

**UNIVERSITE DU QUEBEC**

**MEMOIRE PRESENTE A  
L'UNIVERSITE DU QUEBEC A TROIS-RIVIERES**

**COMME EXIGENCE PARTIELLE  
DE LA MAITRISE EN MATHÉMATIQUES ET INFORMATIQUE  
APPLIQUÉES**

**PAR  
SERGE JONATHAN KOGBETI-WONGO**

**DÉTECTION DES PIÈCES DE VÉHICULES INTACTES ET ENDOMMAGÉES À  
L'AIDE D'ALGORITHMES D'APPRENTISSAGE PROFOND (YOLOV8)  
AMÉLIORÉS PAR LA LOGIQUE FLOUE**

**SEPTEMBRE 2025**

# **Université du Québec à Trois-Rivières**

## **Service de la bibliothèque**

### **Avertissement**

L'auteur de ce mémoire, de cette thèse ou de cet essai a autorisé l'Université du Québec à Trois-Rivières à diffuser, à des fins non lucratives, une copie de son mémoire, de sa thèse ou de son essai.

Cette diffusion n'entraîne pas une renonciation de la part de l'auteur à ses droits de propriété intellectuelle, incluant le droit d'auteur, sur ce mémoire, cette thèse ou cet essai. Notamment, la reproduction ou la publication de la totalité ou d'une partie importante de ce mémoire, de cette thèse et de son essai requiert son autorisation.



# 1 DÉDICACE

À mon père,

Je te dédie ce mémoire, en hommage à ton amour, ta sagesse et ta force qui ont été pour moi des sources inestimables d'inspiration. Depuis toujours, tu as cru en moi, même lorsque le chemin semblait incertain, et cette confiance a nourri ma volonté d'aller au bout de mes ambitions.

Ton sens du devoir, ton engagement envers le travail bien fait et ta persévérance dans l'adversité m'ont profondément marqué. Tu m'as appris à ne jamais renoncer, à rester humble dans la réussite et à tirer des leçons de chaque épreuve. Ces valeurs que tu m'as transmises ont guidé chacun de mes pas et ont été le socle sur lequel j'ai construit mon parcours.

Que tu sois à mes côtés pour célébrer ce moment ou que tu le découvres à distance, je ressens ta présence dans chacune de mes réussites. Chaque avancée, chaque effort et chaque victoire que j'ai connus portent la trace de ton influence et de ton soutien indéfectible.

Ce mémoire représente bien plus qu'un accomplissement académique : il est le reflet des sacrifices silencieux que tu as consentis pour m'offrir les meilleures chances de réussir, et le témoignage de l'amour et de la confiance que tu as toujours placé en moi.

Je suis profondément reconnaissant pour tout ce que tu as fait et continues de faire, et j'espère que ce travail te rendra fier.

Avec toute mon affection, mon admiration et ma gratitude.



## 2 REMERCIEMENTS

Avant tout, je rends grâce à Dieu, source de toute sagesse et de toute force, pour m'avoir accordé la persévérance, le courage et la clarté d'esprit nécessaires à l'aboutissement de ce projet. Sa bienveillance m'a accompagné tout au long de ce parcours, et c'est avec humilité que je lui exprime ma profonde reconnaissance.

Je souhaite exprimer toute ma gratitude à mes parents, qui demeurent mon plus grand soutien. Leur amour inconditionnel, leurs encouragements et leurs innombrables sacrifices ont été le socle de mes aspirations et le moteur de ma détermination. Leur confiance en moi a nourri ma volonté de toujours aller plus loin et de donner le meilleur de moi-même.

J'adresse mes sincères remerciements à mes encadrants, le Dr Usef FAGHIHI et Dr Fadel TOURE, pour leur accompagnement exemplaire, leur disponibilité et leurs conseils avisés qui ont grandement enrichi ce travail. Leur rigueur scientifique, leur bienveillance et leur passion pour la recherche ont été pour moi une véritable source d'inspiration et ont joué un rôle déterminant dans l'orientation et la maturation de ce projet.

Mes remerciements vont aussi à mes collègues, amis et partenaires de recherche, pour leur précieuse collaboration, leurs échanges enrichissants et leur soutien indéfectible. Leur esprit d'équipe, leurs idées et leur expertise ont contribué à élargir mes horizons et à nourrir ma réflexion.

Enfin, je remercie toutes les personnes qui, de près ou de loin, ont apporté leur aide, leur soutien ou leurs encouragements durant cette aventure. Leur présence, parfois discrète mais toujours sincère, a été essentielle à la réussite de ce travail.

Ce parcours, au-delà de l'accomplissement scientifique qu'il représente, a été pour moi une expérience humaine et intellectuelle marquante. Il m'a permis d'acquérir des compétences précieuses, mais surtout d'apprendre à persévérer, à collaborer et à croire en mes capacités. Je me réjouis à l'idée de poursuivre ce chemin et de continuer à contribuer, avec passion et engagement, au progrès de la recherche dans mon domaine.

### 3 Résumé

Ce travail présente un pipeline hybride, basé sur un raisonnement, pour l'évaluation automatisée des dommages sur les véhicules. En s'appuyant sur YOLOv8, nous proposons FuzzyYOLOv8, qui remplace les recouvrements stricts utilisés dans le calcul des scores et le Non-Maximum Suppression (NMS) par une fonction concave  $IoU^\alpha$  ( $\alpha \in (0,1]$ ). Cette approche permet d'intégrer une tolérance aux preuves partielles et de stabiliser les décisions dans des conditions complexes telles que le faible contraste, l'occlusion ou le bruit d'annotation. Les boîtes de détection issues de FuzzyYOLOv8 permettent non seulement d'identifier les pièces intactes et endommagées, mais aussi de localiser avec précision les zones affectées sur des composants spécifiques du véhicule.

Le système intègre également la classification latérale (gauche/droite), un mécanisme léger de filtrage des régions d'intérêt afin de supprimer les éléments distracteurs, ainsi qu'une étape d'inférence floue qui évalue la gravité des dommages à l'aide d'indicateurs interprétables (par exemple, le ratio de dommages, le contexte spatial et la confiance du modèle).

Une évaluation a été réalisée sur un jeu de données composé de plus de 700 images annotées couvrant des véhicules intacts et endommagés, avec des étiquettes relatives aux pièces, aux dommages et aux côtés latéraux. Les résultats montrent que FuzzyYOLOv8 améliore la détection par rapport à la version de base YOLOv8. Pour la détection des pièces intactes, le  $mAP@50:95$  est passé de 0,44 à 0,58, tandis que la mesure F1 a légèrement progressé de 0,87 à 0,88. Concernant la détection des dommages, le  $mAP@50:95$  a augmenté de 0,315 à 0,340 et la mesure F1 de 0,64 à 0,69.

Ces résultats suggèrent que l'intégration de la logique floue renforce la robustesse du détecteur et améliore la fiabilité de la localisation des dommages. Ainsi, la solution proposée se révèle à la fois pratique et interprétable, et s'inscrit comme une approche prometteuse pour les processus réels d'évaluation des dommages automobile en assurance.

**Mots-clés :** Apprentissage profond, Logique floue, Estimation locale versus globale des dommages, Raisonnement, Segmentation, Évaluation des dommages véhicules, YOLOv8.

## 4 Abstract

This work presents a hybrid, reasoning-aware pipeline for automated vehicle damage assessment. Building on YOLOv8, we propose FuzzyYOLOv8, which replaces hard overlaps used in scoring and Non-Maximum Suppression (NMS) with a concave  $IoU^\alpha$  ( $\alpha \in (0,1]$ ) function. This approach introduces tolerance to partial evidence and stabilizes decisions under challenging conditions such as low contrast, occlusion, or annotation noise. The detection boxes produced by FuzzyYOLOv8 not only identify intact and damaged vehicle parts but also precisely localize the affected regions on specific components.

The system further integrates left/right side classification, lightweight region-of-interest filtering to suppress distractors, and a fuzzy inference stage that evaluates damage severity using interpretable indicators (e.g., damage ratio, spatial context, model confidence). An evaluation was conducted on a dataset of more than 700 annotated images covering intact and damaged vehicles, with labels for parts, damages, and lateral sides. The results show that FuzzyYOLOv8 improves detection compared to the baseline YOLOv8. For intact part detection,  $mAP@50:95$  increased from 0.44 to 0.58, while the F1-score slightly improved from 0.87 to 0.88. For damage detection,  $mAP@50:95$  rose from 0.315 to 0.340 and the F1-score from 0.64 to 0.69. These results demonstrate that integrating fuzzy logic enhances detector robustness and improves the reliability of damage localization. The proposed solution is both practical and interpretable, making it a promising approach for real-world insurance and appraisal workflows.

**Keywords:** Deep learning, Fuzzy logic, Local versus global damage estimation, Reasoning, Segmentation, Vehicle damage assessment, YOLO8.

## 5 Acronymes

- **3D** : Trois dimensions
- **AP** : Average Precision
- **CIoU** : Complete Intersection over Union
- **CNN** : Convolutional Neural Network
- **CPU** : Central Processing Unit
- **CSV** : Comma-Separated Values
- **FIoU** : Fuzzy Intersection over Union
- **F1-score** : Moyenne harmonique de la précision et du rappel
- **FPN** : Feature Pyramid Network
- **FP** : False Positive
- **FN** : False Negative
- **GPU** : Graphics Processing Unit
- **IA** : Intelligence Artificielle
- **IoU** : Intersection over Union
- **JSON** : JavaScript Object Notation
- **mAP@0,5** : mean Average Precision à un seuil IoU de 0,5
- **mAP@50:95** : mean Average Precision pour des seuils IoU de 0,50 à 0,95
- **Mask R-CNN** : Mask Region-based Convolutional Neural Network
- **NMS** : Non-Maximum Suppression
- **PR** : Precision-Recall
- **PyTorch** : Framework de deep learning en Python
- **ROI** : Region of Interest
- **TN** : True Negative
- **TP** : True Positive
- **TPU** : Tensor Processing Unit
- **YOLO** : You Only Look Once
- **YOLOv8** : You Only Look Once version 8

# 6 Table des matières

|        |   |     |
|--------|---|-----|
| 1      | DÉDICACE.....   | II  |
| 2      | REMERCIEMENTS.....  | III |
| 3      | Résumé .....  | IV  |
| 4      | Abstract .....  | V   |
| 5      | Acronymes.....  | VI  |
| 6      | Table des matières .....  | VII |
| 7      | Liste des figures .....   | IX  |
| 8      | Listes des tableaux.....  | X   |
| 9      | CHAPITRE 1.....   | 1   |
| 9.1    | Introduction.....   | 1   |
| 10     | CHAPITRE 2 – Etat de l’art.....   | 3   |
| 10.1   | Introduction générale sur l’état de l’art.....                                | 3   |
| 10.1.1 | Détection et classification des dommages.....                                 | 3   |
| 10.1.2 | Segmentation et la localisation des pièces et des dommages.....               | 4   |
| 10.1.3 | Robustesse et performances en conditions réelles .....                        | 6   |
| 10.1.4 | Applications en temps réel et segmentation légère .....                       | 6   |
| 10.1.5 | Synthèse et proposition d’une approche hybride .....                          | 7   |
| 11     | CHAPITRE 3 – Méthodologie proposée .....                                      | 8   |
| 11.1   | YOLO (You Only Look Once).....  | 8   |
| 11.1.1 | Architecture de YOLO.....   | 8   |
| 11.2   | Logique floue : concepts et applications.....                                 | 10  |
| 11.3   | Intégration de la logique floue dans l’architecture YOLOv8.....               | 11  |
| 11.4   | FuzzyYOLOv8 pour l’évaluation locale des dommages.....                        | 17  |
| 11.5   | FuzzyYOLOv8 pour l’évaluation globale des dommages .....                      | 18  |
| 11.6   | Filtrage des pièces détectées à l’aide du clustering et du seuil spatial..... | 20  |

|      |   |     |
|------|---|-----|
| a.   | Clustering basé sur la surface .....  | 20  |
| b.   | Filtrage par position horizontale .....   | 20  |
| 11.7 | Capacité du modèle à distinguer les pièces gauches et droites du véhicule ..... | 21  |
| 12   | CHAPITRE 4 - Expérimentations .....   | 23  |
| 12.1 | Description des jeux de données .....   | 23  |
| b.   | Traitement de données.....  | 23  |
| c.   | Étiquetage des pièces et zones de dommages dans le jeu de données .....         | 23  |
| 12.2 | Autres outils.....  | 25  |
| a.   | Python : Langage de programmation .....   | 25  |
| b.   | Google Drive : Stockage et gestion des données .....                            | 25  |
| c.   | Google Colaboratory : Environnement de calcul.....                              | 25  |
| 13   | CHAPITRE 5 – Résultats .....  | 26  |
| 13.1 | Détection des pièces intactes : comparaison entre YOLOv8 et FuzzyYOLOv8.....    | 26  |
| 13.2 | Détection des pièces endommagées : YOLOv8 et FuzzyYOLOv8.....                   | 27  |
| 14   | CHAPITRE 6 - Menaces à la validité de l'étude .....                             | 29  |
| 14.1 | Validité interne .....  | 29  |
| 14.2 | Validité externe.....   | 29  |
| 14.3 | Validité de construit .....   | 29  |
| 15   | CHAPITRE 7- Conclusion.....   | 30  |
| 16   | REFERENCES .....  | VII |
| 17   | Annexe A .....  | IX  |
| 18   | Annexe B .....  | IX  |

## 7 Liste des figures

|   |    |
|---|----|
| <b>Figure 1:</b> Architecture simplifiée de YOLO .....  | 10 |
| <b>Figure 2:</b> Vue synthétique de l'architecture FuzzyYOLOv8 intégrant une couche de logique floue à l'architecture YOLOv8 .....    | 12 |
| <b>Figure 3:</b> Comparaison des performances des différents modèles de détection .....   | 12 |
| <b>Figure 4:</b> Gravité des dommages .....   | 18 |
| <b>Figure 5:</b> Détection de dommages : modèle Yolov8 vs Yolov8+logique floue .....  | 19 |
| <b>Figure 6:</b> Filtrage pièces véhicule principal .....   | 21 |
| <b>Figure 7:</b> Précision côté et dommages .....   | 22 |
| <b>Figure 8:</b> Amélioration de la convergence et des performances de détection avec FuzzyYOLOv8 par rapport au modèle de base. .... | 27 |

## 8 Listes des tableaux

|   |    |
|---|----|
| <b>Table 1:</b> Liste des classes .....                                 | 24 |
| <b>Table 2:</b> Comparaison de la détection des pièces intactes .....   | 26 |
| <b>Table 3:</b> Comparaison de la détection des pièces endommagées..... | 27 |

# 9 CHAPITRE 1

## 9.1 Introduction

L'évaluation précise et à grande échelle des dommages automobiles constitue un enjeu central pour le traitement des réclamations d'assurance, l'expertise technique et la planification des réparations. L'inspection manuelle demeure coûteuse, lente et subjective ; parallèlement, l'augmentation du nombre de véhicules et la généralisation des flux numériques exigent des outils automatisés, fiables et capables de détecter les pièces endommagées tout en estimant la gravité des dommages.

La vision par ordinateur moderne, en particulier les réseaux de neurones convolutifs, a profondément transformé la détection d'objets. La famille YOLO (You Only Look Once) [1] offre un compromis performant entre précision et rapidité dans des contextes temps réel, et ses itérations récentes (par ex. YOLOv8) sont largement adoptées en pratique [1, 2]. En revanche, les détecteurs purement fondés sur les données se révèlent fragiles face à l'ambiguïté : dommages subtils ou partiellement occultés, salissures, reflets ou faible luminosité. Dans ces cas, les seuils de confiance et le Non-Maximum Suppression (NMS) produisent des décisions instables.

Nous proposons de répondre à ces limites à l'aide d'une architecture hybride utilisant le raisonnement qui intègre la logique floue dans YOLOv8. Nous introduisons ainsi **FuzzyYOLOv8**, une amélioration de YOLOv8 dans laquelle une couche floue adoucit les frontières de décision et intègre une tolérance aux indices partiels grâce à une transformation simple et paramétrable de l'IoU. Cette fonction concave, caractérisée par un paramètre unique  $\alpha$ , est différentiable et directement compatible avec les mécanismes standards de scoring/NMS. Elle permet ainsi d'accroître la robustesse sans recourir à une lourde [3] ingénierie de règles.

Le système intègre également une classification latérale (gauche/droite) et un mécanisme de filtrage léger des régions d'intérêt (fondé sur le clustering par aire et les contraintes de position horizontale) afin de supprimer les distracteurs issus des véhicules environnants [4, 5]. Enfin, des indicateurs interprétables tels que le ratio de dommages dans chaque pièce, le contexte spatial et la confiance du modèle sont calculés, puis exploités par une inférence floue pour classer la gravité en quatre niveaux : intact, modéré, sévère ou très sévère [6-9].

Ce travail apporte deux contributions principales.

Nous avons mis en place un détecteur raisonnement-sensibilisé, **FuzzyYOLOv8**, qui remplace les recouvrements stricts par un recouvrement flou ajustable, stabilisant les décisions en présence d'indicateurs partiels ou ambigus et améliorant la sélection dans des conditions de faible contraste, d'occlusion et de bruit d'annotation [1, 2].

Nous proposons un système opérationnel de détection des pièces intactes ou endommagées, intégrant la classification latérale ainsi que le filtrage des distracteurs. Les évaluations réalisées sur des images réelles suggèrent une amélioration constante des performances selon plusieurs métriques de validation.

L'évaluation a été menée sur un jeu de données de plus de 700 images annotées, comprenant des véhicules intacts et endommagés, avec des étiquettes pour les pièces, les dommages et les côtés latéraux. La majorité des images proviennent d'un jeu de données fourni par un partenaire industriel, complété par des sources publiques afin de couvrir une plus grande variété de situations visuelles. En détection, **FuzzyYOLOv8** surpasse la version de base YOLOv8 : pour la détection des pièces intactes, le **mAP@50 : 95** progresse de 0,44 à 0,58 avec un F1 stable (0,87 → 0,88), tandis que pour la détection des dommages, le **mAP@50 : 95** augmente de 0,315 à 0,340 avec un F1 de 0,64 à 0,69, indiquant une meilleure performance sous des seuils IoU plus stricts.

Ces résultats montrent que l'intégration de la logique floue améliore la robustesse du détecteur et renforce la fiabilité de la localisation des dommages, offrant une solution pratique et interprétable pour les processus réels d'assurance automobile [1, 2, 10, 11].

La suite de ce mémoire présente d'abord les travaux connexes sur l'analyse des dommages et la détection des pièces, avant de détailler l'architecture proposée et de discuter les résultats quantitatifs et qualitatifs obtenus.

# 10 CHAPITRE 2 – Etat de l’art

## 10.1 Introduction générale sur l’état de l’art

L’automatisation de l’estimation des dommages sur les véhicules implique plusieurs objectifs clés : la détection et la classification des dommages, la segmentation précise des pièces du véhicule, l’évaluation contextuelle de la gravité des dommages, ainsi que l’optimisation pour des applications en temps réel. Cette section présente les travaux importants selon ces axes, en mettant en évidence leurs contributions et leurs limites.

### 10.1.1 Détection et classification des dommages

Dwivedi et al. [12] ont proposé un modèle combinant un simple CNN et YOLOv3 afin de détecter et classifier sept types de dommages automobiles (bosse sur pare-chocs, rayures, bosse sur portière, vitre brisée, phare avant cassé, feu arrière cassé, véhicule écrasé), en plus d’une classe sans dommage. Leur jeu de données, constitué de 1 861 images collectées par web crawling, a été porté à environ 9 300 images après augmentation (rotation, zoom, flip, etc.), puis séparé en 80 % pour l’entraînement et 20 % pour le test. Les résultats montrent qu’avec des CNN préentraînés sur ImageNet, la précision atteint jusqu’à 96,39 % avec VGG19 en classification, et que YOLOv3 obtient un mAP de 77,78 % pour la détection des zones endommagées.

Cependant, l’utilisation exclusive de boîtes englobantes a conduit à de nombreux faux positifs et une faible performance sur certaines classes peu représentées (ex. rayures, phares cassés). De plus, bien qu’ils évoquent la perspective d’une approche combinant classification et détection, cette dernière n’a pas été implémentée, ce qui limite la portée pratique de leur méthode.

Van Ruitenbeek et Bhulai[13] ont développé un modèle CNN entraîné sur un jeu de données propriétaire particulièrement vaste et diversifié, dépassant 10 000 images annotées, afin de classer les dommages automobiles en douze catégories (rayures, fissures, vitres brisées, bosses, rouille, etc.). Leur corpus combine plusieurs sources : le jeu Damage Dossiers constitué de 2 499 dossiers regroupant près de 19 907 images réduites à 3 513 après prétraitement, le jeu Damage Web constitué de 2 500 images collectées sur Internet et ramenées à 1 338 après filtrage, ainsi qu’un ensemble expérimental capturé dans un environnement contrôlé (« Light Street ») simulant des conditions réelles d’éclairage et de réflexion.

Grâce à des techniques d'augmentation de données (rotation, flou gaussien, ajustement de luminosité, etc.) et au transfert d'apprentissage depuis ImageNet, leur modèle a atteint une précision globale de 87 % et un F1-score moyen de 0,84, démontrant une bonne généralisabilité aux différents contextes. Mais, des disparités de performance subsistent selon les classes : par exemple, les catégories scratch, rust ou bend affichent des résultats inférieurs. De plus, les tests menés dans l'environnement « Light Street » ont révélé que de fortes réflexions lumineuses dégradaient la robustesse du modèle, limitant sa fiabilité dans des scénarios réels complexes.

### **10.1.2 Segmentation et la localisation des pièces et des dommages**

Pasupa et al. [14] ont évalué cinq algorithmes d'apprentissage profond pour la segmentation sémantique et l'instance segmentation des pièces automobiles, à savoir Mask R-CNN, HTC, CBNet, PANet et GCNet, avec ResNet-50 et ResNet-101 comme backbones. Leur étude s'est appuyée sur le jeu de données DSMLR Car Part, composé de 500 images de berlines, pickups et SUV, collectées sur Internet et dans des parkings publics, puis annotées avec 18 catégories de pièces (pare-chocs, portières, vitres, rétroviseurs, phares, capot, roues, etc.). Les images étaient prises sous plusieurs angles (avant, arrière, latéraux).

Les expériences ont montré que HTC avec ResNet-50 était le plus performant pour la segmentation des pièces, atteignant un mAP de 55,2 % lorsque les côtés gauche et droit étaient annotés séparément, et 59,1 % lorsqu'ils étaient regroupés sous un même label. Concernant la robustesse face à des conditions réelles (pluie, neige, brouillard, givre, variations de luminosité), GCNet s'est révélé le plus stable, avec une performance moyenne sous corruption de mPC = 35,2 % (rPC = 64,4 %) pour les côtés différenciés, et mPC = 38,1 % (rPC = 69,6 %) lorsque les côtés étaient regroupés.

Cette étude constitue une contribution importante, car elle introduit une base de données annotée publiquement disponible au format COCO Challenge et évalue rigoureusement plusieurs architectures de segmentation avancées. Toutefois, ses limites résident dans la taille relativement réduite du dataset (500 images seulement) et l'absence d'une évaluation spécifique sur des véhicules endommagés, ce qui restreint la portée des conclusions pour la détection réelle des dommages.

Mohammed et al [15], ont proposé un système basé sur un double Mask R-CNN, conçu pour localiser les côtés du véhicule (gauche/droite) et identifier les zones de dommages. Leur objectif était non seulement de détecter les parties endommagées, mais aussi d'estimer le coût associé aux réparations.

Pour cela, ils ont construit un jeu de données spécialisé de plus de 2 000 images de voitures accidentées, collectées principalement via le site Copart et enrichies par des images issues de Kaggle (COCO). Ce dataset comporte 19 classes de dommages (ex. pare-chocs avant/arrière, portières, phares, feux arrière, capot, coffre, ailes), mais présente un déséquilibre marqué entre les catégories : par exemple, la classe *Dent Door* compte seulement 79 images d'entraînement contre 197 pour *Crash Door*.

Afin d'accroître la diversité et limiter le surapprentissage, ils ont appliqué des techniques d'augmentations de données (rotations entre  $-20^\circ$  et  $20^\circ$ , retournements horizontaux), multipliant ainsi artificiellement la taille du dataset. La répartition des données a été faite en 80 % pour l'entraînement et 20 % pour les tests.

Les résultats expérimentaux montrent que la combinaison des deux Mask R-CNN atteint une précision globale de 98,5 % sur l'ensemble des classes. Cette performance montre l'efficacité de leur approche pour détecter et localiser les dommages, ainsi que pour contribuer à une estimation automatique des coûts.

Cependant, malgré cette précision élevée, leur travail présente certaines limites :

- le déséquilibre des classes fragilise la détection de certains types de dommages,
- le nombre total d'images reste relativement réduit au regard de la diversité réelle des scénarios (angles, luminosité, météo),
- et l'évaluation se limite à l'exactitude (score d'accuracy), sans rapport détaillé de métriques standards comme le mAP ou l'IoU.

Qaddour et Siddiqa [16] ont proposé un cadre combinant l'apprentissage par transfert et Mask R-CNN avec segmentation d'instances pour la détection et l'évaluation des dommages automobiles. Leur approche utilise deux ensembles de données, l'un constitué de véhicules endommagés et non endommagés collectés sur Internet, et l'autre annoté selon trois niveaux de gravité des dommages

(mineur, modéré et sévère). Afin d'améliorer la robustesse, les auteurs ont appliqué des techniques d'augmentation de données (rotations, flips, zooms), et ont divisé leur jeu de données en 80 % pour l'entraînement et 20 % pour les tests. Les résultats expérimentaux montrent que l'utilisation de Mask R-CNN avec Inception-ResNetV2 comme backbone surpasse les modèles VGG-16 et VGG-19, avec une précision atteignant 92 % pour la détection, 85 % pour la localisation et 80 % pour la classification de la gravité, évalués à l'aide des métriques de précision, rappel, F1-score et exactitude.

Toutefois, la taille réduite et le déséquilibre du jeu de données, ainsi que l'absence de prise en compte de la classification latérale (gauche/droite) ou du filtrage des pièces en arrière-plan, pourraient limiter l'applicabilité en conditions réelles.

### **10.1.3 Robustesse et performances en conditions réelles**

Les modèles présentés rencontrent souvent des difficultés liées à la variabilité des conditions d'éclairage, des angles de vue ou des surfaces réfléchissantes. Par exemple, Van Ruitenbeek et Bhulai [13] signalent des erreurs causées par des réflexions intenses. Mohammed et al. [15] n'ont pas testé la robustesse de leur approche dans ces conditions, ce qui laisse un vide dans l'évaluation pratique.

### **10.1.4 Applications en temps réel et segmentation légère**

Dans les scénarios temps réel, Zhang et al. [10] ont proposé YOLOv8-seg-CP, une variante optimisée de YOLOv8 pour la segmentation en temps réel. Leur modèle intègre le backbone StarNet et un module C2f-Star, permettant une meilleure capture des petites cibles tout en réduisant la charge computationnelle. Ils ajoutent aussi une tête de segmentation légère (LSCSH) et un module d'attention CGCAFusion, améliorant la fusion multimodale.

Les auteurs ont construit un jeu de données industriel de 1006 images de pads de puces collectées en conditions réelles (6017 chip pads "rig", 1151 chip pads "wro" et 3566 probes), complété par de l'augmentation de données. Le modèle atteint une précision de 89,8%, avec un FPS de 1399,3 et un temps d'inférence de 0,72 ms, surpassant le modèle de base de 25,8 %. La taille du modèle (3,7M) et le nombre de paramètres (1,7M) sont réduits respectivement de 45,6% et 50% par rapport à la version standard.

Malgré ces résultats très compétitifs, l'évaluation se limite au domaine industriel spécifique des chip pads. L'absence de comparaison avec des architectures récentes basées sur les transformers

réduit la portée de la validation dans des contextes plus généraux, comme la détection de dommages automobiles en conditions variées.

Li [17] a proposé un modèle multi-étape pour le suivi et l'estimation de la vitesse des véhicules, intégrant la segmentation automatique via Segment Anything Model (SAM), le suivi avec StrongSORT, et la reconnaissance optique de caractères (PP-OCR) pour extraire les horodatages. Ce pipeline hybride permet de combiner détection, suivi multi-caméras et estimation de vitesse, avec une robustesse accrue face aux occlusions et aux scènes de circulation dense. L'approche a été évaluée sur plusieurs jeux de données : UA-DETRAC (138 252 images, scènes variées et conditions météorologiques), un jeu privé de 163 165 images collectées sur autoroute, BrnoCompSpeed (20 685 véhicules avec vitesses réelles mesurées par capteurs), et VeRi-776 (37 778 images pour la ré-identification). Les résultats montrent que le modèle surpasse plusieurs méthodes de pointe, atteignant une précision plus élevée en détection, en suivi et en estimation de vitesse, validée par des métriques telles que le mean Average Precision (mAP), la précision de suivi, ainsi que les erreurs MAE et RMSE pour l'estimation de vitesse. Cependant, la complexité computationnelle du pipeline et sa dépendance à plusieurs modules peuvent limiter son intégration en temps réel dans des systèmes embarqués.

### 10.1.5 Synthèse et proposition d'une approche hybride

En résumé, les travaux existants apportent des contributions significatives à la détection, la classification et la segmentation des dommages automobiles, mais présentent encore des limites en termes de robustesse, de précision contextuelle et d'efficacité dans des conditions réelles. Les approches classiques privilégient souvent la rapidité au détriment de la précision fine, tandis que les méthodes plus complexes offrent une meilleure segmentation mais demeurent coûteuses et peu adaptées aux applications temps réel.

Pour surmonter ces compromis, nous proposons une approche hybride centrée sur **FuzzyYOLOv8**. Cette méthode conserve la rapidité et l'efficacité de YOLOv8 tout en intégrant un raisonnement flou pour renforcer la robustesse des décisions. Elle permet ainsi de stabiliser la détection des pièces intactes et endommagées dans des contextes difficiles (faible contraste, occlusion, bruit d'annotation) et d'offrir une évaluation plus fiable de la gravité des dommages, adaptée aux contraintes pratiques et aux scénarios variés rencontrés dans l'analyse automobile.

# 11 CHAPITRE 3 – Méthodologie proposée

## 11.1 YOLO (You Only Look Once)

YOLO est une technologie d'intelligence artificielle, largement utilisée pour la détection d'objets en temps réel. Son principe consiste à diviser l'image en une grille et à analyser chaque cellule afin de prédire, en un seul passage, la présence d'objets, leurs dimensions et leurs catégories. Cette approche permet de réaliser simultanément la localisation et la classification avec une grande efficacité[1, 18]. Depuis sa première version[2, 19], YOLO a connu de nombreuses améliorations, chacune visant à accroître la précision, la rapidité et la capacité à détecter des objets de tailles variées et dans des contextes complexes. Les versions récentes, telles que **YOLOv8**, offrent une meilleure précision et une robustesse accrue face aux scénarios complexes, ce qui en fait une solution de référence dans des domaines tels que la conduite autonome, la surveillance vidéo et l'évaluation des dommages automobiles. Bien que des versions plus récentes comme **YOLOv10** aient été publiées, YOLOv8 demeure un choix pertinent pour cette étude en raison de son équilibre entre précision, efficacité computationnelle et maturité. Son adoption large dans la communauté scientifique ainsi que sa documentation abondante en font une base solide pour l'intégration de raisonnements flous dans l'évaluation précise des pièces automobiles intactes et endommagées.

Cependant, YOLO présente également certaines limites, notamment en matière de détection d'objets de petite taille ou dans des environnements encombrés [20]. De plus, un déséquilibre des données ou l'utilisation d'ensembles de données provenant de domaines différents de ceux utilisés pour l'entraînement peut réduire ses performances [21].

### 11.1.1 Architecture de YOLO

L'architecture de YOLO est généralement divisée en trois modules principaux : le **backbone**, le **neck** et la **head** [19].

#### *a. Le Backbone*

Le backbone est la partie du réseau chargée d'**extraire les caractéristiques visuelles** à partir de l'image d'entrée.

- Dans YOLOv8, il repose sur un enchaînement de couches convolutionnelles et sur les blocs **C2f**, qui améliorent la transmission des gradients par rapport aux blocs C3 de YOLOv5.

- Le backbone permet de capturer à la fois des informations de bas niveau (contours, textures) et de haut niveau (formes complexes, pièces de véhicules, dommages).
- Il constitue donc la base de l'apprentissage de représentations discriminantes nécessaires à la détection et à la classification.

### *b. Le Neck*

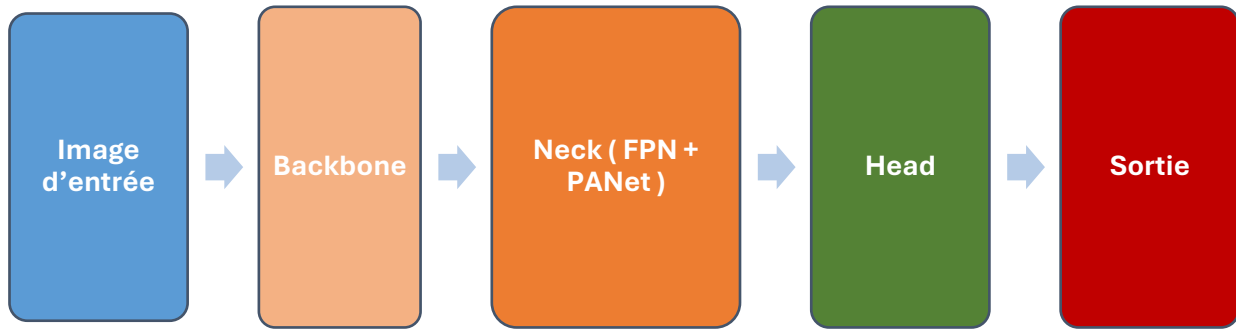
Le neck a pour rôle d'**agréger et de fusionner les caractéristiques** extraites à différents niveaux de profondeur du backbone.

- YOLOv8 intègre des structures inspirées de **Feature Pyramid Networks (FPN)** et de **Path Aggregation Network (PANet)**.
- Cette combinaison permet de mieux gérer la détection d'objets de **tailles variées**, en particulier les petits dommages automobiles, souvent difficiles à identifier.
- En regroupant des informations globales (vision d'ensemble) et locales (détails fins), le neck assure une meilleure robustesse face aux variations de perspective et de conditions d'éclairage.

### *c. La Head*

La head est la partie finale du modèle, responsable de la **prédiction des objets détectés**.

- Elle réalise simultanément la **régression des boîtes englobantes** (localisation) et la **classification des catégories** (pièce du véhicule intacte ou endommagée).
- YOLOv8 adopte une approche **anchor-free**, ce qui élimine la nécessité de définir manuellement des ancres et améliore la flexibilité.
- De plus, la **tête est découplée**, séparant les branches de localisation et de classification afin de réduire les interférences entre ces deux tâches.
- Cette conception optimise la précision tout en maintenant une rapidité élevée, ce qui est essentiel pour des applications en temps réel comme l'évaluation automatisée des dommages automobiles.



**Figure 1:** Architecture simplifiée de YOLO

## 11.2 Logique floue : concepts et applications

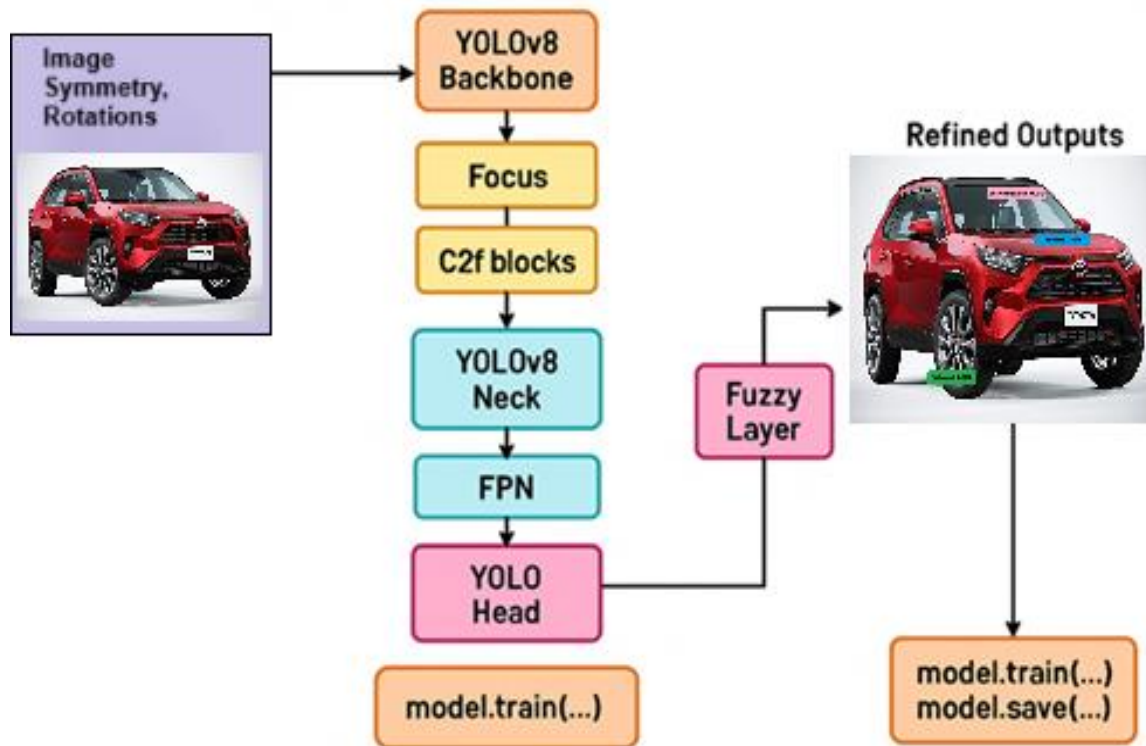
Toutefois, malgré son efficacité, **YOLO présente certaines limites**, notamment la difficulté à détecter les objets de petite taille, la baisse de performance dans les scènes encombrées, ainsi que la sensibilité au déséquilibre des données ou aux changements de domaine entre l’entraînement et les images réelles.

Pour pallier ces limites, nous proposons une amélioration de l’architecture standard de YOLOv8 à travers l’intégration de la logique floue, donnant naissance à une variante que nous nommons FuzzyYOLOv8 (voir Figure 2). La logique floue, introduite par Zadeh en 1965 [6], permet de traiter l’incertitude en attribuant des degrés de vérité compris entre 0 et 1, contrairement à la logique binaire traditionnelle. Elle repose sur l’utilisation de variables linguistiques, de fonctions d’appartenance, de règles conditionnelles et d’un mécanisme d’inférence pour faciliter la prise de décision. Grâce à sa capacité à représenter des concepts vagues ou qualitatifs, elle a trouvé des applications dans de nombreux domaines, notamment l’automatique, la médecine et l’intelligence artificielle [22, 23]. Néanmoins, cette approche présente aussi certaines limites : la définition des règles et fonctions d’appartenance dépend fortement de l’expertise humaine, ce qui peut introduire des biais subjectifs [7, 24]. En d’autres termes, la logique floue excelle dans le raisonnement mais reste limitée en apprentissage, tandis que les réseaux de Deep Learning apprennent efficacement mais manquent de capacités de raisonnement. C’est dans cette complémentarité que s’inscrit notre approche : nous introduisons un cadre hybride, **FuzzyYOLOv8**, qui combine l’apprentissage profond de YOLOv8 et les capacités de raisonnement de la logique floue (voir Section 4).

### 11.3 Intégration de la logique floue dans l'architecture YOLOv8

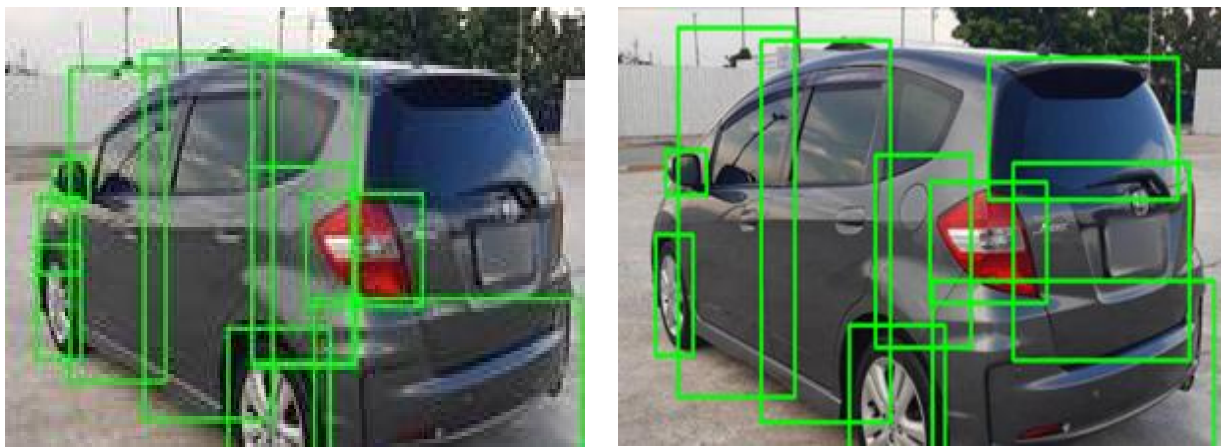
Dans le modèle **FuzzyYOLOv8** proposé, une couche dédiée de logique floue implémentée comme un module personnalisé sous PyTorch est intégrée au sein du pipeline de détection (voir Figure 2). Cette couche affine les scores de confiance bruts produits par la tête de détection de YOLOv8 en appliquant des **fonctions sigmoïdes paramétrées**. Contrairement aux seuils binaires stricts utilisés dans les architectures YOLO classiques, ces fonctions floues introduisent des transitions plus progressives entre les prédictions de faible confiance (incertaines) et celles de forte confiance. Ce mécanisme permet au système de mieux refléter les ambiguïtés inhérentes aux données visuelles réelles [6].

De plus, la couche floue introduit des ajustements dynamiques pénalisant les boîtes englobantes présentant des anomalies géométriques. Par exemple, les boîtes avec des rapports d'aspect extrêmes ou situées à proximité des bords de l'image se voient systématiquement attribuer des valeurs de confiance plus faibles. En outre, des informations contextuelles locales sont exploitées : la confiance d'une boîte est modulée en fonction de la moyenne des confiances de ses boîtes voisines. Ce mécanisme renforce le raisonnement contextuel, améliorant ainsi la robustesse face au bruit, aux environnements encombrés et aux incohérences de détection.



**Figure 2:** Vue synthétique de l'architecture FuzzyYOLOv8 intégrant une couche de logique floue à l'architecture YOLOv8

Figure 3 compare les performances des modèles : le panneau (a) montre que le modèle de base manque la détection du pare-brise arrière, ce qui met en évidence des limites de reconnaissance, tandis que le panneau (b) illustre que le modèle amélioré par la logique floue détecte avec succès tous les composants, suggérant ainsi une meilleure robustesse et une couverture accrue.



**Figure 3:** Comparaison des performances des différents modèles de détection

Figure 3. a) Dans cette image, le modèle YOLOv8 n'a pas réussi à détecter le pare-brise arrière, ce qui met en évidence une limitation de la reconnaissance des pièces dans certaines conditions. b) En revanche, le second modèle (FuzzyYOLOv8), qui utilise la logique floue, a détecté avec succès toutes les pièces du véhicule, montrant une meilleure robustesse et une couverture accrue dans la détection des pièces.

Dans ce qui suit, nous fournissons une explication détaillée de notre approche.

En nous appuyant sur les idées de recouvrement flou proposées par Khelifi [3] pour YOLOv5 à savoir une extension floue du CIoU fondée sur des règles linguistiques appliquées à plusieurs indicateurs géométriques nous identifions plusieurs limitations qui freinent son déploiement en temps réel et dans des contextes spécifiques. Premièrement, cette approche implique une ingénierie lourde des règles : la conception et l'ajustement de nombreuses règles floues et fonctions d'appartenance sont coûteux et fragiles d'un jeu de données à l'autre. Deuxièmement, la méthode manque de modularité prête à l'emploi : elle n'est pas présentée comme un opérateur de recouvrement léger et universel pouvant être directement inséré dans les têtes de détection standards ou dans le Non-Maximum Suppression (NMS). Troisièmement, la dépendance à une base explicite de règles et à une fusion multi-indicateurs accroît la complexité du pipeline, ce qui complique son intégration en temps réel de bout en bout. Enfin, Khelifi [3] ne fournit pas de validation spécifique au domaine pour des applications critiques comme l'évaluation de pièces automobiles endommagées, où les recouvrements partiels et les indices visuels faibles sont fréquents.

En revanche, nous remplaçons cette fuzzification lourde en règles par une simple transformation concave et paramétrable de l'IoU standard. Cette transformation est à la fois simple et différentiable (pour  $\text{IoU} > 0$ ), totalement compatible avec les calculs existants de scores, de NMS et de métriques, négligeable en termes de coût computationnel par rapport à l'IoU lui-même, et contrôlable à l'aide d'un seul paramètre  $\alpha$ , qui permet de moduler la tolérance au recouvrement partiel. Nous définissons ainsi le recouvrement flou comme une transformation en puissance de l'IoU.

$$\text{Fuzzy IoU} = (\text{IoU})^\alpha$$

et, plus formellement, pour deux boîtes  $b_1, b_2$ ,

$$FIoU_{\alpha(b_1, b_2)} = IoU(b_1, b_2)^\alpha, \alpha \in (0, 1]$$

Lorsque  $\alpha < 1$ , la transformation est concave et monotone, ce qui atténue les pénalités pour les recouvrements faibles mais significatifs, correspondant précisément au cas rencontré avec des dommages subtils ou partiellement visibles. En pratique, nous fixons  $\alpha=0.1$  sur la base d'expérimentations empiriques afin d'améliorer la robustesse dans des conditions de faible recouvrement, de faible résolution ou d'indices visuels faibles. Bien que cette valeur n'ait pas été déterminée via une optimisation formelle, elle offre un bon compromis entre sensibilité et tolérance au bruit dans des scénarios typiques. Des travaux futurs pourraient explorer des méthodes plus systématiques pour ajuster  $\alpha$ .

Pour plus de précision, étant donné deux boîtes  $b_1 = [x_1, y_1, x_2, y_2]$  et  $b_2 = [x'_1, y'_1, x'_2, y'_2]$ , soit  $A_\cap$  la surface d'intersection et  $A_\cup$  la surface de l'union de  $b_1$  et  $b_2$ . Le recouvrement standard est défini par :

$$IoU_{(b_1, b_2)} = \frac{A_\cap}{A_\cup} ;$$

et le recouvrement flou proposé est :

$$FIoU_{\alpha(b_1, b_2)} = \left( \frac{A_\cap}{A_\cup} \right)^\alpha .$$

Cette formulation améliore l'approche de [3] en regroupant de nombreuses règles heuristiques en un seul paramètre interprétable  $\alpha$ ; en préservant l'interprétabilité du seuil (un seuil de décision  $\tau$  sur FIoU correspond à un seuil IoU  $\tau^{1/\alpha}$ ; en s'intégrant de manière transparente aux composants existants du détecteur sans nécessiter de base de règles ; et en améliorant la robustesse au domaine grâce à l'inflation des valeurs de faible recouvrement lorsque  $\alpha < 1$ , ce qui permet de mieux gérer les occlusions, les reflets, le flou de mouvement et le bruit d'annotation typiques des images de pièces endommagées.

L'algorithme de calcul du FIoU est résumé ci-dessous :

### **Algorithm FuzzyConfidenceAdjustment :**

### Inputs:

x : input image  
pred\_boxes : list of predicted boxes (x1, y1, x2, y2, confidence, cls)  
a, b : fuzzy layer parameters  
alpha : fuzzy exponent for IoU

### Output:

adjusted\_boxes : list of boxes with adjusted confidence

---

```
1  Function FuzzyLayer(value, a, b)
2      return clamp((value - a) / (b - a), 0, 1)

3  Function ComputeIoUFuzzy(box1, box2, alpha)
4      x1 ← max(box1.x1, box2.x1)
5      y1 ← max(box1.y1, box2.y1)
6      x2 ← min(box1.x2, box2.x2)
7      y2 ← min(box1.y2, box2.y2)
8
9      intersection ← max(0, x2 - x1) × max(0, y2 - y1)
10     area1 ← (box1.x2 - box1.x1) × (box1.y2 - box1.y1)
11     area2 ← (box2.x2 - box2.x1) × (box2.y2 - box2.y1)
12     union ← area1 + area2 - intersection
13
14     if union > 0 then
15         iou ← intersection / union
```

```

16     else
17         iou ← 0
18     return iou^alpha

19 For each box in pred_boxes:
20     extract (x1, y1, x2, y2, confidence, cls)
21     ratio ← (x2 - x1) / (y2 - y1)
22
23     if ratio < 1 then confidence ← confidence + 0.1
24     else if ratio > 2 then confidence ← confidence - 0.1
25
26     if box is near the image borders then
27         confidence ← confidence - 0.1
28
29     iou ← ComputeIoUFuzzy(box, box)
30     if iou > 0.8 then confidence ← confidence + 0.1
31     else if iou < 0.4 then confidence ← confidence - 0.1
32
33     neighbors ← boxes with IoU > 0.5
34     avg_conf ← mean confidence of neighbors
35
36     if avg_conf > 0.85 then confidence ← confidence + 0.05
37     else if avg_conf < 0.5 then confidence ← confidence - 0.05
38
39     box.confidence ← FuzzyLayer(confidence, a, b)
40
41 return adjusted_boxes

```

Dans les sous-sections suivantes, nous expliquons comment l'architecture **FuzzyYOLOv8** est utilisée à la fois pour l'évaluation locale des pièces automobiles endommagées et pour l'évaluation et la classification globales des dommages du véhicule.

## 11.4 FuzzyYOLOv8 pour l'évaluation locale des dommages

Lors de la phase d'inférence, chaque modèle produit un ensemble de boîtes englobantes [2], associées à des classes d'objets (pièces intactes ou endommagées) et à des scores de confiance. Ces boîtes définissent l'étendue spatiale des entités détectées et servent de base à la classification.

Pour classer de manière fiable la gravité des dommages détectés, **FuzzyYOLOv8** commence par fusionner les boîtes englobantes qui se chevauchent en utilisant un seuil de IoU flou prédéfini (FIoU=0,5). Les boîtes dépassant ce seuil sont fusionnées en une seule boîte représentative, offrant une correspondance spatiale cohérente pour les pièces intactes et endommagées du véhicule.

La gravité des dommages est ensuite évaluée à l'aide d'un raisonnement flou, où des fonctions d'appartenance floues permettent des transitions progressives entre les classes de gravité (intacte, modérée, sévère, très sévère). Cette approche de classification floue tient compte des ambiguïtés visuelles et de l'incertitude du modèle, offrant ainsi une meilleure interprétabilité et fiabilité dans l'évaluation de la gravité des dommages [25].

De plus, l'intersection entre les boîtes représentant les pièces intactes et les dommages détectés fournit un **ratio de dommages quantitatif**, calculé comme suit :

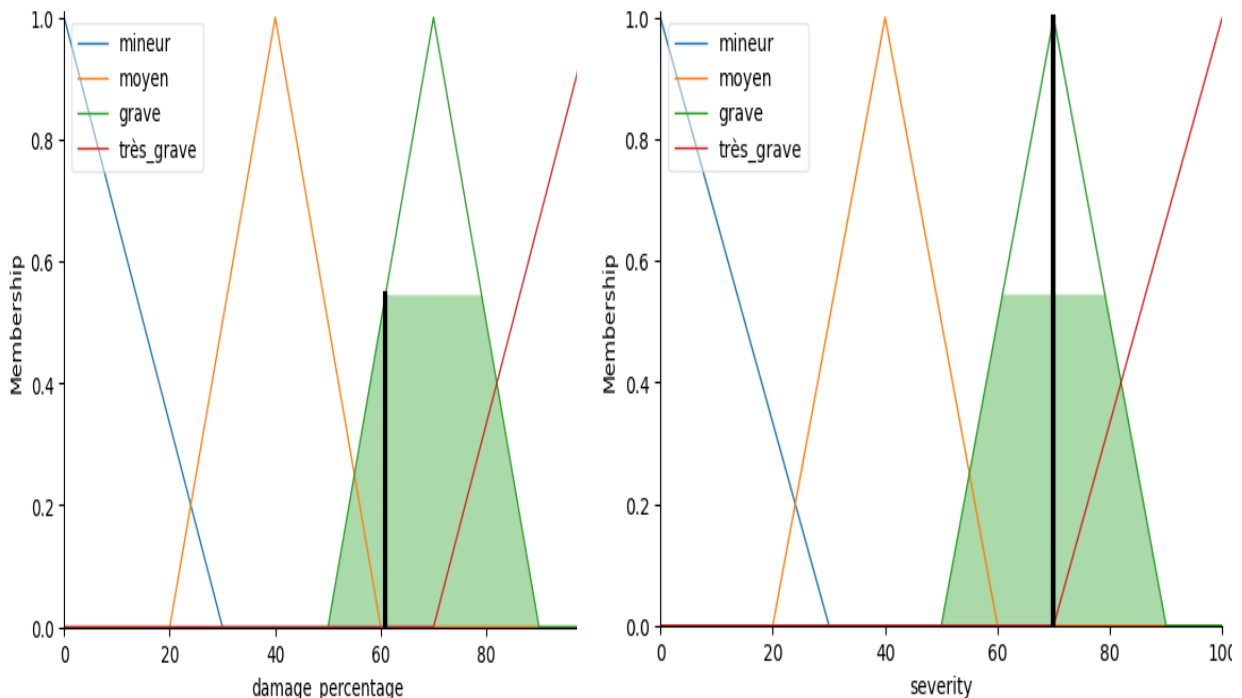
$$\text{Damage Ratio} = \left( \frac{\text{Area of Intersection}}{\text{Area of the Part}} \right) \times 100$$

Ce ratio permet de classer systématiquement chaque pièce détectée comme endommagée dans des catégories de sévérité prédéfinies (voir ci-dessous pour plus de détails) : **mineure** ( $\leq 25\%$ ), **modérée** (25–50 %), **sévère** (50–75 %), et **très sévère** ( $> 75\%$ ), permettant ainsi une évaluation structurée et priorisée des dommages.

## 11.5 FuzzyYOLOv8 pour l'évaluation globale des dommages

Des règles de raisonnement flou sont définies afin de relier les niveaux de dommages localisés à une évaluation globale. Par exemple, si la majorité des pièces présentent uniquement des dommages mineurs, la gravité globale est considérée comme faible. À l'inverse, si une grande proportion de pièces présente des dommages sévères ou critiques, la gravité globale est considérée comme élevée. Ce raisonnement flou introduit une capacité d'interprétation proche de celle de l'humain, dépassant les limites d'une simple moyenne numérique. De plus, les pièces automobiles peuvent être pondérées afin de mieux refléter leur importance réelle dans l'analyse des dégâts après accident, renforçant ainsi la pertinence de l'évaluation finale.

Le score final de gravité globale (Figure 4) est dérivé grâce aux mécanismes d'inférence floue, où les pourcentages moyens de dommages servent d'entrées. Après l'inférence et la défuzzification, une sortie numérique ou catégorielle est produite pour résumer le niveau global de dommages du véhicule.



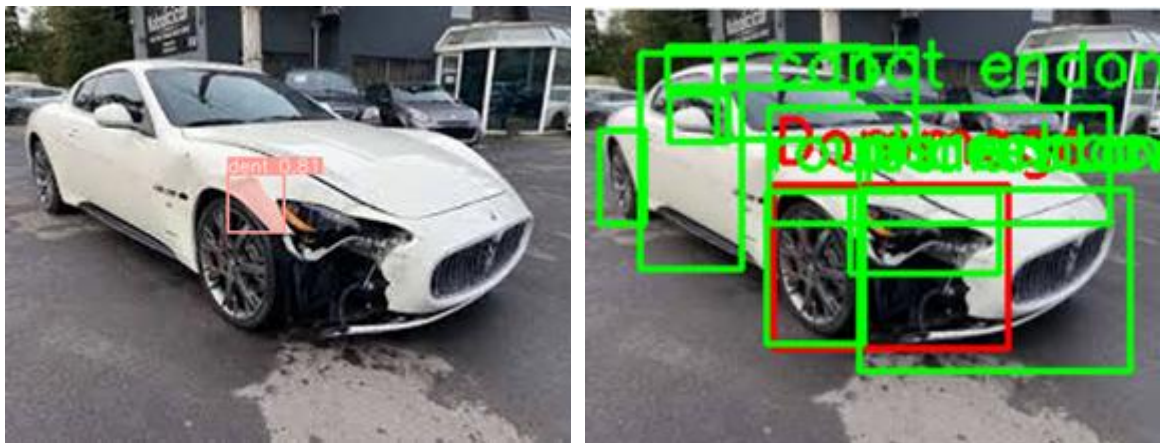
**Figure 4:** Gravité des dommages

Les figures 4.a et 4.b illustrent la phase de défuzzification dans le cadre de la logique floue appliquée à la quantification de la gravité des dommages sur les composants du véhicule. Chaque graphique montre des fonctions d'appartenance triangulaires représentant les catégories

linguistiques **mineur**, **modéré**, **sévère** et **très sévère**, mappées sur un score de dommage continu allant de 0 à 100. La zone verte correspond aux memberships flous agrégés associés à une entrée donnée, tandis que la ligne noire verticale indique la valeur de sortie défuzzifiée. Dans les deux figures, le pic d'appartenance se situe dans la catégorie **sévère** (environ 65–70 %), indiquant un fort alignement de l'entrée avec cette classe. Cette cohérence entre l'appartenance de l'entrée et la sortie finale souligne la fiabilité et l'interprétabilité du système flou dans la gestion des prédictions incertaines ou imprécises du modèle de détection des dommages.

Plus précisément, un dommage est classé comme **mineur** lorsque la zone de recouvrement entre la boîte englobante de la pièce et la boîte de dommage est inférieure ou égale à 25 %, **modéré** pour des recouvrements compris entre 25 % et 50 %, **sévère** entre 50 % et 75 %, et **très sévère** lorsque le recouvrement dépasse 75 %. Cette catégorisation fournit un cadre standardisé et objectif pour évaluer et rapporter l'étendue des dommages sur le véhicule.

Grâce à l'intégration de la logique floue, notre modèle démontre une performance améliorée dans la détection des dommages, comme illustré à la **Figure 5**.



**Figure 5:** Détection de dommages : modèle Yolov8 vs Yolov8+logique floue

Figure 5. a) Montre le modèle Yolov8 simple fait une détection avec une confiance de 0,81 qui ne parvient pas à identifier le dommage. En revanche, b) illustre comment notre modèle YOLOv8 + logique floue localise avec précision la zone endommagée malgré la faible résolution de l'image.

## 11.6 Filtrage des pièces détectées à l'aide du clustering et du seuil spatial

Comme illustré à la **Figure 6**, un défi avec YOLO pour la détection des pièces endommagées d'un véhicule est sa tendance à détecter également des pièces sur les véhicules environnants, non endommagés. Cela se produit car YOLO traite l'image dans son ensemble et ne distingue pas intrinsèquement l'objet principal d'intérêt des objets similaires situés à proximité. Cela entraîne des imprécisions dans l'évaluation des dommages, les résultats étant biaisés par des pièces non liées au véhicule cible.

Pour résoudre ce problème, une technique de filtrage combinant deux stratégies a été mise en œuvre : le **clustering basé sur la surface** [4] et la **sélection en fonction de la position horizontale** [5]

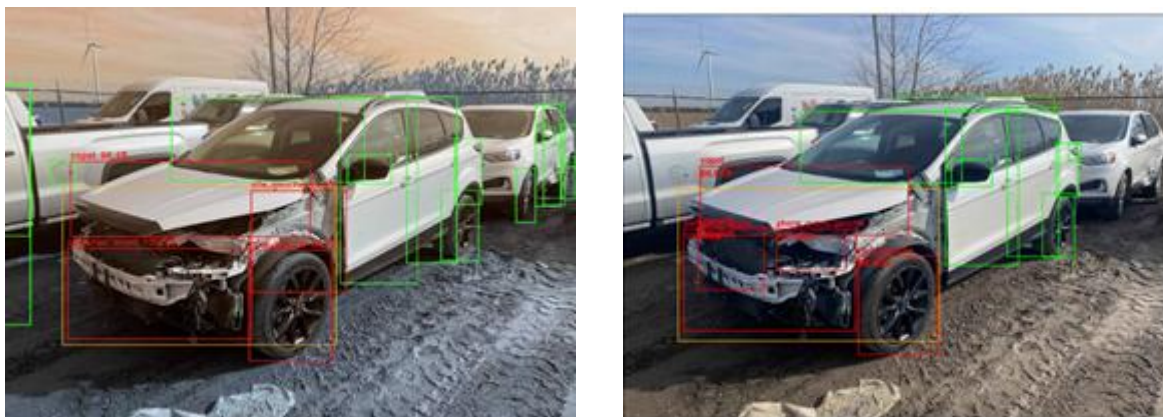
### a. Clustering basé sur la surface

Servant de proxy pour l'association au niveau des instances, en supposant que les pièces du véhicule principal ont une surface plus grande que les pièces détectées sur les véhicules de fond. En regroupant les boîtes englobantes selon leur surface et en conservant le cluster avec la plus grande surface moyenne, on filtre efficacement les pièces appartenant probablement à des véhicules secondaires ou à des objets non pertinents. Cette méthode permet d'éliminer la majorité des éléments de petite taille, souvent associés à des véhicules secondaires ou à du bruit de fond.

### b. Filtrage par position horizontale

Le filtrage par position horizontale agit comme un filtre de région d'intérêt (ROI) basé sur le positionnement spatial. Il suppose que le véhicule principal est centré dans l'image et utilise une marge seuil autour du centre de l'image pour écarter les détections sur les véhicules périphériques.

La Figure 6b montre l'application des algorithmes ci-dessus et les résultats obtenus. Ainsi, le système se concentre exclusivement sur les pièces du véhicule principal, améliore la précision de la détection et assure une évaluation des dommages plus fiable.



**Figure 6:** Filtrage pièces véhicule principal

La figure 6. a) YOLOv8 détecte des pièces sur les véhicules environnants proches du sujet principal. b) Montre comment les méthodes de filtrage décrites ci-dessus permettent de se concentrer exclusivement sur le véhicule principal, améliorant ainsi la précision de la détection des pièces.

## 11.7 Capacité du modèle à distinguer les pièces gauches et droites du véhicule

L'une des caractéristiques de notre modèle de détection est sa capacité à différencier avec précision les côtés gauche et droit des composants du véhicule. À notre connaissance, aucun autre travail antérieur n'a abordé cette problématique. Nous avons également interrogé ChatGPT version 4 (voir Annexe A), et celui-ci n'est pas capable de détecter les côtés gauche et droit du véhicule, ce qui est pourtant particulièrement critique dans des contextes où la précision topologique est essentielle, tels que les systèmes automatisés d'évaluation des dommages, les analyses liées à l'assurance, ou encore les inspections robotiques de véhicules.

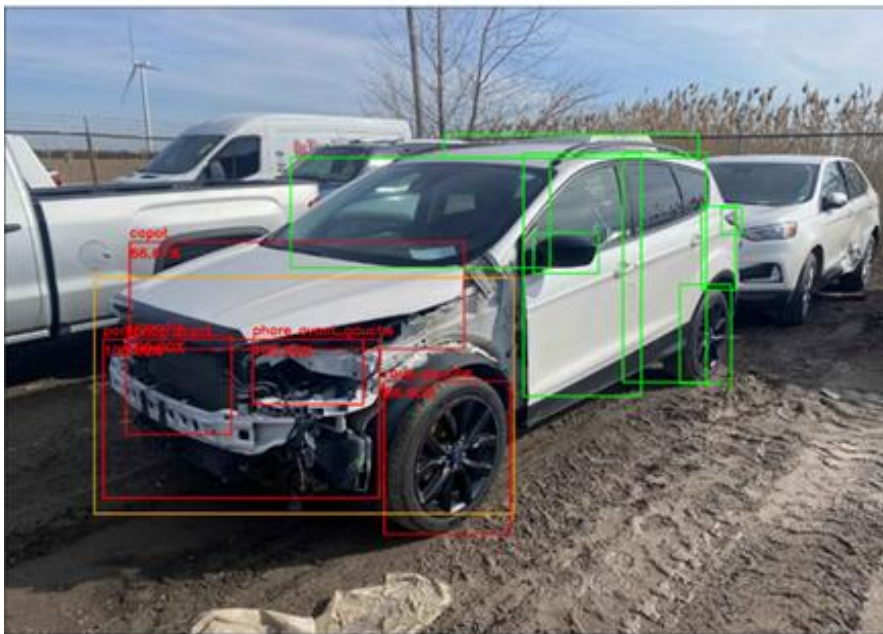
Notre modèle réalise cette classification en comparant la position horizontale du centre de la boîte englobante (bounding box) de chaque composant détecté à une référence spatiale. Lorsque le pare-chocs avant est détecté, son centre sert d'ancrage directionnel permettant de résoudre la symétrie latérale de l'image. La règle de classification s'exprime ainsi :

$$\text{Side} = \begin{cases} \text{"left"} & \text{if } x_c < x_f \\ \text{"right"} & \text{if } x_c \geq x_f \end{cases}$$

Dans les cas où un tel composant de référence n'est pas disponible, le centre de l'image  $x_m = \frac{W}{2}$ , où W représente la largeur de l'image, est utilisé comme solution de repli.

$$\text{Side} = \begin{cases} \text{"left"} & \text{if } x_c < x_m \\ \text{"right"} & \text{if } x_c \geq x_m \end{cases}$$

Cette approche permet la classification des côtés, même en présence d'occlusions partielles ou de points de vue non standards, et assure une meilleure cohérence lors de l'association des dommages détectés aux côtés correspondants du véhicule (Figure 7).



**Figure 7:** Précision côté et dommages

La Figure 7 illustre les zones de dommages détectées, par exemple le rétroviseur gauche avec les coordonnées [957, 433, 1075, 510]. Pour une liste complète des pièces, veuillez-vous référer à l'Annexe B

# 12 CHAPITRE 4 - Expérimentations

## 12.1 Description des jeux de données

Le développement de notre système repose sur un jeu de données comprenant plus de 700 images annotées, représentant à la fois des véhicules intacts et endommagés. Environ 80 % des images ont été collectées à partir de cas réels au Canada, en collaboration avec **Progi**, une entreprise partenaire spécialisée dans les technologies d'assurance automobile [20]. Les 20 % restants proviennent de dépôts publics en ligne, principalement de l'Université de Stanford, permettant une plus grande variété de conditions environnementales (par exemple, éclairage, complexité de l'arrière-plan, angles de vue), ce qui augmente la diversité du jeu de données.

### b. Traitement de données

Nous avons réalisé plusieurs opérations de prétraitement afin de nettoyer et d'optimiser notre collection de données. Dans un premier temps, nous avons supprimé les images de qualité insuffisante ainsi que les doublons. Ensuite, nous avons écarté les images où les dommages étaient difficiles à distinguer, en raison de l'angle de prise de vue ou de la faible résolution.

Nous avons également retiré les images de véhicules entièrement détruits, pour lesquels aucune partie identifiable (roues, portes, rétroviseurs, etc.) n'était visible. Après ce filtrage, notre base de données se compose désormais d'environ 60 % d'images de véhicules endommagés, présentant des dommages sur les côtés, l'avant ou l'arrière, et de 40 % d'images de véhicules intacts, prises sous différents angles. Nous avons choisi de privilégier les véhicules endommagés, car nos tests ont montré que le modèle détecte aisément les pièces non endommagées, indépendamment de l'angle de prise de vue.

Une fois cette étape de collecte et de filtrage achevée, nous avons pu passer à l'étape suivante : l'annotation précise des images.

### c. Étiquetage des pièces et zones de dommages dans le jeu de données

Chaque image a été annotée manuellement via la plateforme **Roboflow** [26], suivant une taxonomie structurée des pièces du véhicule (capot, pare-chocs, portes, phares, roues, pare-brise, etc.) et des zones de dommages associées. Des boîtes englobantes (*bounding boxes*) ont été attribuées à chaque

objet, accompagnées de labels précis, incluant l'orientation gauche/droite pour les composants latéraux.

Notre base de données se compose de 24 classes, détaillées dans le tableau ci-dessous

|    |  |
|----|--|
| 0  | Parechoc avant                                     |
| 1  | Parechoc arrière                                   |
| 2  | Calandre   |
| 3  | Aile droite  |
| 4  | Aile gauche  |
| 5  | Capot  |
| 6  | Parebrise  |
| 7  | Toit   |
| 8  | Phare avant droit                                  |
| 9  | Phare avant gauche                                 |
| 10 | Phare arrière droit                                |
| 11 | Phare arrière gauche                               |
| 12 | Porte avant droite                                 |
| 13 | Porte avant gauche                                 |
| 14 | Porte arrière droite                               |
| 15 | Porte arrière gauche                               |
| 16 | Rétroviseur droit                                  |
| 17 | Rétroviseur gauche                                 |
| 18 | Roues (spécifier la quantité de roues endommagées) |
| 19 | Panneau latéral droit                              |
| 20 | Panneau latéral gauche                             |
| 21 | Coffre   |
| 22 | Vitre arrière                                      |
| 23 | Endommagée   |

**Table 1:** Liste des classes

Selon le protocole standard de gestion des données de Roboflow, le jeu de données a été divisé en 70 % pour l'entraînement, 20 % pour la validation et 10 % pour les tests, garantissant une approche équilibrée pour l'apprentissage du modèle et l'évaluation de sa généralisation.

Les annotations ont été exportées au format **YOLOv8** pour une intégration fluide dans le pipeline d'entraînement. Pour améliorer davantage la robustesse du modèle dans des conditions réelles, des techniques d'augmentation de données ont été appliquées, telles que le retournement horizontal, l'ajustement de la luminosité et du contraste, les rotations et le zoom.

## **12.2 Autres outils**

Pour mettre en œuvre notre méthodologie, nous avons utilisé plusieurs outils et plateformes complémentaires, chacun jouant un rôle clé dans la gestion des données, le développement et l'entraînement des modèles, ainsi que dans l'annotation des images.

### **a. Python : Langage de programmation**

Python a été utilisé pour développer les scripts, créer et entraîner les modèles d'apprentissage automatique, gérer les pipelines de traitement d'images et automatiser les étapes d'évaluation. La richesse de ses bibliothèques scientifiques et frameworks d'IA a rendu son usage indispensable [27].

### **b. Google Drive : Stockage et gestion des données**

Google Drive a servi de plateforme centralisée pour stocker et organiser l'ensemble des fichiers, permettant un accès sécurisé et partagé par les différents scripts et collaborateurs du projet [28].

### **c. Google Colaboratory : Environnement de calcul**

Google Colaboratory a fourni l'infrastructure nécessaire pour entraîner nos modèles sur de larges ensembles d'images. L'accès aux GPU et TPU a permis d'accélérer significativement l'apprentissage et les expérimentations [29].

## 13 CHAPITRE 5 – Résultats

Les résultats sont présentés sous plusieurs formats complémentaires. Les images d'entrée originales sont superposées avec les boîtes englobantes annotées, indiquant les pièces détectées et les pourcentages de dommages correspondants. Ce retour visuel facilite une compréhension rapide et intuitive de l'état du véhicule.

Dans ce qui suit, nous discuterons des résultats pour la détection et la classification des pièces intactes (Tableau 2) et endommagées (Tableau 3), en comparant les performances du YOLOv8 et de FuzzyYOLOv8 intégrant la logique floue.

### 13.1 Détection des pièces intactes : comparaison entre YOLOv8 et FuzzyYOLOv8

La comparaison repose sur Trois métriques clés : la précision moyenne (mAP) à un seuil FIoU de 0,5 (mAP@0.5), la précision moyenne sur plusieurs seuils (mAP@50:95), et le F1-score.

| Modèles /Métriques | <u>mAP@0.5</u> | mAP@50:95 | Score F1 |
|--------------------|----------------|-----------|----------|
| YOLOv8             | 0,899          | 0,440     | 0,87     |
| FuzzyYOLOv8        | 0,92           | 0,58      | 0,88     |

*Table 2: Comparaison de la détection des pièces intactes*

Le modèle de base atteint une **mAP@0,5** de 0,899 et une **mAP@50:95** de 0,440, avec un **F1-score** de 0,87 et une **précision** de 95,2 %. Après intégration de la logique floue, ces valeurs passent à 0,92 pour la mAP@0.5, 0,58 pour la mAP@50:95 et 0,88 pour le F1-score.

Ces améliorations soulignent que la logique floue permet une meilleure gestion des incertitudes dans la localisation et la classification des pièces. Elle aide à mieux traiter les cas complexes où les contours sont flous, les pièces partiellement visibles ou les objets très similaires. Le gain significatif observé sur la **mAP@50:95** démontre également que le FIoU confère une plus grande robustesse du modèle face aux seuils d'IoU, garantissant ainsi une détection plus précise et fiable dans des scénarios réels.

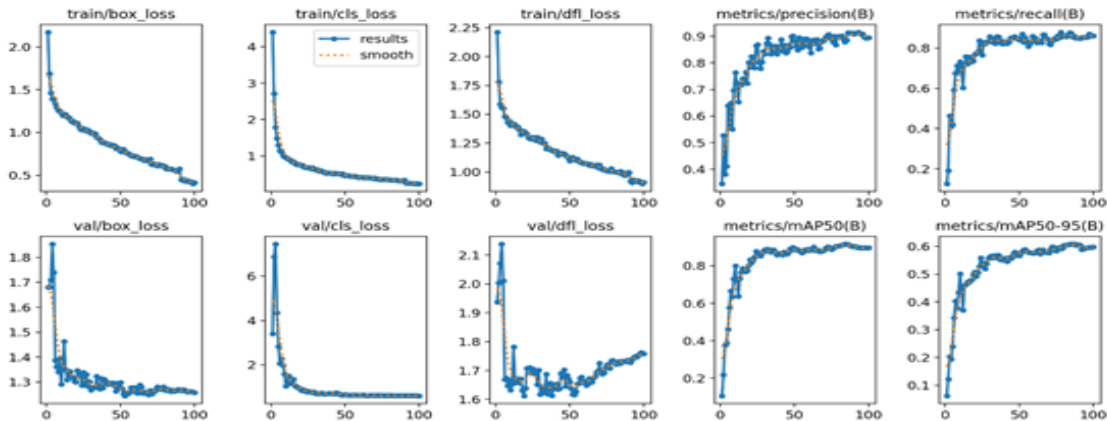
En résumé, l'intégration de la logique floue améliore la qualité des prédictions tout en augmentant la confiance dans la détection, ce qui est crucial pour des applications industrielles telles que l'inspection automatisée des véhicules ou les systèmes d'assurance.

## 13.2 Détection des pièces endommagées : YOLOv8 et FuzzyYOLOv8

| Modèles /Métriques | <u>mAP@0.5</u> | mAP@50:95 | Score F1 |
|--------------------|----------------|-----------|----------|
| YOLOv8             | 0.637          | 0.315     | 0.64     |
| FuzzyYOLOv8        | 0.68           | 0.34      | 0.69     |

**Table 3:** Comparaison de la détection des pièces endommagées

Table 3 présente une comparaison détaillée entre l'architecture YOLOv8 seule et l'architecture FuzzyYOLOv8 pour la détection et la classification des pièces de voiture endommagées. Le modèle de base, YOLOv8-Damages, atteint un mAP@0.5 de 0,637 et un mAP@50:95 de 0,315, avec un F1-score de 0,64. En comparaison, le modèle FuzzyYOLOv8 améliore l'ensemble des métriques, atteignant un mAP@0.5 de 0,680, un mAP@50:95 de 0,340, et un F1-score de 0,69. Ces résultats mettent en évidence une amélioration significative des performances grâce à l'intégration de la logique floue, confirmant sa capacité à renforcer la robustesse et la précision de la détection des dommages.



**Figure 8:** Amélioration de la convergence et des performances de détection avec FuzzyYOLOv8 par rapport au modèle de base.

La Figure 9 montre la progression de l'entraînement et les métriques de détection d'objets pour le modèle FuzzyYOLOv8. Les courbes de perte d'entraînement et de validation diminuent de manière parallèle et cohérente tout au long des itérations, sans divergence significative. Par exemple, la perte de localisation en validation (*val/box\_loss*) suit de près celle de l'entraînement (*train/box\_loss*), les deux convergeant vers environ 1,3. Cette valeur indique une réduction substantielle par rapport à la perte initiale et se situe dans une plage acceptable pour des tâches de détection d'objets avec YOLOv8, en particulier compte tenu de la complexité liée à la détection de multiples pièces automobiles présentant des conditions de dommages variées. Le fait que les deux

courbes se stabilisent autour de cette valeur suggère que le modèle minimise efficacement les erreurs de localisation tout en conservant une bonne capacité de généralisation. De même, la perte de classification (*val/cls\_loss*) chute rapidement et se stabilise selon un schéma qui reflète celui de son équivalent en entraînement, ce qui renforce le comportement d'apprentissage cohérent du modèle. Dans l'ensemble, cette dynamique de convergence soutient la conclusion selon laquelle le modèle apprend efficacement sans surapprentissage des données d'entraînement.

Deuxièmement, les métriques de précision et de rappel augmentent progressivement au cours de l'entraînement, se stabilisant respectivement autour de 0,91 et 0,86 sur l'ensemble de validation. L'absence de baisses brutales indique une forte capacité de généralisation et des performances de détection robustes sur des données inédites.

Enfin, les scores mAP50 et mAP50-95 indicateurs clés de la qualité de détection selon différents seuils de FIoU atteignent respectivement 0,9 et 0,6. Ces valeurs confirment que le modèle conserve une haute précision spatiale même sous des critères d'évaluation plus stricts.

En comparaison avec le YOLOv8 de base, le FuzzyYOLOv8 présente non seulement une meilleure stabilité et des dynamiques d'apprentissage plus solides, mais aussi une précision de détection supérieure, mettant en évidence l'avantage d'intégrer le raisonnement flou pour gérer l'incertitude dans les tâches de localisation et de classification d'objets.

## **14 CHAPITRE 6 - Menaces à la validité de l'étude**

### **14.1 Validité interne**

La taille limitée du jeu de données d'entraînement, composé de seulement 700 images, réduit la diversité des cas rencontrés par le modèle. Cette limitation peut entraîner une couverture insuffisante des différentes configurations de véhicules, types de dommages et conditions d'éclairage, limitant ainsi la capacité du système à s'adapter à des scénarios réels variés. De plus, les entrées sont basées sur une seule image capturée sous un angle unique du véhicule. Par conséquent, le modèle ne peut pas détecter simultanément des pièces ou des dommages situés sur les côtés opposés (par exemple, avant et arrière). Cette limitation réduit la robustesse et la pertinence dans des situations réelles où plusieurs zones d'intérêt peuvent être visibles en même temps, ce qui compromet la capacité du système à produire une estimation globale et cohérente des dommages.

### **14.2 Validité externe**

Les véhicules présents dans le jeu de données proviennent exclusivement d'Amérique du Nord. Les variations de conception, de dimensions ou de matériaux des véhicules d'autres régions pourraient réduire la précision du modèle. L'intégration de véhicules provenant d'autres marchés nécessiterait probablement des ajustements de paramètres ou un réentraînement pour maintenir les performances.

### **14.3 Validité de construit**

L'évaluation de la gravité des dommages se base uniquement sur la partie externe des composants du véhicule, sans tenir compte des dommages internes, ce qui peut sous-estimer la gravité réelle. De plus, le modèle ne prend pas en considération la position des pièces (avant ou arrière) lors de l'évaluation de la gravité. Cela peut introduire un biais dans l'interprétation, car certaines pièces ont une importance stratégique ou des coûts de réparation plus élevés selon leur emplacement. Une solution d'Atténuation peut résider dans la pondération des pièces dans le calcul de l'estimation globale des dégâts.

## 15 CHAPITRE 7- Conclusion

Dans cette étude, nous avons présenté un modèle innovant associant YOLOv8 et la logique floue afin de renforcer la fiabilité de la détection des pièces de véhicules endommagées et d'améliorer la classification de la gravité des dommages. Entraîné sur un jeu de données de 700 images, notre modèle a surpassé la version standard de YOLOv8 sur des métriques clés telles que le mAP@50 et le score F1, confirmant l'apport significatif de l'intégration de la logique floue dans les systèmes de détection automatisée.

Pour approfondir ce travail, plusieurs perspectives se dessinent. Tout d'abord, l'élargissement et la diversification du jeu de données, incluant des images capturées sous différents angles et provenant de contextes géographiques variés, permettront d'accroître la robustesse et la capacité de généralisation du modèle. Par ailleurs, l'intégration de méthodes multi-vues ou de techniques de reconstruction 3D pourrait offrir la possibilité de détecter simultanément les dommages présents sur plusieurs côtés d'un même véhicule, renforçant ainsi la précision de l'évaluation. De plus, la prise en compte des dommages internes, souvent invisibles à l'œil nu, ainsi que l'introduction d'une pondération basée sur la localisation des pièces (avant, arrière) constitueraient des pistes intéressantes pour affiner l'estimation de l'impact et des coûts de réparation. Enfin, l'adaptation du modèle à une diversité de types et de modèles de véhicules, à travers des approches d'apprentissage par transfert, garantirait une applicabilité plus large et flexible.

Ces orientations ouvrent la voie vers un système d'évaluation automatisée des dommages automobiles à la fois plus complet, robuste et pertinent, contribuant à accroître l'efficacité et la précision des processus d'inspection dans des contextes réels, tels que l'assurance, l'expertise technique ou la maintenance automobile

# 16 REFERENCES

1. Redmon, J. *You only look once: Unified, real-time object detection*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
2. Sohan, M., T. Sai Ram, R. Reddy, and C. Venkata. *A review on yolov8 and its advancements*. in *International Conference on Data Intelligence and Cognitive Informatics*. 2024. Springer.
3. Khelifi, E., *Règles associatives floues et logique de raisonnement dans les modèles de vision par ordinateur (Fuzzy YOLO-V5)*. 2023, Université du Québec à Trois-Rivières.
4. Madhulatha, T.S., *An overview on clustering methods*. arXiv preprint arXiv:1205.1117, 2012.
5. Zhu, C., W. Zhang, T.H. Li, S. Liu, and G. Li, *Exploiting the value of the center-dark channel prior for salient object detection*. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2019. **10**(3): p. 1-20.
6. Zadeh, L.A., *Fuzzy sets*. *Information and Control*, 1965.
7. Lee, K.H., *First course on fuzzy theory and applications*. Vol. 27. 2004: Springer Science & Business Media.
8. Redmon, J. and A. Farhadi. *YOLO9000: better, faster, stronger*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
9. Rezatofighi, H., N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese. *Generalized intersection over union: A metric and a loss for bounding box regression*. in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019.
10. Zhang, Z., Y. Zou, Y. Tan, and C. Zhou, *YOLOv8-seg-CP: a lightweight instance segmentation algorithm for chip pad based on improved YOLOv8-seg model*. *Scientific Reports*, 2024. **14**(1): p. 27716.
11. Kirillov, A., E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A.C. Berg, and W.-Y. Lo. *Segment anything*. in *Proceedings of the IEEE/CVF international conference on computer vision*. 2023.
12. Dwivedi, M., H.S. Malik, S. Omkar, E.B. Monis, B. Khanna, S.R. Samal, A. Tiwari, and A. Rathi. *Deep learning-based car damage classification and detection*. in *International Conference on Artificial Intelligence and Data Engineering*. 2019. Springer.
13. Van Ruitenbeek, R. and S. Bhulai, *Convolutional Neural Networks for vehicle damage detection*. *Machine Learning with Applications*, 2022. **9**: p. 100332.

14. Pasupa, K., P. Kittiworapanya, N. Hongngern, and K. Woraratpanya, *Evaluation of deep learning algorithms for semantic segmentation of car parts*. Complex & Intelligent Systems, 2022. **8**(5): p. 3613-3625.
15. Mohammed, N.A., M.Y. Potrus, and A.M. Ali, *Deep learning based car damage classification and cost estimation*. Zanco Journal of Pure and Applied Sciences, 2023. **35**(1): p. 1-9.
16. Qaddour, J. and S.A. Siddiq, *Automatic damaged vehicle estimator using enhanced deep learning algorithm*. Intelligent Systems with Applications, 2023. **18**: p. 200192.
17. Li, R., *A multi stage deep learning approach for real-time vehicle detection, tracking, and speed measurement in intelligent transportation systems*. Scientific Reports, 2025. **15**(1): p. 22531.
18. Ultralytics, *YOLOv8 Documentation*. Documentation en ligne 2025.
19. Terven, J., D.-M. Córdova-Esparza, and J.-A. Romero-González, *A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas*. Machine Learning and Knowledge Extraction, 2023. **5**(4): p. 1680-1716.
20. Xu, L., Y. Zhao, Y. Zhai, L. Huang, and C. Ruan, *Small Object Detection in UAV Images Based on YOLOv8n*. International Journal of Computational Intelligence Systems, 2024. **17**.
21. Potter, K., A. Egon, and A. Gracias, *MACHINE LEARNING ALGORITHMS FOR IMAGE CLASSIFICATION AND OBJECT DETECTION*.
22. Takagi, T. and M. Sugeno, *Fuzzy identification of systems and its applications to modeling and control*. IEEE transactions on systems, man, and cybernetics, 1985(1): p. 116-132.
23. Mamdani, E.H. and S. Assilian, *An experiment in linguistic synthesis with a fuzzy logic controller*. International journal of man-machine studies, 1975. **7**(1): p. 1-13.
24. Murofushi, T. and M. Sugeno, *Fuzzy measures and fuzzy integrals*. Fuzzy measures and integrals: theory and applications, 2000. **2000**: p. 3-41.
25. Shokri, D., C. Larouche, and S. Homayouni, *Proposing an efficient deep learning algorithm based on segment anything model for detection and tracking of vehicles through uncalibrated urban traffic surveillance cameras*. Electronics, 2024. **13**(14): p. 2883.
26. Roboflow. 2025 [cited 10 août 2025; Available from: : <https://roboflow.com/>].
27. Foundation, P.S. *Python* 2025; Available from: <https://www.python.org/>.
28. Google. *Google Drive*. 2025 [cited 2025 12 Septembre]; Available from: <https://workspace.google.com/products/drive/>.
29. Google. *Google Colaboratory*. 2025 [cited 2025 12 Septembre]; Available from: <https://colab.research.google.com/>.

# 17 Annexe A

## LIMITATIONS OBSERVÉES AVEC OPENAI

- Identification erronée des pièces (pièces manquantes, mauvaise identification de certains pièces, résultats inconsistants)
- Difficulté à déterminer l'orientation des pièces (droite/gauche). Complexité accrue lorsque le véhicule est de dos, face, de côté, etc.

It seems the AI is incorrectly identifying left and right sides from the driver's perspective. This happens because computer vision models often interpret images in a default left-right manner based on the image orientation, not necessarily the real-world driver's perspective.

### • Why is This Happening?

1. AI relies on image orientation rather than contextual spatial awareness. → The AI may not correctly infer the driver's perspective.
2. AI might use the image as seen from a human viewer rather than from inside the car. → This means it can mix up left and right.
3. Angle of the photo affects perspective interpretation. → If the image is taken from the front-left side of the car, it may confuse "driver's left" and "passenger's right."

- L'engin de ChatGPT nous renvoyait des erreurs de façon sporadique (ça fonctionne un moment, et 2 minutes après non - on reçoit alors une erreur de l'engin sans aucun détail)
- Difficulté à générer une réponse fiable au format JSON (le JSON généré n'était souvent pas valide, et les résultats ne sont pas consistants entre deux exécutions) - nous avons dû opter pour un format CSV

I understand the intermittent issue you're experiencing: occasionally, the GPT-4o model returns responses that cannot be parsed as valid JSON, leading to errors in your application. This inconsistency can be challenging, especially when the model works correctly at times and fails at others.

## Potential Causes:

1. **Model Output Variability:** Even with explicit instructions to return JSON-formatted responses, the model might occasionally include extra text, comments, or formatting inconsistencies that render the JSON invalid.
2. **Response Truncation:** The model's response might be cut off due to token limits or other factors, resulting in incomplete JSON structures.

Même en essayant de raffiner le prompt itérativement, on arrive à une impasse et OpenAI lui-même suggère d'utiliser des modèles mieux adaptés tels que YOLO

## 🔧 Next Steps

- Run the script with multiple images to validate accuracy.
- Fine-tune OpenCV edge detection for better results.
- Improve damage classification by integrating YOLOv8 for object detection.

Would you like me to help integrate YOLOv8 with GPT-4o for even more precise damage analysis?

# 18 Annexe B

## Liste des pièces endommagées d'une voiture accidentée dans la Figure 5 :

- **Capot**, Coordonnées : [227, 450, 832, 654]
- **Pare-chocs avant**, Coordonnées : [179, 636, 678, 938]
- **Panneau latéral gauche**, Coordonnées : [1261, 382, 1323, 541]
- **Porte arrière gauche**, Coordonnées : [1119, 279, 1270, 717]
- **Roue gauche**, Coordonnées : [687, 716, 915, 1007]
- **Roue gauche**, Coordonnées : [1222, 531, 1315, 720]
- **Rétroviseur gauche**, Coordonnées : [957, 433, 1075, 510]
- **Calandre**, Coordonnées : [220, 629, 411, 817]
- **Pare-brise**, Coordonnées : [518, 289, 987, 499]
- **Toit**, Coordonnées : [795, 245, 1256, 290]
- **Porte avant gauche**, Coordonnées : [939, 281, 1152, 744]
- **Phare avant gauche**, Coordonnées : [449, 636, 649, 760]
- **Phare arrière gauche**, Coordonnées : [1297, 384, 1334, 436]