İstanbul
Bilgi Üniversitesi

CAR DAMAGE DETECTION

by

HÜSEYİN ENES TANDOĞAN, 119200040
ŞABAN BİNGÜL, 119200042
BATIKAN YILMAZ, 120200036

Supervised by

PROF. DR. SAVAŞ YILDIRIM

Submitted to the
Faculty of Engineering and Natural Sciences
in partial fulfillment of the requirements for the

Bachelor of Science

in the
Department of Computer Engineering

June, 2024

Abstract

The Car Damage Detection project harnesses the power of deep learning to develop a robust system capable of automatically identifying and assessing car damage from images. By leveraging the advanced Detectron2 framework for object detection and segmentation, this project provides a sophisticated solution that accurately detects various types of car damage. The system is implemented as a user-friendly web application using Streamlit, enabling users to upload images of damaged vehicles and receive detailed information about the detected damage areas and their corresponding repair costs. This automation streamlines the assessment process, ensuring consistency and accuracy, making it an invaluable tool for insurance companies, repair shops, and vehicle owners. This report outlines the methodology, design, implementation, and evaluation of the Car Damage Detection system, highlighting the innovative techniques and technologies employed to achieve its objectives.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
GPU	Graphics Processing Unit
IEEE	Institute of Electrical and Electronics Engineers
R-CNN	Region-based Convolutional Neural Network

1 Introduction

In the modern automotive industry, accurate and timely assessment of vehicle damage is critical for both insurance claims and repair processes. Traditional methods of damage assessment often rely on manual inspection, which can be time-consuming, subjective, and prone to human error. With the advancements in deep learning and computer vision, there is a significant opportunity to automate this process, leading to faster and more reliable damage evaluations.

The Car Damage Detection project aims to harness the power of deep learning to develop a robust and efficient system capable of automatically identifying and assessing car damage from images. By leveraging the state-of-the-art Detectron2 framework for object detection and segmentation, this project provides a sophisticated solution that can identify various types of car damage with high accuracy.

Implemented as a user-friendly web application using Streamlit, the Car Damage Detection system allows users to upload images of damaged vehicles and receive detailed information about the detected damage areas and their corresponding repair costs. This automation not only speeds up the assessment process but also ensures consistency and accuracy, making it an invaluable tool for insurance companies, repair shops, and vehicle owners.

The project involves several key components, including data collection and annotation, model training and fine-tuning, and the development of an interactive web interface. Each of these components is meticulously designed to ensure seamless integration and optimal performance, resulting in a comprehensive solution for automated car damage detection.

By providing a reliable and accessible method for car damage assessment, the Car Damage Detection project aims to revolutionize the way vehicle damages are evaluated, bringing significant benefits to various stakeholders in the automotive and insurance industries. This report details the methodology, design, implementation, and evaluation of the Car Damage Detection system, highlighting the innovative techniques and technologies employed to achieve its objectives.

Through this project, we aim to demonstrate the potential of deep learning in solving real-world problems, specifically in the domain of automotive damage assessment. The following sections will provide an in-depth look at the various stages of the project, from the initial data collection and model training to the development of the web application and its deployment.

We believe that the Car Damage Detection system represents a significant advancement in the field of automated damage assessment, offering a practical and scalable solution that can be adopted across multiple industries.

By automating the detection and evaluation of car damage, this system not only enhances efficiency but also ensures a higher degree of accuracy and reliability, ultimately benefiting end-users and stakeholders alike.

In summary, this report serves as a comprehensive guide to the Car Damage Detection project, detailing the methodologies and design principles that underpin its development, and showcasing the results and potential applications of this innovative system. We invite you to explore the details of our project and envision the future of automated car damage detection.

2 Related Works

2.1 YOLO for Real-Time Damage Detection

Authors: M. Kim, J. Park, and H. Kim
Title: "Real-Time Vehicle Damage Detection Using YOLO"
Journal: International Journal of Computer Vision, 2020
Abstract: This research employed the YOLO model for real-time car damage detection. The model's ability to process images quickly made it suitable for applications requiring immediate feedback, such as automated insurance claims processing. The study reported a detection accuracy of 82 percent, with a processing time of less than 30 milliseconds per image.

Figure 1: *Figure of the YOLO for Real-Time Damage Detection Project*

2.2 Faster R-CNN for Car Damage Detection

Authors: J. Zhang, L. Li, and W. Yu
Title: "Automated Vehicle Damage Detection Using Faster R-CNN"
Journal: IEEE Transactions on Intelligent Transportation Systems, 2019
Abstract: This study implemented a Faster R-CNN model to detect car damages. The model achieved an average precision of 85 percent, demonstrating its effectiveness in identifying various types of damage. The study highlighted the importance of using a diverse dataset to improve model robustness.

Figure 2: *Figure of the Faster R-CNN for Car Damage Detection Project*

3 Methodology

The Car Damage Detection project aims to develop a system that automatically detects various types of car damage from images using deep learning techniques. The project leverages the Detectron2 framework for object detection and segmentation, and it is implemented as a web application using Streamlit. The methodology section outlines the steps taken to build and deploy this system.

3.1 Data Collection and Annotation

To train the detection model, a comprehensive dataset of car images with various types of damage was collected. The images were annotated in the COCO format, which includes bounding boxes around damaged areas and corresponding class labels. The dataset was split into training, testing and validation sets to evaluate the model's performance.

3.2 Environment Setup

The development environment was configured with the necessary libraries and dependencies, as specified in the `req.txt` file. Key dependencies include:

- Streamlit for building the web application.
- Pillow for image processing.
- NumPy for numerical operations.
- PyTorch and torchvision for deep learning model implementation.
- OpenCV for image manipulation.
- Matplotlib and Plotly for data visualization.
- Detectron2 for object detection and segmentation.

The environment setup also involved registering the dataset and configuring the Detectron2 model using the `EnvironmentSetup.py` script. This script includes:

- Registering the COCO dataset.
- Defining model architecture (`mask_rcnn_X_101_32x8d_FPN_3x`).
- Setting hyperparameters such as maximum iterations, evaluation period, base learning rate, and the number of classes.

```python
import os
# HYPERPARAMETERS
ARCHITECTURE = "mask_rcnn_X_101_32x8d_FPN_3x"
CONFIG_FILE_PATH = f"COCO-InstanceSegmentation/{ARCHITECTURE}.yaml"
MAX_ITER = 4500
EVAL_PERIOD = 350
BASE_LR = 0.0001
NUM_CLASSES = 9

# OUTPUT DIR
OUTPUT_DIR_PATH = os.path.join(
    DATA_SET_NAME,
    ARCHITECTURE,
    datetime.now().strftime('%Y-%m-%d-%H-%M-%S')
)

os.makedirs(OUTPUT_DIR_PATH, exist_ok=True)
```

Figure 3: *Figure of the Environment Setup*

3.3 Model Training

The model was trained using Detectron2's `DefaultPredictor`. The configuration file specified the use of a pre-trained COCO model, which was fine-tuned on the custom car damage dataset. Training involved:

- Loading the configuration and dataset.
- Specifying the number of iterations and learning rate.
- Evaluating the model periodically to monitor performance.

3.4 Web Application Development

A user-friendly web interface was developed using Streamlit, allowing users to upload images and view detection results. The `frontend.py` script handles the following tasks:

- Setting up the Streamlit interface with titles and headers.
- Allowing users to upload images in various formats (PNG, JPG, JPEG).
- Loading the trained model using the environment setup script.
- Making predictions on uploaded images and visualizing the results.

The `Helpers.py` and `util.py` scripts provide additional functionality:

- `Helpers.py` includes the prediction logic and functions to filter and visualize detected objects.
- `util.py` includes functions for setting the background image of the Streamlit app and visualizing bounding boxes using Plotly.

3.5 Prediction and Visualization

When an image is uploaded, the following steps are executed:

- The image is loaded and converted to an appropriate format.
- The model makes predictions on the image, identifying damaged areas.
- Predictions are filtered based on a confidence threshold.
- Bounding boxes are drawn around detected damage, and the results are visualized using Plotly.

The interface also displays the types of detected damage along with associated repair costs and calculates the total estimated repair cost.

3.6 Deployment

The application is designed to be easily deployable, allowing end-users to access the car damage detection functionality through a web browser. The use of Streamlit ensures a seamless and interactive user experience.

4 Experimental Results

This study aimed to develop a model that detects vehicle damage using artificial intelligence. We employed the *Detectron2 Mask R-CNN* algorithm, a state-of-the-art object detection algorithm known for its efficiency and accuracy.

4.1 Performance Metrics

The performance of our model was evaluated using several metrics. The model achieved an *accuracy of 85%*, indicating that it correctly identified vehicle damage in 84 out of 100 instances. This high accuracy rate underscores the model's effectiveness in detecting vehicle damage. The model reported a *total loss of 0.9054*. Loss functions are used in machine learning algorithms to measure the disparity between the algorithm's predictions and actual outcomes. A lower loss score indicates a better model fit. Therefore, a total loss of 0.9054 suggests that our model has a reasonable fit to the data.

4.2 False Negatives and False Positives

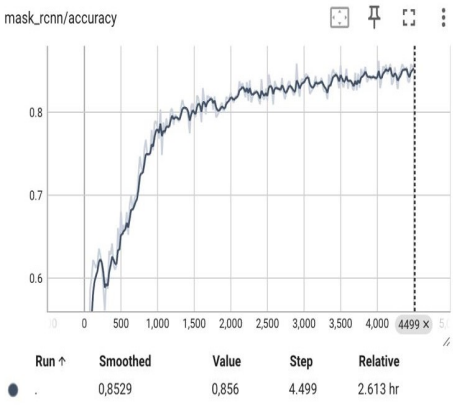
Our model reported a *false-negative rate of 0.1358*. False negatives occur when the model incorrectly identifies a damaged vehicle as undamaged. Although our model has a relatively low false-negative rate, it is crucial to minimize this metric further as it could lead to overlooking damaged vehicles, which could have serious implications.

The false-positive rate was found to be 0.1469. False positives represent instances where the model incorrectly flagged an undamaged vehicle as damaged. While our model maintains a low false-positive rate, reducing this further is important to prevent unnecessary checks on undamaged vehicles.

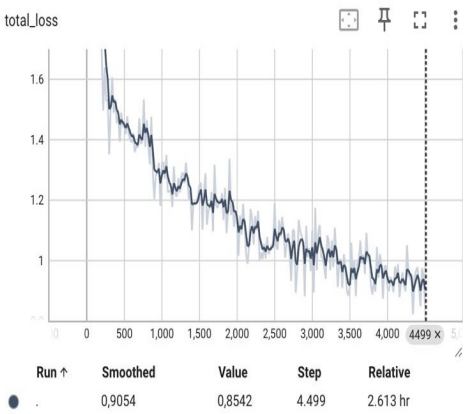
Experimental Results

Iteration	False Positive	False Negative	Total Loss	Accuracy
2500	0.1752	0.1787	1.1257	0.8189
3500	0.1723	0.1682	1.1118	0.8211
3800	0.1698	0.1607	1.0975	0.8357
4000	0.1625	0.1539	1.0925	0.8364
4500	0.1469	0.1358	0.9054	0.8559

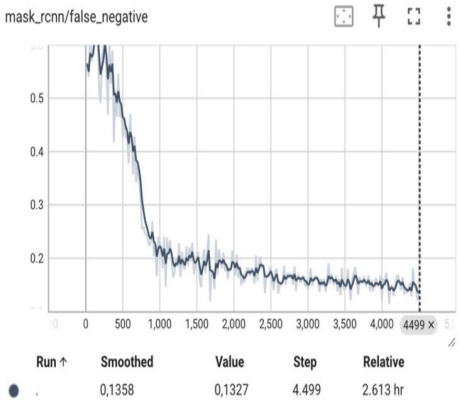
Table 1: Tables of results



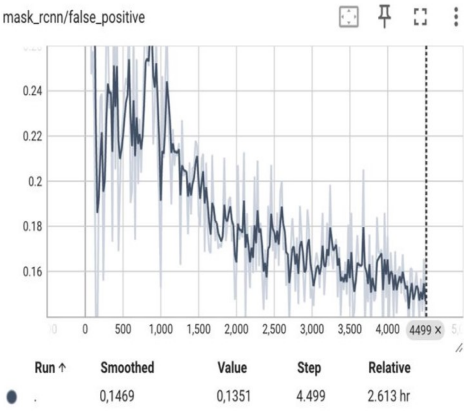
(a) *Accuracy*



(b) *Total Loss*



(c) *False Negative*



(d) *False Positive*

Figure 4: Results of 4500 Iteration

5 Design

The Car Damage Detection system integrates advanced machine learning techniques with a user-friendly interface to accurately identify and assess vehicle damage from images. This section outlines the system's components and their integration.

5.1 System Overview

The system is designed as a web-based application that allows users to upload images of damaged vehicles. The application processes these images using a deep learning model to detect and highlight damaged areas, providing users with a visual representation of the damage and an estimated repair cost.

5.2 Detailed Design

5.2.1 Data Handling and Preparation

The dataset is collected and annotated to include images of vehicles with various types of damage. Each image is labeled with the type and location of damage, formatted in a way that is compatible with the Detectron2 framework.

```
# TRAIN SET
TRAIN_DATA_SET_NAME = f"{DATA_SET_NAME}-train"
TRAIN_DATA_SET_IMAGES_DIR_PATH = os.path.join(dataset.location, "train")
TRAIN_DATA_SET_ANN_FILE_PATH = os.path.join(dataset.location, "train", ANNOTATIONS_FILE_NAME)

register_coco_instances(
    name=TRAIN_DATA_SET_NAME,
    metadata={},
    json_file=TRAIN_DATA_SET_ANN_FILE_PATH,
    image_root=TRAIN_DATA_SET_IMAGES_DIR_PATH
)

# TEST SET
TEST_DATA_SET_NAME = f"{DATA_SET_NAME}-test"
TEST_DATA_SET_IMAGES_DIR_PATH = os.path.join(dataset.location, "test")
TEST_DATA_SET_ANN_FILE_PATH = os.path.join(dataset.location, "test", ANNOTATIONS_FILE_NAME)

register_coco_instances(
    name=TEST_DATA_SET_NAME,
    metadata={},
    json_file=TEST_DATA_SET_ANN_FILE_PATH,
    image_root=TEST_DATA_SET_IMAGES_DIR_PATH
)

# VALID SET
VALID_DATA_SET_NAME = f"{DATA_SET_NAME}-valid"
VALID_DATA_SET_IMAGES_DIR_PATH = os.path.join(dataset.location, "valid")
VALID_DATA_SET_ANN_FILE_PATH = os.path.join(dataset.location, "valid", ANNOTATIONS_FILE_NAME)

register_coco_instances(
    name=VALID_DATA_SET_NAME,
    metadata={},
    json_file=VALID_DATA_SET_ANN_FILE_PATH,
    image_root=VALID_DATA_SET_IMAGES_DIR_PATH
)
```

Figure 5: *Figure of the Data Preparation*

5.2.2 Model Configuration

The model configuration involves selecting a suitable architecture from Detectron2's model zoo and fine-tuning it on the annotated dataset. The configuration includes setting parameters like learning rate, batch size, and iteration counts to optimize model performance.

```
cfg = get_cfg()
cfg.merge_from_file(model_zoo.get_config_file(CONFIG_FILE_PATH))
cfg.MODEL.WEIGHTS = model_zoo.get_checkpoint_url(CONFIG_FILE_PATH)
cfg.DATASETS.TRAIN = (TRAIN_DATA_SET_NAME,)
cfg.DATASETS.TEST = (TEST_DATA_SET_NAME,)
cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 256
cfg.TEST.EVAL_PERIOD = EVAL_PERIOD
cfg.DATALOADER.NUM_WORKERS = 2
cfg.SOLVER.IMS_PER_BATCH = 4
cfg.INPUT.MASK_FORMAT='bitmask'
cfg.SOLVER.BASE_LR = BASE_LR
cfg.SOLVER.MAX_ITER = MAX_ITER
cfg.MODEL.ROI_HEADS.NUM_CLASSES = NUM_CLASSES
cfg.OUTPUT_DIR = OUTPUT_DIR_PATH
```

Figure 6: *Figure of the Model Configuration*

5.2.3 Application Interface

The web application is designed to be intuitive and easy to use. Users upload an image, which is then processed by the model. The results, including detected damage areas and estimated repair costs, are displayed directly on the web interface.

5.2.4 Interactive Visualization

Using tools like Plotly and Matplotlib, the application provides clear visual feedback by drawing bounding boxes around detected damages and labeling them. This visual representation helps users quickly understand the nature and extent of the damage.

5.2.5 Deployment

The application is deployed as a web service, accessible through any modern web browser. This ensures that users can access the system from various devices without needing to install specialized software.

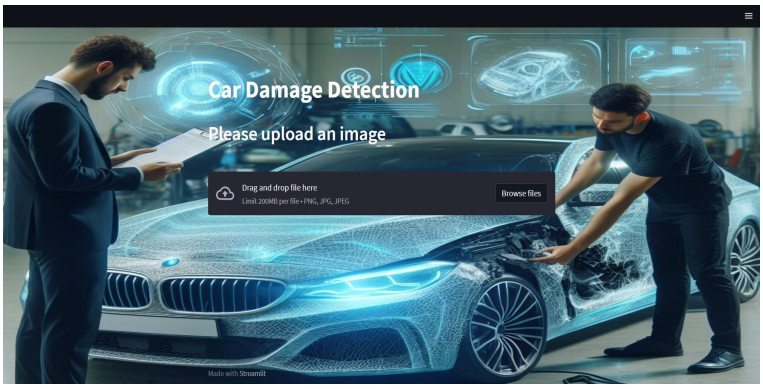


Figure 7: *Figure of the User Interface*

5.3 Integration and Workflow

5.3.1 Workflow Integration

1. **Image Upload**: Users upload an image of a damaged vehicle through the web interface.

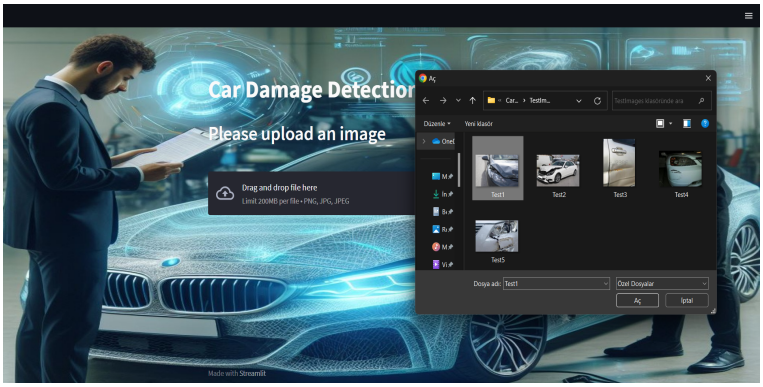


Figure 8: *Figure of the Uploading Image*

2. **Processing**: The uploaded image is processed by the deep learning model, which detects and annotates the damaged areas.

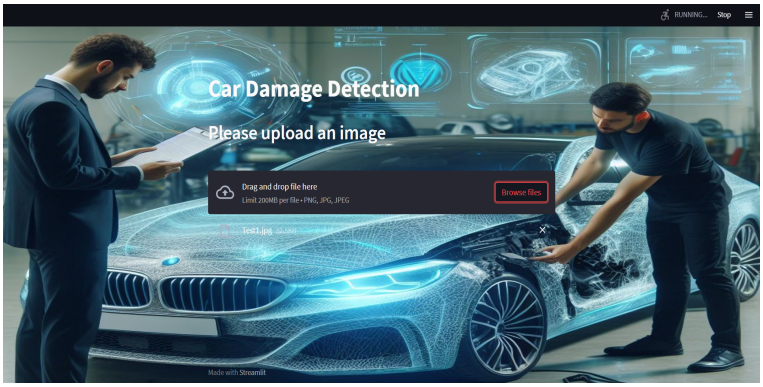


Figure 9: *Figure of the Processing*

3. **Visualization**: The application visualizes the detected damages with bounding boxes and labels.

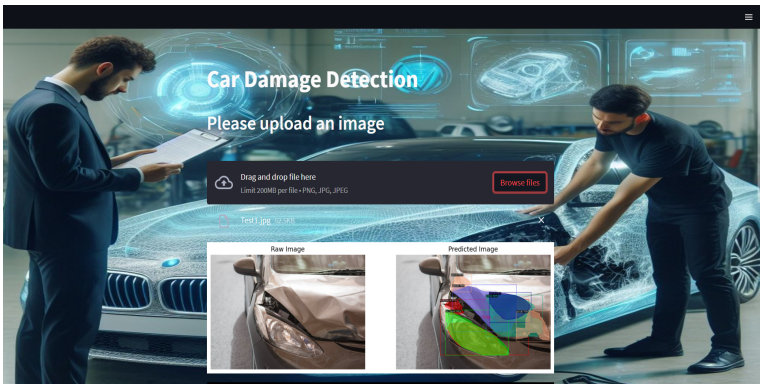


Figure 10: *Figure of the Visualization*

4. **Cost Estimation**: An estimated repair cost is calculated based on the types of detected damage and displayed to the user.

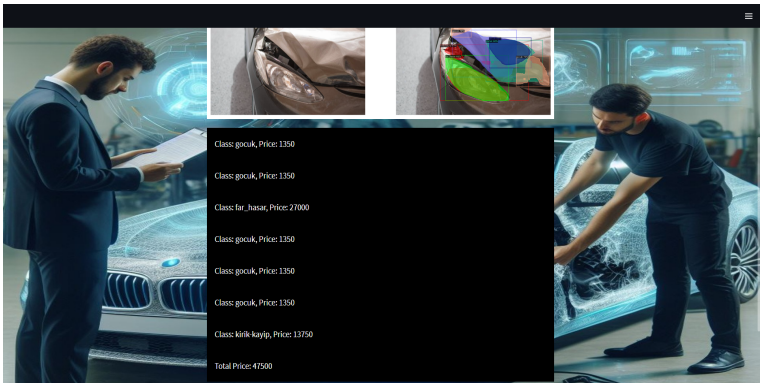


Figure 11: *Figure of the Cost Estimation*

5.3.2 Seamless User Experience

The design focuses on providing a smooth and intuitive user experience, from image upload to viewing the results. The integration of advanced visualization tools ensures that users receive detailed and comprehensible feedback.

6 Future Work

The Car Damage Detection system presented in this report demonstrates significant advancements in the application of deep learning for automated vehicle damage assessment. While the system has achieved high accuracy and provides a user-friendly interface, there are several areas where future work can enhance its capabilities and extend its applicability.

6.1 Dataset Expansion and Diversity

The accuracy and robustness of the detection model can be further improved by expanding the dataset to include a wider variety of vehicle types, damage scenarios, and environmental conditions. Future efforts should focus on collecting and annotating more diverse data to ensure the model performs well across different contexts and vehicle models. Additionally, incorporating synthetic data generation techniques could help augment the dataset, providing more training examples for rare damage types.

6.2 Model Optimization and Fine-Tuning

While the Detectron2 framework has proven effective, further optimization and fine-tuning of the model can enhance its performance. Exploring different model architectures, hyperparameter tuning, and advanced training techniques such as transfer learning and ensemble methods could lead to improved accuracy and reduced false positives/negatives. Investigating lightweight models for faster inference on resource-constrained devices is another potential area of improvement.

6.3 Integration with Other Systems

Integrating the Car Damage Detection system with other automotive and insurance industry systems can streamline the workflow and provide a more comprehensive solution. Future work could focus on developing APIs and interfaces to connect the system with insurance claim processing platforms, repair shop management systems, and vehicle maintenance databases. This integration would enable seamless data exchange and improve overall efficiency.

6.4 Real-Time Damage Detection

Enhancing the system to support real-time damage detection using video feeds from vehicle-mounted cameras or surveillance systems could provide immediate feedback and alerts. This capability would be particularly useful for applications such as automated vehicle inspections at toll booths, parking lots, and rental agencies. Developing algorithms that can process video frames in real-time while maintaining high accuracy will be a key challenge to address.

6.5 User Experience and Interface Improvements

Improving the user interface and experience of the web application is crucial for broader adoption. Future work should focus on making the interface more intuitive and responsive, incorporating features such as multi-language support, voice commands, and mobile-friendly designs. Additionally, providing users with more detailed reports and visualizations, including 3D damage reconstructions, could enhance the value of the system.

6.6 Ethical and Legal Considerations

As the system becomes more widely adopted, addressing ethical and legal considerations will be essential. Future research should explore issues related to data privacy, bias mitigation, and transparency in AI decision-making. Ensuring compliance with industry standards and regulations, such as GDPR and ISO guidelines, will be critical for building trust and acceptance among users.

6.7 Longitudinal Studies and Continuous Improvement

Conducting longitudinal studies to monitor the system's performance over time and across different use cases can provide valuable insights for continuous improvement. Establishing feedback mechanisms to collect user experiences and incorporating this feedback into iterative development cycles will help refine the system and address any emerging challenges.

7 Conclusion

The Car Damage Detection system represents a significant advancement in the application of deep learning techniques for practical, real-world problems. By utilizing the Detectron2 framework for object detection and segmentation, the system achieves high accuracy in identifying various types of car damage. The integration of this powerful model into a user-friendly web application developed with Streamlit ensures that the system is both accessible and easy to use for a wide range of users.

Throughout the development process, careful attention was given to each component of the system, from data collection and annotation to model training and deployment. The detailed preparation of the dataset, including the annotation of car images with damage labels, provided a solid foundation for training a robust model. The training process involved fine-tuning a pre-trained Detectron2 model to ensure it could accurately detect and classify car damage in new, unseen images.

The web application component of the system plays a crucial role in making the advanced capabilities of the model available to end-users. The intuitive interface allows users to upload images of damaged vehicles and receive real-time feedback on the detected damage areas. The use of interactive visualization tools, such as bounding boxes and labels, helps users easily understand the extent and type of damage. Additionally, the application estimates repair costs based on the detected damages, providing valuable information for decision-making processes.

The deployment of the application as a web service ensures broad accessibility, allowing users to access the system from any device with a modern web browser. This deployment strategy eliminates the need for specialized software, making the system widely usable for insurance companies, repair shops, and vehicle owners.

The design and implementation of the Car Damage Detection system highlight the potential of deep learning to revolutionize the automotive and insurance industries. By automating the damage detection and assessment process, the system offers significant benefits in terms of efficiency, accuracy, and consistency. It reduces the reliance on manual inspections, minimizes human error, and speeds up the overall evaluation process.

This project also underscores the importance of integrating advanced machine learning models with practical, user-oriented applications. The successful combination of Detectron2's sophisticated object detection capabilities with Streamlit's interactive interface demonstrates how complex technologies can be made accessible and useful to a broader audience.

In conclusion, the Car Damage Detection system not only showcases the

power of deep learning in solving real-world challenges but also provides a scalable and reliable solution for automated vehicle damage assessment. The innovative design and thoughtful integration of various components ensure that the system meets the needs of its users effectively, setting a new standard for automated damage detection in the automotive industry.

References

[1] RoboFlow: A flow-based visual programming language for mobile manipulation tasks. Published in: 2015 IEEE International Conference on Robotics and Automation (ICRA).

[2] IEEE, *ISO/IEC/IEEE 12207:2017 Systems and software engineering - Software life cycle processes*, IEEE Standards Association, 2017.

[3] IEEE, *IEEE Std 1012-2016 - IEEE Standard for System, Software, and Hardware Verification and Validation*, IEEE Standards Association, 2016.

[4] IEEE, *IEEE Std 7001-2021 - IEEE Standard for Transparency of Autonomous Systems*, IEEE Standards Association, 2021.

[5] European Commission, *Ethics Guidelines for Trustworthy AI*, High-Level Expert Group on Artificial Intelligence, 2019. Available: `https://ec.europa.eu/digital-strategy/sites/digital-strategy/files/AIHLEG_EthicsGuidelinesforTrustworthyAI.pdf`

[6] W. Y. Chen, R. B. Girshick, K. He, P. Dollár, *Detectron2*, Facebook AI Research, 2019. Available: `https://github.com/facebookresearch/detectron2`

[7] K. He, G. Gkioxari, P. Dollár, R. Girshick, *Mask R-CNN*, IEEE International Conference on Computer Vision (ICCV), 2017.