OPENCAMPUS.sh

Binary Car Damage Classification

Max Borm, Nicolas Liebau

OPENCAMPUS.sh

PROJECT IDEA

Objective

The aim of the project is to develop a machine learning model that can accurately classify images of cars into two categories: undamaged cars and heavily damaged cars, as well as undamaged cars and cars with minor damage (such as scratches).

Severe Damage

- Try to detect severe damage to the car
- Classification to whole and damaged

Minor Damage

- Try to detect minor damage to the car
- Classification to whole and scratched

OPENCAMPUS.sh

BUSINESS CASE

Motivation

Relevance to the real world:
- Processing insurance claims
- Pre-purchase vehicle inspections
- Automated damage assessment for car rental or leasing services

Potential Benefits

- Improving the efficiency and accuracy of damage assessment processes
- Automation of damage assessment processes
- Reducing manual inspection costs
- Help with buying a used car
- Enhancing customer experience in insurance or car rental businesses

Market Need

- Manual identification and classification of vehicle damage can be costly and prone to human error, resulting in financial losses and customer dissatisfaction for businesses
- As the insurance industry experiences an increase in competition, image-based processing of vehicle claims is gaining importance, particularly for smaller but more frequent claims
- The growing automotive industry is also contributing to the expanding car insurance market
- Currently, adjusters physically inspect vehicles to assess damage and provide estimates, which introduces the possibility of inaccurate settlements
- However, automating this process through machine learning and remote usage can offer convenience for both claimants and insurance companies, improving productivity and customer satisfaction

OPENCAMPUS.sh

DATA COLLECTION

Image Scraping

1

- Use of an existing Google Image Scraping Application to download images automatically
- https://github.com/talleyhoe/google-image-scraper
- Cloned the repo, manually install the dependencies, and run main with python
- The images were then checked manually to find suitable images
- The result was 622 images for no damage, 455 images for severe damage and 338 for minor damage
- Expanded the data set for severe damages and no damages with images from another project https://cainvas.ai-tech.systems/use-cases/car-damage-detection-app/

Undersampling

2

… to equalize the class distribution of a classification dataset that has a skewed class distribution

Severe Damage:
Train (70%): 2246 files
Validation (20%): 644 files
Test (10%): 320 files

3

Minor Damage:
Train (70%): 444 files
Validation (20%): 130 files
Test (10%): 62 files

OPENCAMPUS.sh

CAR DAMAGE CLASSIFICATION - SEVERE DAMAGE

Deeper Model

Model Overview

Overview:

- Type: Convolutional Neural Network (CNN)

Customization & Architecture:

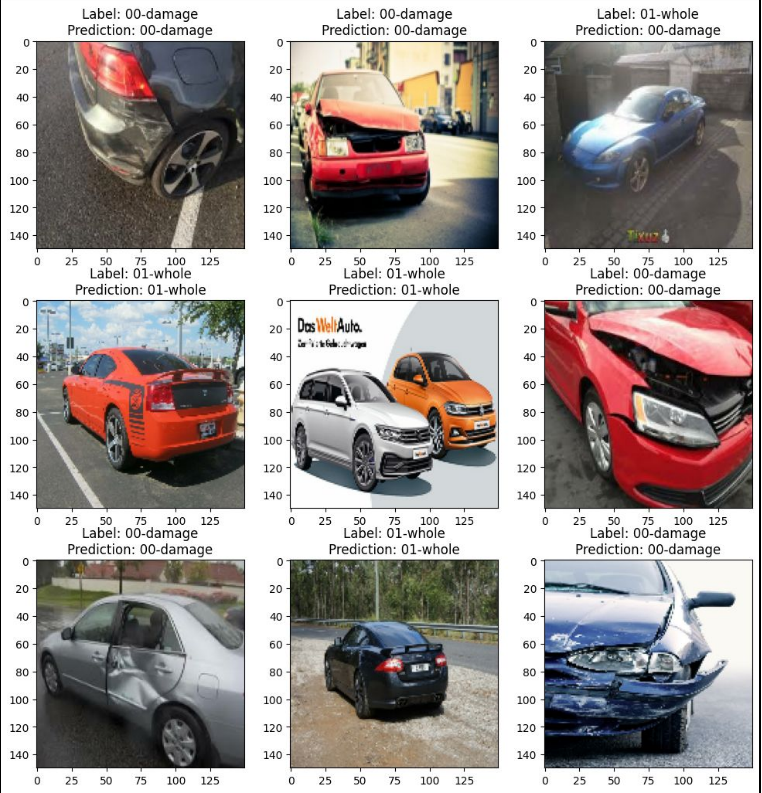
- Input: 150x150 pixel images

Layers:

- Conv2D layers (4x)
- MaxPooling2D
- Batch Normalization
- Dropout layers (0.3 and 0.5 rates) to prevent overfitting
- L2 regularization
- Output layer: 2 neurons (softmax activation) for binary classification

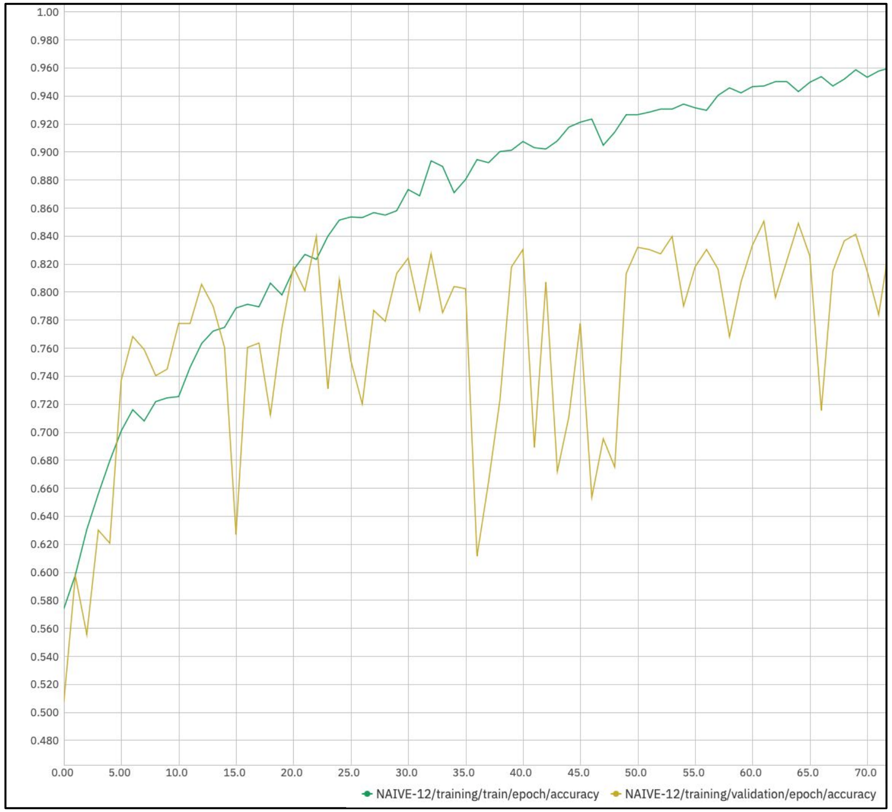
Training & Evaluation:

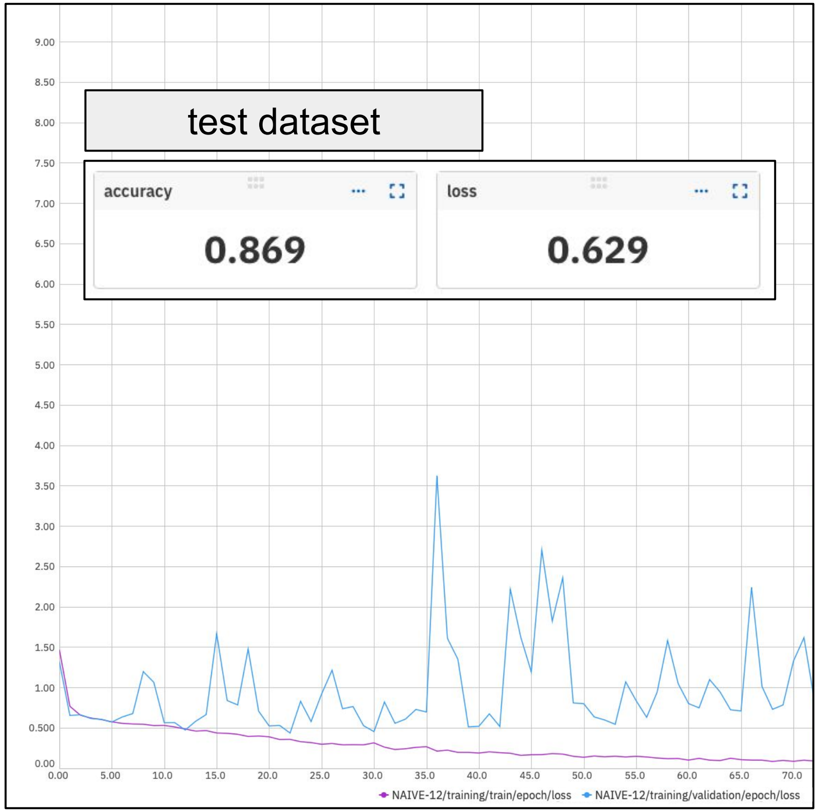
- Epochs: ≈100
- Callbacks: Checkpoint saving and early stopping for efficient training
- Manually selecting checkpoint for testing



CAR DAMAGE CLASSIFICATION - SEVERE DAMAGE

Deeper Model





test dataset

accuracy	loss
0.869	0.629

CAR DAMAGE CLASSIFICATION - SEVERE DAMAGE

Transfer Model

Model Overview

Overview:

- Type: Convolutional Neural Network (CNN) using TensorFlow & Keras
- Base: VGG16 (pre-trained on ImageNet)

Customization & Architecture

- Input: 224x224 pixel images

Layers:

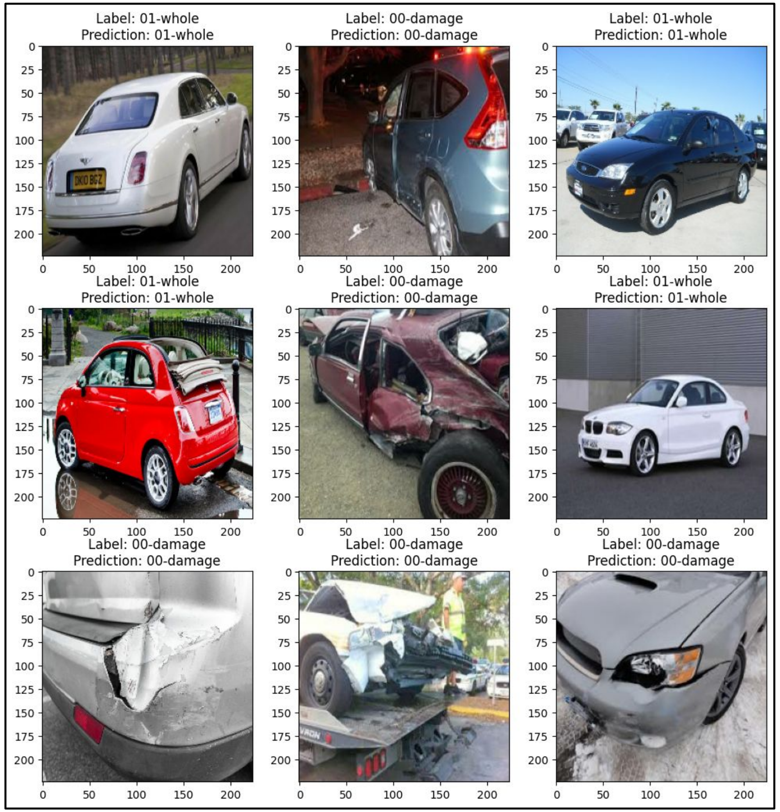
- Frozen VGG16 base for feature extraction
- Flatten layer
- Two Dense layers (128 and 64 neurons) with ReLU activation
- Dropout layers (0.5 rate) after each Dense layer
- Output layer: 2 neurons (softmax activation)

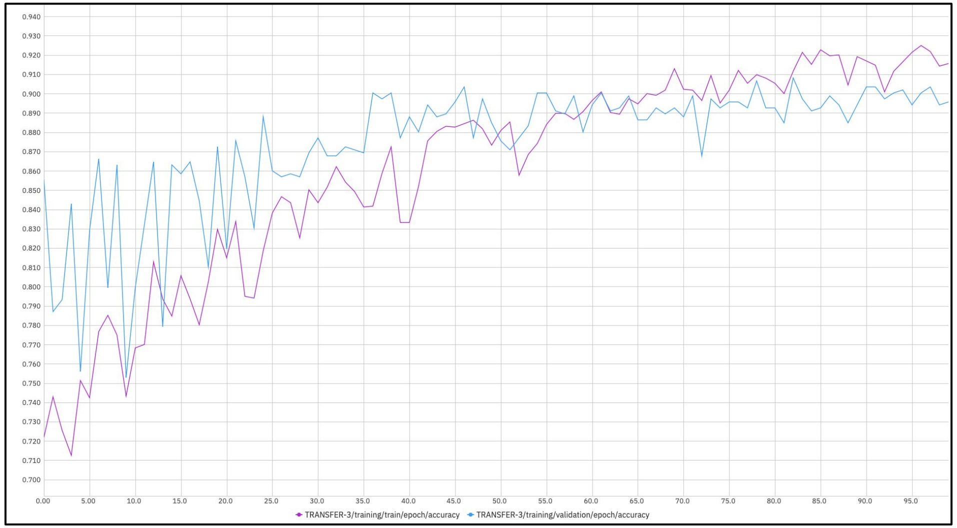
Training & Evaluation

- Optimizer: Adam
- Loss Function: Categorical Crossentropy
- Metrics: Accuracy
- Epochs: 100
- Evaluation: Tested on separate dataset

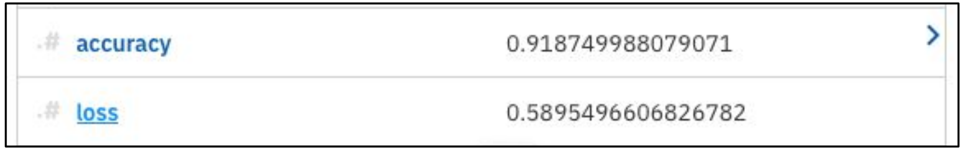
CAR DAMAGE CLASSIFICATION - SEVERE DAMAGE

Transfer Model





test dataset



accuracy	0.918749988079071
loss	0.5895496606826782

OPENCAMPUS.sh

CAR DAMAGE CLASSIFICATION - MINOR DAMAGE

Transfer Model

Model Overview

Overview:

- Type: Convolutional Neural Network (CNN) using TensorFlow & Keras
- Base Model: Pre-trained VGG16 (weights from ImageNet)

Customization & Architecture:

- Input Size: 224x224 pixels

Layers:

- VGG16 base (frozen layers for feature extraction)
- Flatten layer
- Dense layers (128 and 64 neurons, ReLU activation)
- Dropout layers (0.5 rate) for regularization
- Output layer: 2 neurons (softmax activation) for binary classification

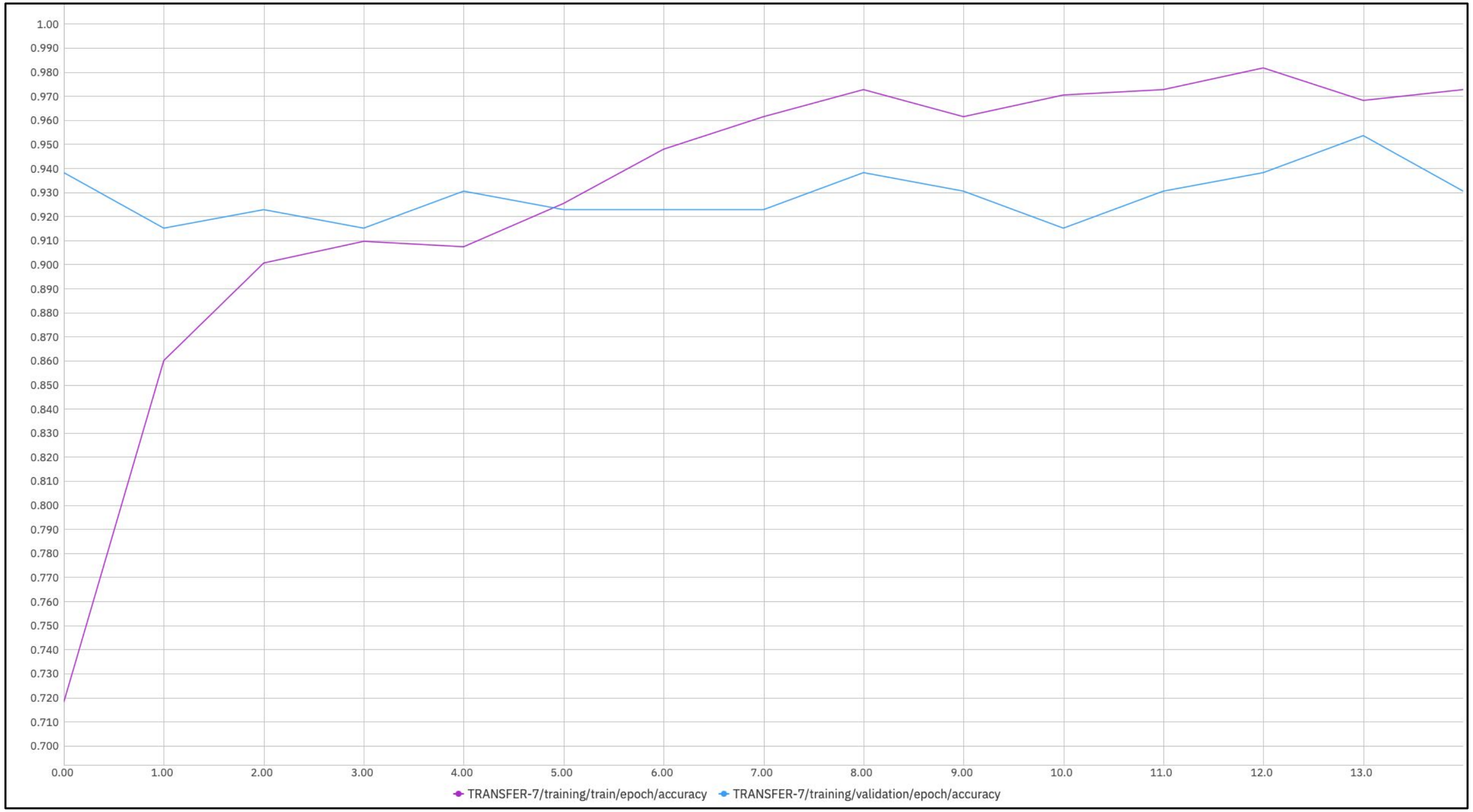
Training and Evaluation:

- Loss Function: Categorical Crossentropy
- Optimizer: Adam
- Metrics: Accuracy
- Training & Evaluation
- Epochs: 15
- Model evaluated on a separate testing set

OPENCAMPUS.sh

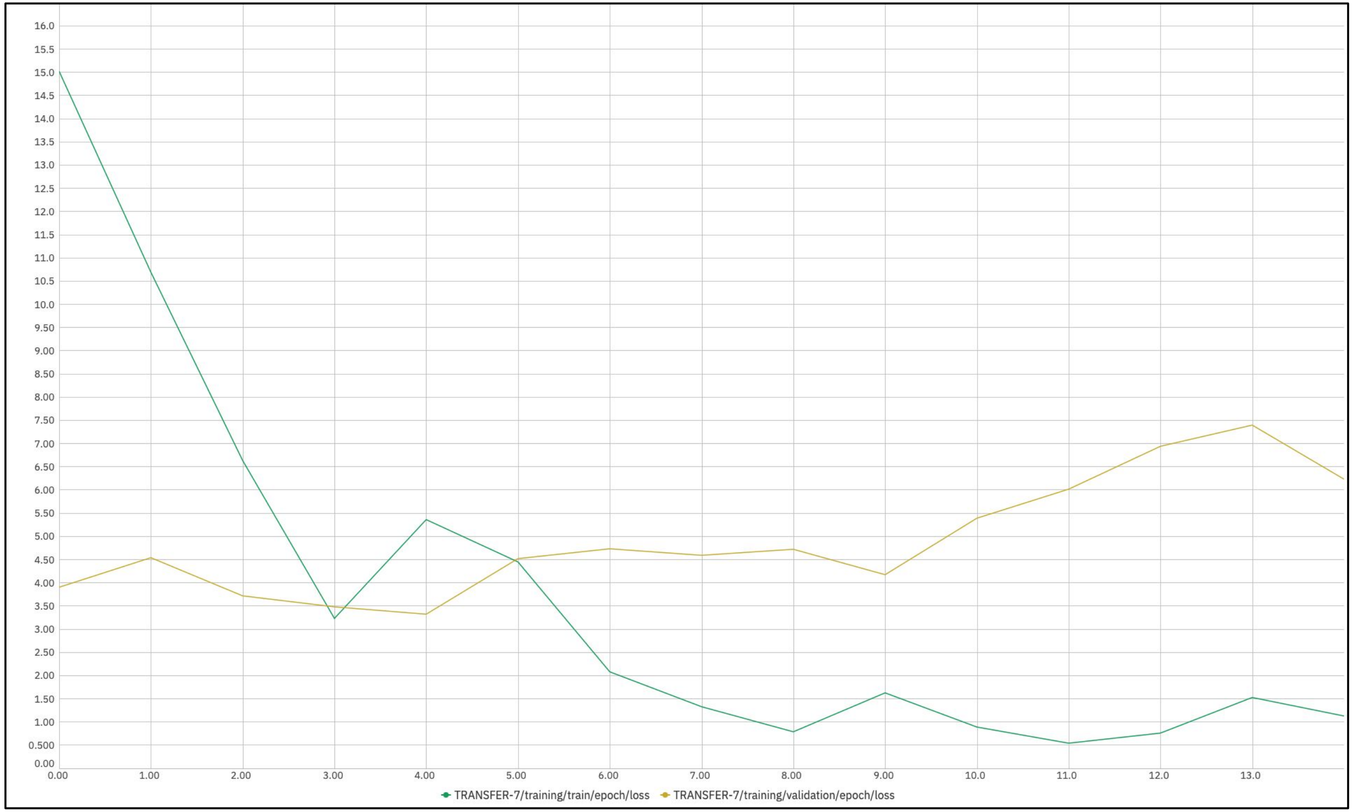
CAR DAMAGE CLASSIFICATION - MINOR DAMAGE

Transfer Model



CAR DAMAGE CLASSIFICATION - MINOR DAMAGE

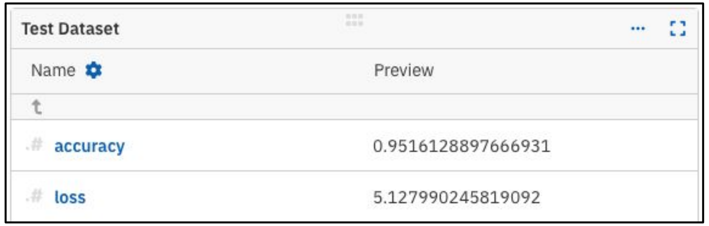
Transfer Model



OPENCAMPUS.sh

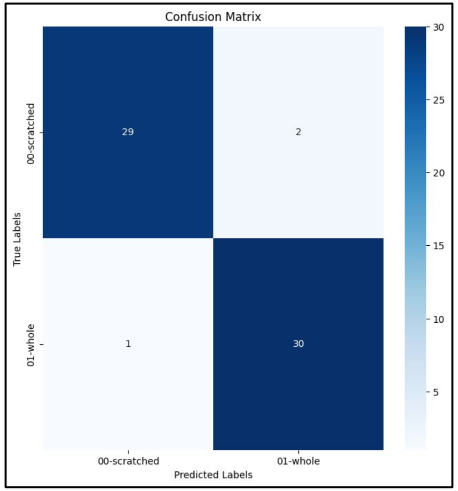
CAR DAMAGE CLASSIFICATION - MINOR DAMAGE

Transfer Model



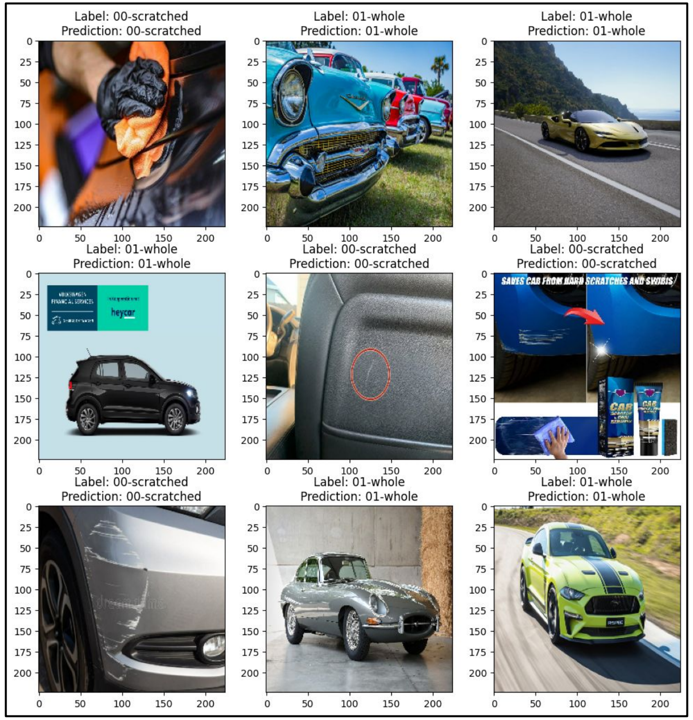
Test Dataset

Name	Preview
accuracy	0.9516128897666931
loss	5.127990245819092



Confusion Matrix

$$\begin{array}{c|cc} & \text{00-scratched} & \text{01-whole} \\ \hline \text{00-scratched} & 29 & 2 \\ \text{01-whole} & 1 & 30 \end{array}$$



OPENCAMPUS.sh

CAR DAMAGE CLASSIFICATION - BOUNDING BOXES

The Idea

WHERE is my classified object/feature
Just trying a naive approach using transfer learning and regression
Tedious creation of dataset due to manual labeling
Found Dataset on Kaggle without any description - What could go wrong? 🥴

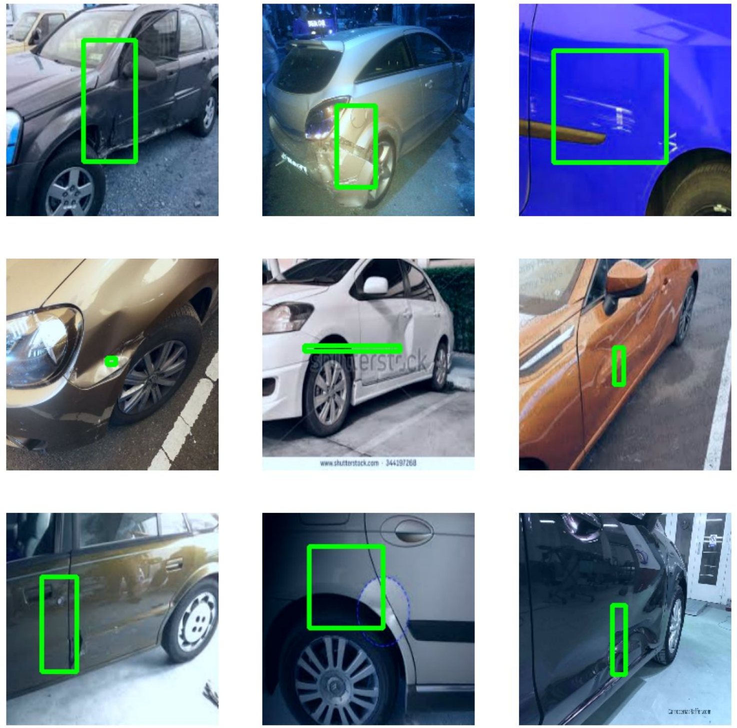
The Model

Single class and 4 output neurons for x_start, y_start, x_end, y_end

Layers:

- VGG16 as base-model
- Dense(512)
- Dropout(0.5)
- Dense(128)
- Dense(2*2)

The Dataset



OPENCAMPUS.sh

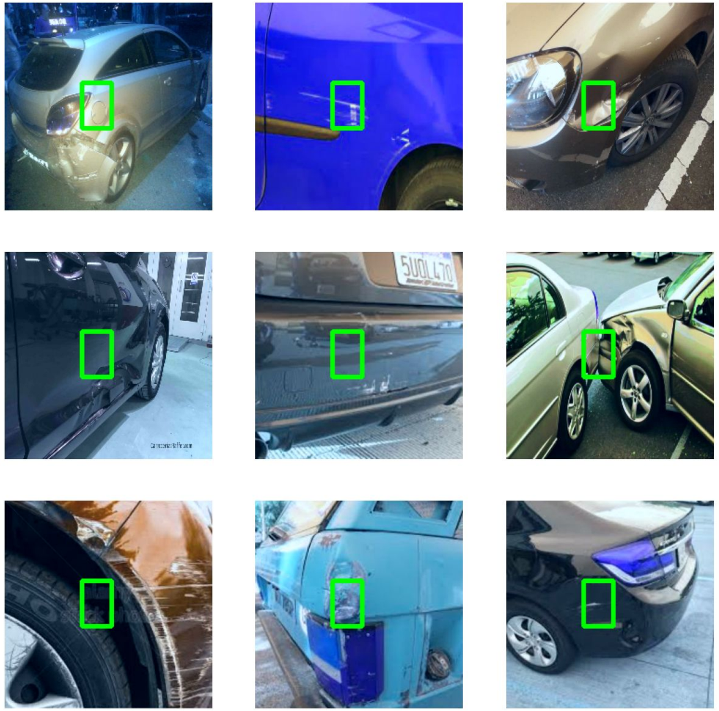
CAR DAMAGE CLASSIFICATION - BOUNDING BOXES

What went wrong?

Model is just averaging start & end positions
Training dataset has low correlation with actual damages
Due to the format of the data, I suspect it's YOLO predictions

- There is a lot to learn 😅

The Predictions



OPENCAMPUS.sh

CONCLUSION AND FUTURE WORK

Conclusion

Severe Damage:

- With a train set of 2246 files, a validation set of 644 files and a test set of 320 files, we obtained
  - a test accuracy of 87% and a loss of 0.629 with the deeper model
  - a test accuracy of 92% and a loss of 0.590 with the transfer model

Minor Damage:

- With a train set of 444 files, a validation set of 130 files and a test set of 62 files, we obtained
  - a test accuracy of 95% and a loss of 5.128 with the transfer model

Future Work

Damage detection:

- Train the model with a diverse data set containing different types of damage
- Label them accurately using polygons or brushes

Damage assessment:

- Create a classification for each labelled damage type, e.g. scratches, chipped parts, chipped paint and broken glass.
- In addition, markings can be added to indicate the severity of the damage, e.g. low, medium or high

Recognition of car parts:

- Creating a car part recognition is another important step in damage detection e.g. boot, tyres or front door
- Identification of the specific part that is damaged and allows to understand the proportion of the damaged area compared to the total area of the vehicle part