Car Damage and Repair Cost Estimation Using Image Segmentation

Alexandre Baptista, nr fc64506 ; Carolina Rodrigues, nr fc62910

1. Problem Statement and motivation:

This project aims to develop a system that detects and classifies vehicle damage based on images as well as estimating the corresponding repair costs.

Given one or more images of the same damaged vehicle as input, the system will:

- Detect and identify the types of damage (such as scratches, dents, broken glass);
- Estimate the total repair cost.

This is particularly interesting for insurance and automotive industries, where this process is often done with manual inspections which are time-consuming and prone to subjectivity. With this project we intend to improve the accuracy and efficiency of damage detection and cost prediction.

2. Research and background

We will review previous academic papers on car damage detection (such as R.E. van Ruitenbeek, S. Bhulai, *Convolutional Neural Networks for vehicle damage detection* [1]. This study develops a model to locate vehicle damages and classify them into twelve categories using multiple deep learning algorithms.
And car industry case study on how AI and image recognition technology can improve insurance claims estimation processes. [2]

3. Dataset

For this project, we are using the **"Damaged Vehicle Images"** dataset available on Roboflow [3], which contains over 5,000 labeled images of damaged vehicles. The dataset provided was exported in **COCO JSON format**, including bounding boxes and class labels for various types of damage, such as **scratches, dents, and broken windshield**.

The original dataset was distributed as follows: Train set: 94% (4,777 images), Validation set: 4% (193 images) and Test set: 2% (102 images).

To ensure a more balanced and standardized evaluation, we created a new project on Roboflow, uploaded the dataset, and used the platform's built-in tools to apply a new **80/10/10 split** for training, validation, and testing.

This choice follows best practices in dataset preparation aimed at balancing model generalization and avoiding overfitting, as discussed in *https://encord.com/blog/train-val-test-split/*.

Using the **Roboflow Python API**, we then exported the dataset and structure to our project folder, as follows:

```python
#dataset from Roboflow
from roboflow import Roboflow
rf = Roboflow(api_key="***")
project = rf.workspace("c-zef0o").project("damaged-vehicle-images-euqbx")
version = project.version(1)
dataset = version.download("folder")
```

```
loading Roboflow workspace...
loading Roboflow project...
Downloading Dataset Version Zip in Damaged-Vehicle-Images-1 to folder:: 100%| | 192548/192548 [00:18<00:00, 10231.67it/s]


Extracting Dataset Version Zip to Damaged-Vehicle-Images-1 in folder:: 100%| | 5160/5160 [00:05<00:00, 998.85it/s]
```

The resulting folder, his generates a folder **Damaged-Vehicle-Images-1/** containing train/, valid/, and test/, where each image is organized by damage category.

Additionally, we will explore complementary datasets, such as the Electric Vehicle Dataset (1997–2024) from Kaggle [4], this information will be used to support the **cost estimation** component of the project, as repair costs can vary significantly depending on these characteristics.

4. Methods and Algorithms

To segment and classify car damage, we will explore deep learning models, particularly:

In a first approach, the initial model:

In a first implementation we used the **Keras** framework (with TensorFlow backend) and a basic CNN model trained with the following configuration:

- **Framework**: Keras (TensorFlow)
- **Optimizer**: Adam
- **Training**: 15 epochs, batch size = 32

The preliminary results from this model were:

- **Training accuracy:** 0.7555
- **Training loss:** 0.7588
- **Validation accuracy:** 0.0533
- **Validation loss:** 12.2260

Next Steps

To improve performance, we plan to re-implement the model in PyTorch, using ResNet-50, an architecture known for its robustness and efficiency in image classification tasks [5].

We will compare the two implementations (Keras vs. PyTorch) in terms of its accuracy, training speed and robustness

In the later phases, we also plan to incorporate object detection models such as **Faster R-CNN** for better damage localization, and explore regression models for cost estimation based on extracted damage features.

5. Evaluation and Expected Results

The performance of the system will be assessed using both **qualitative** and **quantitative** evaluation methods, focusing on two core tasks: **damage detection/classification** and **repair cost estimation**.

Qualitative Evaluation

We will visually inspect the model outputs to verify if the system correctly identifies damaged areas and focuses attention on relevant regions.

These visualizations will help validate whether the model is learning meaningful features and whether the predictions are interpretable and trustworthy.

Quantitative Evaluation

For the **damage classification** component, we will compute standard classification metrics:

- **Accuracy; Precision; Recall; F1-score; Confusion matrix**

For the **repair cost estimation** component, we will consider two scenarios:

- Categorical estimation (such as low/medium/high cost range). Evaluation using classification **accuracy** and **F1-score.**
- **Continuous estimation** exact monetary value. Evaluation using **Mean Squared Error (MSE)** and **Mean Absolute Error (MAE)**

These metrics will allow us to measure both the classification effectiveness and the accuracy of the cost prediction component.

We expect to produce performance plots such as learning curves and confusion matrices, along with segmented image examples and cost prediction distributions, to fully document and communicate the system's results.

References:

[1] https://doi.org/10.1016/j.mlwa.2022.100332

[2]https://celebaltech.com/case-studies/leading-insurance-provider-leverages-ai-powered-vehicle-damage-detection

[3] https://universe.roboflow.com/project/damaged-vehicle-images/dataset/3

[4] https://www.kaggle.com/datasets/iottech/electric-vehicle-data-1997-2024-update-version

[5] https://blog.roboflow.com/what-is-resnet-50