Analysis Performance of One-Stage and Two Stage Object Detection Method for Car Damage Detection

Harum Ananda Setyawan[1], Alhadi Bustamam[2]*, Rinaldi Anwar Buyung[3]
Department of Mathematics, University of Indonesia, Depok, Indonesia[1, 2]
Department of Data Science, Global Risk Management, Jakarta, Indonesia[3]

***Abstract*—The large use of private cars is directly proportional to the number of insurance claims. Therefore, insurance companies need a breakthrough or new approach that is more effective and efficient to be able to compete for the trust of their customers. One approach that can be taken is to use artificial intelligence to detect damage to the car body to speed up the claims process. In this research, several experiments will be carried out using various types of models, namely Mask-R-CNN, ResNet50, MobileNetv2, YOLO-v5, and YOLO-v8 to detect damage to the car body. Furthermore, in the experiments that were carried out, the best results were obtained using the YOLO-v8x model with precision, recall, and F1-score values of 0.963, 0.951, and 0.936 respectively.**

***Keywords*—Car damage detection; insurance claim; deep learning; object detection**

I. Introduction

The prevalence of private vehicle usage in Indonesia is notably substantial. As per data extracted from the [1] for the year 2022, an estimated 17,175,632 units of private automobiles were in circulation across the nation. This proliferation of private vehicles in Indonesia corresponds directly with the incidence of traffic accidents. According to [2], approximately 25,144 cases of traffic accidents occurred during the first semester of 2022, resulting in a cumulative loss nearing three billion Indonesian rupiahs (Pusiknas Bareskrim Polri, 2023). Prevalent accident types include head-on collisions, rear-end collisions, and instances of vehicles veering off roads, with respective occurrences numbering 3,503, 3,066, and 2,951 cases in sequence.

The World Health Organization (WHO) identifies several primary factors contributing to traffic accidents and the heightened risk of resultant losses. These factors encompass driving at speeds surpassing the average, driving under the influence of alcohol, operating vehicles devoid of essential safety equipment such as helmets, seat belts, and child car seats, engaging in communication activities while driving, encountering inadequacies in road infrastructure, possessing vehicles with substandard maintenance, experiencing deficient post-accident care, and encountering lax enforcement of traffic laws [3].

Motor vehicle insurance has become a popular means of reducing the damages that people suffer from traffic accidents. To reduce the losses from unanticipated events like theft, accidents, and disasters, a motor vehicle insurance plan is required. As a result, a large number of insurance firms have surfaced that provide services related to motor vehicle insurance.

Indonesia has witnessed a tremendous increase in the motor vehicle insurance market. The company in this sector showed a premium growth of 19.4% in the third quarter of 2022 compared to the third quarter of 2021, according to a study [4]. To be competitive, motor vehicle insurance companies must have a solid and superior business strategy compared to other businesses, given the industry's rapid expansion. The plan for processing insurance claims is one of the most important factors to take into account in the insurance industry. Research in study [5] asserts that tangibility, reliability, assurance, responsiveness, and empathy are the five essential aspects of service quality that can be assessed. Research in study [6] states that each of the previously listed dimensions is defined. First of all, tangibility describes how the actual buildings, tools, staff, and marketing materials seem in relation to the level of service that the insurance firm offers. The capacity to provide promised services consistently and precisely is the second definition of reliability. Next, assurance comprises staff members' expertise, politeness, and capacity to inspire trust in customers. Moreover, being responsive means being ready to help clients and offer timely assistance. Finally, empathy is showing consideration and care for the people who serve clients. Fig. 2 describes how customers feel about an insurance product's satisfaction and quality. According to a study [5], in the PT Multi Artha Guna Insurance motor vehicle insurance market, responsiveness has the lowest score, followed by dependability, empathy, assurance, and tangibility. Furthermore, the research indicates that the ease of the process scheme for filing auto insurance claims to the insurance company is the aspect that most determine consumer satisfaction in insurance services.

One of the procedures used by customers in the motor vehicle insurance sector when they sustain losses on their cars in accordance with the arrangements they have with the insurance provider is filing a claim. The claims procedure is still primarily completed by hand at this time. Because set and consistent parameters are not applied uniformly among insurance firms, this leads to consumer dissatisfaction with the damage assessment process.

This procedure can be made digital and automated with data science and artificial intelligence (AI) techniques, especially deep learning. AI models can be trained using data from prior claims to comprehend and forecast car damage, resulting in a more transparent and objective evaluation. These

AI models can also be trained and enhanced over time, which will speed up and improve the efficiency of the claims process.

Moreover, the procedure for submitting claims currently in place takes a long time. This process can be accelerated by utilizing AI and data science. Sophisticated data analysis methods can be used to find trends in the claims, speed up the verification process, and even help find fraudulent activity. This may result in a quicker settlement of claims, which would eventually increase client satisfaction. It is therefore necessary to develop a more widely accepted, widely adopted, and well-standardized claim submission procedure. Customers will be able to file claims more conveniently, get their cars fixed quickly, and benefit from more transparent and objective damage assessment procedures, all of which will increase customer satisfaction. To overcome the previously described problems, a data science and deep learning strategy will be used in this study.

By employing methodologies rooted in data science and deep learning, it is anticipated that the efficiency and effectiveness of the claims process can be improved while minimizing subjectivity. Data science, an interdisciplinary domain, encompasses scientific techniques, algorithms, and systems to glean insights from data in diverse formats, structured or unstructured. Drawing upon principles from various fields such as mathematics, statistics, information science, and computer science, data science is closely associated with practices like data mining, machine learning, and big data analysis. It facilitates the development of predictive models, identification of trends and patterns, and aids in data-driven decision-making. Deep learning, as described in study [7], is a branch of machine learning that enables computers to learn from prior training and grasp complex concepts. As computers acquire knowledge through experience, human operators are not required to explicitly provide all necessary information. Consequently, this approach holds the potential to enable real-time claims processing for consumers. In the studies by [8] and [9], Deep Learning, particularly Convolutional Neural Networks, was utilized to detect Diabetic Retinopathy from fundus images.

This study will use image data to apply the deep learning approach further to the detection of damaged objects in cars. It is anticipated that using this method will enable effective, real-time damage identification, which will benefit clients. Furthermore, since AI can now automate damage detection—a task that was formerly completed by qualified professionals—insurance firms can save money on their operations.

There are two primary categories of frameworks for generic object detection techniques, according to study [10]. The first kind operates according to the standard object detection procedure, first generating region proposals and subsequently classifying each proposal according to several item types. The second style uses integrated frameworks to produce final findings that include both categories and locations simultaneously, seeing object detection as a classification or regression problem. R-CNN, Mask R-CNN, SPP-net, Fast R-CNN, Faster R-CNN, and FPN are some of the models for the first type; SSD, YOLO (You Only Look Once), DSSD, and DSOD are some of the models for the second type.

Mask R-CNN is an extension of the Faster R-CNN model designed for instance segmentation, i.e., the ability to classify objects in images and simultaneously produce accurate masks for each object instance. Mask R-CNN adds a branch for mask prediction to Faster R-CNN, enabling it to distinguish between objects at the pixel level. In the context of detecting damage on car bodies, studies have used Mask R-CNN to accurately identify and localize damages such as scratches, dents, and scrapes. For example, research in [11] used Mask R-CNN to detect and classify various types of damage on car bodies using a dataset consisting of thousands of car images taken from various angles, demonstrating a high level of precision in damage detection. Subsequently, [12] applied an improved Mask RCNN by optimizing ResNet and FPN as feature extraction. In this study, the use of the improved Mask RCNN method increased accuracy by almost 2%, from 94.53% to 96.38%.

Once the damage has been successfully detected using Mask R-CNN, the classification of the type of damage is performed. Research in study [13] used CNN with EfficientNet and MobileNetV2 architectures. In this study, the results from Mask R-CNN segmentation were used as input for the classification model. The use of this technique in the classification model can improve the F1-score value of the model by up to 9%. The best classification model is MobileNetV2, which uses this technique with an F1-score of up to 91%. Previously, damage detection studies with transfer learning models have also been conducted by comparing various architectures such as Inception V3, VGG16, VGG19, Xception, MobileNet, and ResNet50. The best accuracy results were obtained by the MobileNet model at 97.28%, followed by ResNet50. Another study conducted by [14] used transfer learning models comparing VGG16, VGG19, ResNet34, and ResNet50 architectures. The results showed that ResNet50 had the best accuracy results at 96.39%, followed by VGG19, VGG16, and ResNet34 at 95.87%, 94.84%, and 93.88%, respectively. Furthermore, [15] added an ensemble method to the transfer learning model created. The best result was obtained when using the ResNet architecture with an accuracy of 88.24%.

YOLO is one of the Convolutional Neural Network (CNN) based object detection models widely used today. YOLO can perform two tasks simultaneously, namely detection and classification of damages. In previous studies, several kinds of research have been conducted using YOLO, such as [16] using the YOLO-v5 model to detect damages and the location of insulators. Then, [17] conducted an Automatic Non-Parking Overload Detection Method based on the YOLO-v5 model. In addition, [18] performed real-time object detection for substation security warnings based on YOLO-v5, [19] used the YOLO-v5 model to detect road damages, and [20] used the YOLO-v5 model to detect car damages. In January 2023, [21], the developer of the YOLO-v5 model, released one of the latest versions of YOLO, namely YOLO-v8, which is an improvement over YOLO-v5, where the YOLO-v8 algorithm can be faster in object detection and also more accurate in detecting small objects.

In this research, the YOLO-v5 and YOLO-v8 algorithm approaches will be conducted as algorithms for detecting

objects that have been previously conducted in previous studies. The reason for choosing the YOLO-v5 algorithm is because it is one of the most widely used object detection algorithms in various fields at present. Then, the YOLO-v8 algorithm, which is one of the latest version of the YOLO algorithm at present, will also be used. Furthermore, the evaluation results obtained from the YOLO algorithm will be compared with the approach that has been previously done in detecting damages on vehicles, namely damage segmentation using Mask R-CNN and damage type classification using CNN. In this research, the two best CNN architectures based on previous studies, namely ResNet and MobileNet, will be used.

II. Related Work

In this segment, preceding research relevant to the current investigation will be elucidated. Numerous inquiries into car damage detection have been conducted previously. These studies can be seen in Table I.

TABLE I. State-of-the-Art

Research	Method	Dataset	Evaluation
[20]	YOLO-v5	767 images	Precision = 95% Recall = 87% F1-score = 91%
[22]	CNN	3000 images	Accuracy = 98%
[23]	YOLO-v3	150 images	Map score = 82%
[13]	Mask R-CNN and CNN (MobileNetV2)	1600 images	F1-score = 91%
[24]	Mask R-CNN and CNN (VGG16)	460 images	Precision = 91% Recall = 90% F1-score = 90%
[14]	CNN (AlexNet, VGG19, InceptionV3, ResNet50, MobileNets V1.0)	1172 images	Accuracy = 96%
[25]	Transfer learning and VGG16	2300 images	Accuracy = 87%
[26]	R-CNN (MobileNet)	600 images	Accuracy = 95%
[12]	Mask R-CNN (ResNet101)	2000 images	Accuracy = 96%

Therefore, this study will compare the performance of yolo-v8, yolo-v5, ResNet50, and MobileNetV2.

III. Data and Methodology

Within this section, the data and methodologies employed to address the focal subjects of inquiry in this study will be elucidated. Fig. 1 delineates the procedural framework underpinning this research endeavor.

A. Dataset

In this study, the acquisition of photographs depicting automobile damages was conducted in collaboration with PT Global Risk Management. The captured images of vehicles encompass various categories, including scratches, dents, tears, shattered or cracked glass, and fractured or cracked lights. The entirety of the data utilized for this investigation amounts to 700 images. Each class's data volume has been organized into five distinct categories, as illustrated in Fig. 2.

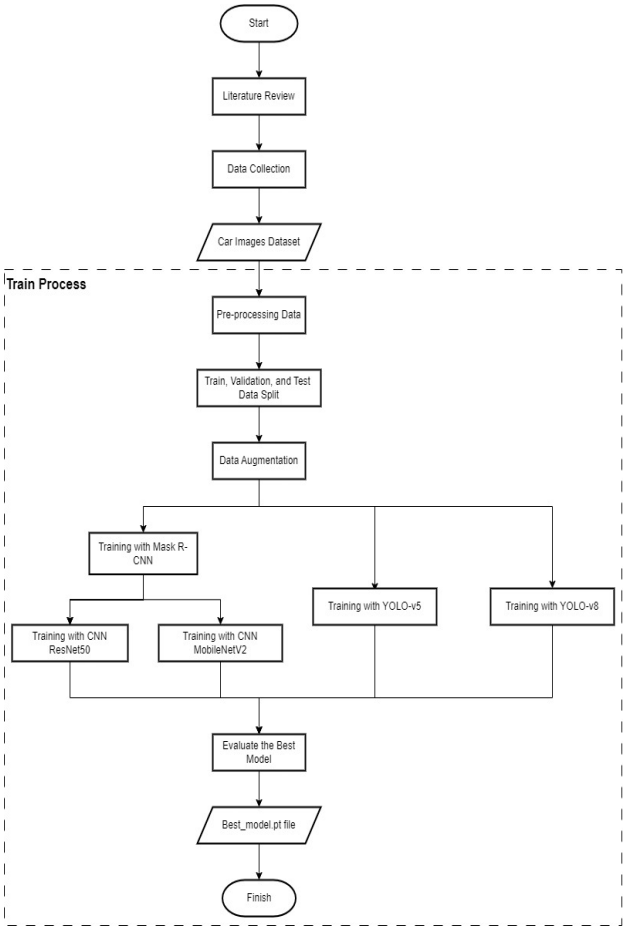

Fig. 1. Research workflow.

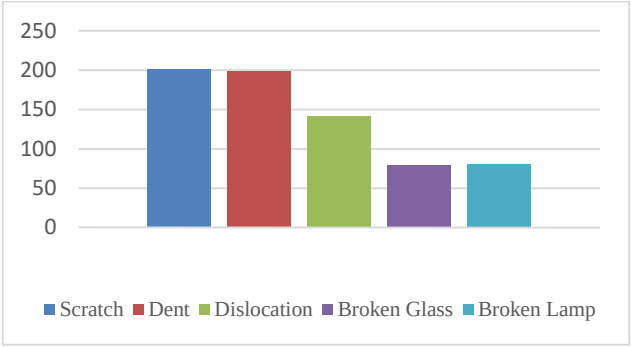

Fig. 2. Damages distribution in the dataset.

B. Data Preprocessing and Augmentations

Several pre-processing data techniques employed in this study include:

- Data Annotation: The annotation process involves marking the location and type of damage on car images. In this study, two annotation approaches were employed: the first utilized bounding boxes for the YOLO algorithm, while the second employed annotation in the Common Object in Context (COCO) format.

- Resizing Data: Resizing images during the data preprocessing stage is a critical step in preparing the dataset for deep learning models. This process involves adjusting the size of images to meet the requirements of the model, with benefits such as computational efficiency, reduced memory load, ensuring consistency in input sizes for the model, and preventing overfitting. In this study, the image sizes of 1024×1024 pixels for the Mask R-CNN algorithm, 224×224 pixels for CNN architectures ResNet and MobileNet, and 640×640 pixels for YOLO-v5 and YOLO-v8 algorithms will be utilized.
- Data Train, Validation, and Test Split: The division of data into training, validation, and test sets is a key stage in the development of deep learning models. The training data, which comprises the majority of the dataset, is used to train the model and enable it to understand patterns and features within the data. Validation data is utilized during training to evaluate the model's performance and prevent overfitting or underfitting. Meanwhile, test data is employed to assess the model's ability against data it has not previously encountered. In this study, the data was divided into train, validation, and test sets with proportions of 70%, 20%, and 10%, respectively.
- Following this, in our research, augmentation techniques were applied to the training data to enhance the dataset's size and optimize algorithm performance. The augmentation methods used include adjusting hue within the range of -90° to 90°, modifying saturation between -50% and 50%, tuning brightness from -20% to 20%, and implementing mosaic augmentation. Below are explanations for each augmentation process:
- Hue Augmentation: Hue augmentation involves adjusting the hue values in an image's color models (HSL or HSV) to enhance data diversity. By rotating hue values, colours shift across the spectrum, promoting learning across various color ranges. This technique prevents models from fixating on specific colors, fostering better adaptation to diverse color variations in new data. Unlike relying solely on memorized colors, hue augmentation encourages focus on object features and shapes. Despite potentially unnatural color shifts, the goal is to improve the model's ability to recognize general visual patterns, reducing reliance on specific color cues.
- Saturation Augmentation: Saturation augmentation is a technique that alters color intensity within an image, affecting brightness and color vibrancy. By randomly adjusting pixel saturation values, it creates varied color tones, prompting machine learning models to discern objects and patterns across different saturation levels. This technique enhances model performance in real-world scenarios with diverse lighting conditions or weather, enabling recognition of objects amidst varying color saturation levels. However, extremes in saturation levels may lead to distorted images, necessitating careful adjustments to maintain visual consistency. When combined with other augmentation methods like hue adjustment or resizing, it enriches dataset variations, aiding models in understanding diverse color schemes and brightness levels in images.
- Brightness Augmentation: Brightness augmentation is a technique used to modify the overall brightness level within an image by applying random changes to pixel brightness values, resulting in either increased or decreased light intensity throughout the image. This adjustment creates diverse lighting variations for objects and backgrounds in the image. The primary aim of brightness augmentation is to enhance machine learning models' adaptability to different lighting conditions encountered in real-world scenarios. However, excessive changes in brightness may lead to the loss of critical image information, necessitating careful adjustments to maintain visual balance. Additionally, local brightness augmentation techniques target specific image areas, offering more precise adjustments. By combining brightness augmentation with other image data augmentation methods, such as contrast enhancement or cropping, a more diverse training dataset can be generated, aiding models in recognizing various lighting conditions.
- Mosaic Augmentation: Introduced in the study [27], this process is a widely used technique in image processing and object detection. It aims to enhance the reliability and robustness of models by combining multiple images into a single mosaic, treating it as a unified image. This method increases the diversity of perspective and scale, thus enriching the model's learning by expanding the variety of training samples. It improves object detection across different scales and viewpoints, as well as enhances the model's ability to handle challenging images, such as those with partially visible or cropped objects. Several studies have utilized mosaic augmentation for object detection, including [28], where it was employed to broaden the variety of training samples and explore the relationship between classification and localization in object detection.

C. Building Models

This study will employ two approaches to construct the best model. The first approach utilizes Mask R-CNN to detect the location of damage, after which the output images from Mask R-CNN will be fed into CNN algorithms with ResNet and MobileNet architectures for the classification of damage types. This initial approach, within the scope of object detection, can be termed as a double-stage method because the tasks of detecting damage location and classifying damage types are performed in two distinct stages. The second approach employs the YOLO algorithm, specifically YOLO-v5 and YOLO-v8 in our research, to simultaneously perform the tasks of detecting damage location and classifying damage types. This second approach, within the scope of object detection, can be referred to as a one-stage method because the detection of damage location and the classification of damage types are carried out in a single stage.

For Mask R-CNN, firstly, an input image is fed into the network. This image undergoes processing in the backbone network, which is a combination of the ResNet101 architecture and is complemented by the Feature Pyramid Network (FPN). The backbone network aims to extract feature maps from the input image. Secondly, the feature maps generated by the backbone network are sent to the Region Proposal Network (RPN). In this process, several operations are performed, including sliding window operation, convolution operation, softmax operation, and bounding box regression (bbox regression). The sliding window operation entails a fixed 3×3 window moving across the feature map to identify candidate regions. At each window position, the convolution operation is utilized to produce scores for each anchor and proposal for bounding box regression. The convolution operation applies a kernel to the feature map by sliding it across and computing the dot product between the kernel values and the corresponding feature map values. Mathematically, the convolution operation formula in the RPN network can be observed in Eq. (1).

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n) \cdot K(m,n) \quad (1)$$

Next, the softmax operation is performed for the classification of two classes (object vs. non-object). In this study, objects refer to damaged parts of the car body, while non-objects are defined as undamaged parts of the car body. For each anchor, the Region Proposal Network (RPN) generates two scores indicating the likelihood of the anchor containing foreground (object) or background (non-object). The softmax function is applied to these scores to obtain the probability of a desired object using the formula in Eq. (2).

$$\sigma(z)_i = \frac{e^{z_i}}{e^{z_0} + e^{z_1}} \quad (2)$$

Furthermore, the bounding box regression (bbox reg) operation is conducted to refine the bounding box proposals. For each anchor, the Region Proposal Network (RPN) generates four values representing predictions of displacement and scale transformation to adjust the anchor closer to the actual object bounding box. The general formula for bbox reg in the RPN context can be observed in Eq. (3) to Eq. (6).

$$t_x = \frac{(x - x_a)}{w_a} \quad (3)$$

$$t_y = \frac{(y - y_a)}{h_a} \quad (4)$$

$$t_w = \log\left(\frac{w}{w_a}\right) \quad (5)$$

$$t_h = \log\left(\frac{h}{h_a}\right) \quad (6)$$

$$L = L_{cls} + L_{box} + L_{mask} \quad (7)$$

This second process yields outputs in the form of Regions of Interest (RoIs). Subsequently, the RoIs generated by the RPN are mapped to extract corresponding target features in the shared feature map, and then forwarded to the Fully Connected Layers and Fully Convolutional Network (FCN), respectively for target classification and instance segmentation. This process generates classification scores, bounding boxes, and segmentation areas. To evaluate these three aspects, Equation (7) is utilized, where L represents the total loss function, L_{cls} is the loss function for object classification, L_{box} is the loss function for bounding box regression, and L_{mask} is the loss function for object segmentation. Furthermore, the output from the Mask R-CNN algorithm, which consists of segmented images and information regarding the coordinates of segmentation bounding boxes, will be used as input for the subsequent algorithm, which is CNN for classifying the type of damage. Initially, only the parts of the image marked with bounding box coordinates, indicating objects, are selected. The objects identified using Mask R-CNN will be used as input for the damage classification algorithm in this study, employing CNN with ResNet-50 and MobileNet-V2 architectures.

Next, the classification process will be explained by constructing the best model using the ResNet-50 architecture. Initially, the input image is resized to 224×224. Then, the image enters the first convolutional layer consisting of 7×7 filters with an output of 64 feature maps and a smaller dimension image of 112×112. In each filter window, convolution operations, as in Eq. (8), are performed, followed by batch normalization using Eq. (9) and Eq. (10), and finally passed through the ReLU activation function using the formula in Eq. (11).

$$Output = W * X + b \quad (8)$$

$$Output\ normalized = \frac{Output - mean(Output)}{\sqrt{var(Output) + \epsilon}} \quad (9)$$

$$Output\_bn = \gamma \times Output\ normalized + \beta \quad (10)$$

$$ReLU(Output\_bn) = \max(0, Output\_bn) \quad (11)$$

Next, the results from the operations in the first convolutional layer enter the second convolutional layer block. This block consists of three convolutional layers. The first layer comprises 1×1 filters with 64 filters, the second layer consists of 3×3 filters with 64 filters, and the third layer consists of 1×1 filters with 256 filters. Convolution operations, as in Eq. (8), are performed in each filter window, followed by batch normalization using Eq. (9) and Eq. (10), and finally passed through the ReLU activation function using the formula in Eq. (11). This block will be executed three times in total. The output of this block is processed in the third convolutional block.

In the third convolutional block, there are three convolutional layers. The first layer comprises 1×1 filters with 128 filters, the second layer consists of 3×3 filters with 128 filters, and the third layer consists of 1×1 filters with 512 filters. Similar to before, convolution operations, batch normalization, and ReLU activation functions are applied. This block will be executed four times in total.

Next, the output from the third convolutional block enters the fourth convolutional block. This block also consists of three convolutional layers. The first layer comprises 1×1 filters with 256 filters, the second layer consists of 3×3 filters with 256 filters, and the third layer consists of 1×1 filters with 1024

filters. Similar operations are applied in each filter window, and this block will be executed six times in total.

Lastly, there is the fifth convolutional block, which also consists of three convolutional layers. The first layer comprises 1×1 filters with 512 filters, the second layer consists of 3×3 filters with 512 filters, and the third layer consists of 1×1 filters with 2048 filters. Operations similar to previous blocks are applied, and this block will be executed three times in total.

In addition to using the ResNet-50 architecture, this study will also employ the MobileNet-v2 architecture for the CNN classification algorithm with the aim of classifying the types of damage based on the predicted damage areas obtained using the Mask R-CNN algorithm. First, an image of size 224×224 with three channels will be used as input. Then, standard convolution operations will be performed with a filter size of 3×3 and a formula as shown in Eq. (12).

$$O_{ij} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I_{(i+m,j+n)} \cdot K_{mn} \tag{12}$$

Next, it will enter the core part that characterizes the MobileNet-v2 architecture, namely depthwise separable convolution, which consists of two convolutional layers: depthwise convolution and pointwise convolution. In depthwise convolution, a single filter is applied to each feature map separately. This means that if there are D feature maps, there will be D depthwise filters. The formula for the depthwise convolution operation can be seen in Eq. (13).

$$O_{i,j,d} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I_{(i+m,j+n,d)} \cdot K_{m,n,d} \tag{13}$$

After that, the 32 output feature maps from the depthwise convolution will enter the pointwise convolution layer. In pointwise convolution, the operation is performed using 1×1 pixel filters on all the output feature maps from the depthwise convolution. The formula for the pointwise convolution operation can be seen in Eq. (14).

$$O_{i,j,l} = \sum_{d=1}^{D} I_{x,y,d} \cdot K_{d,l} \tag{14}$$

In this process, the output is 64 feature maps with a size of 112×112 pixels. These three convolution processes will be repeated 4 times until an output of 1024 feature maps with a size of 7×7 is obtained. Then, global average pooling operation will be performed with the formula as shown in Eq. (15) below.

$$O_k = \frac{1}{WH} \sum_{i=1}^{H} \sum_{j=1}^{W} I_{i,j,k} \tag{15}$$

As a result, the overall outcome of global average pooling will yield a vector with a dimension size of 1024. Next, this vector will be used as input to the final layer, namely the fully connected layer, with the formula as shown in Eq. (16).

$$\boldsymbol{O_j} = activation\left(\sum_{i=1}^{N} \boldsymbol{W_{i,j}} \cdot \boldsymbol{X_i} + \boldsymbol{b_j}\right) \tag{17}$$

For each layer, an activation function is employed to process the weighted calculations and generate the output of the best model. Typically, the ReLU activation function is used during the process, with the formula as shown in Eq. (18), while the sigmoid activation function is utilized as the activation function in the final layer to perform the classification task, with the formula as depicted in Eq. (19).

$$ReLU(x) = \max(0, x) \tag{18}$$

$$\sigma(\boldsymbol{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{19}$$

In this study, detection and classification of car body damage will also be performed using the YOLO algorithm. YOLO is an algorithm capable of simultaneously performing detection and classification functions. Therefore, the YOLO algorithm approach can be referred to as a case of object detection with a one-stage method. In our study, two versions of the YOLO algorithm will be used, namely YOLO-v5 and YOLO-v8. The reason for choosing these two versions is that both were developed by the same team, Ultralytics. YOLO-v5 was released in May 2020 and has been widely used in various fields for object detection cases. It will then be compared with YOLO-v8, which is one of the latest versions of YOLO released in January 2023. First, the implementation of YOLO-v5 in this study will be explained.

YOLO-v5 consists of several sub-versions, namely YOLO-v5n, YOLO-v5s, YOLO-v5m, YOLO-v5l, and YOLO-v5x. All sub-versions have the same input size, which is an image with dimensions of 640×640 pixels. Each version has a similar overall architecture, with differences only in the number of layers and parameters used. For example, YOLO-v5n has the fewest layers and parameters, resulting in the fastest computation time compared to all other sub-versions, but its accuracy is relatively lower compared to the other four sub-versions. Typically, the choice of sub-version is based on the complexity of the case or problem and the available resources. The performance of these five sub-versions will be compared in this study. Fig. 3 shows YOLO-v5 architecture.

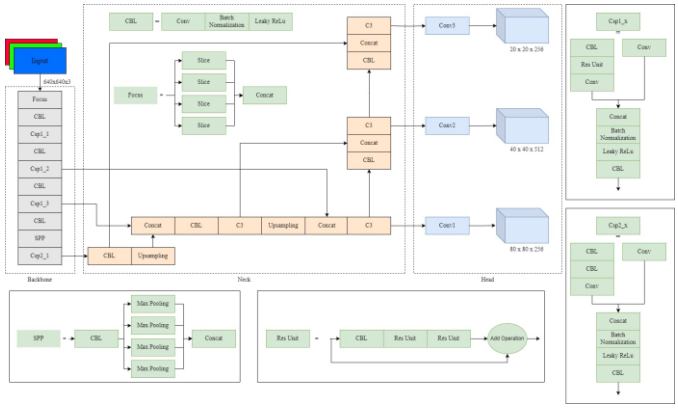

Fig. 3. YOLO-v5 architecture.

The YOLO architecture is divided into three main parts: backbone, neck, and head. In the backbone part, the first step involves the input image with dimensions of 640×640×3 pixels entering the focus layer. In this layer, the image is divided into four equally sized parts and combined into a new image tensor with dimensions of 320×320×12. The purpose of this operation is to reduce the spatial dimensions of the image and focus more on the important features within it. The output of this layer then enters the subsequent CBL layer.

In the next process in the CBL layer, three main operations are performed: convolution operation, batch normalization, and Leaky ReLU. The convolution operation generates new feature maps using a 3×3 filter applied to the entire spatial area of the input feature map. This operation is used to extract important and meaningful features from the image. Next, batch normalization is applied to normalize the output of the convolution layer and speed up convergence towards the best model. Finally, the Leaky ReLU operation is used to prevent dead neurons due to negative input values. The formulas for each operation can be seen respectively in Eq. (20) – Eq. (22).

$$O_{ijk} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \sum_{c=0}^{C-1} I_{i+m,j+n,c} \cdot K_{mnck} \tag{20}$$

$$O'_k = \gamma \left(\frac{O_k - \mu_b}{\sqrt{\sigma_B^2 + \epsilon}} \right) + \beta \tag{21}$$

$$f(x) = \begin{cases} x, & jika\ x > 0 \\ \alpha x, & jika\ x \leq 0 \end{cases} \tag{22}$$

Then, there are several layers that need to be passed through such as Residual Unit (Res unit), CSP bottlenecks, and Spatial Pyramid Pooling (SPP). In the Res Unit, the concept from the ResNet architecture is used where the input feature map can perform skip connections to convolutional layers, and then their results are combined with the output of those convolutional layers. This is done to preserve some important original features in the image. Then, in the CSP bottlenecks, there are two types of operations: $csp1_x$ and $csp2_x$. Finally, in the SPP operation, max pooling is performed in parallel on the output from the CBL with varying window pooling scale intensities. Then, the results of each max pooling operation are combined again to deepen the features.

In the neck part, there are two operations that have not been performed in the backbone process, namely upsampling, concatenation, and C3. In this part, the model focuses more on processing the feature maps generated in the previous backbone part, before continuing to the head part for the prediction process. Upsampling is the process of increasing the resolution of the feature map. This is usually done through techniques such as nearest neighbor or bilinear interpolation. To perform an upsampling operation, given a feature map with size $W \times H$, upsampling with a scale factor s will result in a new feature map with size $sW \times sH$. This operation is carried out to increase the resolution of the feature map to match the dimension of the feature map that will undergo concatenation operation. Concatenation operation is the process of appending one or more tensors along a certain dimension. In this case, feature maps of various resolutions that have been upsampled will be concatenated along the channel dimension. For example, if there are two feature maps A and B with sizes $W \times H \times C_A$ and $W \times H \times C_B$ respectively, then the result of the concatenation operation will yield a feature map with size $W \times H \times (C_A + C_B)$.

After the upsampling and concatenation operations, there is another block operation called C3. The C3 block aims to further process the concatenated feature maps, using the bottleneck CSP technique for computational efficiency while still extracting relevant features. In this block, there are two main operations: convolution operation and bottleneck CSP operation. The convolution operation consists of three layers followed by batch normalization and Leaky ReLU operations as shown in Eq. (20) – Eq. (22). After passing through the convolution layers, the feature map will enter the bottleneck CSP operation. The C3 block is responsible for processing the combined features and generating richer feature maps that will be used for prediction.

Finally, in the head part, the output from the first C3 block will enter the last convolution layer for the bounding boxes prediction process, the output from the second C3 block will enter the last convolution layer for predicting the confidence or probability of the damage being detected inside the bounding boxes or not, and the output from the last C3 block will enter the last convolution layer and used for classifying the type of damage. These three outputs will be evaluated to minimize the loss using Eq. (23) – Eq. (27).

$$Loss = \lambda_1 L_{cls} + \lambda_2 L_{obj} + \lambda_3 L_{loc} \tag{23}$$

$$L_{cls} = L_{obj} = -\frac{1}{N}(y_n \times ln_{x_n} + (1 - y_n) \times \ln(1 - x_n)) \tag{24}$$

$$L_{loc} = 1 - IOU + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v \tag{25}$$

$$v = \frac{4}{\pi^2}(\arctan^2 \frac{w^{gt}}{h^{gt}} - arctan \frac{w}{h})^2 \tag{26}$$

$$\alpha = \frac{v}{(a - IOU) + v} \tag{27}$$

In Eq. (24), N represents the number of classification classes, y_n represents the ground truth for the presence of each class within the bounding boxes (1 if class n of damage is within the bounding boxes and 0 if class n of damage is not within the bounding boxes), and x_n represents the predicted probability for that class.

Next, the YOLO-v8 model will be explained. This study compares several sub-versions of YOLO-v8, specifically YOLO-v8n, YOLO-v8s, YOLO-v8m, YOLO-v8l, and YOLO-v8x. Similar to YOLO-v5, all sub-versions of YOLO-v8 use input images of size 640×640. YOLO-v8n is the version with the fastest training time as it uses the fewest parameters and layers. On the other hand, YOLO-v8x is the sub-version with an average longest training time but is expected to yield the most accurate results due to the higher number of parameters and layers used. In this study, all the aforementioned sub-versions of YOLO-v8 are compared.

First, an image with dimensions of 640×640×3 pixels is taken as input into the algorithm. Then, the image is processed

within the Convolutional Block (CBS Module). In this CBS block, several operations are performed, including convolution operation, Batch Normalization, and Sigmoid activation function. The formulas for performing these three operations can be seen in Eq. (28) – Eq. (30).

$$O_{ijk} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \sum_{c=0}^{C-1} I_{i+m,j+n,c} \cdot K_{mnck} \quad (28)$$

$$O'_k = \gamma \left(\frac{O_k - \mu_b}{\sqrt{\sigma_B^2 + \epsilon}} \right) + \beta \quad (29)$$

$$\sigma(O'_k) = \frac{1}{1 + e^{(-O'_k)}} \quad (30)$$

The CBS block has the main function of extracting feature maps and reducing spatial dimensions. In addition, there are other operation blocks, namely C2F and SPPF. In the C2F operation block, the feature map is divided into n parts, where n represents the number of bottleneck layers. Then, each layer undergoes two operations as described in Eq. (28) – Eq. (30). For the SPPF operation block, there is a uniqueness where the feature map is first divided into four parts, and each part undergoes processing as shown in Fig. 4. Furthermore, the neck and head parts are not much different from those in YOLO-v5. All operations performed are almost the same, with only differences in the order of operation processes and the size of the feature map in the final detection block.

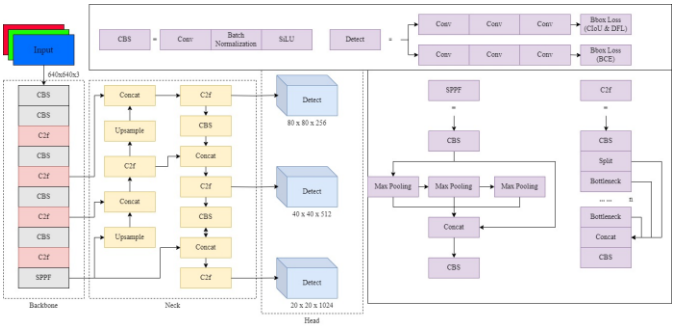

Fig. 4. YOLO-v8 architecture.

IV. Experiments and Result

A. The Evaluation Parameters

Several evaluation criteria were used in this study, including the F1-score, precision, and recall. A prominent tool for evaluating the effectiveness of classification models is the confusion matrix, which offers a thorough comparison between the predictions made by the model and the actual labels. False Positives (FP), False Negatives (FN), True Positives (TP), and True Negatives (TN) are all part of it. Precision is a metric that measures the proportion of true positive predictions compared to the total positive predictions. Precision provides information on how many of the positive predictions are correct out of the total positive predictions, assisting in identifying the rate of false positive errors. Recall computes the percentage of successfully predicted positive instances among all actual positive instances, whereas precision quantifies the percentage of correctly predicted positive instances among all cases projected as positive. Recall provides information on the extent to which a model can detect actual positive instances, aiding in identifying the rate of false negative errors. The F1-score provides a fair evaluation of the model's performance since it is a harmonic mean of precision and recall. This provides a balanced measure of the model's ability to identify positive instances with minimal false positive and false negative errors. The given formulas can be used to calculate these measures.

$$Precision = \frac{TP}{TP + FP} \quad (31)$$

$$Recall = \frac{TP}{TP + FN} \quad (32)$$

$$F1 - Score = \frac{2 \times (presisi \times recall)}{presisi + recall} \quad (33)$$

B. Experimental Analysis

In this research endeavor, to conduct simulations and model development aimed at detecting damages on automobile bodies, as well as to create a digital image-based system for detecting such damages, the authors employed machinery and software characterized by specifications delineated in the ensuing table. For comprehensive details regarding the specifications of the equipment utilised by the authors, kindly consult Table II.

TABLE II. Device Specification

Specifications	
GPU	NVIDIA RTX A4000
GPU Memory	16 GB
RAM	16 GB
Disk	1 TB
Programming Language	Phyton 3.10

The authors set the number of epochs to 150 iterations, the learning rate to 0.01, and the image sizes to 1024×1024×3 for Mask R-CNN, 224×224×3 pixels for ResNet-50 and MobileNet-v2, and 640×640×3 pixels for YOLO-v5 and YOLO-v8. Additionally, a batch size of 16 was assigned to each model employed in this study.

According to Table III, it is evident that the best results were achieved by the YOLO-v8x model, with precision, recall, and F1-score values of 0.963, 0.951, and 0.936, respectively. Training the YOLO-v8x model to its optimal performance required 3 hours and 48 minutes. Additionally, the model with the shortest training time was YOLO-v8n, which trained in 2 hours and 10 minutes, achieving precision, recall, and F1-score values of 0.867, 0.821, and 0.814. YOLO-v8n demonstrates superior performance compared to the Mask R-CNN + MobileNet-v2 model, despite the significant difference in training duration. However, the Mask R-CNN + ResNet-50 model still outperforms several sub-versions of the YOLO-v5 and YOLO-v8 models. Furthermore, the one-stage object detection method which is YOLO-v8x has better performance than the two-stage method in this research.

Additionally, Fig. 5 and Fig. 6 illustrate a comparison between the manually labeled images and the model's predicted

results. Observations indicate that the model generally performs well in detecting damages to the vehicle's body.

TABLE III. MODELS PERFORMANCE

Model		Evaluation Parameters			Training Time
		Precision	*Recall*	*F1-Score*	
Mask R-CNN + ResNet-50		0.908	0.885	0.860	4 hours 35 minutes
Mask R-CNN + MobileNet-v2		0.866	0.842	0.815	3 hours 20 minutes
YOLO-v5	YOLO-v5n	0.841	0.813	0.820	2 hours 23 minutes
	YOLO-v5s	0.849	0.842	0.842	2 hours 48 minutes
	YOLO-v5m	0.893	0.878	0.875	3 hours 9 minutes
	YOLO-v5l	0.918	0.891	0.893	3 hours 40 minutes
	YOLO-v5x	0.922	0.903	0.899	4 hours 32 minutes
YOLO-v8	YOLO-v8n	0.867	0.821	0.814	2 hours 10 minutes
	YOLO-v8s	0.892	0.842	0.839	2 hours 24 minutes
	YOLO-v8m	0.924	0.892	0.893	2 hours 58 minutes
	YOLO-v8l	0.954	0.914	0.909	3 hours 21 minutes
	YOLO-v8x	**0.963**	**0.951**	**0.936**	3 hours 48 minutes

Fig. 5. Manually labelled images.

Fig. 6. Predicted result of YOLO-v8x model.

V. CONCLUSION AND FUTURE WORK

In this research, the best results were obtained using the YOLO-v8x model. This can be explained because YOLO-v8 is one of the newest versions of YOLO, which is stable and was released in early 2023. Furthermore, YOLO-v8 is also a development of YOLO-v5, which results in even better model performance. Further research is needed by trying various batch sizes, epochs, increasing the amount of data, trying various other data augmentation combinations, and using the latest YOLO model, namely YOLO-v9, which was only released in early 2024.

ACKNOWLEDGMENT

This Research is funded by FMIPA UI 2023 Research Grant, number: PKS-055/UN2.F3.D/PPM.00.02/2023 and supported by Data Science Center FMIPA UI, Laboratory of Advance Computing, and PT Global Risk Management.

REFERENCES

[1] "Badan Pusat Statistik." Accessed: May 12, 2023. [Online]. Available: https://www.bps.go.id/indikator/indikator/view_data_pub/0000/api_pub/V2w4dFkwdFNLNU5mSE95Und2UDRMQT09/da_10/1

[2] "Statistik Laka Lantas | Pusiknas Bareskrim Polri." Accessed: May 04, 2023. [Online]. Available: https://pusiknas.polri.go.id/laka_lantas

[3] World Health Organization, "Road traffic injuries." Accessed: May 15, 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries

[4] Analisa Industri Asuransi & Reasuransi, "Analisa Industri Asuransi & Reasuransi - AAUI." Accessed: May 04, 2023. [Online]. Available: https://aaui.or.id/analisa-industri-asuransi-reasuransi/

[5] F. Efendi, "Analysis Of Insurance Customer Satisfaction In The Process (Claim) of Vehicle Damage To The Quality Of Services At PT Multi Artha Guna Insurance, Tbk Batam," May 2019. doi: 10.2991/icaess-19.2019.18.

[6] V. A. Zeithaml, M. J. Bitner, and D. D. Gremler, Services marketing : integrating customer focus across the firm, 7th ed. McGraw Hill, 2018.

[7] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.

[8] A. Bustamam, D. Sarwinda, R. H. Paradisa, A. A. Victor, A. R. Yudantha, and T. Siswantining, "Evaluation of convolutional neural network variants for diagnosis of diabetic retinopathy," Communications in Mathematical Biology and Neuroscience, 2021, doi: 10.28919/cmbn/5660.

[9] A. Salma, A. Bustamam, A. Yudantha, A. Victor, and W. Mangunwardoyo, "Artificial Intelligence Approach in Multiclass Diabetic Retinopathy Detection Using Convolutional Neural Network and Attention Mechanism," International Journal of Advances in Soft Computing and its Applications, vol. 13, no. 3, pp. 101–114, Dec. 2021, doi: 10.15849/IJASCA.211128.08.

[10] Z.-Q. Zhao, P. Zheng, S. Xu, and X. Wu, "Object Detection with Deep Learning: A Review," Jul. 2018, [Online]. Available: http://arxiv.org/abs/1807.05511

[11] S. Pathari, S. M, S. A, S. Kumar, and K. Devaki, "Assessing Car Damage using Mask R-CNN," Nov. 2020.

[12] 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, doi: 10.1109/ACCESS.2020.2964055.

[13] D. Widjojo, E. Setyati, and Y. Kristian, "Integrated Deep Learning System for Car Damage Detection and Classification Using Deep Transfer Learning," in 2022 IEEE 8th Information Technology International Seminar (ITIS), IEEE, Oct. 2022, pp. 21–26. doi: 10.1109/ITIS57155.2022.10010292.

[14] M. Dwivedi et al., "Deep Learning-Based Car Damage Classification and Detection," 2021, pp. 207–221. doi: 10.1007/978-981-15-3514-7_18.

[15] K. Patil, M. Kulkarni, A. Sriraman, and S. Karande, "Deep Learning Based Car Damage Classification," in 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, Dec. 2017, pp. 50–54. doi: 10.1109/ICMLA.2017.0-179.

[16] Q. Li, F. Zhao, Z. Xu, J. Wang, K. Liu, and L. Qin, "Insulator and Damage Detection and Location Based on YOLOv5," in 2022 International Conference on Power Energy Systems and Applications (ICoPESA), IEEE, Feb. 2022, pp. 17–24. doi: 10.1109/ICoPESA54515.2022.9754476.

[17] G. Lu and Y. Wang, "The Improved YOLO-V5 Based Automatic Non-parking Overloading Detection Method," in 2022 5th International Symposium on Autonomous Systems (ISAS), IEEE, Apr. 2022, pp. 1–6. doi: 10.1109/ISAS55863.2022.9757325.

[18] Y. Xiao, A. Chang, Y. Wang, Y. Huang, J. Yu, and L. Huo, "Real-time Object Detection for Substation Security Early-warning with Deep Neural Network based on YOLO-V5," in 2022 IEEE IAS Global Conference on Emerging Technologies (GlobConET), IEEE, May 2022, pp. 45–50. doi: 10.1109/GlobConET53749.2022.9872338.

[19] S. Wang et al., "An Ensemble Learning Approach with Multi-depth Attention Mechanism for Road Damage Detection," in 2022 IEEE International Conference on Big Data (Big Data), IEEE, Dec. 2022, pp. 6439–6444. doi: 10.1109/BigData55660.2022.10021018.

[20] H. A. Setyawan, A. Bustamam, and R. Anwar, "Detection and Assessment of Damaged Objects on the Car Body Based on YOLO-V5," in 2023 3rd International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS), IEEE, Aug. 2023, pp. 508–513. doi: 10.1109/ICE3IS59323.2023.10335327.

[21] "Home - Ultralytics YOLOv8 Docs." Accessed: May 24, 2023. [Online]. Available: https://docs.ultralytics.com/

[22] J. S. Thomas, S. Ejaz, Z. Ahmed, and S. Hans, "Optimized Car Damaged Detection using CNN and Object Detection Model," in 2023 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), IEEE, Mar. 2023, pp. 172–174. doi: 10.1109/ICCIKE58312.2023.10131804.

[23] K. Meenakshi and S. Sivasubramanian, "An Intelligent System to Assess the Exterior Vehicular Damage based on DCNN," in 2023 International Conference on Computer Communication and Informatics (ICCCI), IEEE, Jan. 2023, pp. 1–5. doi: 10.1109/ICCCI56745.2023.10128579.

[24] A. Shirode, T. Rathod, P. Wanjari, and A. Halbe, "Car Damage Detection and Assessment Using CNN," in 2022 IEEE Delhi Section Conference (DELCON), IEEE, Feb. 2022, pp. 1–5. doi: 10.1109/DELCON54057.2022.9752971.

[25] H. Bandi, S. Joshi, S. Bhagat, and A. Deshpande, "Assessing Car Damage with Convolutional Neural Networks," in 2021 International Conference on Communication information and Computing Technology (ICCICT), IEEE, Jun. 2021, pp. 1–5. doi: 10.1109/ICCICT50803.2021.9510069.

[26] U. Waqas, N. Akram, S. Kim, D. Lee, and J. Jeon, "Vehicle Damage Classification and Fraudulent Image Detection Including Moiré Effect Using Deep Learning," in 2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), IEEE, Aug. 2020, pp. 1–5. doi: 10.1109/CCECE47787.2020.9255806.

[27] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," Apr. 2020.

[28] Y. Wu et al., "Rethinking Classification and Localization for Object Detection," Apr. 2019.