IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 1 June 2023, accepted 21 July 2023, date of publication 26 July 2023, date of current version 2 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3299223

APPLIED RESEARCH

Vehicle Damage Severity Estimation for Insurance Operations Using In-The-Wild Mobile Images

DIMITRIOS MALLIOS[1], LI XIAOFEI[1], NIALL MCLAUGHLIN[2], JESUS MARTINEZ DEL RINCON[2], CLARE GALBRAITH[1], AND RORY GARLAND[1]

[1]Liberty IT, Adelaide Exchange, BT2 8GD Belfast, U.K.
[2]Centre for Secure Information Technologies (CSIT), Queen's University Belfast, BT3 9DT Belfast, U.K.

Corresponding author: Niall McLaughlin (n.mclaughlin@qub.ac.uk)

This work was supported in part by the U.K. Research and Innovation (UKRI) via Knowledge Transfer Partnership (KTP) under Grant 1026544, and in part by the Liberty Information Technology (IT) Collaboration funded by Liberty Mutual Insurance. Data and code supporting the results reported in this paper have been withheld for commercial reasons.

ABSTRACT Following a car accident, an insurance company must assess the level of damage to each vehicle to decide on the compensation paid to the insurance customer. This assessment is usually performed by manual inspection, which is costly and time-consuming. Automatic car damage assessment using image data is an under-addressed problem highly relevant to the insurance industry. Although there have been many attempts at solving particular aspects of this problem, we are unaware of any complete solutions available. In this work, we propose a pipeline that uses photographs of a damaged car, collected by the users from multiple angles, together with structured data about the vehicle, to estimate damage severity following an accident. Our proposed pipeline consists of several computer-vision models for the detection of damage and the determination of its extent. Unlike existing approaches in car damage assessment, we use semantic segmentation to understand which parts of the car are damaged, and to what extent. We then extract computer-vision features, indicating the location and severity of damage to each exterior panel, together with structured data, to arrive at an accurate damage cost estimation. We train and evaluate this model on a large dataset of historical insurance claims with known outcomes, all captured in the wild with mobile-phone hardware.

INDEX TERMS Claim operations, insurance, vehicle damage estimation, computer-vision, deep-learning.

I. INTRODUCTION

The field of machine learning and the deep-learning (DL) area [1] have gone through tremendous progress over the last years. Everything started with the acquisition of massive datasets and the application of pre-existing algorithms [2] on big data led to a revolution of the field, finally reaching human-level performance on computer-vision (CV) tasks [3], [4].

One of the industries that could benefit from the deep-learning revolution is the insurance sector. Every day, thousands of claimants send media content to their insurance companies in order to have the company assess their damaged vehicles online. The process of assessing damage severity is handled by experienced adjusters and can be time-consuming. By leveraging machine learning models to predict the severity of the damage, it may be possible to expedite the claims handling process, reducing the expense of rental and storage, and improving the customer experience. Using machine learning in this process is an ideal way to facilitate the claimant process.

In this paper, we tackle vehicle damage severity estimation using in-the-wild mobile phone imagery, computer vision and deep-learning techniques. We show that image data and deep learning can model the state of a damaged car. We show that our methods surpass the existing approaches that use only structural features that are derived from information supplied by the customer after the accident.

A. OVERVIEW

Our contribution is an end-to-end pipeline that uses deep learning to extract damage estimations from in-the-wild

The associate editor coordinating the review of this manuscript and approving it for publication was Charalambos Poullis.

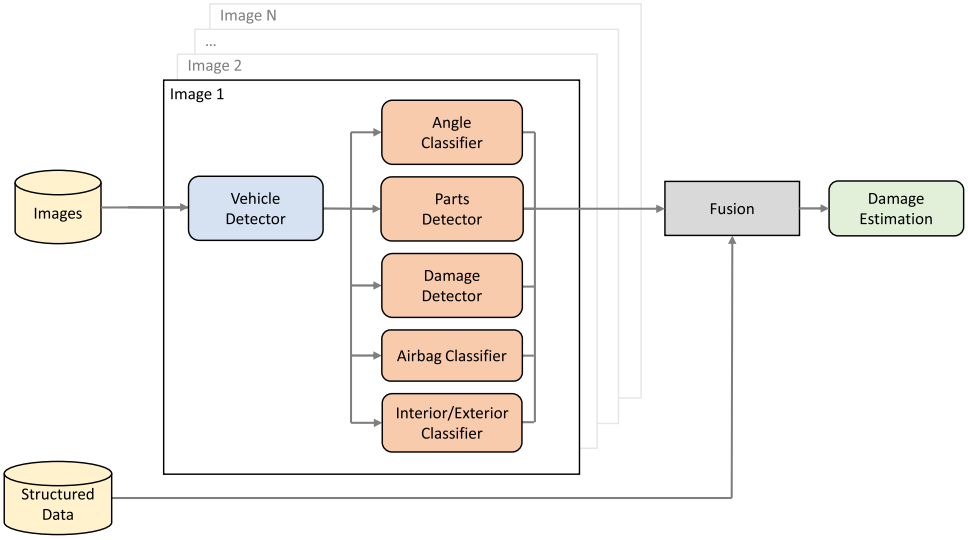


FIGURE 1. An overview of the Damage Estimation Pipeline. Computer-vision features are extracted from every image using DNNs. The computer-vision features from all images are then fused with additional features from structured data e.g., the age of the car, to produce an overall feature vector. The overall feature vector is used by a final XGBoost model for severity prediction.

mobile imagery data. The damage information is then combined with structured information supplied by the customer to produce the final cost estimate for insurance purposes. Existing approaches that try to directly estimate damage costs from images are not scalable. They need to be continually re-trained and adapted as new models of cars are released. By separating damage estimation from cost estimation our model is more robust. The computer-vision pipeline for damage estimation can remain stable. In contrast, the cost estimation (fusion) model, which is computationally cheaper to train, can be updated to incorporate new structured information such as changing car models or relevant business considerations.

Our proposed system works as follows. After a car is detected in the image, we use semantic segmentation to build a detailed picture of the individual car parts. We then detect and location and severity of the damage to each car panel. The pipeline handles multiple images for each claim, extracting and combining damage features from all images. The damage information from all images is then combined with structured features, including information about the claim supplied by the customer, to pass to the fusion module. The damage features from all images are then combined with the structured data in the fusion module to produce an accurate damage cost estimate for the vehicle. The contributions of this paper are:

- We propose the first comprehensive multi-image car damage estimation system for insurance companies.
- A feature fusion mechanism for combining multiple computer-vision-based damage estimates, from multiple unconstrained images, to arrive at an accurate damage cost estimate.
- We show how to decompose the problem of estimating the cost of car damage in a way that allows it to be applied to real-world datasets. This involves separating the methods that estimate car damage from images from the system for producing the cost estimate.
- Our system is the first deep-learning-based damage estimation model able to provide an estimate of the severity of damage to each body panel from unconstrained in-the-wild imagery.
- We show that our computer-vision features improve the performance of damage assessment models, compared to using structured information alone.

II. RELATED WORK

The problem of car damage assessment has received increasing attention in the literature. Many of the existing approaches use a similar pipeline consisting of car detection, damage detection and damage classification [5], [6], [7]. There is a divide in the literature about whether the damage detection problem should be treated as an object detection [6] or a segmentation problem [8], with more recent methods generally favouring segmentation. A common concern, mentioned by several authors, is the difficulty of consistently and accurately classifying damage type and severity when creating ground-truth labels [6], [9]. In reality, there are no hard boundaries between different damage classes or types of damage. Failure to consider this issue can lead to inconsistent labels in the training data, which makes training a model to classify damage more challenging. Another common theme in the literature is the need to cope with small datasets as no large publicly available dataset exists for this problem [6], [10]. Since there is no standard dataset for car damage assessment, it is hard to directly and fairly compare the different approaches in the literature.

One of the earliest proposed methods for car damage classification [10] studied fine-tuning from a network trained on ImageNet. Fine-tuning is compared with pre-training using a convolutional auto-encoder and using an ensemble. Both fine-tuning and pre-training are intended to help deal with the small dataset sizes. This paper classifies damage into several classes including dents, shatter, broken, scratch, smash, and no damage. The damage class is predicted for the whole image using the Convolutional Neural Network (CNN) classifier and localisation can be performed by cropping and classifying small image patches. Alternately damage detection can be treated as an object detection problem [7]. Damage is located using a YOLO-based object detector. This work is concerned with anti-fraud so re-identification features are extracted from damaged areas to allow for damage patch retrieval from an image database.

More recent approaches tend to use a pipeline consisting of car detection, followed by damage assessment, which is similar to our proposed method. Pipeline-based approaches become relevant when there is a need to automatically perform the damage assessment task in user-submitted images. When following this approach a detection network is used to locate and crop the car from the broader image. Another network then identifies the class, location, severity and size of the damaged area [6]. Data augmentation can be used to improve classifier performance despite the low number of training images, or alternatively, transfer learning from a network pre-trained on COCO can be used [8].

Pipeline-based methods differ in how they perform the damage assessment. Some methods require specialised camera setups [9]. Bandi et al. [5] classify the severity and location of the damage. However, location is limited to front, side or rear and damage is not explicitly localised, instead, the network is trained to predict an overall location label, front, side or rear, for each image. Several recent pipeline-based approaches treat damage detection as a segmentation problem [8], [11], [12]. The task is to predict a segmentation mask covering the damaged areas, which may better model these irregular regions than the bounding boxes produced by object detection-based approaches. The Mask R-CNN network [13] can be used to perform damage detection and predict segmentation masks [8], [11]. Mask R-CNN is specifically useful, as it is an instance-segmentation model, allowing different areas of damage to be identified independently. However, one recent work has questioned whether Mask R-CNN is the optimal approach. Instead proposing a specialised two-stream network for this task [14].

The above-mentioned approaches often focus specifically on assessing car damage. Our work, in contrast, proposes a complete end-to-end pipeline for automating the assessment of damages that should be paid to insurance claimants. There exist several related works in the literature. The approach of Imaam et al. [15] is limited to six car models and is focused on predicting customer churn. fernando et al. [16] add natural language processing to extract relevant information from the claimant. They propose a pipeline to automate the whole insurance claims process, although the cost estimate is not performed as part of the pipeline. Poon et al. propose an end-to-end pipeline where mask R-CNN is used to identify damaged areas. Metadata and image features are then combined to produce a final appraisal. However, the damage description is not extracted in a fine-grained parts-based manner.

In this work, we develop a novel pipeline for car damage assessment as follows.

- Our method uses unconstrained in-the-wild mobile images, captured by the customer, of the car from multiple angles. For each image, our system detects the vehicle and focuses on this region of interest only.
- Unlike previous methods in the literature, we semantically analyse the car to segment all car parts and all damaged areas. We treat parts and damage detection as instance segmentation problems using the Mask R-CNN network. We measure the relative area of damage affecting each car part. We also detect additional features such as airbag deployment.
- We fuse the information from all the images to accurately estimate the damage to each car part. The estimated damage to each part is used, together with structured information about the vehicle, to arrive at a final estimated damage severity. This additional structured information allows us to produce an accurate cost estimate for the damage useful for real-world insurance purposes.
- Our pipeline separates the process of using images to estimate the level of damage to each part, from the process of using this information to arrive at a final cost estimate. Decomposing the problem in this way allows the damage estimation model to remain fixed, as the cost estimation model is updated to cope with ongoing changes such as new car models and changing business and economic factors.

In the following section, we will provide details of the various models that make up the damage estimation pipeline and show how they have been combined to perform the damage estimation task. We also provide details of the data annotation method used to create our dataset.

III. METHODS

In this section, we outline our approach to creating a system for vehicle damage severity estimation. We first outline how we created and annotated the necessary datasets. We then outline how we created computer-vision models for each sub-task.

We note that vehicle damage severity estimation has not yet been widely studied. Therefore there are no publicly available datasets encompassing the complete end-to-end task. We, therefore, used a combination of publicly available and private data to create bespoke datasets for each sub-task. Where possible, public datasets were used. This made it easier to build baseline models to facilitate comparisons with the literature. We also sourced additional private data to enhance

TABLE 1. Distribution of images in the car detector dataset for each of the vehicle class (Car, Truck, Bus) and the total amount of images.

	Car	Truck	Bus	Total
Train	43867	9973	6069	59909
Validation	1932	415	285	2632

the model's performance as needed or to tackle aspects of the problem for which no public dataset existed.

A. DATA ANNOTATION PROCESS

One of the challenges in the annotation process is to ensure consistency between different human annotators. Consistency across annotators, leading to a consistently labelled dataset, is crucial for obtaining reliable deep-learning models, especially in fine-grained annotation tasks such as instance segmentation [17]. However, ensuring good quality and consistent annotation is expensive and time-consuming.

To ensure high annotation quality, we used an iterative process. Firstly, small batches of 20 images were passed to three human annotators. Our aim was to refine the instructional materials given to the annotators to ensure consistent labelling of samples. For each batch of images, we assessed the inter-annotator consistency. We did this by measuring the intersection over union (IoU) over the annotations to check whether the annotations from different annotators were consistent. A low IoU means that the annotators have not provided consistent annotations and the instructional materials should be revised. Within a few iterations, we arrived at a labelling approach that ensured consistency between annotators and could mitigate miss-annotation issues. We then proceeded to provide the images in large batches to the annotators while reviewing the results of every labelled batch.

We later incorporated elements of active learning into the annotation process. For example, in the case of care damage annotation, at first, we randomly delivered images containing vehicles to the annotators, who would then label the damage or indicate that the image did not contain any damage. Our process was later refined by using our existing models to suggest annotations that could then be refined by human annotators. This eventually led to a performance increase of over 20% on the damage detector.

B. DATASET CREATION

In this section, we outline the datasets used for each sub-task of the overall damage severity evaluation problem.

1) CAR DETECTOR DATASET

For training the car detector we used the COCO [18] dataset. For evaluation, we used only images containing relevant vehicle objects classes i.e., car, truck and bus. The distribution of images within each class is shown in Table 1.

2) INTERIOR-EXTERIOR DATASET

For the Interior-Exterior dataset, we collected around 3500 exterior images and 1600 interior images from publicly available online sources. We used the Google-API to search for images of vehicle interiors over the time period 1980 - 2020. This ensures that the dataset contains a wide diversity of car interiors from different eras, allowing the model to generalize well. The interior images include dashboards as well as front and rear seat interior views. Figure 2 shows some interior images sampled from the dataset. The dataset was split into 80% for training and 20% for validation.

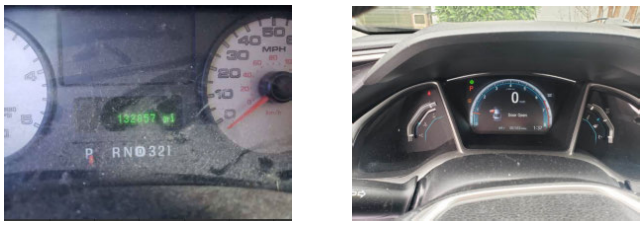


FIGURE 2. Interior samples of the Interior/Exterior Dataset.

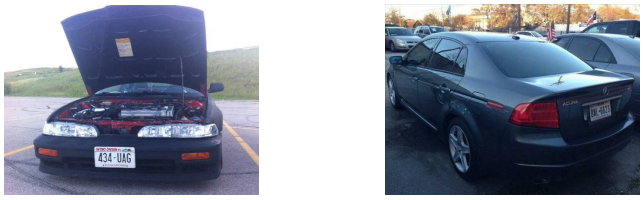


FIGURE 3. Samples of the car angle classifier's dataset. Front view (Left) and Rear Left view (Right).

TABLE 2. Distribution of car images in the Car Angle Dataset. Images are shown with respect to the angle label for the train, validation and testing sets.

Angle Class	Total	Train	Validation	Test
Rear	802	641	82	79
Rear-Left	619	518	65	36
Rear-Right	613	518	65	30
Front-Left	443	371	48	24
Front-Right	442	371	48	23
Front	212	168	21	23
Left	225	189	22	14
Right	221	189	22	10
TOTAL	3577	2965	373	239

3) CAR ANGLE DATASET

We used a set of public car images retrieved from online sources for the car angle dataset (See Figure 3). Images were manually labelled with one of 8 angle labels. Table 2 shows the angle labels used and the distribution of samples in the train, validation and test sets. The dataset was augmented by applying horizontal flipping to every image. For non-frontal/rear images this gives the view from the opposite side of the car e.g., flipping a front-right view creates a new image with the front-left view. This assumes a symmetrical exterior appearance, which is true in most non-damage cases. The dataset had a total of 3577 images and was split into 82% for training, 11% for validation and 7% for testing.

4) PARTS AND DAMAGE DATASET

The parts and damage datasets are described together as both datasets are similar. The parts dataset contained 8005 images

FIGURE 4. Example image showing the segmentation masks for annotated car parts used in the car parts dataset.

and the damage dataset 9226 images. For part classification, each of the car parts was labelled with one of 24 classes. Damage was labelled with one of 5 different types. Segmentation masks were created for all visible damage as well as the part boundaries. Images are annotated and labelled with the segmentation masks for each part, using the process described in Section III-A. To create both datasets, images were collected from private sources and annotated using the AWS GroundTruth annotator. The same tool was used for parts and damage annotation. The damage samples were distributed according to Figure 8. Figure 4 shows an example image of the car parts annotation used in the parts dataset, and Figure 6 shows examples of damage annotations.

Figure 7 and Figure 8 show the distribution of the class labels for each detector. In both cases, a small validation set was used. The validation sets were small because they were created as soon as the first small pool of annotated data was collected and kept intact throughout the data collection and model creation process to ensure consistency (see Section III-A). The parts dataset had 917 validation images and the damage dataset had 1408 validation images.

5) AIRBAG DATASET

The airbag dataset contains 2560 airbag and 2381 non-airbag car images. These images were collected from a combination of public and private sources. Example interior and exterior airbag deployment images are shown in Fig. 5.

6) CLAIMS DATASET

The purpose of our overall pipeline is to predict the damage cost estimate by combining structured information provided by the claimant with knowledge about the extent of damage measured using the computer-vision pipeline. To train a model to predict the cost estimate given this information, we use a dataset of around 18,000 claims containing over 200,000 images for training and a dataset of 2,600 claims containing 30,000 images for validation.

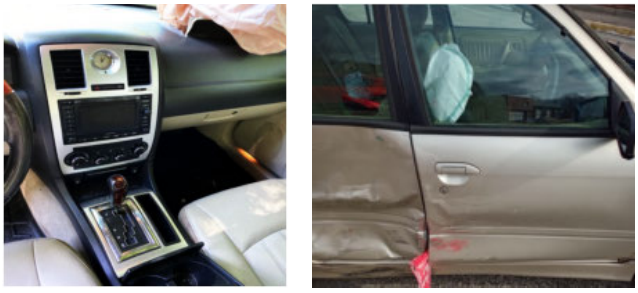


FIGURE 5. Example images of a deployed airbag from interior and exterior views used in the airbag dataset.

C. DAMAGE ASSESSMENT PIPELINE

The Damage Assessment Pipeline consists of six independent deep-learning computer-vision models and a model that combines computer vision and structured features to predict the severity of the damage. Given a set of unconstrained images, the features from all computer-vision models are combined into a single fixed-length feature vector by the feature fusion step. A binary model uses the fused feature vector, in combination with structured data associated with the client's account, to predict the damage cost estimate. The overall pipeline is illustrated in Figure 1.

The input to the pipeline is a customer claim. The claim may contain one or more images and an additional set of structured data. The images are unconstrained in-the-wild colour images collected by an untrained user with a mobile device. Under ideal circumstances, the images will show the vehicle from a variety of exterior angles, encompassing all damaged areas. For best results, a view of the whole car, or a big portion of it, should be included in each image. As a first step, the image orientation is corrected using the image metadata, and images are down-sampled to a lower resolution for processing. The pipeline is independently applied to each image to produce a set of features, and the features from all images are combined in the fusion step (See Section III-C6).

The question may be asked as to why the system was designed as a pipeline of independent models rather than training a single model end-to-end. Training such an end-to-end system would require a much larger and more diverse dataset. By decomposing the problem into multiple smaller steps we can take advantage of existing pre-trained models and available smaller datasets to solve each of the sub-problems and/or fine-tune the pre-trained models. Trying to tackle the complete problem end-to-end with a single model appears infeasible at this time. Our approach also has the advantage of being more interpretable. The analyst can look at the output from each intermediate model to understand how the final damage estimate was reached, which is crucial within a regulated industry such as the insurance sector.

In the following section, we will discuss each part of the pipeline in more detail. Details of the datasets used to train the models are provided in Section III-B.

1) VEHICLE DETECTOR

The vehicle detection network is the first step in our pipeline. Detected vehicle bounding boxes are provided as input for all later steps. Images not containing vehicles are discarded. It is crucial that this step meets certain requirements. Firstly, the detector needs to be reliable when used on realistic images. Secondly, it should be efficient enough to cope with a high volume of customer images.

The vehicle detection step uses the YOLOv4 object detector [19], [20] pre-trained on the public COCO object detection dataset [18]. The pre-trained model can detect and classify over 80 individual classes of objects, including various vehicle classes such as cars, buses and trucks, relevant to this work. In Section IV we describe the experiments used to select a pre-trained detector meeting these requirements.

2) INTERIOR/EXTERIOR CLASSIFIER

The purpose of this module is to detect whether an image shows an interior or exterior view of a car. Understanding whether the view in a particular image shows the interior or exterior of the vehicle helps to determine the damage assessment. This module is a binary classifier that takes an image as input and produces a probability score as output. In this binary classifier, the exterior class is considered as default due to being the most common one in a claim.

Similarly to the car detector (See Section III-C1), this step must be both reliable and efficient. This model produces a binary output, indicating whether an image contains an interior/exterior view, that is used by the final feature fusion step. We use a VGG-16 network for this classifier, although other architectures were also evaluated. In Section IV we outline the experiments used to select the optimal model for the interior/exterior classification task.

3) ANGLE CLASSIFIER

This module estimates the angle of the vehicle with respect to the camera e.g., frontal view, side view, frontal-left etc. We bin the estimated pose angles into one of eight possible views separated by 45 degrees, listed in Table 2.

Understanding vehicle angle is important for localising damage on the body of the vehicle and for differentiating symmetrical body parts where damage may appear. Frontal damage may be indicative of more severe damage due to the higher likelihood that vital mechanical parts, located at the front of the vehicle, are damaged. A vehicle angle classifier is also useful for helping to identify specific vehicle panels such as telling the difference between left and right door panels.

While it is possible to treat vehicle angle prediction as a simple regression problem, this causes some issues. Angles are fundamentally a measure of position on a circle i.e., the angles 0 and 359 degrees are close on the circle but appear to be very different to the regression loss. We, therefore, follow the approach of [21] and instead transform each angle into an (x,y) coordinate on the unit circle. We then train a network to regress the (x,y) coordinates corresponding to the vehicle angle using mean squared error (MSE) loss. This approach cleanly maps the angle prediction problem into a natural embedding space without any discontinuities. The final angle output by the model is then binned into the closest view from our eight possible viewing angles. A ResNet-18 model [22] was trained to perform the angle prediction task. See Section IV-B for experimental results on angle prediction.

4) PARTS AND DAMAGE DETECTION MODELS

The twin problems of detecting parts and detecting damage are the heart of the overall damage estimation problem. The majority of vehicles tend to follow a similar layout and have similar parts, which allows us to train a model to identify similar parts across a range of models of vehicles e.g., windscreen, front grill, rear-left door panel etc. We also take advantage of the similarities between the parts and damage detection problems. We treat both as segmentation problems where the objective is to predict a mask corresponding to the shape/area of the part or damage. Specifically, an instance segmentation approach is useful here. We use instances of parts and instances of damage to measure the extent of damage on each panel of the vehicle. This is something that would not be feasible using semantic segmentation. Instance segmentation identifies each distinct object of the same category, whereas semantic segmentation groups together all objects of the same category. Our approach allows the extent of damage to each body panel to be measured independently.

The damage detection task imposes some unique challenges compared with object detection or parts segmentation. For instance, the damage detection model has to deal with detecting damaged areas with arbitrary shapes. We classify damages into five classes; Damage, Broken Light, Broken Window, Dent and Missing Part. Figure 6 illustrates each type of damage, with the corresponding annotation. The damage class is the most severe. It is used when a panel is destroyed or when over half the panel is severely crumpled. Damaged vehicles can be very deformed, making the task of damage detection more challenging. We, therefore, used deformable convolutions [23] as kernels in the damage detection network backbone, which improves performance.

For the parts detection task, a total of 24 different car parts are detected as indicated in Fig. 7. Both the parts detection and damage detection models use a modified version of the MaskR-CNN network [13]. The backbone is a ResNet-50 [22] with a feature pyramid pooling network on top [24]. This helps the models learn different features across different scales. During the training of both models, we use the generalised IoU loss function [25] in the bounding box regressor. This provides a smooth derivative

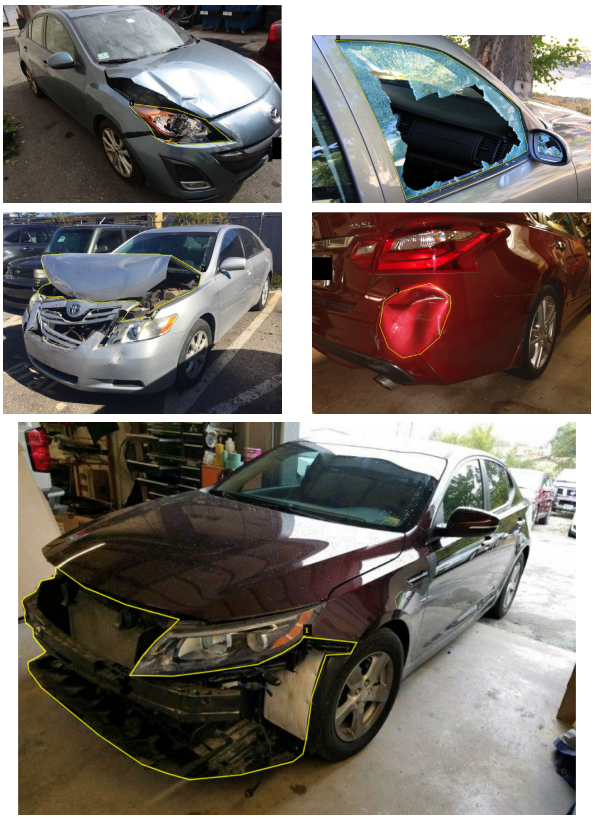


FIGURE 6. Example annotations from the car damage dataset of the five classes of damage. The damage classes are: broken light, broken window, damage, dent and missing part.

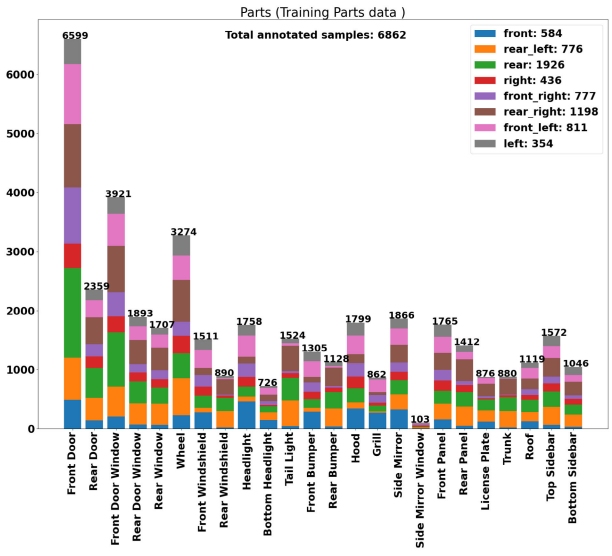


FIGURE 7. Parts training dataset annotations distribution. On the x-axis are the labels of the parts the colour illustrates the angles of the car in the image. The height presents the amount of annotated instances.

for non-overlapping bounding boxes as well providing better convergence.

5) AIRBAG CLASSIFIER - EXTERIOR AND INTERIOR

In historical data, airbag deployment has proven to be a crucial clue for human insurance analysts to quickly assess the

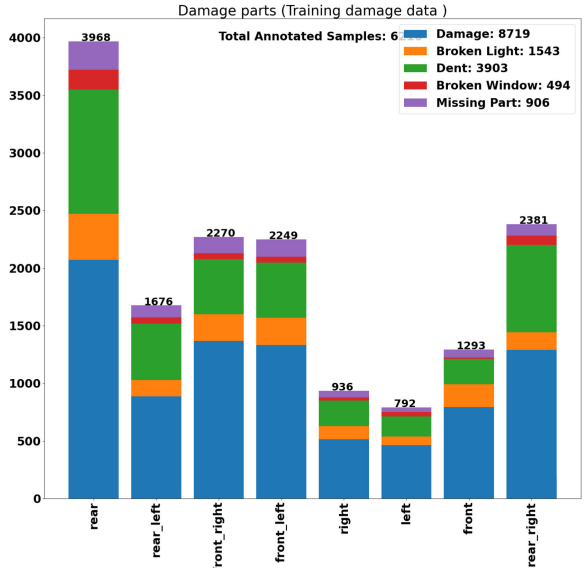


FIGURE 8. Damage training dataset annotations distribution. On the x-axis, we have the angle of the car in the image and the colours present the damage labels. The height presents the amount of annotated damage labels.

severity of the damage. Therefore we add this feature to our model. The airbag deployment classifier architecture is based on the ResNet-34 CNN and produces a binary output. This model was trained on a combination of public and private images. A web scraper was used to collect public data images from the internet. These images were then manually filtered and annotated. To increase the model's accuracy and generalization, data augmentation was used including rotation, brightness, contrast enhancement, random pixel dropout, horizontal flip, zoom and random scale.

6) FEATURE FUSION

The features from all the input images must be combined to produce an overall feature vector for damage severity prediction. The final cost prediction model requires a single input vector of fixed dimensionality regardless of the number of input images. Therefore, the feature fusion module aggregates the computer-vision features from all unconstrained input images into a single feature vector describing the overall damage observed on the vehicle.

To construct a vector of computer-vision features, we combine the outputs of the parts and damage segmentation models on all images. The parts segmentation model produces a mask for each car panel, while the damage segmentation model produces a mask for each damaged area. Given the predicted mask of each car panel and its corresponding predicted damage mask, we compute the Intersection over the Union score (IoU) of the damaged area with each panel. Doing this for all images from all views gives a measure of the level of damage of each car panel of the vehicle. The damage IoU score is then normalised with respect to part size i.e., the percentage of the

part damaged, to be independent of the size of the car in the image.

A single feature vector is produced to describe the overall level of damage of the car by combining the damage IoU scores for all car panels in all input images as well as their corresponding confidence scores given by the damage detector. For each of the 24 car panels (see Fig.7), the maximum damage IoU score for that panel across all images - and its corresponding confidence score - are retained to produce the overall damage feature vector. Although other simple aggregation functions could be used, e.g., mean, min etc, using the maximum score across all images achieved the best results in our evaluation.

The final feature vector output by the fusion module concatenates the computer-vision features with structured information. The computer-vision features include the damage feature vector described above plus the airbag detection probability. The structural features include the vehicle make and model, an estimation of the current vehicle model in the market, a binary feature indicating whether the vehicle was towed from the scene of the accident, and a binary feature indicating whether the vehicle was operable after the accident. The final feature vector containing all the relevant information for damage cost prediction has 50 dimensions.

7) FINAL DAMAGE COST PREDICTION MODEL

Once all the relevant data from the computer-vision pipeline and structured data have been gathered together, one final model is used to estimate the vehicle damage severity estimate. The final prediction is performed by an XGB (extreme gradient boosted tree model) [26], implemented using the LightGBM framework [26]. The XGB model takes a static feature vector as input, derived from the earlier computer-vision models, plus structural data from the insurance company database, and produces the vehicle damage severity estimate, conditioned on the input images. The model combines computer-vision features, such as the amount of damage detected on each body panel, and combines these with structured features, such as the age of the car, model, vehicle history etc., to reach an overall vehicle damage severity prediction.

IV. EXPERIMENTS

In this section, we describe in detail the experiments conducted for each model along with the details of model training and evaluation. For classification models, we measure performance using F1-score, Precision, Recall and Accuracy. For detection models, we measure performance using Mean Average Precision (mAP) at various recall values, as is standard practice in the literature.

A. CAR DETECTOR

As mentioned in Section III-B1, the car detector was trained on a subset of the COCO dataset. The distribution of images in the training and validation sets is shown in table 1.

TABLE 3. Car detector model performance comparison in terms of frame-per-second (FPS) processed on an Nvidia Tesla V100 GPU and mAP on the vehicle classes of the COCO dataset. Note TTA indicates the model used for test time augmentation. Two Yolo variants(M and XL) and three Efficient-net variants (D0, D6 and D7x), as explained in [19] and [27], were tested.

Model	Bounding Box mAP	FPS
Yolov4 M	0.445	250
Yolov4 M + TTA	0.460	81
Yolov4 XL	0.488	135
Yolov4 XL + TTA	0.507	40
EfficientDet D0	0.270	14.2
EfficientDet D6	0.520	6
EfficientDet D7x	0.556	3.6

To develop the car detector model we experimented with several well-known computer-vision architectures, YOLO and EfficientNet [28]. We used the pre-trained versions of these models available from public repositories online. For each model, we benchmarked its object detection performance on the vehicle classes from the COCO dataset.

With the YOLO architecture, we used the Test-Time-Augmentation (TTA) technique during inference. This helps to improve the prediction results. To perform TTA, we pass 6 augmented variations of the image to the model, then aggregate the bounding box predictions using non-maxima suppression (NMS) to keep the most prominent predictions. For TTA augmentations we flip the image horizontally and use slight variation in scale.

In Table 3 we compare the YOLO and EfficientDet models in terms of vehicle detection performance and the number of frames-per-second processed on an Nvidia Tesla V100 GPU. Speed is as important as detection accuracy, due to the large number of customer images that need to be processed. Looking at the results, we can see that the YOLO architecture is able to process many more images per second than the EfficientDet variants tried, making YOLO a better candidate for our use case. EfficientDet utilises 3 parameterized FPN [24] stages called BiFPN which makes the network slow for our case.

Following this initial experiment, we decided to further explore variants of the YOLO detector with and without test-time-augmentation. In these experiments, we break down the performance on each vehicle class individually. We also experiment with different input image resolutions i.e., either 640 or 1280 pixels on the long edge. The results of these experiments are shown in Table 4. We can see that the use of higher-resolution images generally improves performance. The results from using TTA are more mixed, but on balance, this technique helps to improve the robustness and stability of the detector. We, therefore, opted for high-resolution input images, the use of TTA and the use of the YOLOv4 XL model in our car detector stage. This system performs car detection well while still meeting our needs in terms of processing speed.

B. CAR ANGLE CLASSIFIER

To develop the car angle classifier we first implemented a baseline solution using Google's AutoML AI platform [29]

TABLE 4. Comparison of car detector models with different input image resolutions. TTA indicates the use of Test Time Augmentation.

Model	Input Resolution	mAP Car	mAP Truck	Prec. Car	Rec. Car	Prec. Truck	Rec. Truck
Yolo-XL	640	0.657	0.322	0.559	0.944	0.193	0.828
Yolo-XL + TTA	640	0.621	0.285	0.726	0.878	0.309	0.69
Yolo-XL	1280	0.701	0.382	0.84	0.812	0.422	0.621
Yolo-XL + TTA	1280	0.723	0.405	0.815	0.862	0.404	0.71

TABLE 5. Angle classifier model comparison between the Google baseline model and our ResNet-18 classifier.

	F1-score	Accuracy
Google Baseline	0.70	0.75
Our Model	0.939	0.949

TABLE 6. Interior exterior classification model performance comparison.

	F1-Score	Accuracy
EfficientNet-B2	0.93	0.88
ResNet-50	0.96	0.94
VGG-16	0.96	0.94

by providing it with our training dataset and allowing its automation tool to develop the model. We compare this baseline solution with our proposed model outlined in Section III-C which is based on Resnet-18. Both models were trained and tested with the same data to provide a fair comparison. The dataset was augmented by applying horizontal flipping to every image. For non-frontal/rear images this gives the view from the opposite side of the car e.g., flipping a front-right view creates a new image with the front-left view. This assumes a symmetrical exterior appearance, which is true in most non-damage cases.Table 2 shows the distribution views of the dataset.

Table 5 presents a comparison of performance on the testing set between the baseline model implemented using Google's AI platform and our solution. It can be seen that our model significantly outperforms the baseline in terms of the F1-score. We also visualise the raw angle i.e., (x,y) coordinates, predicted by the model in Figure 9. This shows that the model's performance is very robust in many scenarios. Most of the incorrectly predicted points belong to adjacent classes showing the robustness of our approach.

C. INTERIOR-EXTERIOR CLASSIFIER

The purpose of the Interior-Exterior classifier is to predict whether an image shows the interior of a car or not. This is then used in feature fusion to either populate the damage panel features or the airbag deployment feature. We compare the performance of three well-known architectures at this task: EfficientNet, ResNet and VGG.

Each of the models tried for this task was trained using the class balance focal loss objective [30] for over 70 epochs using SGD optimizer [31]. Table 6 presents the results of the models on a validation set of 900 images.

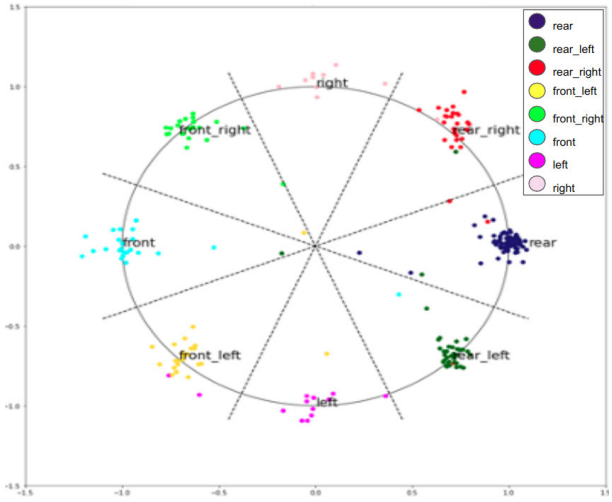


FIGURE 9. Plotting the raw output of the angle classifier to verify its robustness. The angle classifier predicts an (x,y) coordinate on the unit circle corresponding to an angle. Colours represent the ground truth class. The majority of predictions cluster close to the centre of each of the 8-angle bins.

TABLE 7. Airbag classifier experimental analysis showing performance with and without the use of data augmentation.

Use-Augmentation	Accuracy	Precision	Recall	F1-score
False	0.931	0.947	0.928	0.937
True	0.952	0.967	0.948	0.957

One of the challenges we have found is dealing with dark exterior images as these may be easily confused for interior images. When a car has been captured in shadow, the colour of the car is black, or the mechanical parts are exposed, then the model can sometimes predict an exterior view with an interior. To mitigate this issue we have found it necessary to use a high probability threshold at 70% so that the model only predicts the interior label when it is very confident.

D. AIRBAG CLASSIFIER

The airbag classifier was trained with a mixture of public and private data (See Section III-B5). The ResNet-34 model was trained for 20 epochs. Early stopping and the *reduce the learning rate on plateau* techniques were used during training. Experiments were performed with and without data augmentation, including rotation, brightness, contrast enhancement, random pixel dropout, horizontal flip, zoom and random scale. In all cases, the validation set remained unchanged.

The results of this experiment are given in Table 7. We can see that the use of data augmentation improves the f1-score. It is well known that data augmentation can facilitate the training process of a model and generalise better.

E. PARTS AND DAMAGE DETECTOR

As mentioned in Section III-C we treat both parts and damage detection as segmentation problems. We train models to predict segmentation maps over the image showing the locations of parts or damage. Specifically, we compared the MaskR-CNN and MaskR-DCN architectures. The MaskR-DCN uses deform-able convolutions [23] as kernels in the network's ResNet-50 backbone, which may help this model to cope with structural deformations, such as those caused by damage.

We trained both models using the Adam optimizer [32], which helps to improve convergence compared with standard Adam. Generalised IoU loss [25] was used in training the bounding box regressor. This provides a smooth derivative for non-overlapping boxes as well giving better convergence.

We used a linear warm-up for 500 iterations with a 10% scaled [33], [34] learning rate. This means that we start with 10% of the full learning rate and scale this up to the original after 500 iterations. This technique avoids numerical instability in the early training steps. Initially, we train the model for 100 epochs with a frozen backbone, after which all layers are unfrozen. In the end, we finalize training freezing the backbone and decreased the learning rate to 10^{-5} for the last 100 epochs for fine-tuning. We used a batch size of 16. We used data augmentation implemented using the Albumentations [35] library. Specifically, for both parts and damage detection we used scale, translation and rotation augmentations along with colour transformations including random brightness enhancement, colour shifts and channel shuffling.

Tables 8 and Table 9 show the results of our initial comparison between the MaskR-CNN and MaskR-DCN architectures for parts and damage detection respectively. This initial comparison used only a small subset of the training data with 2139 training images and 917 validation images. We can see that the MaskR-DCN consistently outperforms the MaskR-CNN across both the parts and damage detection tasks at AP@0.5:0.95. We, therefore, decided to use the MaskR-DCN in our final production system.

Given our choice to use the MaskR-DCN network, we then performed additional experiments on parts and damage detection using the complete dataset. The complete parts dataset contained 7088 training images and 917 validation images. The complete damage dataset contained 7818 training images and 1408 validation images. One of the main problems that we faced with damage detection was the sensitivity of our model to Out-of-Distribution (OOD) data [36]. For instance, the model may predict areas of damage when presented with an out-of-distribution image, such as a car interior. To address this issue, we also added 900 hard-negative samples containing interior views during the training process [37]. Those images were collected from private sources and were used in the interior/exterior model training.

TABLE 8. Part detection model comparison between MaskR-CNN, MaskR-DCN and the MaskR-DCN with additional training data. Note that both models were trained using data augmentation, Generalized IoU Loss and Soft NMS.

	AP@0.5:0.95		AP@0.5	
	Bbox	Mask	Bbox	Mask
MaskR-DCN	0.449	0.466	0.70	0.68
MaskR-CNN	0.42	0.44	0.73	0.708
MaskR-DCN & Train Data++	0.86	0.83	-	-

TABLE 9. Damage detection models comparison between MaskR-CNN, MaskR-DCN and the MaskR-DCN with additional training data. Note that both models were trained using data augmentation, Generalized IoU Loss, Soft NMS.

	AP@0.5:0.95		AP@0.5	
	Bbox	Mask	Bbox	Mask
MaskR-DCN	0.723	0.751	0.862	0.863
MaskR-CNN	0.652	0.715	0.869	0.865
MaskR-DCN & Train Data++	0.76	0.77	-	-

TABLE 10. Final prediction models compared. Our proposed model, using structured and image features is compared with a baseline approach using only structured features.

	Precision	Recall	F1-score	AUC
Structure data only	0.86	0.66	0.70	0.896
Structure + image data	0.86	0.70	0.75	0.918

Tables 8 and Table 9 show the results of the final MaskR-DCN network trained with the additional data (Training data++ in the tables) compared with the earlier models. We can see that the use of additional training data gives a dramatic improvement to the performance of the parts detection model and a modest improvement to the damage detection model, which was already performing well.

F. FINAL DAMAGE SEVERITY PREDICTION MODEL

Our proposed XGBoost model combines structured data about the car with features extracted from the images, such as the level of damage to each panel, to predict a high or low level of severity for a particular claim.

To validate our approach, we compare the performance of our proposed model combining image features with structured data to a baseline model that uses structured features only. Our proposed XGBoost model was trained using both the structured-data features and image-based features, while the baseline model only had access to the structured-data features. The results of this comparison are shown in Table 10.

We can see that our proposed model combining the structured features along with the image-based features outperforms the model using structured features by over 2% AUC. The use of image features achieves the same precision while reducing the number of false negatives, resulting in a better f1-score. This boost in the recall is down to the strength

of our new model at detecting more loss cases than the baseline.

V. CONCLUSION

The estimation of car damage costs by using image data has been a research challenge in the insurance industry. Our efforts prove that there is a lot of potential for using images to estimate the damage severity of a vehicle which could revolutionise the insurance user experience. Our method could help by shortening the length of time needed for the insurance company to resolve a claim. Thus saving time and money for both parties, while increasing customer satisfaction. We have shown that our proposed system can reduce the number of false positives, which has the potential to save significant amounts of money for a large insurance company. We have demonstrated that current improvements in deep learning and the adequate use of proprietary and public data can help an insurance organisation develop a full pipeline to improve its services. One of the main remaining challenges is to further refine the pipeline to give a more granular prediction about the damage to vehicle parts in order to improve prediction accuracy.

REFERENCES

[1] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[2] H. J. Kelley, "Gradient theory of optimal flight paths," *ARS J.*, vol. 30, no. 10, pp. 947–954, Oct. 1960.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1026–1034.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[5] H. Bandi, S. Joshi, S. Bhagat, and A. Deshpande, "Assessing car damage with convolutional neural networks," in *Proc. Int. Conf. Commun. Inf. Comput. Technol. (ICCICT)*, Jun. 2021, pp. 1–5.

[6] J. De Deijn, "Automatic car damage recognition using convolutional neural networks," in *Internship report MSc Business Analytics*. 2018. [Online]. Available: https://fdocuments.in/document/automatic-car-damage-recognition-using-convolutional-automatic-car-damage.html?page=1

[7] P. Li, B. Shen, and W. Dong, "An anti-fraud system for car insurance claim based on visual evidence," *Int. Res. J. Eng. Technol. (IRJET)*, vol. 7, no. 7, pp. 3875–3884, 2018.

[8] Q. Zhang, X. Chang, and S. B. Bian, "Vehicle-damage-detection segmentation algorithm based on improved mask RCNN," *IEEE Access*, vol. 8, pp. 6997–7004, 2020.

[9] R. E. van Ruitenbeek and S. Bhulai, "Convolutional neural networks for vehicle damage detection," *Mach. Learn. Appl.*, vol. 9, Sep. 2022, Art. no. 100332.

[10] K. Patil, M. Kulkarni, A. Sriraman, and S. Karande, "Deep learning based car damage classification," in *Proc. 16th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2017, pp. 50–54.

[11] H. V. Yashaswini and V. Karthik, "Car damage detection and analysis using deep learning algorithm for automotive," *Int. J. Sci. Res. Eng. Trends*, vol. 5, no. 6, pp. 1896–1898, 2019.

[12] D. Widjojo, E. Setyati, and Y. Kristian, "Integrated deep learning system for car damage detection and classification using deep transfer learning," in *Proc. IEEE 8th Inf. Technol. Int. Seminar (ITIS)*, Oct. 2022, pp. 21–26.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[14] M. Parhizkar and M. Amirfakhrian, "Recognizing the damaged surface parts of cars in the real scene using a deep learning framework," *Math. Problems Eng.*, vol. 2022, Aug. 2022, Art. no. 5004129.

[15] F. Imaam, A. Subasinghe, H. Kasthuriarachchi, S. Fernando, P. Haddela, and N. Pemadasa, "Moderate automobile accident claim process automation using machine learning," in *Proc. Int. Conf. Comput. Commun. Informat. (ICCCI)*, Jan. 2021, pp. 1–6.

[16] N. Fernando, A. Kumarage, V. Thiyaganathan, R. Hillary, and L. Abeywardhana, "Automated vehicle insurance claims processing using computer vision, natural language processing," in *Proc. 22nd Int. Conf. Adv. ICT for Emerg. Regions (ICTer)*, Nov. 2022, pp. 124–129.

[17] S. Wang, C. Li, R. Wang, Z. Liu, M. Wang, H. Tan, Y. Wu, X. Liu, H. Sun, R. Yang, X. Liu, J. Chen, H. Zhou, I. Ben Ayed, and H. Zheng, "Annotation-efficient deep learning for automatic medical image segmentation," *Nature Commun.*, vol. 12, no. 1, p. 5915, Oct. 2021.

[18] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Computer Vision—ECCV*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham, Switzerland: Springer, 2014, pp. 740–755.

[19] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection, 2020, *arXiv:2004.10934*.

[20] G. Jocher et al., "Ultralytics/YOLOv5: v6.2—YOLOv5 classification models, Apple M1, reproducibility, ClearML and Deci.ai integrations," Tech. Rep., Aug. 2022. [Online]. Available: https://github.com/ultralytics/yolov5/discussions/8996

[21] S. S. Mukherjee and N. M. Robertson, "Deep head pose: Gaze-direction estimation in multimodal video," *IEEE Trans. Multimedia*, vol. 17, no. 11, pp. 2094–2107, Nov. 2015.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[23] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 764–773.

[24] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," 2016, *arXiv:1612.03144*.

[25] H. Rezatofighi, 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 658–666.

[26] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2016, pp. 785–794.

[27] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," 2019, *arXiv:1911.09070*.

[28] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[29] Google. *AutoML—Vision*. Accessed: Jun. 2022. [Online]. Available: https://cloud.google.com/automl

[30] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9260–9269.

[31] S. Ruder, "An overview of gradient descent optimization algorithms," 2016, *arXiv:1609.04747*.

[32] I. Loshchilov and F. Hutter, "Fixing weight decay regularization in Adam," 2017, *arXiv:1711.05101v2*.

[33] P. Goyal, P. Dollár, R. B. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, large minibatch SGD: Training ImageNet in 1 hour," 2017, *arXiv:1706.02677*.

[34] T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li, "Bag of tricks for image classification with convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 558–567.

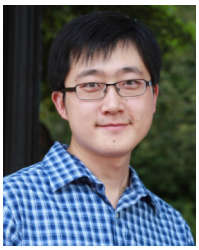
[35] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, p. 125, Feb. 2020.

[36] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. A. DePristo, J. V. Dillon, and B. Lakshminarayanan, "Likelihood ratios for out-of-distribution detection," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2019, pp. 1–12.

[37] M. Li, L. Wu, A. Wiliem, K. Zhao, T. Zhang, and B. Lovell, "Deep instance-level hard negative mining model for histopathology images," in *Medical Image Computing and Computer Assisted Intervention—MICCAI*. Berlin, Germany: Springer-Verlag, 2019, pp. 514–522.

DIMITRIOS MALLIOS received the M.Eng. degree in computer engineering from the University of Thessaly, Greece, in 2018, and the M.Sc. degree in artificial intelligence from the University of Southampton, U.K. His exceptional dedication and passion for the subject were reflected in his remarkable academic performance, earning him an honors distinction.

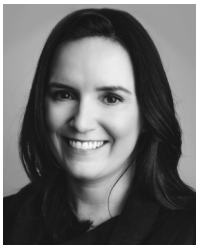
A significant milestone in his academic journey was the publication of a section of his master's thesis dissertation at the prestigious European Signal Processing Conference (EUSIPCO). His research work was recognized for its groundbreaking insights and was presented orally at the conference. Driven by a desire to address real-world challenges, he has focused his research interests on the applications of computer vision as an end product. With a deep understanding of this field, he has sought to leverage computer vision technologies to tackle diverse practical challenges. His expertise spans various domains, including medical applications, insurance, and urban areas taking into account hardware constraints.



LI XIAOFEI received the M.S. degree in statistics and the Ph.D. degree in applied economics from the University of Georgia, in 2014 and 2015, respectively. He is currently a Data Scientist Director of Liberty Mutual Insurance and leading machine learning model and computer vision model development to support claims operation.

NIALL MCLAUGHLIN received the M.Eng. and Ph.D. degrees in computer science from Queen's University Belfast, in 2008 and 2012, respectively. He is currently a Lecturer with the School of Electronics, Electrical Engineering and Computer Science, Queen's University of Belfast. His research interests include deep learning applied to cyber-security and computer vision.

JESUS MARTINEZ DEL RINCON received the B.Sc. degree in telecommunication engineering and the Ph.D. degree in computer vision from the University of Zaragoza, Zaragoza, Spain, in 2003 and 2008, respectively, for his work into the development of tracking algorithms for video surveillance and human motion analysis. He is currently a Senior Lecturer with the School of Electronics, Electrical Engineering and Computer Science, Queen's University of Belfast, Belfast, U.K. His research interests include deep learning applied to video surveillance and cybersecurity.



CLARE GALBRAITH received the B.Sc. degree in mathematics and computer science from Queens University Belfast. She is currently a Senior Engineering Manager with Liberty IT. She was the Project Manager for the Vehicle Damage Estimation Project.

RORY GARLAND received the B.Sc. degree in physics from Hull University, in 2012, the M.S. degree in physics from York University, in 2013, and the Ph.D. degree in physics from Queens University Belfast (QUB), in 2018. He was a Lead Data Scientist with Liberty IT and acted as a Technical Lead for the Vehicle Damage Estimation Project. His research interests include semi-supervised learning and model quantization.