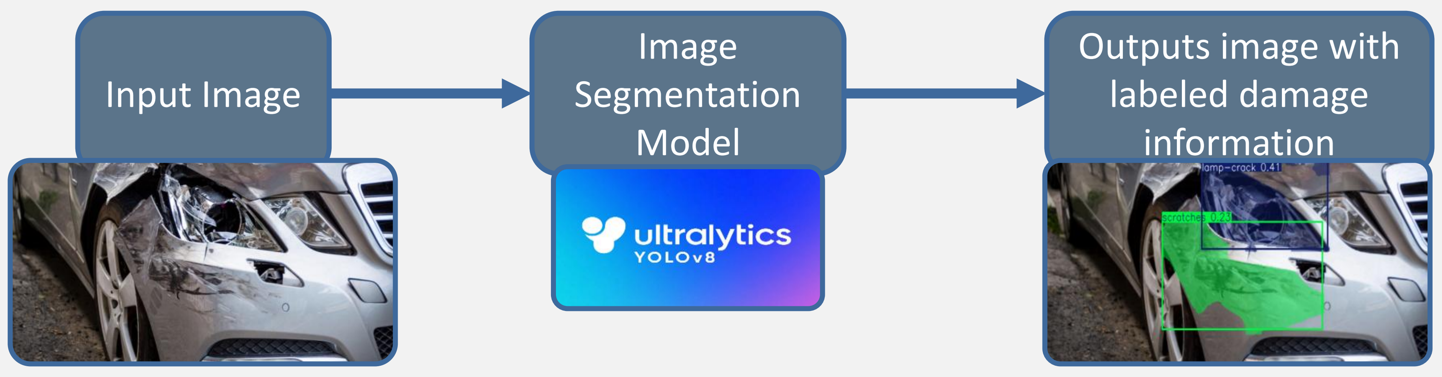
Car Damage and Repair Cost Estimation Using Image Segmentation

Ciências ULisboa
M.Sc. Data Science
Deep Learning class 2024/2025
Alexandre Baptista (nº 64506) ; Carolina Rodrigues (nº 62910)

Objectives

- **Image segmentation task** and **damage detection** from vehicle photos.
- **Aim**: Develop an AI-based application that analyzes vehicle images to:
  - ✓ **Detect and localize vehicle damage** precisely on the vehicle;
  - ✓ **Categorize types** of damage (e.g., dents, scratches, cracks);
  - ✓ **Estimate repair costs** based on damage severity.



Problem

- <u>Problem statement</u>: Assessing vehicle damage is essential for insurance companies, yet manual inspections are slow, costly, and error-prone.
- An AI-based system, from customer-taken photo, can **locate** and **classify exterior damages** and produce a report with **cost-range estimates**.
- This makes inspection process faster, more reliable and cost-effective, cutting the on-site physical inspections expenses.
- <u>**Common approaches and drawbacks:**</u> Some early digital solutions are being developed [1, 2] but remain largely unused by insurers. These tools often miss uncommon damage patterns and rarely provide reliable cost estimates.

Methodology

1. **Dataset Download**
   - "Car DD - Car Damage Detection"
   - From Roboflow
   - Structured in Yolov8 format
2. **Split**
   - Dataset Split in 70/20/10 ratio
3. **Sanity checks**
   - Data Leakage
   - data Corruption
   - Label Balance
4. **Configuration file**
   Edit the config.yaml file for corresponding labels, so that the model can read the labels correctly
5. **Train Models**
   - Tain models yolov8 seg in n, m and l
   - and for 10, 25 and 50 epochs
6. **Evaluate Models**
   Evaluate each model and compare its Precision, Recall mAP50 and F1-Score.

The methodology integrates YOLOv8 segmentation models. The experiments:

1. **Baseline + Hyper-parameter search**: Tune learning rate and optimizer for YOLOv8m baseline.
2. **Augmentation test:** Baseline Vs. Albumentations enabled.
3. **Model-size comparison**: Best YOLOv8m against YOLOv8s and YOLOv8l, each trained for 10, 25 and 50 epochs.

Conclusions

Although we expected the larger models (**YOLOv8l**) to outperform the smaller ones, results were that: only after 50 epochs it improved **mAP** only marginally over the medium model (**YOLOv8m**) yet required far **more training time** and **GPU memory**.

<u>**Future Work**</u>: Extend the model for precise cost-estimation → using mask size to predict damage-specific repair-cost estimates based on the size, location and severity of the affected area.

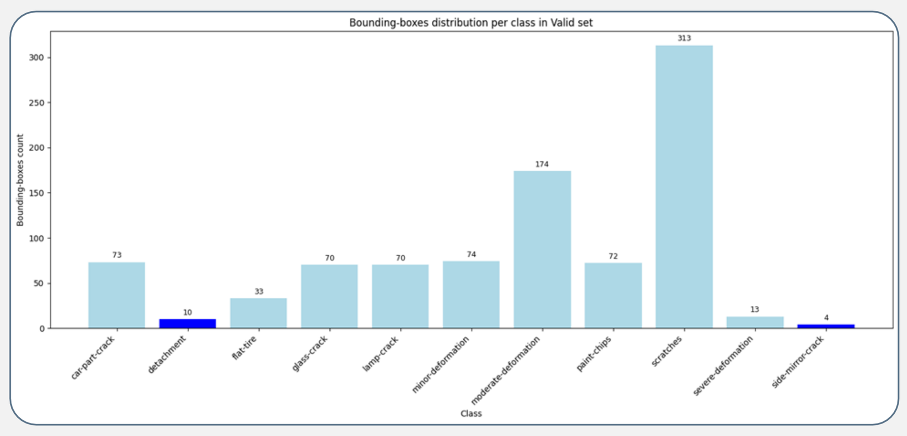
Dataset

- **Source:** *"Car Damage Detection (CarDD)"* dataset from Roboflow [3];
- **Format**: YOLOv8 (data.yaml + .txt file per image (polygons and class IDs) → **multiple labels**
- **Split**: 70/20/10 → ***Train*** = 1 395, ***Valid*** = 401, ***Test*** = 204 images and labels;

Image sample from valid set

Classes (11): *car-part-crack, detachment, flat-tire, glass-crack, lamp-crack, minor-deformation, moderate-deformation, paint-chips, scratches, severe-deformation, side-mirror-crack*

- **Sanity Checks:**
  - ✓ No leakage (SHA-1 & pHash);
  - ✓ No corrupted images (PIL verify);
  - × Moderate class imbalance.



Bar chart of the inference's distribution, highlighting the unbalanced classes

Results

<u>Experiment 1:</u> Baseline model and Hyper-parameter Search

- YOLOv8 variants: **n, s** (small, for mobile/edge), **m** (balanced), and **l, x** (large, hight capacity);
- Baseline model → **YOLOv8m :** Balance between speed and accuracy;
- Hyper-parameter sweep → **(lr ∈ {0.005, 0.01, 0.05}, Adam vs RAdam)**.

Model	Learning rate:	Optimizer:	time(s):	precision(B)	recall(B)	F1(B)	mAP50(B)	precision(M)	recall(M)	F1(M)	mAP50(M)
YOLOv8m-seg	0.01	* not specified	584,54	0,415	**0,412**	0,413	**0,391**	0,426	<u>0,396</u>	0,411	**0,384**
	0.01	**Adam**	565,15	0,462	0,345	0,395	0,323	0,470	0,332	0,389	0,304
	0.05	**Adam**	541,25	<u>0,554</u>	0,253	0,347	0,223	<u>0,562</u>	0,244	0,340	0,211
	0.005	**Adam**	525,50	0,342	0,374	0,357	0,348	0,338	0,364	0,350	0,340
	0.01	SGD	**519,65**	0,425	<u>0,461</u>	<u>0,442</u>	<u>0,412</u>	0,423	0,451	<u>0,437</u>	<u>0,403</u>
	0.01	**Radam**	<u>508,08</u>	**0,510**	0,376	**0,433**	0,380	**0,511**	**0,369**	**0,428**	0,363

Highlighted in **blue** → baseline model; in **green** → best model parameters
(underlined and in bold, are the first- and second-best performing metrics, respectively; (B) = bounding box detection, (M) = segmentation masks)

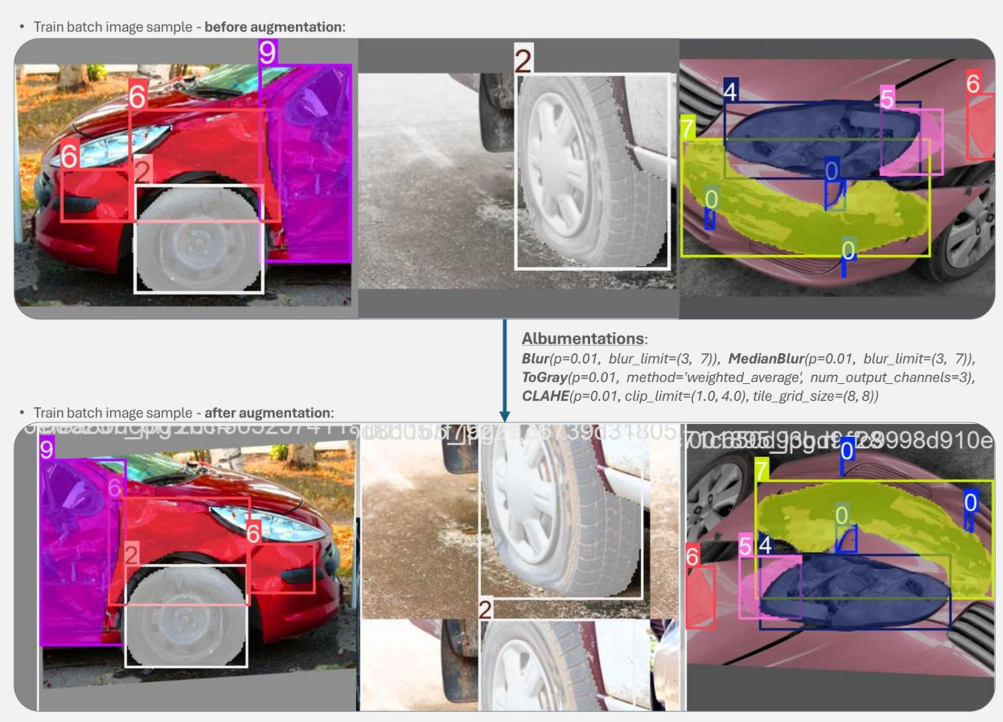
<u>Experiment 2:</u> Baseline model and Hyper-parameter Search

- Sanity checks revealed moderate class imbalance, we tested YOLOv8's built-in **Albumentations** 6ẻºɔừ ì ửâℓŇỷºì 7 and compared results to the non-augmented baseline.

Yolov8m

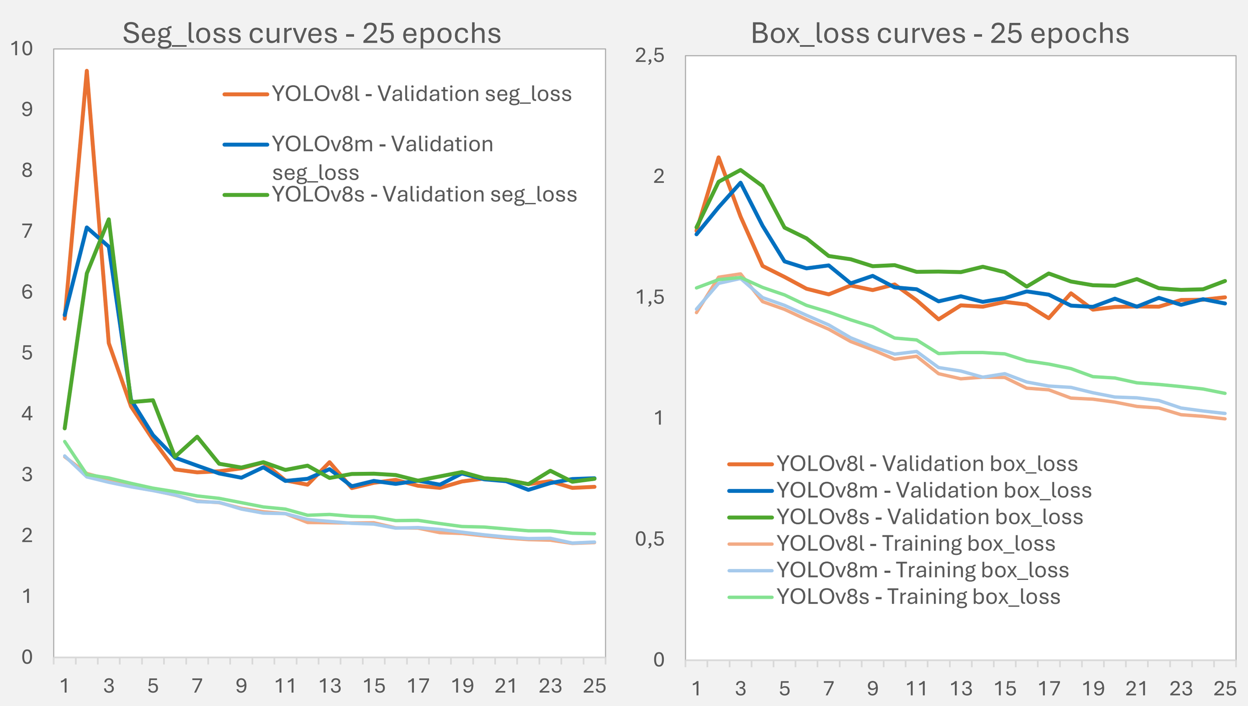
Yolov8m baseline	precision(B)	recall(B)	F1(B)	mAP50(B)	precision(M)	recall(M)	F1(M)	mAP50(M)
10ep - No augmentation	0,51017	0,37605	0,4330	0,38042	0,51122	0,36858	0,4283	0,36333
10ep - With augmentation	0,47033	0,39487	0,4293	0,38585	0,46334	0,38714	0,4218	0,37576
25ep -No Augmentation	0,51905	0,42047	0,4646	0,40831	0,52865	0,40696	0,4599	0,40015
25ep -With Augmentation	**0,57255**	<u>0,39713</u>	0,4690	**0,40846**	**0,58204**	<u>0,39231</u>	**0,4687**	**0,40623**
50ep -With Augmentation	**0,57475**	0,39379	0,4674	0,41227	<u>0,55029</u>	0,38597	0,4537	0,39725

In bold metric improvement and underlined metric underperformed.



- In 25 epochs, Albumentations slightly improved precision and mAP but decreased recall.
- In 50 epochs, improved in precision and mAP for object detection (B) but decreased for the masks (M).

<u>Experiment 3:</u> Best YOLOv8m Vs. YOLO (s and l) versions



Among all models tested, **YOLOv8-l trained for 50 epochs** delivered the most consistent and best overall performance.

Therefore, the one used for the predictions.

Yolov8s

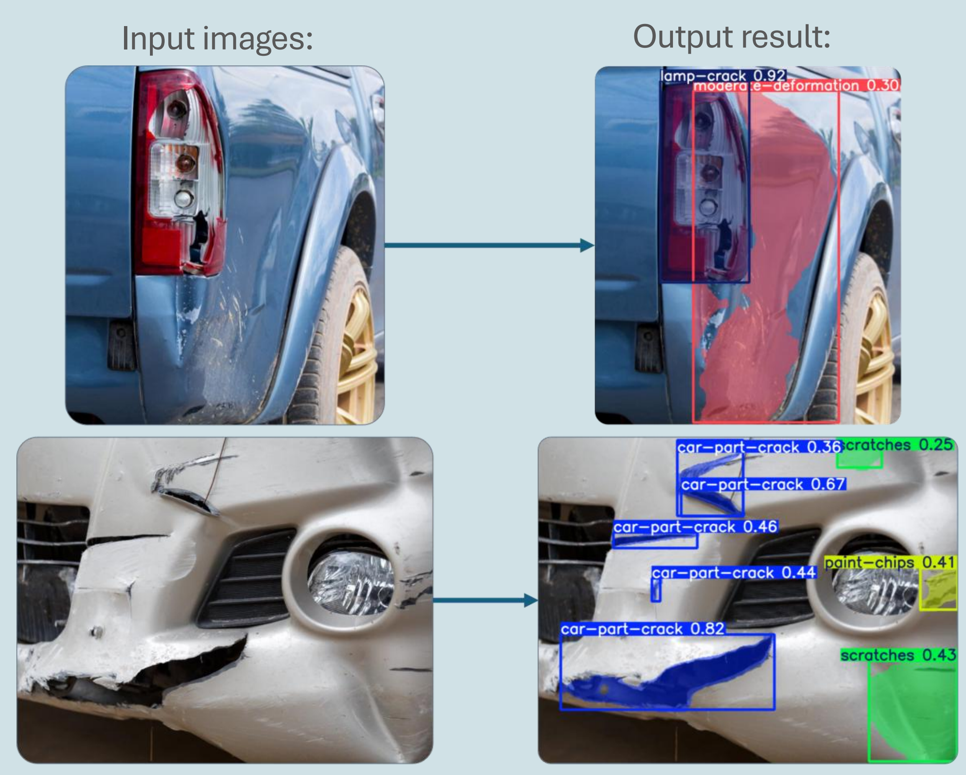
Yolov8s	precision(B)	recall(B)	F1(B)	mAP50(B)	precision(M)	recall(M)	F1(M)	mAP50(M)
10ep -With Augmentation	0,31202	0,41965	0,3579	0,36638	0,57452	0,34099	0,4280	0,35275
25ep -With Augmentation	0,40805	0,49807	0,4486	0,40804	0,39865	0,47358	0,4329	0,39366
50ep - With Augmentations	0,4838	0,4082	0,4428	0,3888	0,4987	0,4051	0,4471	0,3832

Yolov8l

Yolov8l	precision(B)	recall(B)	F1(B)	mAP50(B)	precision(M)	recall(M)	F1(M)	mAP50(M)
10ep -With Augmentation	0,41829	0,39295	0,4052	0,3635	0,37203	0,34402	0,3575	0,25432
25ep -With Augmentation	0,40631	0,51403	0,4539	0,4207	0,39572	0,49504	0,4398	0,40881
50ep - With Augmentations	**0,50113**	0,45456	**0,4767**	**0,44807**	**0,50591**	**0,45789**	**0,4807**	**0,43559**

In bold metric improvement.
Model chosen for prediction yolov8l with 50 epochs

<u>Predictions:</u> YOLOv8l trained for 50 epochs



Refferences

[1] Celebal Technologies, "Leading Insurance Provider Leverages AI-Powered Vehicle Damage Detection," Case Study. [Online]. Available: https://celebaltech.com/case-studies/leading-insurance-provider-leverages-ai-powered-vehicle-damage-detection
[2] Solera. "Guided Image Capture – Product Overview." Vehicle Claims & Collision, https://www.claims.solera.com/products/guided-image-capture/
[3] Roboflow dataset, "Car Damage Severity Detection – CarDD" [Online]. Available: https://universe.roboflow.com/car-damage-detection-cardd/car-damage-severity-detection-cardd
[4] YOLOv8.org. "How to Fine-Tune YOLOv8." [Online]. Available: https://yolov8.org/how-to-use-fine-tune-yolov8/
[5] Ultralytics, "YOLOv8 Models — Documentation." [Online]. Available: https://docs.ultralytics.com/models/yolov8.

LISBOA UNIVERSIDADE DE LISBOA
Acknowledgements: This poster session is kindly sponsored by: NTT DATA