diagonale** traduisent des **erreurs de classification** (confusions entre classes).

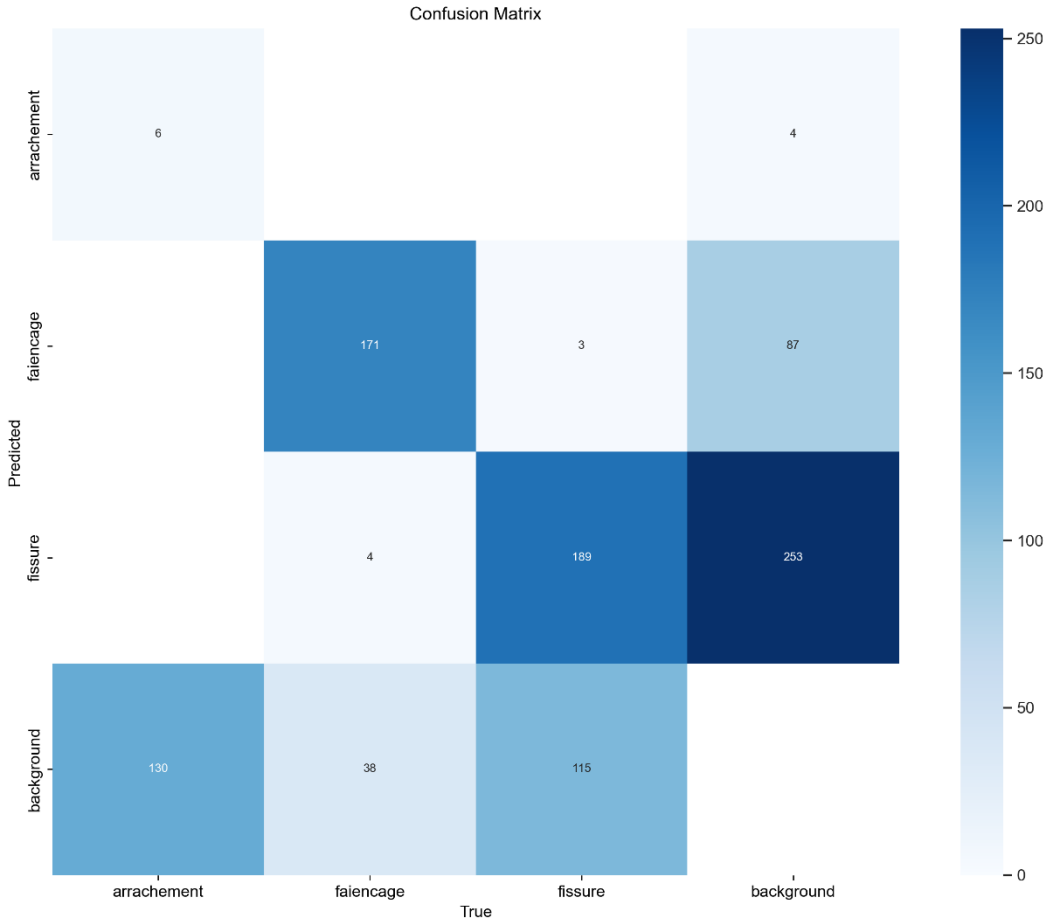


Figure V.15: matrice de confusion de Yolov8m

La matrice de confusion est un outil fondamental pour évaluer les performances d'un modèle de classification. Elle met en évidence la qualité des prédictions effectuées par le modèle en comparant les classes réelles (vérités terrain) aux classes prédites.

- Les colonnes de la matrice correspondent aux classes réelles, telles qu'annotées dans les données.
- Les lignes correspondent aux classes prédites par le modèle.
- Chaque cellule contient le nombre d'instances appartenant à une classe réelle (colonne) et ayant été classées dans une certaine catégorie prédite (ligne).
- Les valeurs situées sur la diagonale principale représentent les prédictions correctes.
- Les valeurs hors diagonale indiquent les erreurs de classification, c'est-à-dire les cas où le modèle a confondu une classe avec une autre.

V.5.1.1. Classes analysées :

Le modèle YOLOv8n a été entraîné pour classifier quatre catégories de données issues d'images aériennes : Arrachement, faïençage, fissure et background (zones sans défaut).

V.5.1.2. Analyse détaillée par classe

V.5.1.2.1. Classe "arrachement"

- **Prédictions correctes** : 6 véritables arrachements ont été correctement détectés. Ce chiffre, très faible, témoigne d'une performance insatisfaisante pour cette classe.

- **Erreurs de classification** :
    - 130 instances ont été classées à tort comme background, ce qui constitue une erreur majeure. Cela montre une confusion profonde entre les arrachements et les zones neutres, réduisant considérablement le rappel pour cette classe.

V.5.1.2.2. Classe "faïençage"

- **Prédictions correctes** : 171 cas correctement identifiés, traduisant une excellente capacité du modèle à reconnaître cette classe.
- **Erreurs de classification** :
    - 4 instances ont été confondues avec fissure (erreur minime).
    - 38 instances ont été mal classées comme background, ce qui constitue une perte partielle d'information, bien que contenue.

V.5.1.2.3. Classe "fissure"

- **Prédictions correctes** : 189 instances ont été correctement classées, confirmant une très bonne performance pour cette catégorie.
- **Erreurs de classification** :
    - 3 cas ont été confondus avec faïençage (erreur négligeable).
    - 115 instances ont été classées comme background, ce qui représente une erreur de classification importante, affectant le taux de rappel de manière significative.

V.5.1.2.4. Classe "background"

- **Prédictions correctes** : 0 instance a été correctement classée comme background. Ce résultat est particulièrement préoccupant, puisqu'il signifie que le modèle échoue totalement à reconnaître les zones sans dégradation.
- **Erreurs de classification** :
    - 4 cas ont été prédits comme arrachement,
    - 87 comme faïençage,
    - 253 comme fissure.

Cette distribution reflète une confusion systématique du fond avec des classes de défauts, notamment la classe fissure, ce qui pourrait entraîner une surodétection artificielle de défauts sur des zones pourtant saines.

V.5.2. Matrice de confusion normalisée:

La matrice de confusion normalisée comme illustré dand la figure V.16 ci-desous représente les performances du modèle YOLOv8n en termes proportionnels plutôt qu'en valeurs absolues. Chaque cellule de la matrice indique la part des instances d'une classe réelle donnée ayant été classées dans une catégorie prédite.

- La valeur (i, j) correspond à la proportion d'instances réellement de la classe j qui ont été prédites comme étant de la classe i.
- Ainsi, la somme des valeurs de chaque colonne est égale à 1 (ou 100 %), car elle traduit la distribution des prédictions effectuées pour une classe réelle spécifique. Les valeurs situées sur la diagonale principale (du haut gauche vers le bas droit) représentent

- le rappel (ou sensibilité) pour chaque classe, c'est-à-dire la capacité du modèle à détecter correctement les instances appartenant à cette classe.
- Les valeurs hors diagonale traduisent les erreurs de classification, c'est-à-dire les cas où le modèle a confondu une classe avec une autre.

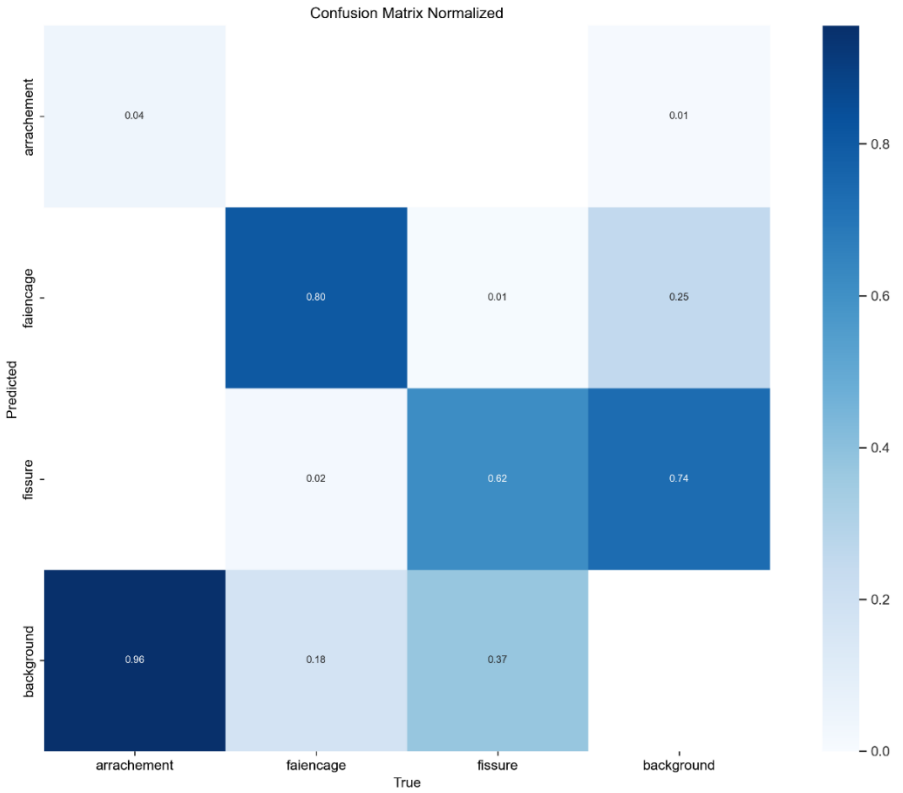


Figure V.16: matrice de confusion normalisée de Yolov8m

V.5.2.1. Classes évaluées

Le modèle YOLOv8n a été entrainé pour classifier les quatre catégories suivantes : (arrachement, faïençage, fissure et background (zone sans défaut).

V.5.2.2. Analyse des performances (rappel/sensibilité)

V.5.2.2.1. Classe "arrachement"

Rappel : 0,04 (soit 4 %)

Le modèle présente une très faible capacité de détection pour cette classe. Il ne parvient à identifier correctement que 4 % des véritables arrachements.

Erreurs principales : 96 % des instances réelles sont erronément classées comme background, ce qui reflète une confusion presque totale avec l'arrière-plan.

V.5.2.2.2. Classe "faïençage"

Rappel : 0,80 (soit 80 %) Il s'agit ici de la meilleure performance obtenue parmi toutes les classes. Le modèle démontre une bonne sensibilité à ce type de dégradation.

Erreurs de classification :

- 2 % sont classées comme fissure (erreur négligeable).

- 18 % sont mal interprétées comme background, ce qui montre que malgré de bonnes performances, le modèle perd encore près d'un cinquième des cas.

V.5.2.2.3. Classe "fissure"

Rappel : 0,62 (soit 62 %) Le modèle identifie correctement un peu plus de la moitié des fissures réelles, ce qui constitue une performance modérée à satisfaisante.

Erreurs de classification :

- 1 % sont confondues avec faïençage.
- 37 % sont classées comme background, ce qui reste un taux d'erreur important.

V.5.2.2.4. Classe "background"

Rappel : 0,00 (soit 0 %) Le modèle échoue totalement à reconnaître le background. Aucune instance de fond n'est correctement identifiée.

Erreurs principales :

- 1 % des backgrounds sont classés comme arrachement,
- 25 % comme faïençage,
- 74 % comme fissure, ce qui témoigne d'une surodétection systématique des défauts sur les zones saines.

V.5.3. La courbe F1-confiance:

La courbe F1-Confidence illustrée par la figure V.17 ci-dessous, permet d'évaluer la performance du modèle YOLOv8n en fonction du seuil de confiance appliqué aux prédictions. Elle constitue un outil essentiel pour analyser l'équilibre entre précision et rappel à différents niveaux de certitude dans la détection.

L'Axe des abscisses (Confidence) : représente le seuil de confiance appliqué aux prédictions du modèle, allant de 0.0 à 1.0. Plus le seuil est élevé, plus le modèle doit être sûr de sa prédiction pour la valider.

L'Axe des ordonnées (F1-score) : indique le score F1, qui combine la précision (précision des détections) et le rappel (taux de détection des objets réels).

Un F1-score élevé traduit un bon compromis entre éviter les faux positifs et minimiser les oublis.

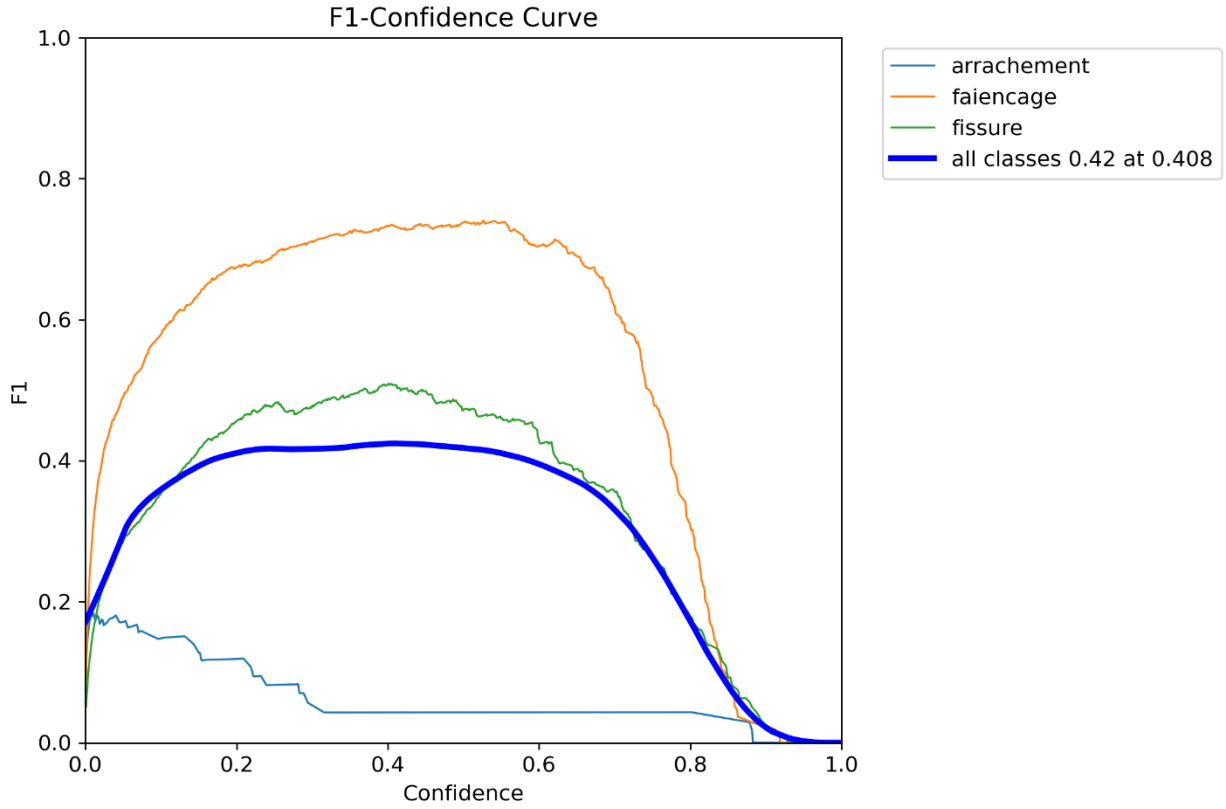


Figure V.17: la courbe de F1 de Yolov8m

V.5.3.1. Analyse des courbes par classe

V.5.3.1.1. Classe faïençage (courbe orange)

Cette courbe présente la meilleure performance parmi toutes les classes. Le pic du score F1 atteint environ 0,72 à 0,74, pour un seuil de confiance situé entre 0,4 et 0,6. La stabilité du score F1 sur une large plage de seuils (jusqu'à ~0,6-0,7) démontre que le modèle est fiable, robuste et confiant pour la détection de faïençage.

V.5.3.1.2. Classe fissure (courbe verte)

Elle se classe en deuxième position en termes de performance. Le score F1 maximal est d'environ 0,48 à 0,50, pour un seuil similaire. Bien que moins performant que pour faïençage, ce résultat demeure suffisant pour certaines applications où une détection partielle est tolérable.

V.5.3.1.3. Classe arrachement (courbe bleu clair)

Il s'agit de la classe la moins bien détectée. Le F1-score reste nettement inférieur à 0,2, atteignant un maximum autour de 0,18-0,19. La courbe chute rapidement avec l'augmentation du seuil de confiance, ce qui indique que le modèle manque de fiabilité pour cette catégorie, même à faible seuil.

V.5.3.1.4. Moyenne toutes classes (All classes, courbe bleue foncée épaisse)

Cette courbe reflète la performance globale du modèle sur l'ensemble des classes. Le point optimal est atteint à un seuil de confiance de 0,408, avec un score F1 de 0,42. Ce point représente le meilleur compromis entre précision et rappel global sur ce jeu de données.

V.5.4. Courbe de précision :

La figure V.18 ci-dessous montre la courbe Précision-Confidence (ou P-curve), qui permet d'évaluer la performance du modèle YOLOv8 en fonction du niveau de confiance associé à ses prédictions. Cette courbe est nécessaire pour comprendre le compromis entre précision et seuil de confiance, et pour déterminer un seuil optimal à partir duquel les prédictions sont fiables.

L'Axe des abscisses (Confidence) : représente le seuil de confiance minimal requis pour qu'une détection soit considérée comme valide. Les valeurs s'échelonnent de 0.0 à 1.0.

L'Axe des ordonnées (Precision) : indique la précision au sens classique :

Une précision élevée reflète une capacité du modèle à limiter les fausses détections.

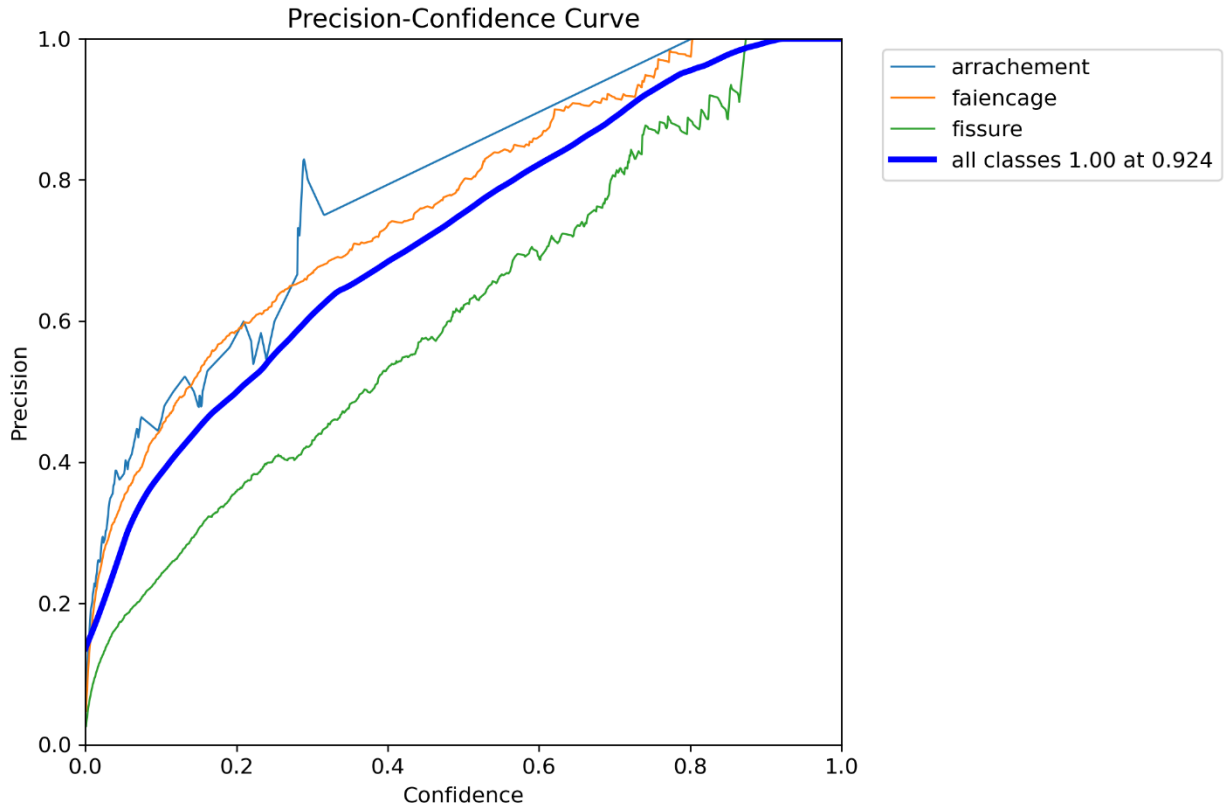


Figure V.18: la courbe de précision de Yolov8m

V.5.4.1. Analyse des courbes par classe

V.5.4.1.1. Classe faïençage

Cette courbe affiche la meilleure performance de précision parmi toutes les classes. La précision initiale est très élevée, avoisinant 1.0 pour des seuils de confiance élevés. Elle demeure supérieure à 0.8 sur une grande plage de seuils (de 0,6 à 0,8), ce qui indique que lorsque le modèle prédit un faïençage, il a très peu de chances de se tromper.

V.5.4.1.2. Classe fissure

La courbe présente également une bonne précision, généralement comprise entre 0.6 et 1.0 sur un seuils de de 0,4-0,8. Cela montre que les prédictions de fissures sont fiables, bien que légèrement moins précises que pour faïençage.

V.5.4.1.3. Classe arrachement

La courbe est irrégulière et montre des valeurs entre0,1-0,8 pour un seuil de confiance inférieure à 0,4, et dépassent 0,8 après une confiance de 0, 4., ce qui indique que le modèle détecte un peut mieux cette classe.

V.5.4.1.4. Courbe globale- toutes classes (épaisse, bleu foncé)

Elle synthétise la précision moyenne du modèle sur l'ensemble des classes. La valeur maximale atteint 1.0 à un seuil de confiance de 0.924. Cela signifie qu'à ce niveau, le modèle ne commet quasiment aucune fausse détection, mais il est alors extrêmement restrictif. Cette situation illustre le compromis classique entre précision et rappel : une précision parfaite s'accompagne souvent d'un taux de détection très faible.

Dans ce cas d'étude, la classe « faïençage » se distingue par une excellente performance, tandis que les classes "fissure" et surtout "arrachage" présentent plus de variabilité. Le seuil optimal situé à 0.924 garantit une précision maximale, critère fondamental dans une application de détection automatique sur infrastructures routières.

V.5.5. La courbe Recall-Confidence:

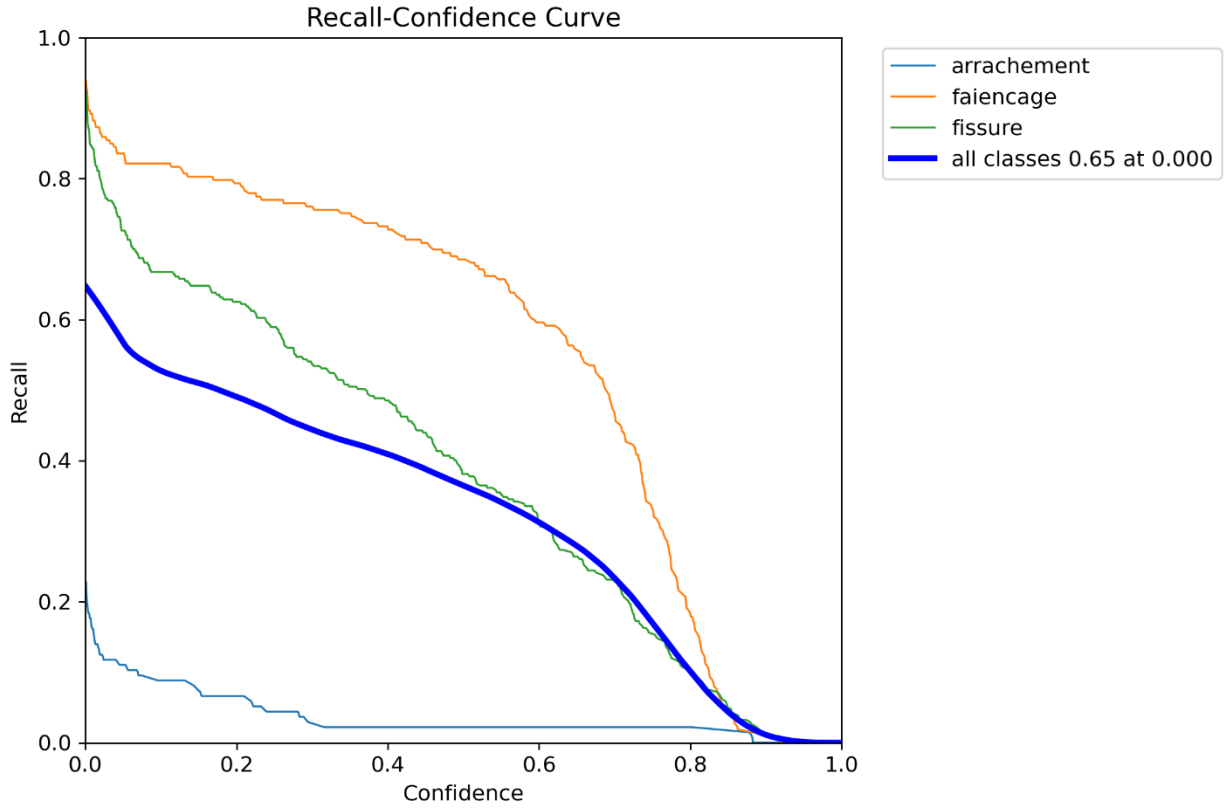


Figure V.19: La courbe Recall–Confidence de Yolov8m

La courbe Recall–Confidence présentée par la figure V.19 ci-dessus, illustre la capacité du modèle YOLOv8n à détecter correctement les objets pertinents en fonction du seuil de confiance appliqué aux prédictions. Elle permet de quantifier, pour chaque classe, la proportion des instances réelles qui ont été correctement identifiées (rappel) en fonction de la rigueur avec laquelle le modèle accepte ou rejette une détection.

L'Axe des abscisses (Confidence) : représente le seuil minimal requis pour valider une prédiction. Un seuil bas permet de retenir davantage de détections, tandis qu'un seuil élevé filtre les prédictions les moins sûres.

Un rappel élevé signifie que le modèle détecte la majorité des objets réellement présents, même si cela se fait parfois au détriment de la précision.

V.5.5.1. Analyse des courbes par classe

V.5.5.1.1. Classe faïençage :

La courbe de rappel pour cette classe est la plus performante. A un seuil de confiance nul (0.0), le rappel est environ 0,95, ce qui indique que le modèle détecte pratiquement toutes les instances de faïençage. Même en augmentant le seuil jusqu'à 0.6, le rappel reste supérieur à 0.6, traduisant une grande robustesse de détection, notamment à faible ou moyenne confiance. Cela confirme la fiabilité du modèle pour cette classe, déjà observée dans les courbes F1 et les matrices de confusion.

V.5.5.1.2. Classe fissure :

Le rappel initial est aussi élevé environ 0,92 pour des seuils proches de zéro. La courbe décroît plus rapidement que celle de faïençage, mais le rappel reste au-dessus de 0.4 jusqu'à un seuil d'environ 0.5, ce qui témoigne d'une capacité modérée à bonne à détecter les fissures, sous réserve d'un seuil de confiance adapté.

V.5.5.1.3. Classe arrachement :

Il s'agit de la classe la plus problématique. Le rappel débute à un niveau très bas environ 0.22 à un seuil nul, et chute brutalement à 0,8 pour les seuils 0.2 à 0.3 . Cela indique une incapacité du modèle à identifier correctement les arrachements, quelles que soient les conditions.

V.5.5.1.4. Courbe globale (All classes) – Moyenne pondérée

À un seuil nul, le modèle atteint un rappel global de 0.65. Autrement dit, en acceptant toutes les détections sans filtrage, YOLOv8m est capable de retrouver 65 % des objets réellement présents. Toutefois, cette approche s'accompagne d'une baisse significative de la précision, rendant ce niveau de rappel difficilement exploitable en pratique sans calibration.

V.5.6. La courbe Precision-Recall :

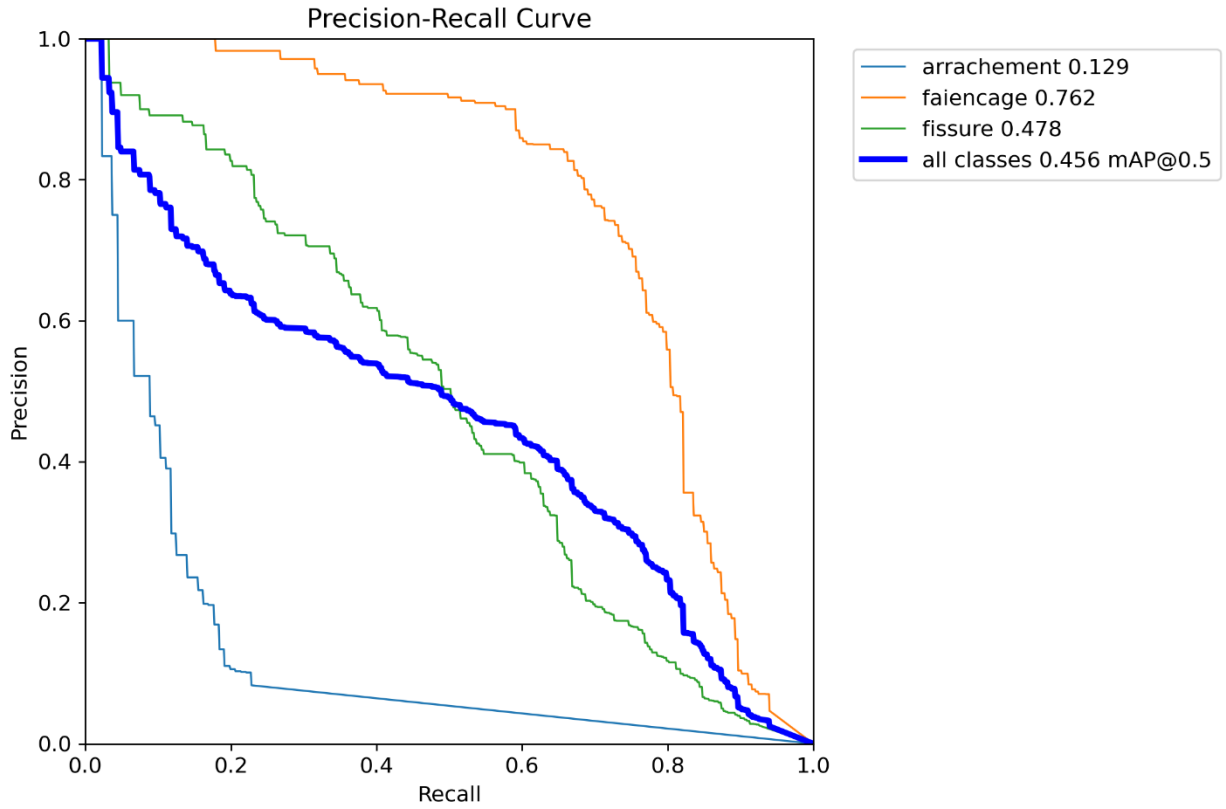


Figure V.20: La courbe Precision–Recall (PR) de Yolov8m

La courbe Precision–Recall (PR) illustrée par la figure V.20 ci-dessus, permet de visualiser le compromis entre la précision (exactitude des prédictions positives) et le rappel (capacité à détecter toutes les instances positives) pour chaque classe. Elle est particulièrement utile dans le contexte des ensembles de données déséquilibrés, où une simple mesure de précision ou d'exactitude peut être trompeuse.

L'Axe des abscisses (Recall) : représente le taux de détection des objets réellement présents. Plus on avance vers la droite, plus le modèle détecte d'instances correctes.

L'Axe des ordonnées (Precision) : mesure la fiabilité des détections positives. Une valeur élevée indique un faible taux de faux positifs.

Une courbe idéale conserve une précision élevée sur une large gamme de rappels. L'aire sous la courbe (Average Precision, AP) est utilisée pour résumer la performance de chaque classe.

V.5.6.1. Analyse par classe

V.5.6.1.1. Classe faïençage :

AP (Average Precision) **: 0.762,** La courbe est la plus performante de toutes, témoignant d'un excellent équilibre entre précision et rappel. La précision reste proche de 0.9 même pour les rappels élevés (0.6 à 0.7), ce qui indique que le modèle détecte efficacement la majorité des faïençages tout en limitant les erreurs.

V.5.6.1.2. Classe fissure :

AP : 0.478, la performance est modérée, mais significative. La précision décroît progressivement à mesure que le rappel augmente. Pour un rappel de 0.6, la précision avoisine 0.4, indiquant une capacité correcte à identifier les fissures, avec un compromis acceptable.

V.5.6.1.3. Classe arrachement :

AP : 0.129, Il s'agit de la classe la moins bien détectée par le modèle. La courbe chute brutalement de 1,0 à 0,1 pour un rappel inférieur à 0,2 : la précision est faible à tous les niveaux de rappel. Ce résultat traduit une incapacité manifeste du modèle à apprendre des représentations discriminantes pour la classe des arrachements.

V.5.6.1.4. Courbe moyenne (all classes) – Performance globale :

mAP@0.5 : 0.458, Cette métrique représente la moyenne des AP de toutes les classes (mean Average Precision). Une valeur de 0.458 indique une performance globale modérée, principalement pénalisée par la très faible performance du modèle sur la classe arrachement.

V.5.7. Train/val :

: Le modèle YOLOv8n a été entraîné sur 100 époques. L'analyse des courbes générées (losses et métriques) permet d'évaluer l'évolution de son comportement en termes d'apprentissage, de généralisation, et de détection par classe. Cette section synthétise les principales observations issues des figures générées dans results.jpg. La figure V.21 ci-après montre l'évolution des pertes et des métriques

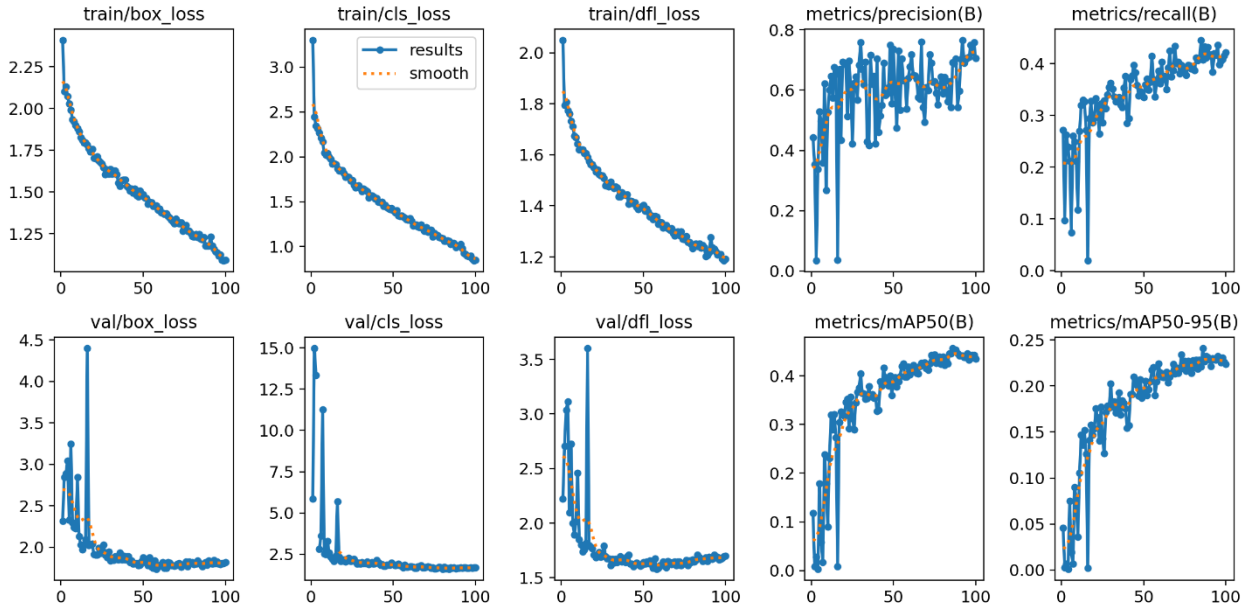


Figure V.21:les courbes losses et métriques de Yolov8m

V.5.7.1. Perte de localisation (box_loss)

V.5.7.1.1. Entraînement (train/box_loss) :

la perte diminue progressivement de 2.4 à environ 1.1, traduisant une amélioration continue dans la précision de localisation des boîtes englobantes sur les données d'apprentissage.

V.5.7.1.2. Validation (val/box_loss) :

la courbe suit une tendance similaire, bien que plus bruitée, se stabilisant autour de 1.75 après 40 époques. Ce qui confirme que la convergence parallèle des pertes d'entraînement et de validation suggère une bonne capacité de généralisation du modèle, sans signe évident de surapprentissage.

V.5.7.2. Perte de classification (cls_loss) :

V.5.7.2.1. Entraînement :

la perte de classification chute de plus de 2.6 à environ 0.6, montrant que le modèle apprend à mieux attribuer la bonne classe aux objets détectés.

V.5.7.2.2. Validation :

initialement très bas environ 2.6 elle diminue rapidement pour se stabiliser vers 1.6–1.7.

Ce qui exprime que la forte décroissance initiale est typique d'une phase d'adaptation rapide.

V.5.7.3. Perte DFL (Distribution Focal Loss)

V.5.7.3.1. Entraînement (train/dfl_loss) :

décroissance continue de 2.05 à environ 1.2.

V.5.7.3.2. Validation (val/dfl_loss) :

valeur initiale moyenne environ 2,6, suivie d'une chute rapide et d'une stabilisation vers 1.7.

Le pic montre une stabilité du modèle au démarrage de l'apprentissage, rapidement corrigée. Ensuite, la courbe suit une tendance similaire aux autres pertes, ce qui reflète une bonne stabilité générale de l'apprentissage.

V.5.7.4. Evolution de la précision et du rappel

V.5.7.4.1. Précision (metrics/precision(B)) :

la courbe évolue de 0.0 vers un plateau autour de 0.32–0.34, Le modèle démontre une précision supérieure à celle de YOLOv8n, traduisant une réduction des faux positifs. Le rappel, quant à lui, continue de croître, suggérant une capacité accrue à détecter les objets réels. Le comportement est caractéristique d'un modèle bien équilibré.

V.5.7.4.2. Rappel (metrics/recall(B)) :

progression régulière de 0.08 à environ 0.42, indiquant une amélioration stable de la capacité du modèle à détecter les objets réels.

V.5.7.5. Evaluation par mAP

V.5.7.5.1. mAP@0.5 :

augmente progressivement jusqu'à 0.43–0.45, ce qui signifie que près de la moitié des prédictions atteignent un chevauchement suffisant avec les objets réels.

V.5.7.5.2. mAP@0.5:0.95 :

atteint environ 0.2–0.22, ce qui est une valeur un peut faible pour un modèle de type "medium".

Ces résultats indiquent un bon compromis entre exactitude et robustesse, avec une capacité à maintenir des performances stables même en présence d'un seuil élevé. YOLOv8m parvient à préserver une bonne sensibilité spatiale tout en conservant une classification pertinente.

V.6. Yolov8l :

V.6.1. Matrice de confusion :

La figure V.22 illustre la matrice de confusion obtenue lors de l'évaluation du modèle YOLOv8l. Cette matrice permet d'analyser les performances du modèle en termes de classification des différentes classes de dégradations.

- Les **lignes** de la matrice correspondent aux **classes prédites** par le modèle.
- Les **colonnes** représentent les **classes réelles**, c'est-à-dire celles effectivement présentes dans les données de test.
- Chaque **cellule** indique le nombre d'exemples d'une classe réelle spécifique (colonne) que le modèle a classés dans une autre classe (ligne).
- Les **valeurs situées sur la diagonale principale** (de haut en bas) correspondent aux **prédictions correctes** (ou vrais positifs), tandis que les **valeurs en dehors de cette diagonale** traduisent des **erreurs de classification** (confusions entre classes).

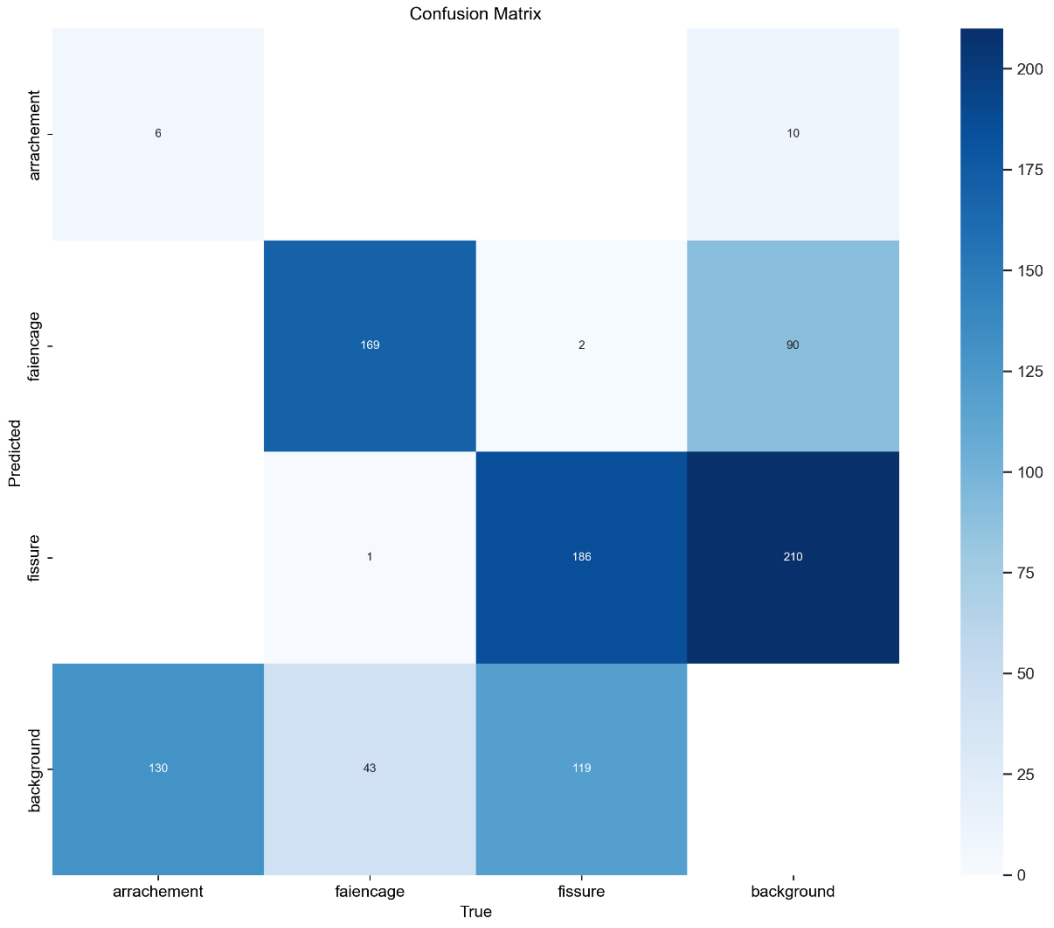


Figure V.22: matrice de confusion de Yolov8l

V.6.1.1. Classes analysées :

Le modèle YOLOv8l a été entraîné pour classifier quatre catégories de données issues d'images aériennes : Arrachement, faïençage, fissure et background (zones sans défaut).

V.6.1.2. Analyse détaillée des performances par classe:

V.6.1.2.1. Classe "arrachement" :

- **Prédictions correctes**: 6 véritables arrachements ont été identifiés correctement.
- **Erreurs de classification**:130 instances d'arrachement ont été erronément classées comme "background".
  Cette erreur est significative et traduit une grande confusion entre l'objet d'intérêt et l'arrière-plan, ce qui limite fortement l'efficacité du modèle sur cette classe.

V.6.1.2.2. Classe "faïençage" :

- **Prédictions correctes**: 169 instances, soit une excellente performance.
- **Erreurs de classification**:
  - 1 instance a été classée à tort comme fissure (erreur négligeable).
  - 43 instances ont été confondues avec le background, ce qui révèle une perte modérée mais non négligeable d'informations.

V.6.1.2.3. Classe "fissure" :

- **Prédictions correctes**: 186 instances, ce qui reflète une très bonne capacité de détection.
- **Erreurs de classification**:
  - 2 instances ont été confondues avec faïençage (erreur mineure).
  - 119 instances ont été prédites comme background, ce qui constitue une erreur de classification importante.

V.6.1.2.4. Classe "background" :

- **Prédictions correctes**: 0 instance. Aucune zone réellement considérée comme background n'a été reconnue comme telle par le modèle.
- **Erreurs de classification**:
  - 10 instances ont été prédites comme arrachement,
  - 90 comme faïençage,
  - 210 comme fissure.

Cette distribution est particulièrement préoccupante, car elle montre que le modèle a systématiquement échoué à identifier les zones de background, les classant à tort comme des défauts.

V.6.2. Matrice de confusion normalisée:

La matrice de confusion normalisée comme illustrée dans la figure V.23 ci-desous représente les performances du modèle YOLOv8l en termes proportionnels plutôt qu'en valeurs absolues. Chaque cellule de la matrice indique la part des instances d'une classe réelle donnée ayant été classées dans une catégorie prédite.

La valeur (i, j) correspond à la proportion d'instances réellement de la classe j qui ont été prédites comme étant de la classe i.

Ainsi, la somme des valeurs de chaque colonne est égale à 1 (ou 100 %), car elle traduit la distribution des prédictions effectuées pour une classe réelle spécifique. Les valeurs situées sur la diagonale principale (du haut gauche vers le bas droit) représentent

le rappel (ou sensibilité) pour chaque classe, c'est-à-dire la capacité du modèle à détecter correctement les instances appartenant à cette classe.

Les valeurs hors diagonale traduisent les erreurs de classification, c'est-à-dire les cas où le modèle a confondu une classe avec une autre.

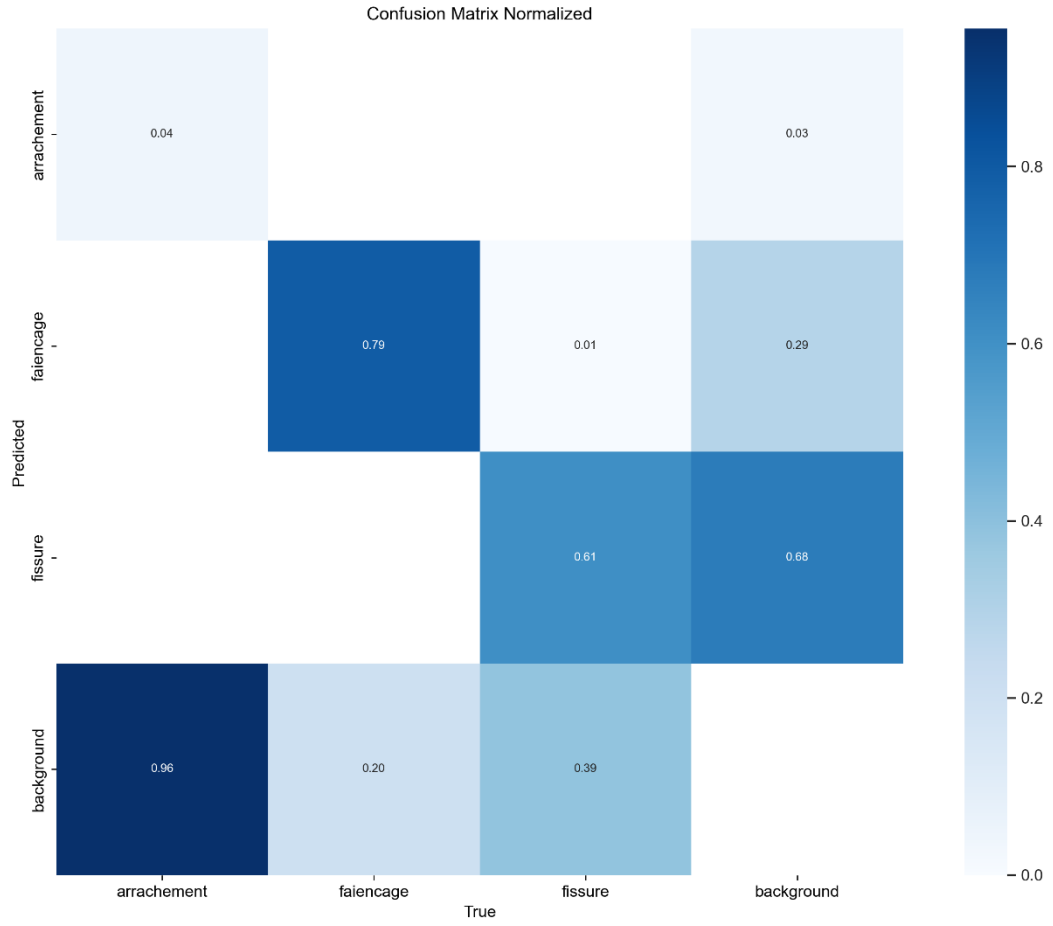


Figure V.23: matrice de confusion normalisée de Yolov8l

V.6.2.1. Classes évaluées:

Le modèle YOLOv8n a été entrainé pour classifier les quatre catégories suivantes : (arrachement, faïençage, fissure et background (zone sans défaut).

V.6.2.2. Analyse détaillée des performances (rappel/sensibilité):

V.6.2.2.1. Classe "arrachement" :

- **Rappel** : 0,04 (soit 4 %) Le modèle présente une capacité extrêmement limitée à détecter cette classe
- **Erreurs de classification** : 96 % des arrachements réels sont mal classés comme background, révélant une confusion majeure avec l'arrière-plan.

Ce faible rappel reflète probablement une carence en données représentatives pour cette classe ou une ressemblance visuelle importante avec le fond, rendant sa détection difficile.

V.6.2.2.2. Classe "faïençage" :

- **Rappel** : 0,79 (soit 79 %) Il s'agit de la meilleure performance enregistrée pour ce modèle. YOLOv8l démontre une excellente capacité de détection du faïençage.
- **Erreurs de classification** :
  - 1 % sont confondus avec fissure (erreur négligeable),
  - 20 % sont mal classés comme background, ce qui reste une proportion non négligeable, traduisant une perte de rappel partielle.

V.6.2.2.3. Classe "fissure" :

- **Rappel** : 0,61 (soit 61 %) Le modèle parvient à détecter près des deux tiers des fissures, ce qui constitue une performance correcte à modérée.
- **Erreurs de classification** :
  - 1 % sont confondues avec faïençage,
  - 39 % sont classées comme background, ce qui représente une source importante de faux négatifs.

V.6.2.2.4. Classe "background" :

- **Rappel** : 0,00 (soit 0 %) Ce score reflète une incapacité totale du modèle à reconnaître les zones sans dégradation. Le background est systématiquement interprété comme un défaut, ce qui engendre une surodétection problématique.
- **Erreurs de classification**:
  - 3 % des background sont classés comme arrachement,
  - 29 % comme faïençage,
  - 68 % comme fissure.

V.6.3. La courbe F1-confiance:

La courbe F1-Confidence illustrée par la figure V.24 ci-dessous, permet d'évaluer la performance du modèle YOLOv8n en fonction du seuil de confiance appliqué aux prédictions. Elle constitue un outil essentiel pour analyser l'équilibre entre précision et rappel à différents niveaux de certitude dans la détection.

L'Axe des abscisses (Confidence) : représente le seuil de confiance appliqué aux prédictions du modèle, allant de 0.0 à 1.0. Plus le seuil est élevé, plus le modèle doit être sûr de sa prédiction pour la valider.

L'Axe des ordonnées (F1-score) : indique le score F1, qui combine la précision (précision des détections) et le rappel (taux de détection des objets réels).

Un F1-score élevé traduit un bon compromis entre éviter les faux positifs et minimiser les oublis.

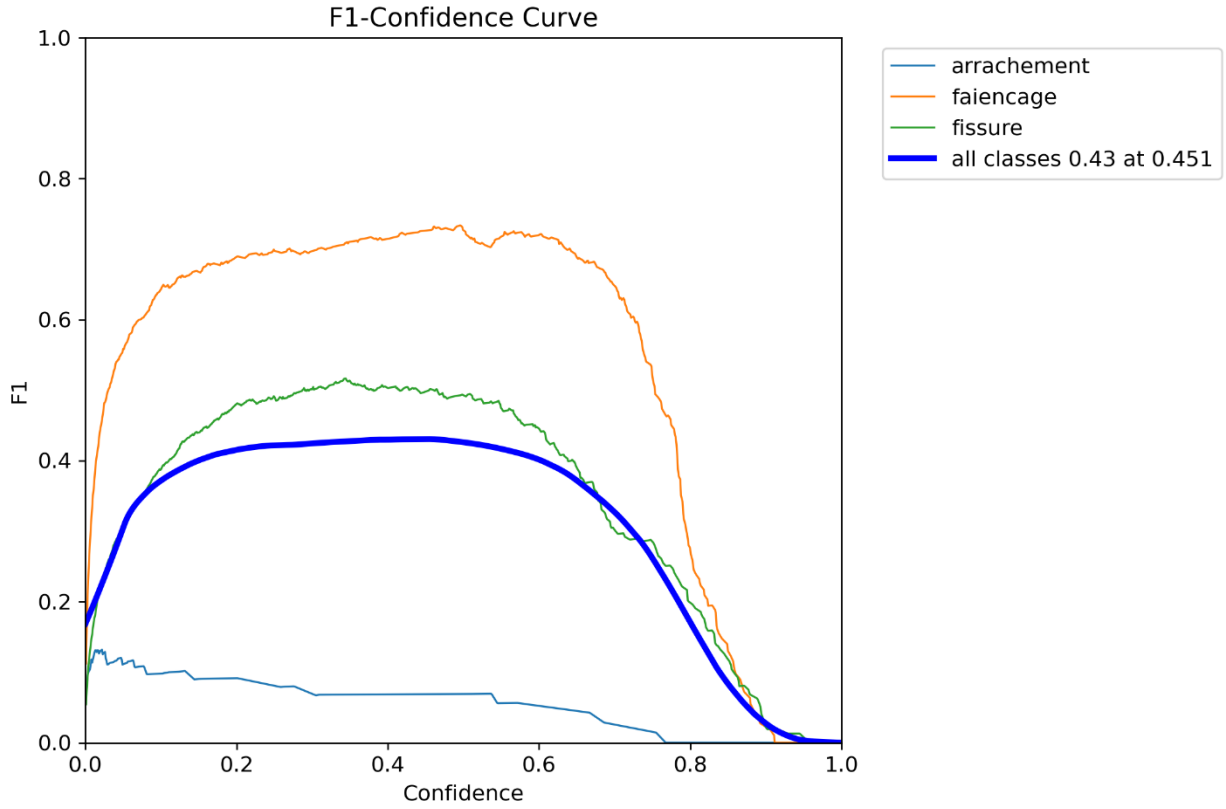


Figure V.24: la courbe F1 de Yolov8l

V.6.3.1. Analyse des courbes par classe

V.6.3.1.1. Classe faïençage (courbe orange)

Cette courbe présente la meilleure performance parmi toutes les classes. Le pic du score F1 atteint environ 0,72 à 0,73, pour un seuil de confiance situé entre 0,4 et 0,6. La stabilité du score F1 sur une large plage de seuils (0,1-0,7) démontre que le modèle est fiable, robuste et confiant pour la détection de faïençage.

V.6.3.1.2. Classe fissure (courbe verte)

Elle se classe en deuxième position en termes de performance. Le score F1 maximal est d'environ 0,50 à 0,52 situé entre 0,2 et 0,4. . Bien que moins performant, le modèle conserve une certaine stabilité, illustrant une capacité modérée à détecter les fissures tout en limitant les erreurs de classification. Cette courbe suggère un équilibre partiel entre les deux composantes du F1-score, mais avec une marge d'optimisation notable.

V.6.3.1.3. Classe arrachement (courbe bleu clair)

Il s'agit de la classe la moins bien détectée. Le F1-score reste nettement inférieur à 0,2, atteignant un maximum autour de 0,12 à 0,14. La courbe chute rapidement avec l'augmentation du seuil de confiance, ce qui indique que le modèle manque de fiabilité pour cette catégorie, même à faible seuil.

V.6.3.1.4. Moyenne toutes classes (All classes, courbe bleue foncée épaisse) :

Cette courbe agrégée synthétise la performance moyenne du modèle sur l'ensemble des classes. Elle atteint son maximum à un F1-score de 0.43 pour un seuil de confiance de 0.451, ce qui constitue le meilleur compromis global observé entre précision et rappel. Toutefois, ce score global demeure relativement faible dans le contexte des tâches de détection d'objets, ce qui suggère un potentiel d'amélioration significatif, notamment dans la gestion des classes complexes.

V.6.4. Courbe de précision :

La figure V.25 ci-dessous montre la courbe Précision-Confidence (ou P-curve), qui permet d'évaluer la performance du modèle YOLOv8 en fonction du niveau de confiance associé à ses prédictions. Cette courbe est nécessaire pour comprendre le compromis entre précision et seuil de confiance, et pour déterminer un seuil optimal à partir duquel les prédictions sont fiables.

L'Axe des abscisses (Confidence) : représente le seuil de confiance minimal requis pour qu'une détection soit considérée comme valide. Les valeurs s'échelonnent de 0.0 à 1.0.

L'Axe des ordonnées (Precision) : indique la précision au sens classique :

Une précision élevée reflète une capacité du modèle à limiter les fausses détections.

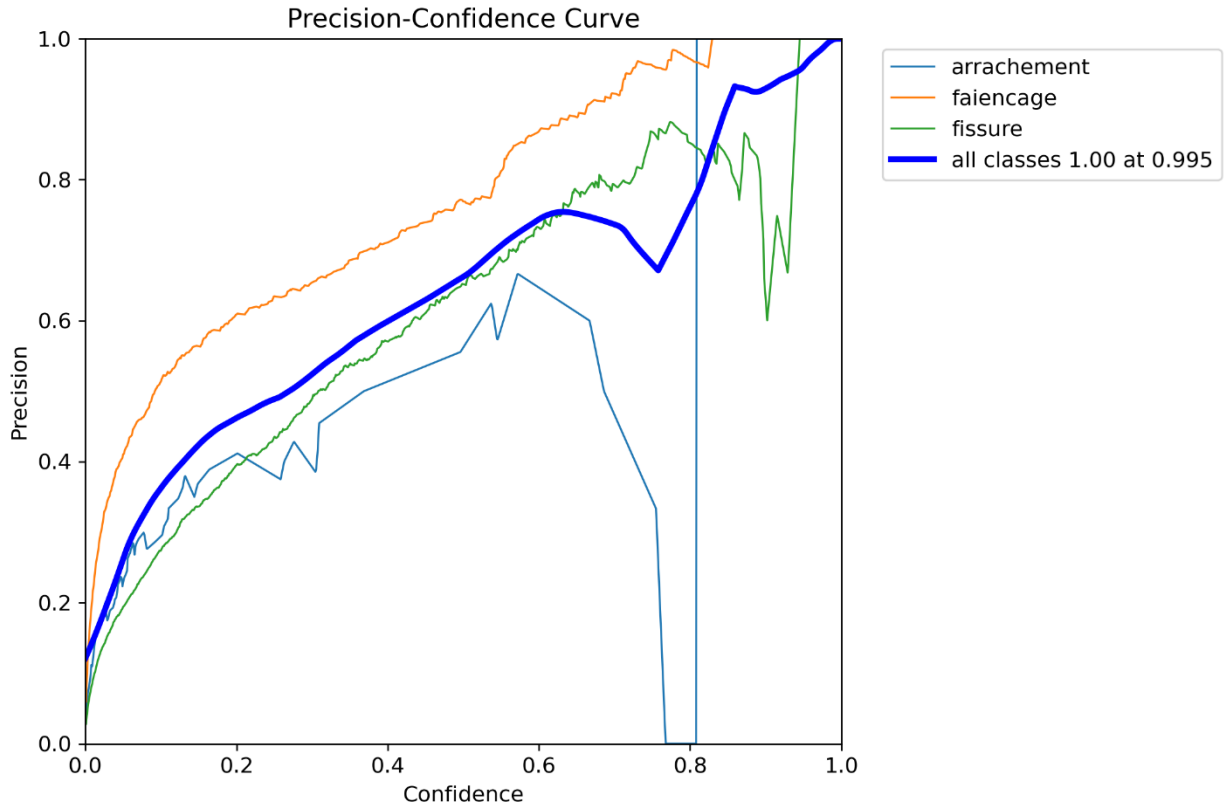


Figure V.25: la courbe de précision de Yolov8l

V.6.4.1. Analyse des courbes par classe

V.6.4.1.1. Classe faïençage

Cette courbe se distingue par sa stabilité et sa précision élevée sur une large gamme de seuils. Dès les niveaux de confiance faibles à modérés (environ 0.2 à 0.8), la précision demeure supérieure à 0.6, et atteint des valeurs proches de 1.0 aux seuils 0,7-0,8. Ce profil indique une grande fiabilité

du modèle pour cette classe : lorsqu'il prédit un faïençage, il est presque toujours exact, même avec un niveau de confiance modéré.

V.6.4.1.2. Classe fissure

La courbe présente également une bonne précision, généralement comprise entre 0.5 et 0.9 sur une plage de seuils 0,4-0,8. Cela montre que les prédictions de fissures sont fiables, bien que légèrement moins précises que pour faïençage.

V.6.4.1.3. Classe arrachement

La courbe correspondant à cette classe révèle une forte instabilité. Si la précision peut atteindre ponctuellement des niveaux acceptables (jusqu'à 0.70) pour des seuils de confiance très élevés, elle chute considérablement pour des seuils plus courants (entre 0.1 et 0.6), oscillant souvent entre 0.4 et 0.6. Cette variabilité importante souligne le manque de robustesse du modèle pour cette classe : aux seuils habituellement utilisés, la fiabilité des prédictions d'arrachements demeure faible.

V.6.4.1.4. Courbe globale- toutes classes (épaisse, bleu foncé)

La courbe moyenne représente la précision globale du modèle sur l'ensemble des classes. Elle atteint un maximum de 1.0 à un seuil de confiance très élevé (0.995). Si ce résultat témoigne de la capacité du modèle à effectuer des prédictions extrêmement précises, il traduit aussi une certaine prudence extrême : à ce niveau, seules très peu de détections sont effectuées, mais elles sont toutes correctes. Cela indique que le modèle devient extrêmement sélectif, au prix probable d'un rappel très faible.

V.6.5. La courbe Recall-Confidence:

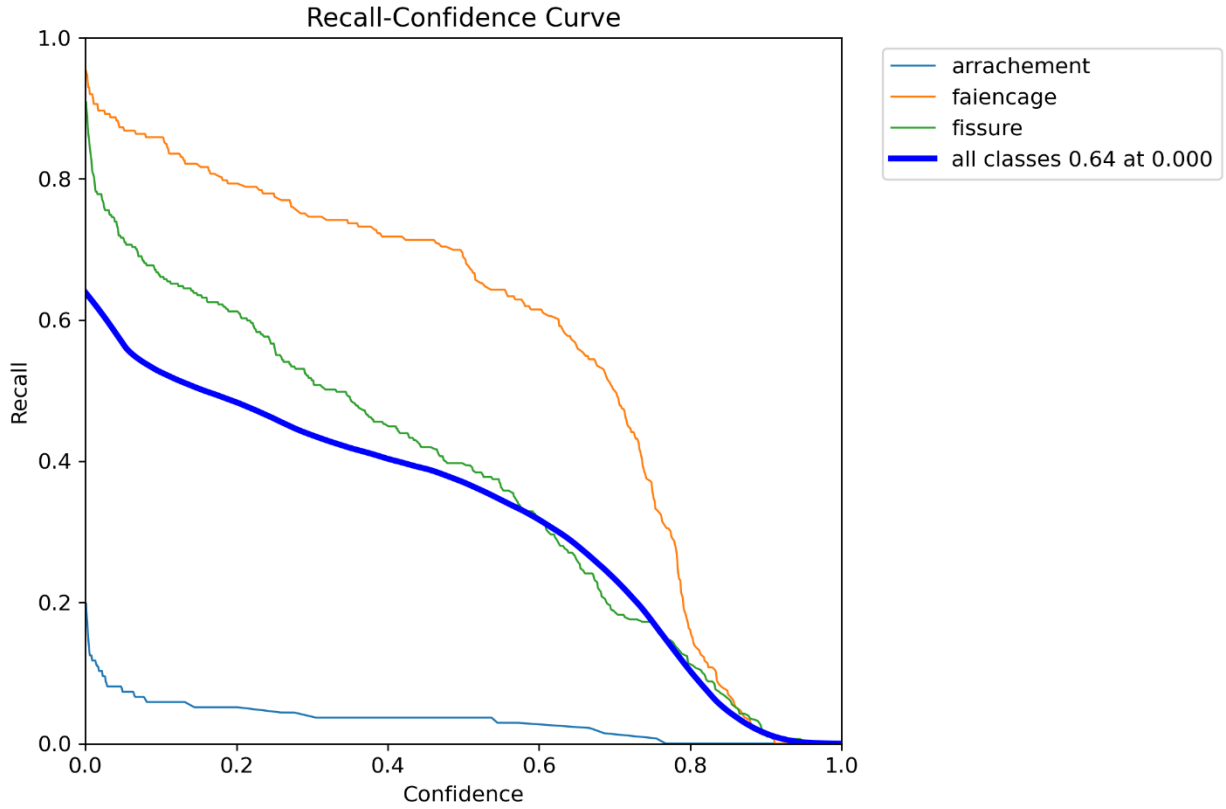


Figure V.26: la courbe de rappel de Yolov8l

La courbe Recall–Confidence présentée par la figure V.26 ci-dessus, illustre la capacité du modèle YOLOv8l à détecter correctement les objets pertinents en fonction du seuil de confiance

appliqué aux prédictions. Elle permet de quantifier, pour chaque classe, la proportion des instances réelles qui ont été correctement identifiées (rappel) en fonction de la rigueur avec laquelle le modèle accepte ou rejette une détection.

L'Axe des abscisses (Confidence) : représente le seuil minimal requis pour valider une prédiction. Un seuil bas permet de retenir davantage de détections, tandis qu'un seuil élevé filtre les prédictions les moins sûres.

Un rappel élevé signifie que le modèle détecte la majorité des objets réellement présents, même si cela se fait parfois au détriment de la précision.

V.6.5.1. Analyse des courbes par classe

V.6.5.1.1. Classe faïençage :

Cette courbe présente les meilleures performances en termes de rappel. Elle démarre très haut, avec un rappel proche de 1.0 pour les seuils de confiance les plus bas, et conserve une valeur supérieure à 0.6 jusqu'à des seuils autour de 0.6–0.7. Cela traduit une excellente capacité du modèle à identifier la grande majorité des faïençages, même en présence d'un seuil de confiance modéré. La décroissance est progressive, ce qui dénote une stabilité appréciable.

V.6.5.1.2. Classe fissure :

La performance du modèle sur cette classe est également satisfaisante. Le rappel débute, là encore, à des valeurs élevées pour des seuils faibles, mais décroît plus rapidement que celui de la classe faïençage. Pour des seuils moyens (environ 0.4–0.5), le rappel demeure autour de 0.4 à 0.5, ce qui reflète une bonne capacité de détection, bien que moins robuste que pour le faïençage.

V.6.5.1.3. Classe arrachement :

Cette courbe révèle une performance nettement inférieure. Le rappel est faible dès les premiers seuils (autour de 0.2 à un seuil de confiance nul) et chute brutalement avec l'augmentation du seuil, atteignant presque zéro dès 0.3. Cette évolution traduit une difficulté manifeste du modèle à localiser correctement les arrachements, même en adoptant une approche permissive du seuil de confiance. La courbe met ainsi en évidence une classe particulièrement mal détectée.

V.6.5.1.4. Courbe globale (All classes) – Moyenne pondérée :

La courbe moyenne indique un rappel global de 0.64 à un seuil de confiance de 0.0. Ce score reflète le comportement agrégé du modèle sur toutes les classes : une sensibilité élevée en l'absence de filtre, mais qui pourrait être associée à une baisse notable de précision dans ce cas extrême. La courbe descend ensuite de manière régulière, illustrant la perte de détections à mesure que le seuil se resserre.

V.6.6. La courbe Precision-Recall :

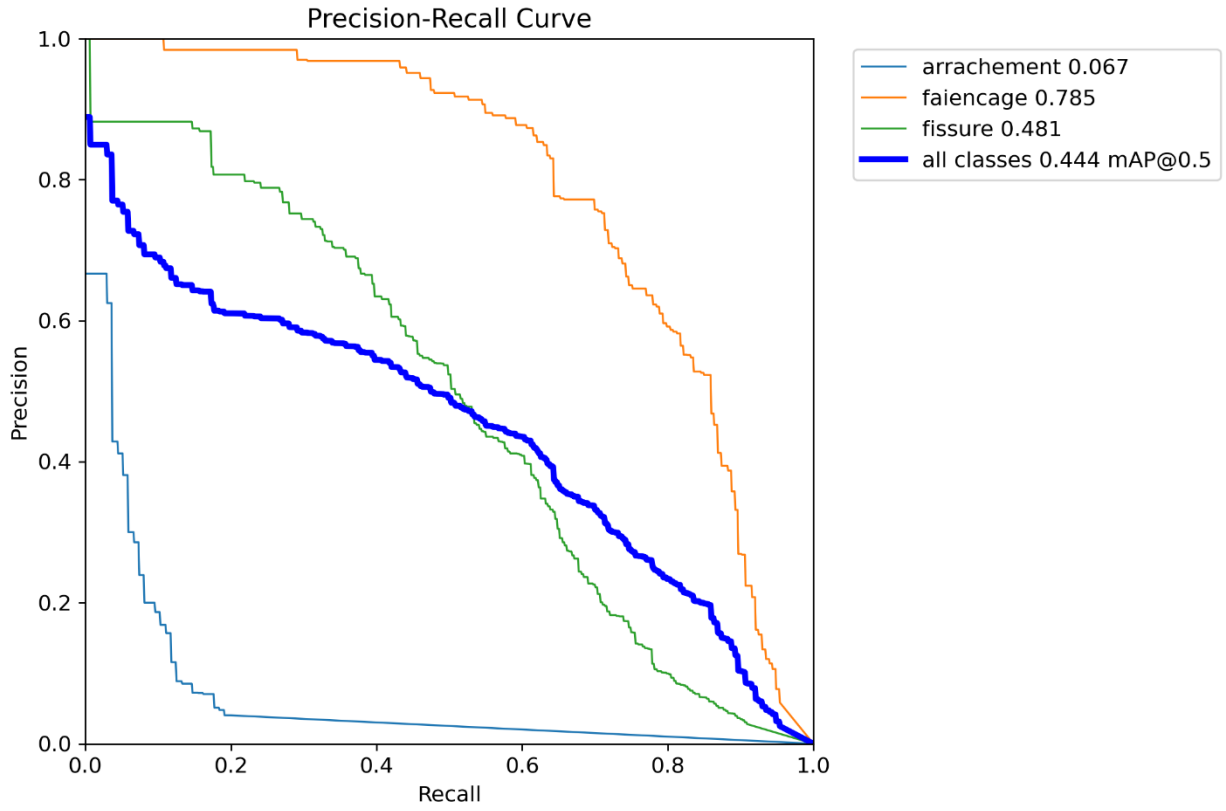


Figure V.27: la courbe de précision-rappel de Yolov8l

La courbe Precision–Recall (PR) illustrée par la figure V.27 ci-dessus, permet de visualiser le compromis entre la précision (exactitude des prédictions positives) et le rappel (capacité à détecter toutes les instances positives) pour chaque classe. Elle est particulièrement utile dans le contexte des ensembles de données déséquilibrés, où une simple mesure de précision ou d'exactitude peut être trompeuse.

L'Axe des abscisses (Recall) : représente le taux de détection des objets réellement présents. Plus on avance vers la droite, plus le modèle détecte d'instances correctes.

L'Axe des ordonnées (Precision) : mesure la fiabilité des détections positives. Une valeur élevée indique un faible taux de faux positifs.

Une courbe idéale conserve une précision élevée sur une large gamme de rappels. L'aire sous la courbe (Average Precision, AP) est utilisée pour résumer la performance de chaque classe.

V.6.6.1. Analyse par classe

V.6.6.1.1. Classe faïençage :

Il s'agit de la classe la mieux détectée par le modèle. La courbe est étendue, régulière et conserve des valeurs de précision très élevées (souvent > 0.9), même pour des niveaux de rappel proches de 0.8. Avec une AP de 0.785, le modèle démontre une capacité remarquable à concilier précision et rappel pour cette classe. Ce résultat témoigne d'une robustesse significative dans la détection des faïençages.

V.6.6.1.2. Classe fissure :

Le comportement est satisfaisant, bien que nettement en retrait par rapport à la classe précédente. La courbe décline plus rapidement, indiquant une diminution progressive de la précision à mesure que le rappel augmente. Pour atteindre un rappel plus élevé, le modèle doit accepter davantage de fausses détections. L'AP obtenue est de 0.481, traduisant une performance modérée avec une marge d'amélioration notable dans la gestion du compromis entre précision et couverture.

V.6.6.1.3. Classe arrachement :

Cette courbe révèle les plus grandes difficultés du modèle. Elle reste proche de l'axe des abscisses, avec une précision extrêmement faible dès les premiers niveaux de rappel. Même pour des rappels très bas (inférieurs à 0.1), la précision chute rapidement vers zéro. L'AP associée est très faible (0.067), reflétant une incapacité du modèle à détecter cette classe de manière fiable. Ce profil met en évidence un déséquilibre marqué dans la performance du modèle entre les différentes catégories de dégradations.

V.6.6.1.4. Courbe moyenne (all classes) – Performance globale :

La courbe moyenne reflète la performance agrégée du modèle sur l'ensemble des classes. Elle aboutit à un mAP@0.5 de 0.444, valeur considérée comme modérée. Ce score résulte d'une moyenne entre des performances élevées pour certaines classes (comme faïençage) et très faibles pour d'autres (comme arrachement), ce qui souligne l'hétérogénéité du comportement du modèle selon les catégories.

V.6.7. Train/val :

Le modèle YOLOv8l a été entraîné sur 100 époques. L'analyse des courbes générées (losses et métriques) permet d'évaluer l'évolution de son comportement en termes d'apprentissage, de généralisation, et de détection par classe. Cette section synthétise les principales observations issues des figures générées dans results.jpg. La figure V.5 ci-après montre l'évolution des pertes et des métriques

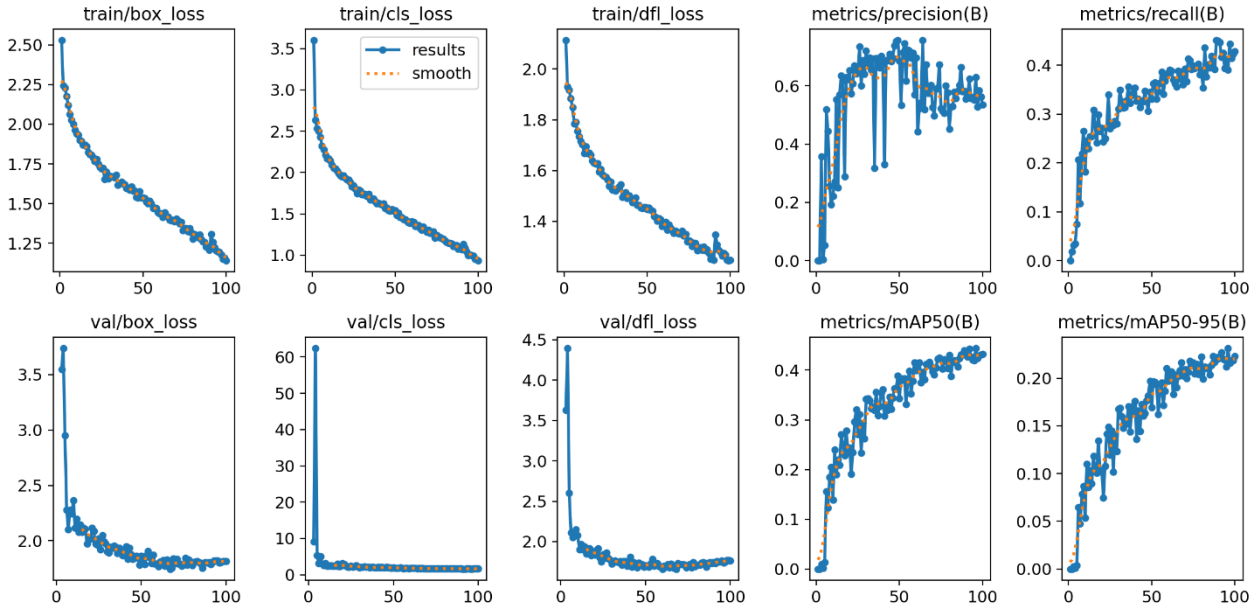


Figure V.28: les courbes de losses et métriques de Yolov8l

V.6.7.1. Perte de localisation (box_loss)

V.6.7.1.1. Entraînement (train/box_loss) :

la perte diminue progressivement de 2.25 à environ 1.15, traduisant une amélioration continue dans la précision de localisation des boîtes englobantes sur les données d'apprentissage.

V.6.7.1.2. Validation (val/box_loss) :

la courbe commence par une valeur de 2,1, et se stabilisant autour de 1.75 à 1.8 après 40 époques. Ce qui confirme que la convergence parallèle des pertes d'entraînement et de validation suggère une bonne capacité de généralisation du modèle, sans signe évident de surapprentissage.

V.6.7.2. Perte de classification (cls_loss) :

V.6.7.2.1. Entraînement :

la perte de classification chute de plus de 3.5 à environ 1.0, montrant que le modèle apprend à mieux attribuer la bonne classe aux objets détectés.

V.6.7.2.2. Validation :

initialement très élevée jusqu'à 60, et elle chute rapidement pour se stabiliser vers 0.5–0.6.

Ce qui exprime que la forte décroissance initiale est typique d'une phase d'adaptation rapide. L'absence de remontée en validation exclut un phénomène d'overfitting. Toutefois, la variabilité restante traduit une possible confusion inter-classes.

V.6.7.3. Perte DFL (Distribution Focal Loss):

V.6.7.3.1. Entraînement (train/dfl_loss) :

décroissance continue de 2.2 à environ 1.2.

V.6.7.3.2. Validation (val/dfl_loss) :

valeur initiale extrêmement haute 4,4 suivie d'une chute rapide et d'une stabilisation vers 1.6.

Le pic au début montre une instabilité temporaire du modèle au démarrage de l'apprentissage, rapidement corrigée. Ensuite, la courbe suit une tendance similaire aux autres pertes, ce qui reflète une bonne stabilité générale de l'apprentissage.

V.6.7.4. Evolution de la précision et du rappel :

V.6.7.4.1. Précision (metrics/precision(B)) :

la courbe évolue de 1.0 vers un plateau autour de 0.65–0.7, mais avec une certaine instabilité (effet de "dents de scie"), signe d'une sensibilité aux fluctuations dans les classes ou les échantillons de validation.

V.6.7.4.2. Rappel (metrics/recall(B)) :

progression régulière de 0.0 à environ 0.45, indiquant une amélioration stable de la capacité du modèle à détecter les objets réels.

Analyse : la croissance conjointe des deux métriques est encourageante. Toutefois, la variabilité de la précision souligne des limites dans la capacité du modèle à distinguer correctement certaines classes, notamment dans les cas de forte similarité visuelle ou de déséquilibre de données.

V.6.7.5. Evaluation par mAP:

V.6.7.5.1. mAP@0.5 :

augmente progressivement jusqu'à 0.41–0.42, ce qui signifie que près de la moitié des prédictions atteignent un chevauchement suffisant avec les objets réels.

V.6.7.5.2. mAP@0.5:0.95 :

atteint environ 0.22–0.23, ce qui reste satisfaisant dans un contexte de détection multi-classes avec des objets potentiellement difficiles à distinguer.

V.7. Yolov8x :

V.7.1. Matrice de confusion :

La figure V.1 illustre la matrice de confusion obtenue lors de l'évaluation du modèle YOLOv8x. Cette matrice permet d'analyser les performances du modèle en termes de classification des différentes classes de dégradations.

- Les **lignes** de la matrice correspondent aux **classes prédites** par le modèle.
- Les **colonnes** représentent les **classes réelles**, c'est-à-dire celles effectivement présentes dans les données de test.
- Chaque **cellule** indique le nombre d'exemples d'une classe réelle spécifique (colonne) que le modèle a classés dans une autre classe (ligne).
- Les **valeurs situées sur la diagonale principale** (de haut en bas) correspondent aux **prédictions correctes** (ou vrais positifs), tandis que les **valeurs en dehors de cette diagonale** traduisent des **erreurs de classification** (confusions entre classes).

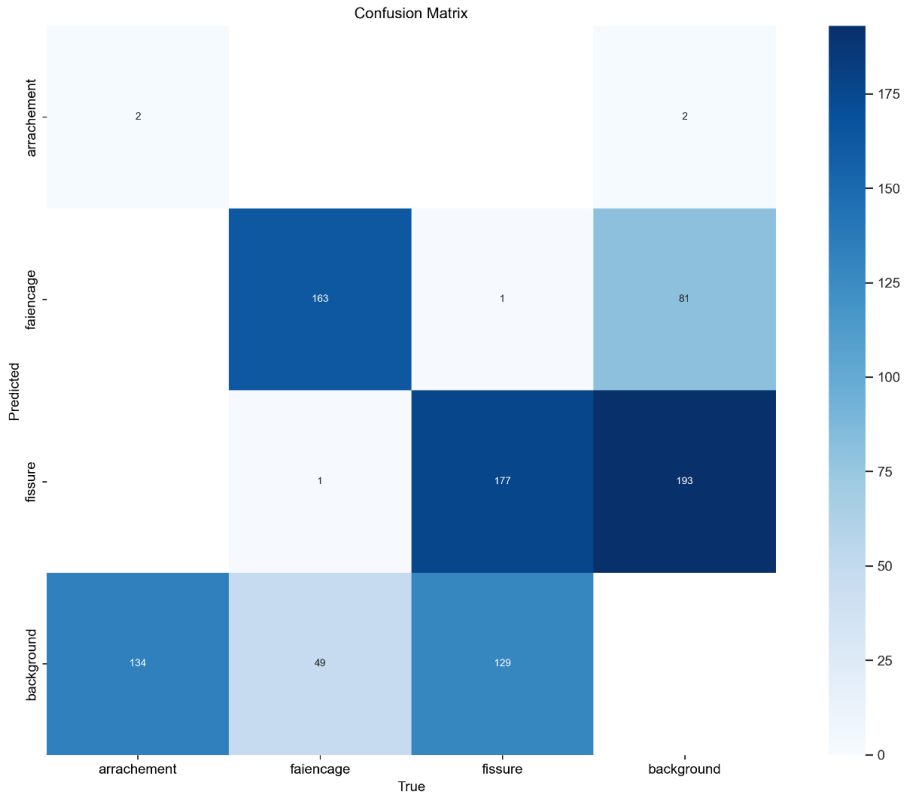


Figure V.29: matrice de confusion de Yolov8x

V.7.1.1. Classes analysées :

Le modèle YOLOv8x a été entraîné pour classifier quatre catégories de données issues d'images aériennes : Arrachement, faïençage, fissure et background (zones sans défaut).

V.7.1.2. Analyse détaillée par classe:

V.7.1.2.1. Classe "arrachement":

- **Prédictions correctes** : 2 véritables arrachements ont été correctement identifiés par le modèle. Ce chiffre reste extrêmement faible, ce qui traduit une très faible sensibilité à cette catégorie.
- **Erreurs de classification** :134 instances ont été classées à tort comme background. Il s'agit d'une erreur critique, car elle montre que la quasi-totalité des arrachements réels sont ignorés par le modèle ou mal interprétés comme étant des zones neutres.

V.7.1.2.2. Classe "faïençage":

a. **Prédictions correctes** : 163 cas ont été correctement détectés, ce qui constitue une excellente performance pour cette classe.

b. **Erreurs de classification** :
   - 1 instance a été confondue avec fissure (erreur négligeable).
   - 49 instances ont été erronément classées comme background, ce qui représente une perte non négligeable.

V.7.1.2.3. Classe "fissure" :

a. **Prédictions correctes** : 177 cas correctement classés, soit une très bonne reconnaissance.

b. **Erreurs de classification** :
   - 1 cas a été confondu avec faïençage (erreur mineure).
   - 129 instances ont été classées comme background, ce qui constitue une erreur majeure de détection.

V.7.1.2.4. Classe "background" :

a. **Prédictions correctes** : Aucune. Le modèle n'a jamais réussi à reconnaître correctement une zone de fond, ce qui est hautement problématique.

b. **Erreurs de classification** :
   - 2 cas ont été classés comme arrachement
   - 81 comme faïençage
   - 193 comme fissure

Ce schéma révèle une confusion systématique du background avec les défauts. Le modèle attribue à tort une anomalie visuelle à des zones neutres, ce qui peut compromettre sérieusement son usage en contexte réel.

V.7.2. Matrice de confusion normalisée:

La matrice de confusion normalisée comme illustrée dans la figure V. représente les performances du modèle YOLOv8x en termes proportionnels plutôt qu'en valeurs absolues. Chaque cellule de la matrice indique la part des instances d'une classe réelle donnée ayant été classées dans une catégorie prédite.

La valeur (i, j) correspond à la proportion d'instances réellement de la classe j qui ont été prédites comme étant de la classe i.

Ainsi, la somme des valeurs de chaque colonne est égale à 1 (ou 100 %), car elle traduit la distribution des prédictions effectuées pour une classe réelle spécifique. Les valeurs situées sur la diagonale principale (du haut gauche vers le bas droit) représentent

le rappel (ou sensibilité) pour chaque classe, c'est-à-dire la capacité du modèle à détecter correctement les instances appartenant à cette classe.

Les valeurs hors diagonale traduisent les erreurs de classification, c'est-à-dire les cas où le modèle a confondu une classe avec une autre.

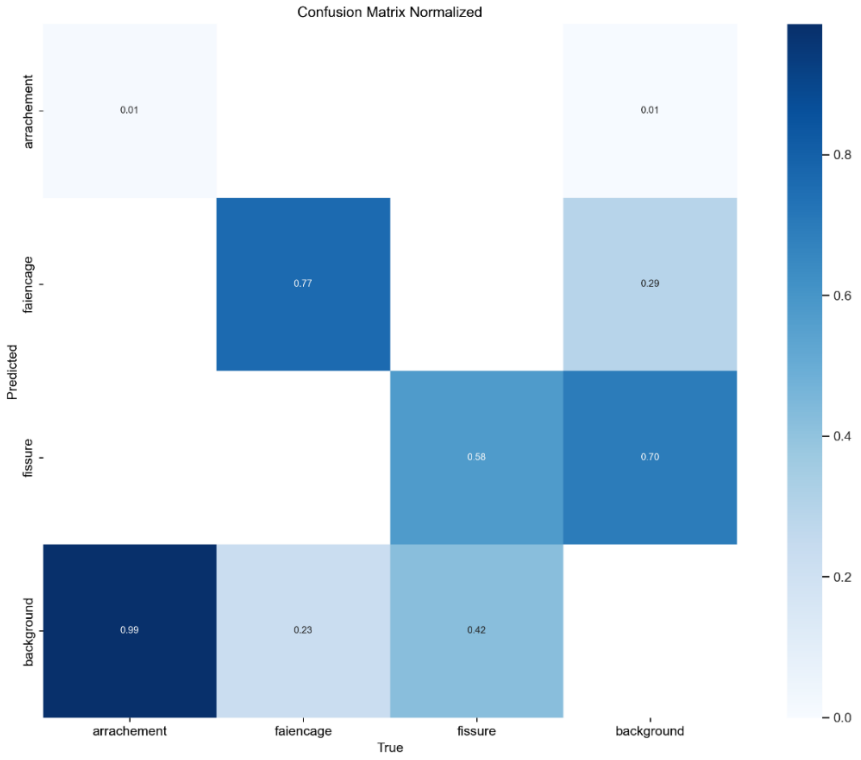


Figure V.30: matrice de confusion normalisée de Yolov8x

V.7.2.1. Classes évaluées:

Le modèle YOLOv8n a été entrainé pour classifier les quatre catégories suivantes : (arrachement, faïençage, fissure et background (zone sans défaut).

V.7.2.2. Analyse des performances (rappel/sensibilité):

V.7.2.2.1. Classe "arrachement" :

a. **Rappel** : 0,01 (soit 1 %) Il s'agit du score le plus bas observé pour cette classe parmi toutes les versions de YOLOv8 analysées. Le modèle échoue presque totalement à détecter les arrachements.

b. **Erreurs de classification**: 99 % des arrachements réels sont classés comme background, démontrant une confusion quasi systématique avec le fond.

V.7.2.2.2. Classe "faïençage"

a. **Rappel** : 0,77 (soit 77 %)
Cette classe affiche une bonne sensibilité, confirmant une capacité robuste du modèle à reconnaître le faïençage.

b. **Erreurs de classification** :

- 0 % sont confondues avec fissure, ce qui est notable (aucune confusion directe).

- 23 % sont mal classées comme background, ce qui constitue encore une perte significative de rappel.

V.7.2.2.3. Classe "fissure"

a. **Rappel** : 0,58 (soit 58 %) Le modèle identifie plus de la moitié des fissures, mais la performance reste modérée.

b. **Erreurs de classification**:

- 0 % sont prédites comme faïençage,
- 42 % sont classées comme background, ce qui révèle une confusion fréquente avec l'arrière-plan, et donc une importante perte de sensibilité.

V.7.2.2.4. Classe "background"

a. **Rappel** : 0,00 (soit 0 %) Le modèle ne parvient pas à identifier la moindre instance de fond. Le background est entièrement absorbé par les classes de défauts, ce qui est particulièrement problématique.

b. **Erreurs de classification**:

- 4 % des vrais background sont prédits comme arrachement,
- 29 % comme faïençage,
- 70 % comme fissure, illustrant une forte tendance à générer des faux positifs pour cette classe.

V.7.3. La courbe F1-confiance:

La courbe F1-Confidence illustrée par la figure V.3 ci-dessous, permet d'évaluer la performance du modèle YOLOv8x en fonction du seuil de confiance appliqué aux prédictions. Elle constitue un outil essentiel pour analyser l'équilibre entre précision et rappel à différents niveaux de certitude dans la détection.

L'Axe des abscisses (Confidence) : représente le seuil de confiance appliqué aux prédictions du modèle, allant de 0.0 à 1.0. Plus le seuil est élevé, plus le modèle doit être sûr de sa prédiction pour la valider.

L'Axe des ordonnées (F1-score) : indique le score F1, qui combine la précision (précision des détections) et le rappel (taux de détection des objets réels).

Un F1-score élevé traduit un bon compromis entre éviter les faux positifs et minimiser les oublis.

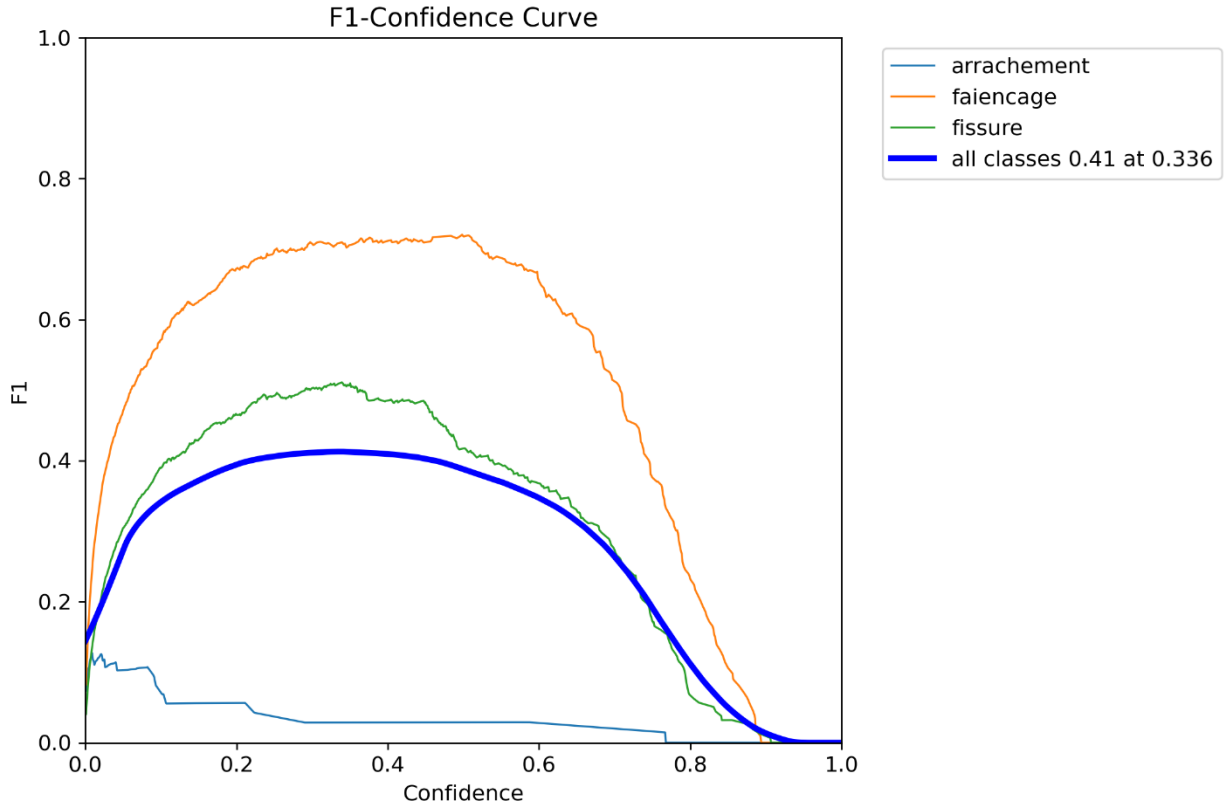


Figure V.31: la corbe F1 de Yolov8x

V.7.3.1. Analyse des courbes par classe

V.7.3.1.1. Classe faïençage (courbe orange)

La classe faïençage affiche la performance la plus élevée sur l'ensemble des seuils. Le score F1 atteint un pic situé entre 0.70 et 0.75, correspondant à un seuil de confiance compris entre 0.4 et 0.6. Cette courbe reste relativement stable et élevée jusqu'à des seuils d'environ 0.7, traduisant une forte capacité du modèle à maintenir un bon équilibre entre exactitude et couverture des détections pour cette classe.

V.7.3.1.2. Classe fissure (courbe verte)

La classe fissure présente une performance intermédiaire. Son score F1 maximal avoisine 0.50, atteint pour des seuils de confiance de 0,2-0,4. La courbe indique une diminution progressive de performance à mesure que le seuil augmente, révélant une efficacité plus modérée, mais néanmoins cohérente, dans la détection de cette catégorie.

V.7.3.1.3. Classe arrachement (courbe bleu clair)

La classe arrachement se distingue par une courbe nettement inférieure aux autres. Le score F1 plafonne à environ 0.15, avec une chute rapide dès que le seuil de confiance augmente au-delà de 0.2. Cette instabilité marque une faiblesse importante du modèle pour cette catégorie, avec une dégradation marquée des performances même à des niveaux de confiance faibles.

V.7.3.1.4. Moyenne toutes classes (All classes, courbe bleue foncée épaisse)

La courbe globale, qui reflète la moyenne des performances sur l'ensemble des classes, atteint un maximum de 0.41 pour un seuil de confiance de 0.336. Ce point correspond au compromis optimal atteint par le modèle entre précision et rappel. Un score F1 global de 0.41 reste cependant modeste, traduisant une efficacité relative dans la détection multi-classes.

V.7.4. Courbe de précision :

La figure V.32 ci-dessous montre la courbe Précision-Confidence (ou P-curve), qui permet d'évaluer la performance du modèle YOLOv8x en fonction du niveau de confiance associé à ses prédictions. Cette courbe est nécessaire pour comprendre le compromis entre précision et seuil de confiance, et pour déterminer un seuil optimal à partir duquel les prédictions sont fiables.

L'Axe des abscisses (Confidence) : représente le seuil de confiance minimal requis pour qu'une détection soit considérée comme valide. Les valeurs s'échelonnent de 0.0 à 1.0.

L'Axe des ordonnées (Precision) : indique la précision au sens classique :

Une précision élevée reflète une capacité du modèle à limiter les fausses détections.

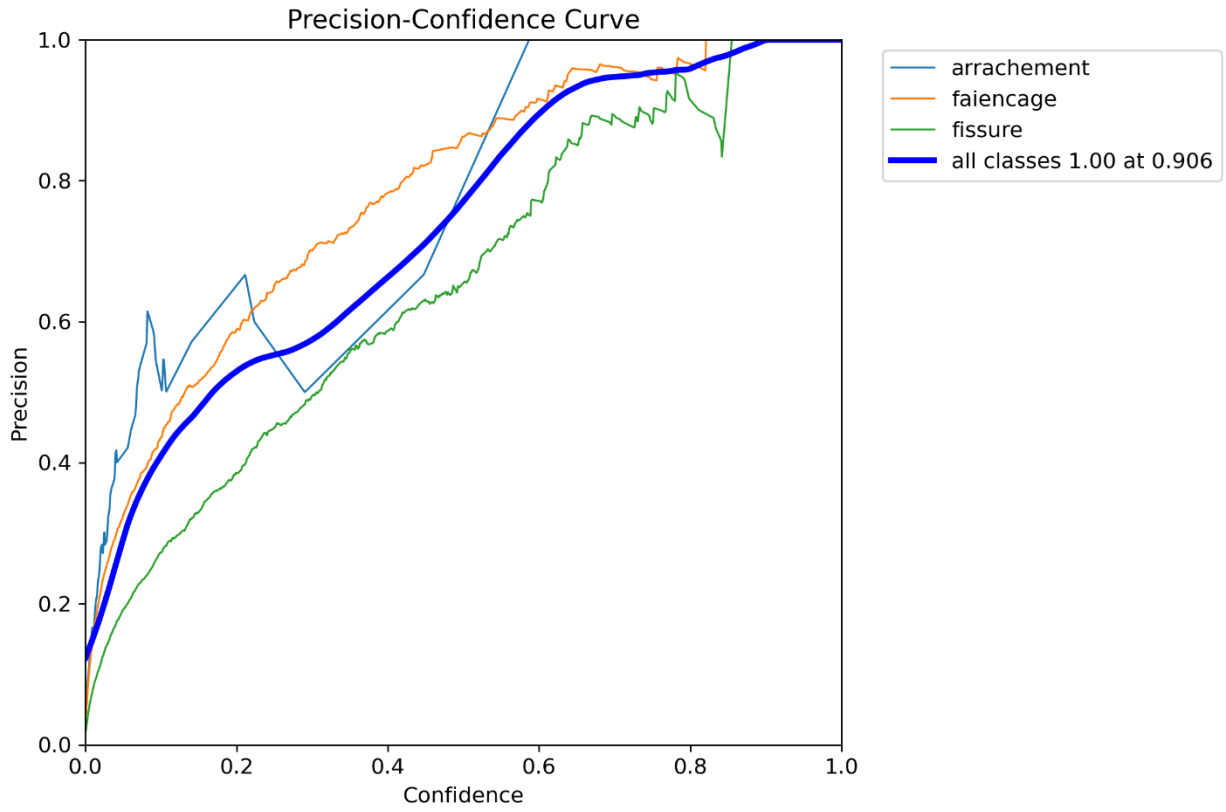


Figure V.32: la courbe de précision de Yolov8x

V.7.4.1. Analyse des courbes par classe

V.7.4.1.1. Classe faïençage

La classe faïençage se distingue par une courbe de précision à la fois élevée et remarquablement stable. Elle démarre à un niveau très haut, proche de 1.0, dès les seuils les plus élevés, et conserve une précision supérieure à 0.8 sur une large plage de seuils, allant jusqu'à environ 0.7–0.8. Cette régularité traduit une grande fiabilité du modèle dans l'identification correcte de cette classe, même à des niveaux de confiance modérés.

V.7.4.1.2. Classe fissure

La courbe correspondant à la classe fissure montre également de bonnes performances, bien que légèrement inférieures à celles observées pour le faïençage. Elle affiche une précision généralement supérieure à 0.7 sur une portion significative de l'intervalle de confiance. Cela traduit une capacité satisfaisante du modèle à limiter les erreurs de classification pour cette catégorie.

V.7.4.1.3. Classe arrachement

La courbe arrachement apparaît comme la plus instable et la moins performante. Pour des seuils de confiance faibles, la précision reste nettement inférieure à 0.4, témoignant d'un nombre important de faux positifs. À mesure que le seuil augmente, la précision peut s'améliorer et atteindre des valeurs élevées, proches de 1.0, mais cela se fait au prix d'une forte réduction du nombre de détections, ce qui limite considérablement l'utilité pratique du modèle pour cette classe. L'irrégularité de cette courbe révèle une grande variabilité dans la capacité du modèle à produire des prédictions fiables pour les arrachements.

V.7.4.1.4. Courbe globale- toutes classes (épaisse, bleu foncé)

La courbe moyenne, synthétisant les performances globales du modèle sur l'ensemble des classes, atteint un maximum de 1.0 à un seuil de confiance de 0.906. Si une telle précision maximale suggère une extrême fiabilité à ce seuil, elle est souvent obtenue au prix d'un nombre très restreint de prédictions, ce qui diminue considérablement la portée opérationnelle du modèle dans les contextes réels.

V.7.5. La courbe Recall-Confidence:

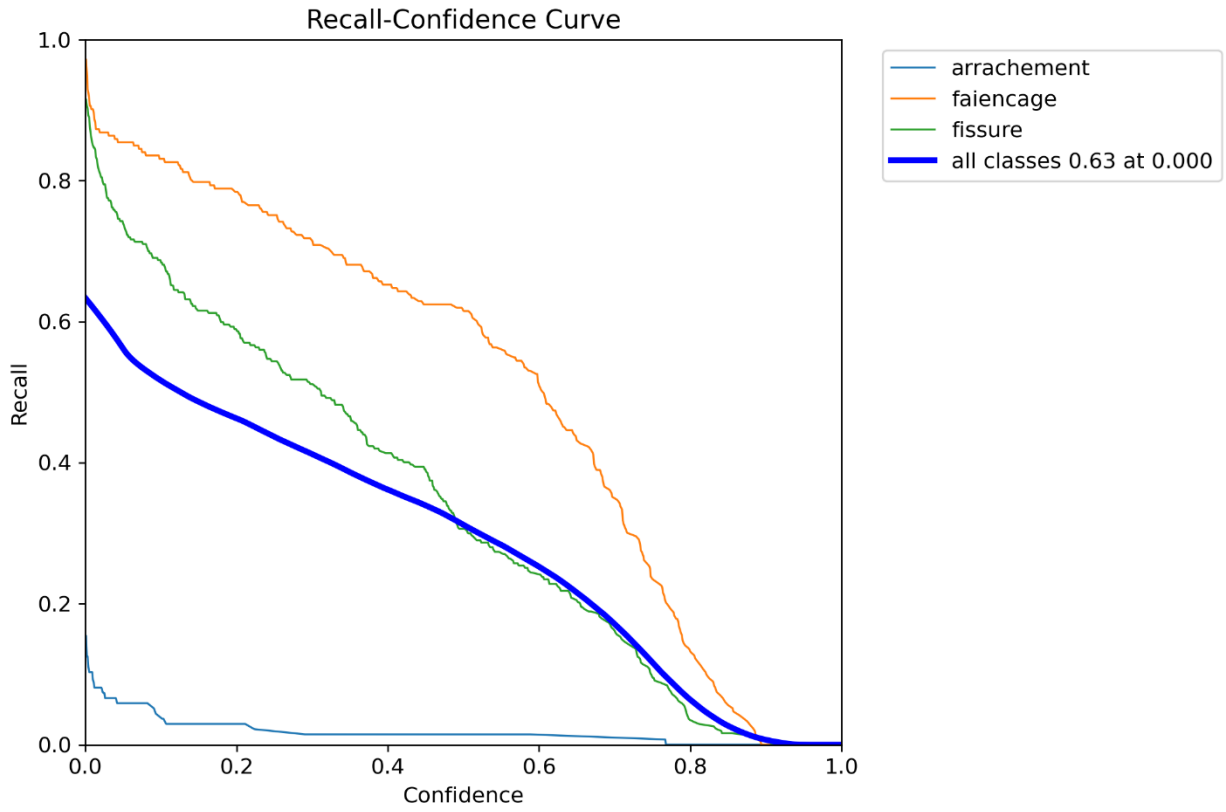


Figure V.33: la courbe de rappel de Yolov8x

La courbe Recall–Confidence présentée par la figure V.33 ci-dessus, illustre la capacité du modèle YOLOv8x à détecter correctement les objets pertinents en fonction du seuil de confiance appliqué aux prédictions. Elle permet de quantifier, pour chaque classe, la proportion des instances réelles qui ont été correctement identifiées (rappel) en fonction de la rigueur avec laquelle le modèle accepte ou rejette une détection.

L'Axe des abscisses (Confidence) : représente le seuil minimal requis pour valider une prédiction. Un seuil bas permet de retenir davantage de détections, tandis qu'un seuil élevé filtre les prédictions les moins sûres.

Un rappel élevé signifie que le modèle détecte la majorité des objets réellement présents, même si cela se fait parfois au détriment de la précision.

V.7.5.1. Analyse des courbes par classe

V.7.5.1.1. Classe faïençage :

La classe faïençage présente la courbe de rappel la plus performante. À des seuils de confiance très bas, le rappel avoisine 1.0, traduisant une détection quasi exhaustive des objets de cette catégorie. Bien que la courbe décline progressivement à mesure que le seuil augmente, elle maintient un niveau de rappel élevé (souvent supérieur à 0.7) jusqu'à un seuil avoisinant 0.4. Cette stabilité indique que le modèle parvient à conserver une bonne sensibilité pour cette classe, même lorsque l'exigence de confiance est accrue.

V.7.5.1.2. Classe de fissure :

Le rappel pour la classe fissure est globalement bon, bien qu'inférieur à celui observé pour le faïençage. Il atteint également des valeurs proches de 1.0 à très faible seuil, mais décline plus rapidement dès que le niveau de confiance augmente. Néanmoins, pour des seuils modérés (environ 0.4–0.5), le rappel se stabilise à un niveau respectable, souvent autour de 0.4–0.5, suggérant une détection partielle mais significative des objets appartenant à cette catégorie.

V.7.5.1.3. Classe arrachement :

La courbe associée à la classe arrachement est la moins performante. Elle démarre à un niveau déjà bas (environ 0.2 au seuil 0.0) et décroît rapidement pour atteindre des valeurs proches de 0 dès que le seuil dépasse 0.2–0.3. Cette chute abrupte traduit une grande difficulté du modèle à détecter efficacement cette classe, même lorsque les critères de confiance sont très permissifs.

V.7.5.1.4. Courbe globale (All classes) – Moyenne pondérée :

La courbe globale, représentant la moyenne des rappels sur l'ensemble des classes, atteint un rappel maximal de 0.63 à un seuil de confiance nul (0.000). Ce niveau globalement modéré traduit une capacité de détection correcte du modèle dans un contexte général, bien qu'il soit important de noter que de telles valeurs à très faible seuil s'accompagnent généralement d'une baisse significative de précision.

V.7.6. La courbe Precision-Recall :

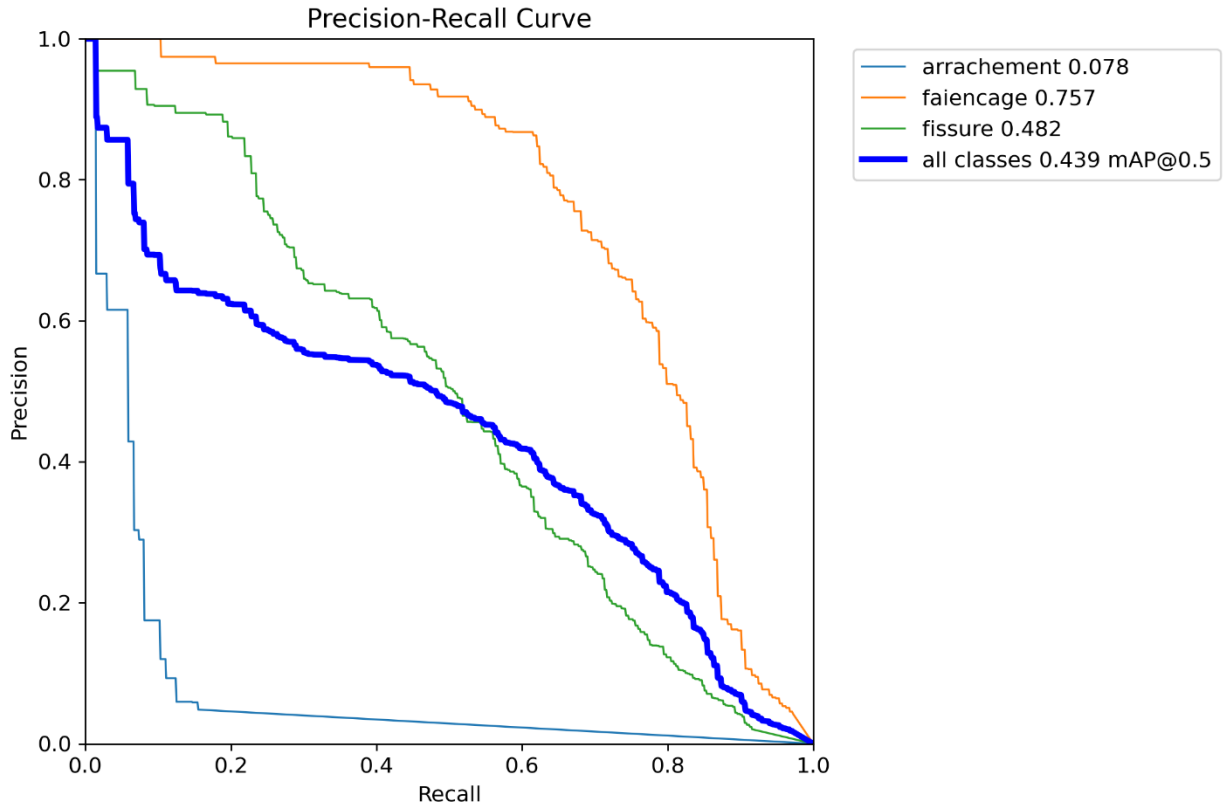


Figure V.34: Courbe Précision -Recall de Yolov8x

La courbe Precision–Recall (PR) illustrée par la figure V.34 ci-dessus, permet de visualiser le compromis entre la précision (exactitude des prédictions positives) et le rappel (capacité à détecter toutes les instances positives) pour chaque classe. Elle est particulièrement utile dans le contexte des ensembles de données déséquilibrés, où une simple mesure de précision ou d'exactitude peut être trompeuse.

L'Axe des abscisses (Recall) : représente le taux de détection des objets réellement présents. Plus on avance vers la droite, plus le modèle détecte d'instances correctes.

L'Axe des ordonnées (Precision) : mesure la fiabilité des détections positives. Une valeur élevée indique un faible taux de faux positifs.

Une courbe idéale conserve une précision élevée sur une large gamme de rappels. L'aire sous la courbe (Average Precision, AP) est utilisée pour résumer la performance de chaque classe.

V.7.6.1. Analyse par classe :

V.7.6.1.1. Classe faïençage :

AP (Average Precision) : 0.757, La courbe est la plus performante de toutes, témoignant d'un excellent équilibre entre précision et rappel. La précision demeure très élevée souvent supérieure à 0.8 même lorsque le rappel atteint des niveaux élevés (aux alentours de 0.6). Cette stabilité témoigne d'une capacité remarquable du modèle à détecter de manière exhaustive les objets de cette classe tout en limitant les fausses détections, ce qui traduit un équilibre optimal entre les deux composantes de l'évaluation.

V.7.6.1.2. Classe fissure :

AP : 0.482, la courbe révèle un déclin plus rapide de la précision à mesure que le rappel augmente. Autrement dit, le maintien d'un bon rappel exige ici une concession notable sur la précision. Cela reflète une certaine instabilité dans la capacité du modèle à conserver une fiabilité constante lorsque l'exigence de couverture des objets devient plus importante.

V.7.6.1.3. Classe arrachement :

AP : 0.078, La précision chute rapidement, même pour de faibles niveaux de rappel (inférieurs à 0.1), et se maintient à des valeurs très basses pour l'ensemble de la courbe. Cette tendance souligne une difficulté manifeste du modèle à identifier cette classe de manière efficace, tant en termes de détection que de fiabilité des prédictions.

V.7.6.1.4. Courbe moyenne (all classes) – Performance globale :

La courbe moyenne, qui agrège les performances sur l'ensemble des classes, affiche un mAP@0.5 de 0.439. Ce score global reflète un niveau de performance modéré. Il met en lumière un bon compromis global entre précision et rappel, bien que la courbe demeure influencée par les disparités de performance observées entre les différentes classes.

V.7.7. Train/val :

Le modèle YOLOv8x a été entraîné sur 100 époques. L'analyse des courbes générées (losses et métriques) permet d'évaluer l'évolution de son comportement en termes d'apprentissage, de généralisation, et de détection par classe. Cette section synthétise les principales observations issues des figures générées dans results.jpg. La figure V.5 ci-après montre l'évolution des pertes et des métriques

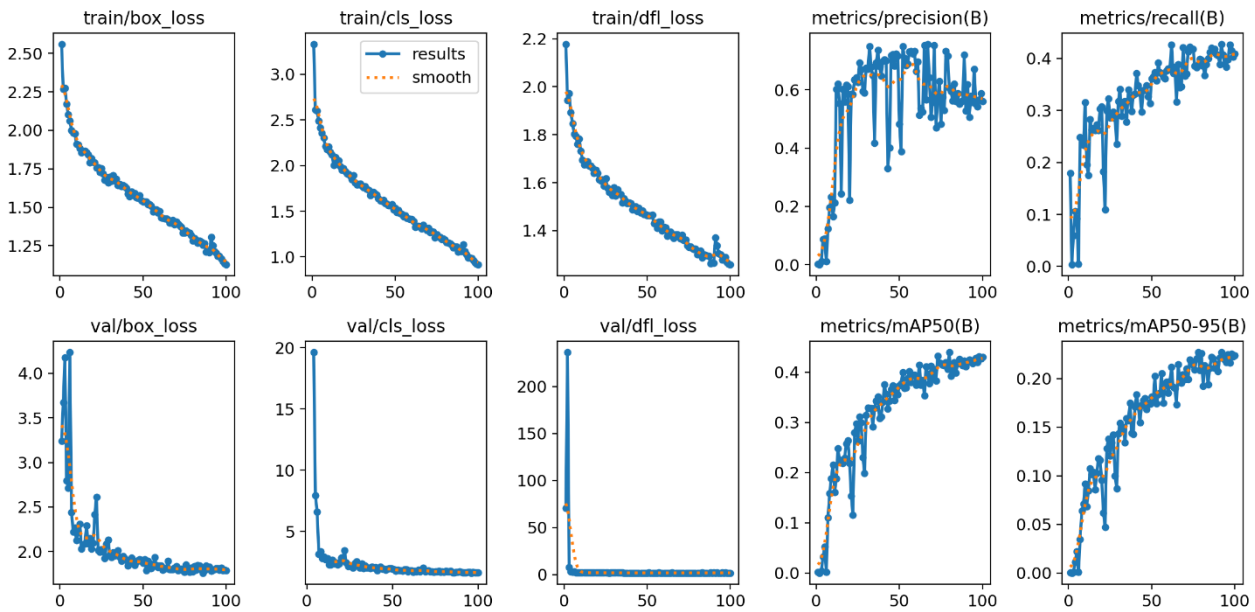


Figure V.35: losses et métriques de Yolov8x

V.7.7.1. Perte de localisation (box_loss)

V.7.7.1.1. Entraînement (train/box_loss) :

la perte diminue progressivement de 2.5 à environ 1.1, traduisant une amélioration continue dans la précision de localisation des boîtes englobantes sur les données d'apprentissage.

V.7.7.1.2. Validation (val/box_loss) :

la courbe suit une tendance similaire avec une valeur initial de 3,25, bien que plus bruitée, se stabilisant autour de 1.75 à 1.8 après 40 époques. Ce qui confirme que la convergence parallèle des pertes d'entraînement et de validation suggère une bonne capacité de généralisation du modèle, sans signe évident de surapprentissage.

V.7.7.2. Perte de classification (cls_loss) :

V.7.7.2.1. Entraînement :

la perte de classification chute de plus de 3.5 à environ 0.8, montrant que le modèle apprend à mieux attribuer la bonne classe aux objets détectés.

V.7.7.2.2. Validation :

initialement très élevée environ 19, elle diminue rapidement pour se stabiliser vers 1.6–1.7.

Ce qui exprime que la forte décroissance initiale est typique d'une phase d'adaptation rapide. L'absence de remontée en validation exclut un phénomène d'overfitting. Toutefois, la variabilité restante traduit une possible confusion inter-classes.

V.7.7.3. Perte DFL (Distribution Focal Loss)

V.7.7.3.1. Entraînement (train/dfl_loss) :

décroissance continue de 2.18 à environ 1.3.

V.7.7.3.2. Validation (val/dfl_loss) :

valeur initiale extrêmement haute environ 230 suivie d'une chute rapide et d'une stabilisation vers 1.

Le pic au début montre une instabilité temporaire du modèle au démarrage de l'apprentissage, rapidement corrigée. Ensuite, la courbe suit une tendance similaire aux autres pertes, ce qui reflète une bonne stabilité générale de l'apprentissage.

V.7.7.4. Evolution de la précision et du rappel :

V.7.7.4.1. Précision (metrics/precision(B)) :

la courbe évolue de 0.2 vers un plateau autour de 0.62–0.634, mais avec une certaine instabilité.

V.7.7.4.2. Rappel (metrics/recall(B)) :

progression régulière de 0.02 à environ 0.43, indiquant une amélioration stable de la capacité du modèle à détecter les objets réels.

V.7.7.5. Evaluation par mAP:

V.7.7.5.1. mAP@0.5 :

augmente progressivement jusqu'à 0.41–0.42, ce qui signifie , indicateur d'une détection correcte avec un recouvrement modéré.

V.7.7.5.2. mAP@0.5:0.95 :

atteint environ 0.22, ce qui traduit une précision acceptable dans la localisation fine, compte tenu de la complexité des données.

V.8. Comparaison entre les résultats:

V.8.1. Comparaison des résultats par classe

Faïençage : une détection maîtrisée et constante

L'ensemble des modèles testés a démontré une remarquable capacité à détecter le faïençage, avec des rappels élevés (75 % à 77 %) et des précisions moyennes (AP) oscillant entre 0.757 et 0.812. Cette stabilité remarquable atteste d'une généralisation robuste et d'une excellente sensibilité aux motifs de dégradation de cette nature. Le modèle YOLOv8s, en particulier, enregistre une performance optimale (AP = 0.812), confirmant l'efficacité du modèle face à des patterns bien représentés et visuellement distincts.

V.8.1.1. Fissure :

une performance solide et généralisable

La détection des fissures s'est avérée globalement fiable, avec des scores AP compris entre 0.474 et 0.485, et des rappels allant jusqu'à 58 %. Ces performances soulignent la capacité des modèles à identifier cette classe malgré une variabilité potentielle dans sa représentation visuelle. YOLOv8x atteint la meilleure précision moyenne sur cette classe (AP = 0.482), démontrant que même les structures plus complexes peuvent maintenir une efficacité appréciable dans des tâches de détection délicates.

V.8.1.2. Arrachement :

un défi clairement identifié

Bien que la détection de la classe « arrachement » demeure la plus complexe, le modèle YOLOv8m parvient à atteindre la meilleure AP (0.129), ce qui constitue un résultat significatif compte tenu de la sous-représentation de cette pathologie dans les données d'apprentissage. Les performances observées sur cette classe mettent en lumière des pistes de renforcement futures, notamment en matière d'équilibrage de la base d'entraînement.

V.8.1.3. Background :

un axe d'amélioration transverse

Le rappel nul constaté sur la classe « background » dans toutes les configurations révèle une difficulté commune à distinguer les zones saines des pathologies. Cette confusion contribue à l'apparition de faux positifs, en particulier sur les classes fissure et faïençage. Toutefois, cette limite, clairement identifiée, constitue une base solide pour l'orientation de travaux futurs.

V.8.2. Comparaison des performances par architecture YOLOv8

V.8.2.1. YOLOv8n :

une base légère et compétitive

Avec un mAP@0.5 de 0.438 et un F1-score global de 0.42, YOLOv8n pose une fondation solide pour une détection efficace avec des ressources computationnelles réduites. Ses performances constituent une référence équilibrée, démontrant qu'un modèle compact peut déjà capturer des motifs significatifs avec pertinence.

V.8.2.2. YOLOv8s:

– le modèle le plus performant

Le modèle YOLOv8s se distingue comme l'architecture affichant les résultats globaux les plus élevés sur ce jeu de données, avec un mAP@0.5 de 0.465 et des scores AP notables sur toutes les

classes. Cette version combine efficacement légèreté, rapidité d'inférence et performance, en exploitant pleinement les régularités présentes dans les données.

V.8.2.3. YOLOv8m:

– une progression mesurée et stable

YOLOv8m présente des résultats comparables à YOLOv8s, avec un mAP@0.5 de 0.456 et la meilleure performance en AP pour la classe arrachement. Sa stabilité générale et sa capacité de généralisation en font une option fiable, notamment pour des environnements où un compromis entre capacité de calcul et précision est recherché.

V.8.2.4. YOLOv8l :

– un F1-score maximal atteint

En atteignant un F1-score global de 0.43, YOLOv8l montre une convergence plus rapide et une robustesse dans ses prédictions, notamment sur les classes dominantes. Son efficacité est manifeste, bien que l'amélioration sur les classes minoritaires reste marginale.

V.8.2.5. YOLOv8x :

– une montée en complexité sans gain décisif

Le modèle le plus profond, YOLOv8x, affiche une performance mAP@0.5 de 0.439. Malgré sa puissance théorique supérieure, il ne surpasse pas les versions précédentes. Cela confirme que, dans le contexte d'un jeu de données limité et déséquilibré, l'augmentation de la profondeur ne garantit pas une amélioration proportionnelle des résultats.

L'étude comparative des différentes versions de YOLOv8 a mis en évidence leur efficacité dans la détection des dégradations de type faïençage et fissure, avec des résultats cohérents, stables et largement exploitables. Le modèle YOLOv8s se distingue comme le plus performant sur ce jeu de données, en raison de son équilibre optimal entre complexité architecturale et généralisation effective.

Au-delà des résultats quantitatifs, cette analyse a permis de cerner précisément les points forts des modèles YOLOv8, ainsi que les limites liées à la distribution des données. Le travail mené constitue donc une base méthodologique fiable, démontrant la pertinence de l'approche par deep learning dans un contexte routier réel, et ouvrant des perspectives d'amélioration concrètes pour des applications futures plus robustes.

V.8.2.6. Évaluation du surapprentissage et du sous-apprentissage

L'examen croisé des courbes d'entraînement et de validation permet de juger de la présence éventuelle de phénomènes de sous-apprentissage (underfitting) ou de surapprentissage (overfitting).

Aucun signe manifeste de sous-apprentissage n'est observé : les pertes d'entraînement décroissent de manière continue, et les performances s'améliorent progressivement sur les deux ensembles, démontrant que le modèle parvient à apprendre des représentations pertinentes.

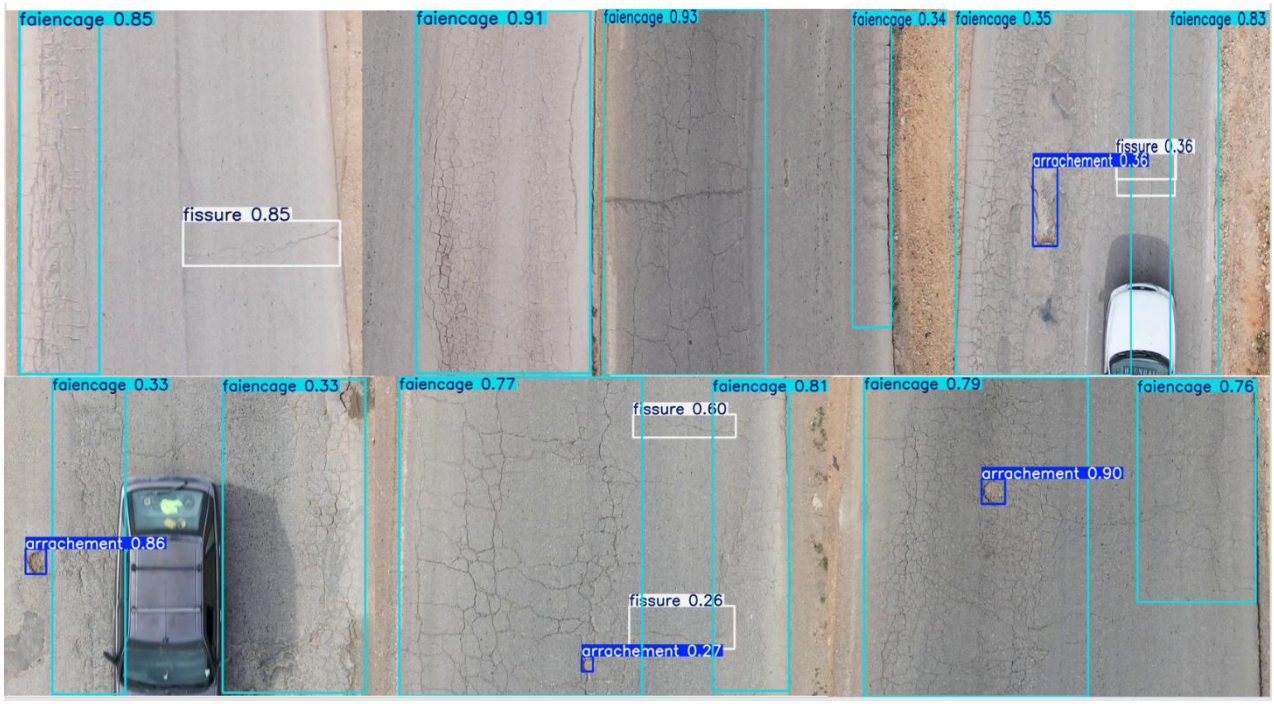
De même, aucun surapprentissage significatif n'est détecté. En effet, après une phase initiale de déséquilibre, les pertes en validation rejoignent une trajectoire parallèle à celles de l'entraînement, et les métriques globales ne montrent pas de dégradation soudaine ni de divergence excessive entre les deux ensembles. Cette cohérence suggère que le modèle ne se spécialise pas excessivement sur les données d'apprentissage et conserve sa capacité de généralisation.

V.9. Le test du modèle:

Suite à l'entraînement exhaustif des différentes architectures YOLOv8 et à l'identification du modèle YOLOv8s comme l'architecture la plus performante sur notre jeu de données (tel que déterminé par son mAP@0.5 global le plus élevé de 0.465), la phase suivante de notre protocole expérimental a consisté en l'évaluation de sa capacité de généralisation sur des données inédites.

À cette fin, le modèle optimal, sauvegardé sous le fichier best.pt, a été déployé pour effectuer des inférences sur un jeu de données indépendant et non préablement vu par le modèle. Cet ensemble d'évaluation comprenait non seulement 113 images statiques, mais également une séquence vidéo, soigneusement sélectionnées pour représenter la diversité des scénarios routiers sans chevauchement avec les ensembles d'entraînement ou de validation.

Les résultats de cette phase d'inférence se sont concrétisés par la génération de nouvelles images et d'une vidéo annotées par des boîtes englobantes (bounding boxes), visualisant directement les détections des différentes dégradations. Cette approche visuelle est fondamentale car elle permet de quantifier la robustesse du modèle face à de nouvelles données (statiques et dynamiques) et de valider sa pertinence opérationnelle dans un contexte réel. Les résultats obtenus lors de cette phase fourniront des indicateurs concrets de la performance du modèle en situation quasi-réelle, en évaluant sa capacité à détecter et localiser les défauts avec les compromis de précision et de rappel établis lors des phases d'entraînement et de validation. La figure V.36 montre quelque images de test de Yolov8s :



& : les résultats de test du modèle Yolov8s

Malgré les limitations identifiées lors de la phase d'entraînement, notamment le déséquilibre des classes et la difficulté à distinguer les zones de fond, le fichier final best.pt du modèle YOLOv8s a démontré une capacité effective à détecter les trois types de dégradations ciblés : faïençage, fissure et arrachement.

Une vérification expérimentale a été conduite sur un échantillon aléatoire de 7 images issues de l'ensemble de test (113 images au total). Les résultats obtenus montrent que le modèle a réussi à prédire la présence des trois classes, avec des scores de confiance notables.

- **Faïençage:** Le modèle a détecté cette classe avec des précisions élevées, comprises entre 0.33 et 0.93, notamment :0.85, 0.91, 0.93, 0.34, 0.83, 0.33, 0.77, 0.81, 0.79, 0.76
  Cela confirme la robustesse de la détection pour cette pathologie, en cohérence avec les métriques quantitatives précédemment observées.

- **Fissure** : Bien que les scores soient plus variables, le modèle a détecté des fissures avec des niveaux de confiance allant jusqu'à 0.85, et des cas plus modérés autour de 0.26 à 0.60, illustrant une capacité de détection efficace mais sensible à la variabilité visuelle des fissures.

- **Arrachement:** Le modèle a également identifié des arrachements avec des précisions comprises entre 0.27 et 0.90, incluant un pic élevé à 0.86, démontrant qu'en dépit d'une faible performance moyenne sur cette classe en entraînement, le modèle est capable de détecter certaines occurrences avec fiabilité sur des images nouvelles.

Ces observations illustrent de manière concrète que le modèle YOLOv8s, bien que confronté à des défis structurels liés à la base de données, généralise suffisamment bien pour détecter toutes les classes de dégradations sur des images inédites. La présence de prédictions fiables sur l'ensemble des classes, y compris l'arrachement, témoigne de la pertinence de l'approche retenue et de la capacité d'adaptation du réseau dans un contexte routier réel.

V.10. Conclusion :

Ce chapitre a permis de porter une analyse approfondie sur les performances obtenues par les modèles YOLOv8 dans le cadre de la détection automatique des dégradations de chaussée. À travers une analyse détaillée des différentes courbes métriques (précision, rappel, F1-score, mAP), des matrices de confusion, ainsi que des prédictions visuelles, nous avons pu identifier les points forts et les faiblesses des variantes testées. Parmi celles-ci, YOLOv8s s'est démarqué par son bon compromis entre légèreté, rapidité et précision, notamment dans la détection du faïençage et des fissures. Toutefois, des limites subsistent concernant la classe 'arrachement', moins bien détectée, probablement en raison du déséquilibre des données ou d'une variabilité visuelle élevée. L'acquisition des données s'est faite à l'aide de deux supports : des drones et des smartphone embarqué sur véhicule. Dans le cadre de cette étude, seules les données captées par drone ont été utilisées. Les images issues de smartphone n'ont pas été exploitées, bien qu'elles aient été collectées, car le temps alloué à ce projet de fin d'études ne permettait pas de mener une analyse comparative

approfondie. Cette comparaison entre sources d'acquisition constitue une piste intéressante pour les travaux futurs

Conclusion générale :

Ce projet de fin d'études a visé à concevoir une approche automatisée pour la détection des dégradations des chaussées souples, en mobilisant les capacités des modèles d'apprentissage profond, et plus particulièrement ceux de la famille YOLOv8. En adoptant une méthodologie rigoureuse allant de la collecte des données à leur annotation, en passant par l'entraînement et l'évaluation de plusieurs variantes du modèle, nous avons démontré que YOLOv8 constitue un cadre efficace pour l'identification automatique des détériorations telles que les fissures, le faïençage et l'arrachement.

L'expérimentation s'est fondée exclusivement sur des données acquises par drone, bien que des vidéos étaient également été enregistrées via un smartphone embarqué sur véhicule. Faute de temps, ces dernières n'ont pu être exploitées. Une étude comparative entre ces deux modalités d'acquisition pourrait, à l'avenir, apporter un éclairage pertinent sur l'influence des conditions de captation sur les performances de détection.

Parmi les variantes testées, YOLOv8s s'est distingué comme étant le plus adapté au contexte algérien, en offrant un bon compromis entre rapidité d'inférence et précision de détection.

Les résultats obtenus ouvrent de nombreuses perspectives, notamment l'enrichissement de la base de données, le rééquilibrage des classes peu représentées, l'intégration dans des systèmes d'information géographique (SIG), ainsi que le déploiement embarqué sur drones ou véhicules pour une surveillance en temps réel du réseau routier.

Enfin, ce travail confirme le potentiel des technologies d'intelligence artificielle appliquées à la gestion routière, et pose les bases d'un système de diagnostic automatisé pouvant contribuer à la maintenance préventive du réseau routier.

Références bibliographique :

[1] : Benabbou, S., & Hamidi, M. (2024). *Dégradations des routes en Algérie*. Université de Tiaret.
[2] : Touahria, N., & Djerad, A. (2022). *Étude des dégradations de chaussée*. Université de Tiaret.
[3] : Malti, M., & Ghorfi, S. (2021). *Pathologie routière : étude de cas*. Université de Tlemcen.
[4] : Benkacem, H., & Zeggai, A. (2021). *La sécurité routière en Algérie*. Université de Témouchent.
[5] : LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444.
[6] : Zhang, L., Yang, F., Zhang, Y. D., & Zhu, Y. J. (2021). Road crack detection using deep convolutional neural network. *IEEE Transactions on Image Processing*, 30, 7665–7677.
[7] : Fang, C., Li, H., & Xu, Y. (2022). Real-time pavement defect detection using YOLOv5. *Journal of Computing in Civil Engineering*, 36(3), 04022015.
[8] : Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. *European Conference on Computer Vision (ECCV)*.
[9] : Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
[10] : Girshick, R. (2014). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
[11] : Girshick, R. (2015). Fast R-CNN. *IEEE International Conference on Computer Vision (ICCV)*.
[12] : Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Advances in Neural Information Processing Systems (NeurIPS)*.
[13] : He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. *IEEE International Conference on Computer Vision (ICCV)*.
[14] : Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv preprint arXiv:2004.10934*.
[15] : Boesch G. A Guide to YOLOv8 in 2024 [Internet]. viso.ai. 2024. Disponible sur : https://viso.ai/deep-learning/yolov8-guide .Consulté le : 19/04/2025
[16] : Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.
[17] : Philippeau X. Traitement d'images - Les filtres usuels [Internet]. Developpez.com. Disponible sur : https://xphilipp.developpez.com/articles/filtres/?page=page_3
[18] : Madeleine S. Normalisation, centrage and standardisation des images TDM [Internet]. IMAIOS. 2024. Disponible sur : https://www.imaios.com/fr/ressources/blog/images-tdm-normalisation-centrage-et-standardisation .Consulté le : 25/01/2025
[19] : Malviya N. Object Detection — Anchor Box VS Bounding Box - Nikita Malviya - Medium. Medium [Internet]. 7 août 2023 ; Disponible sur :

https://medium.com/@nikitamalviya/object-detection-anchor-box-vs-bounding-box-bf1261f98f12 .Consulté le : 29/01/2025
[20] : https://www.linkedin.com/pulse/les-chauss%C3%A9es-souples-hugues-alexandre-castanou/
[21] : Mémoire Pour l'obtention du diplôme de Master Domaine : des Sciences et de la Technologie Filière : Gini Civil Spécialité : Structures, Thème Dégradation des routes en Algérie Diagnostic – causes et solution, Déposé le : 06/06/2022 Par :**DAHAM Eddine** et **GHEZAL Amel,** Page 22.
[22] : Manuel d'identification des dégradations des chaussées souples Québec
[23] : https://www.reseau-charon-creation.fr/tout-savoir-sur-les-nids-de-poule/
[24] : Guide méthodologique « Diagnostic et conception des renforcements de chaussées, Mai 2016 » rédigé par l'Institut Des Routes, des Rues et des Infrastructures de Mobilité (IDRRIM), est édité par le Centre d'études et d'expertise sur les risques, l'environnement, la mobilité et l'aménagement (Cerema), dans le cadre d'une convention partenariale.
[25] : Apprentissage profond non-supervisé : Application à la détection de situations anormales dans l'environnement du train autonome, spécialité : Informatique Par AMINE BOUSSIK Le 15/12/2023, à Valenciennes Ecole doctorale : Polytechnique Hauts-de-France (ED PHF n°635) Laboratoire : Laboratoire d'Automatique, de Mécanique et d'Informatique Industrielles et Humaines (LAMIH- UMR CNRS 8201)
[26] : Mémoire pour l'Obtention de Diplôme Master : en Informatique Option : Administration et Sécurité des Réseaux THEME : Apprentissage profond pour l'analyse et la classification d'imageries médicales Par : Chiter Yasmine et Hafiane Aicha 2021/2022
[27] : Deep Learning vs Machine Learning : The Ultimate Battle [Internet]. 2022. Disponible sur : https://www.turing.com/kb/ultimate-battle-between-deep-learning-and-machine-learning
[28] : Salim Khazem. Apprentissage profond et traitement d'images pour la détection et la prédiction des nœuds au cœur des rondins. Intelligence artificielle [cs.AI]. CentraleSupélec, 2024. Français.
[29] : Mémoire Master Professionnel, Domaine : Informatique et Technologie de l'Information, Filière : Informatique, Par : Boughaba Mohammed et Boukhris Brahim Thème : L`apprentissage profond (Deep Learning) pour la classification et la recherche d'images par le contenu
[30] : Bento C. Multilayer Perceptron Explained with a Real-Life Example and Python Code : Sentiment Analysis. Medium [Internet]. 5 janv 2022 ; Disponible sur : https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141. Consulté le : 17/03/2025
[31] : Christopher A. What is Perceptron : A Beginners Tutorial For Perceptron. Medium [Internet]. 6 janv 2022 ; Disponible sur : https://antoblog.medium.com/what-is-perceptron-a-beginnerstutorial-for-perceptron-632539884146 .Consulté le : 13/03/2025
[32] : Pandey B. How Do Neural Networks Make Decisions ? A Look at Activation Functions [Internet]. Goglides Dev 🌱. 2023. Disponible sur : https://www.goglides.dev/bkpandey/how-do-neural-networks-make-decisions-a-look-at-activation-functions-141e . Consulté le : 5/04/2025
[33] : Zambare S. Object detection : using non-max supression over YOLOv2. Medium [Internet]. 8 déc 2021 ; Disponible sur : https://medium.com/@sarangz/object-detection-using-non-max-supression-over-yolov2-382a90212b51.Consulté le : 03/02/2025

[34] : Lunge N. A deep architecture : Multi-Layer Perceptron - Ninad Lunge - Medium. Medium [Internet]. 24 mars 2024 ; Disponible sur : https://medium.com/@nlunge786/a-deep-architecture-multi-layer-perceptron-164bc5ff3842.Consulté le : 7/05/2025 [30] Bento C. Multilayer Perceptron Explained with a Real-Life Example and Python Code : Sentiment Analysis. Medium [Internet]. 5 janv 2022 ; Disponible sur : https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141. Consulté le : 7/04/2025

[35] : https://www.ultralytics.com/fr/glossary/object-detection

[36] : Mémoire de Fin d'Etudes Master Filière : Informatique Option : Sciences et Technologie de l'Information et de la Communication Thème : Détection d'objets par Deep Neural Network à l'aide du modèle YOLO en temps réel, Par : Mesbah Fethia 2021

[37] : Boesch G. What is Intersection over Union (IoU) ? [Internet]. viso.ai. 2024. Disponible sur : https://viso.ai/computer-vision/intersection-over-union-iou/ .Consulté le : 08/02/2025

[38] : J LC. A new activation function for Neural Networks - logmoid [Internet]. 2022. Disponible sur : https://www.linkedin.com/pulse/activation-functions-neural-networks-leonardo-calderon-j-/ Consulté le : 13/05/2024 [35] Banoula M. What is Cost Function in Machine Learning [Internet]. Simplilearn.com. 2023. Disponible sur : https://www.simplilearn.com/tutorials/machine-learning-tutorial/cost-function-in-machine-learning Consulté le : 03/05/2025

[39] : YOLO Algorithm : Real-Time Object Detection from A to Z [Internet]. Kili-website. Disponible sur : https://kili-technology.com/data-labeling/machine-learning/yolo-algorithm-real-time-object-detection-from-a-to-z#what-is-yolo? .Consulté le : 28/02/2025

[40] : Belaidi N. YOLO : détection sur les images avec TensorFlow [Internet]. Formation Tech et Data en ligne | Blent.ai. Disponible sur : https://blent.ai/blog/a/detection-images-yolo-tensorflow .Consulté le : 02/03/2025

[41] : Boesch G. A Guide to YOLOv8 in 2024 [Internet]. viso.ai. 2024. Disponible sur : https://viso.ai/deep-learning/yolov8-guide/ .Consulté le : 09/03/2025

[42] : Zambare S. Object detection : using non-max supression over YOLOv2. Medium [Internet]. 8 déc 2021 ; Disponible sur : https://medium.com/@sarangz/object-detection-using-non-max-supression-over-yolov2-382a90212b51.Consulté le : 03/06/2024

[43] : Toward Accurate Fused Deposition Modeling 3D Printer Fault Detection Using Improved YOLOv8 With Hyperparameter Optimization - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/The-improved-YOLOv8-network-architecture-includes-an-additional-module-for-the-head_fig2_372207753 .Consulté le : 05/06/2025

[44] : YOLO Algorithm : Real-Time Object Detection from A to Z [Internet]. Kili-website. Disponible sur : https://kili-technology.com/data-labeling/machine-learning/yolo-algorithm-real-time-object-detection-from-a-to-z#what-is-yolo? .Consulté le : 03/06/2024