T.C.

MARMARA UNIVERSITY

FACULTY of ENGINEERING

COMPUTER ENGINEERING DEPARTMENT

CSE4197 Engineering Project I

PROJECT SPECIFICATION DOCUMENT

Title of the Project

DAMAGEWIZ: COST CALCULATION OF CARS DAMAGED USING ARTIFICIAL INTELLIGENCE

December 1, 2022

Group Members

150120531 Hasan ŞENYURT

150119761 Cem ANARAL

150119727 Melisa DURMUŞ

Supervised by

Assoc. Prof. Mustafa AĞAOĞLU

TABLE OF CONTENTS

1. PROBLEM STATEMENT

Traffic accidents are such a big problem for people in the modern world. Damage assessment of the car that is involved in the accident is the problem that our project focuses on. Damage assessment of the car takes lots of time because the car may be taken by tow truck to an expert. If this is not the case, the owner of the car has to take the car to an expert somehow to predict the cost of damaged parts. These options are waste of time, expensive and prone to human error.

In addition, some mechanics and experts may be unreliable in terms of cost estimation of the damage detection. They may overprice during the assessment of the damage or try to deceive the accident victim to repair undamaged parts of the car. Also, people who are involved in traffic accidents want to search the closest mechanic to purchase parts of a car or call them to ask for help, and search for the cheapest car parts to change with damaged ones.

2. PROBLEM DESCRIPTION AND MOTIVATION

People go to mechanics when they have an accident or their car is damaged, but the damage assessment process progresses very slowly. The driver who is involved in an accident loses lots of time during damage assessment of exterior parts of his/her car in expert investigation. We aim to solve this problem with the help of photographs of damaged car with the application we will develop.

Another problem besides losing time during damage assessment is the cost of prediction of damaged parts. Different prices for the same parts of a car exist in different mechanic shops. So, drivers cannot find the cheapest or most suitable parts for their car easily. How much the car parts cost in the mechanics will be shown to the user in our project. In this way, the accident victim will be ready for what he/she will encounter when he/she goes to the mechanic.

In order not to have trouble for finding a good and qualified mechanic, some features will be added to the application. For example, the user will be able to rate a mechanic he/she has visited before. Users who are not satisfied with the service will give bad ratings, while those who are satisfied will give good ratings. The app will show nearby mechanics using GPS also.

Importance of this project can be explained shortly: making the process faster. In cities such as Istanbul, traffic is a huge problem itself. Drivers complain about the process which will be going on after an incident that they are involved in. With our application, things will be faster when drivers learn the cost of damaged parts of the car immediately.

In this project, we will solve the problem of the cost calculation of cars that are damaged by using machine learning and image processing. Firstly, we will detect the car brand and model using the photos taken by the accident victim. Secondly, we will detect the place of the damaged part and show the accident victim the percentage of the damage. Rating system will be used to evaluate mechanics. Reliability problem will be solved by using this feature.

3. MAIN GOAL AND OBJECTIVES

Our main goal is to develop an application that will help the accident victim to repair his/her damaged car in the cheapest and fastest way possible by showing them their damaged parts.

We will do the following to fulfill our main aim:

3.1 Developing of an artificial intelligence system to extract information from damaged car photos.

a. Detecting the car brand using damaged car photos.

We will develop an artificial intelligence model to detect the car brand with over 65% accuracy.

b. Detecting the damaged car parts using damaged car photos.

We will develop an artificial intelligence model to detect the damaged car parts with over 70% accuracy.

c. Detecting the percentage of damage from car part photos.

We will develop an artificial intelligence model to detect the percentage of damage of the damaged car part. With this percentage the accident victim will decide to buy a brand-new part or repair the damaged part. Our model will work with over 70% accuracy.

3.2 Developing the DamageWiz Application.

a. Developing a marketplace for car mechanics.

By this marketplace, car mechanics will be able to list their car parts prices with repairment costs. The accident victim will use this marketplace after uploading the photo and decide where to buy the new car parts or completely where to repair the car from.

b. Developing a rating mechanism of car mechanics.

After the interaction between the car mechanic and the accident victim ends, the accident victim will be able to rate the mechanic out of five and optionally write about the repairment process as a whole. This way users of the system will be able to warn each other if a mechanic is dishonest. We will also be able to detect and ban these dishonest mechanics from the system.

c. Developing a filtering mechanism to find the most suitable mechanic.

We will provide several ways of finding the most suitable mechanic for the victim. Accident victims will be able to sort the mechanics in different ways. For instance, the user will be able to sort the mechanics from the cheapest to most expensive, closest to farthest or from highest rated one to lowest rated.

4. RELATED WORK

4.1 Vehicle Brand Detection Using Deep Learning Algorithms

Kunduracı and Kahramanli [1] have proposed a solution for detecting car brands by implementing a classification method based on deep neural networks. They have found out that the Faster R-CNN method (which is based on deep neural networks) works with 67.66% accuracy.

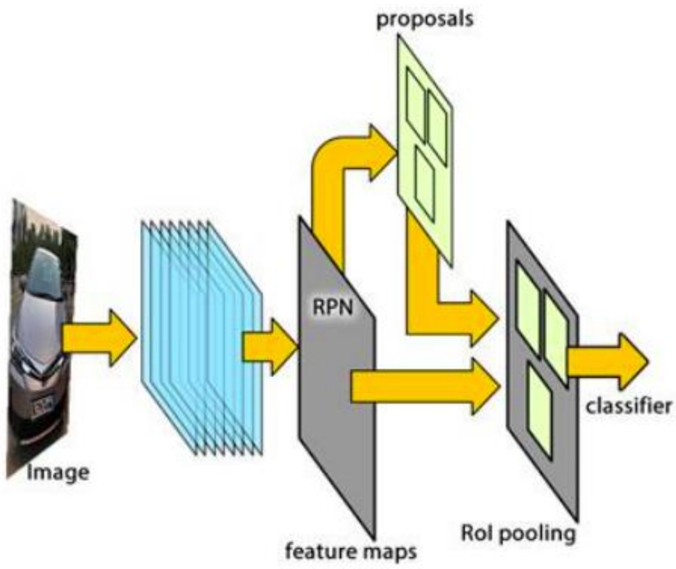

Figure 1: Faster R-CNN architecture [1].

Faster R-CNN is composed of two submodules. First module is a deep fully convolutional network that proposes regions, and the second module uses these regions. Second module is the Faster R-CNN part. Feature network is obtained in the first module which is called RPN. Proposed regions are resized using RoI pooling. The work they have done only detects the brands of the car, but we need to detect both the brand and the model of the damaged car.

4.2 CarNet.AI

CarNet.AI [2] is an enterprise level API that can detect models and brands of the cars, developed by LastStar Ltd. Since it is not an open-source project, we could not find much about how its internals work, but they have provided some little information on their description page.

At first, they detect every car inside of the image and draw a bounding box around them. Next step is to filter out the bounding boxes that are not suitable for detection based on the dimensions of the bounding boxes. Third step is to select a detection strategy. There are three strategies to choose from. They either detect all bounding boxes, the one closest to the center of the image or the largest bounding box only. Fourth step is detecting the subclasses. They process every bounding box selected in the third step and find the classes of them. Only bounding boxes that have “vehicle” class are sent to the last step. In the last step they find the features of the vehicles. These features are car model, brand, color and angle.

Actually CarNet.AI exactly does what we need to do in detecting brand and model phase, but we cannot use it since it is not an open-source project. Also, this project does not detect the damaged car parts or the percentages of the damages.

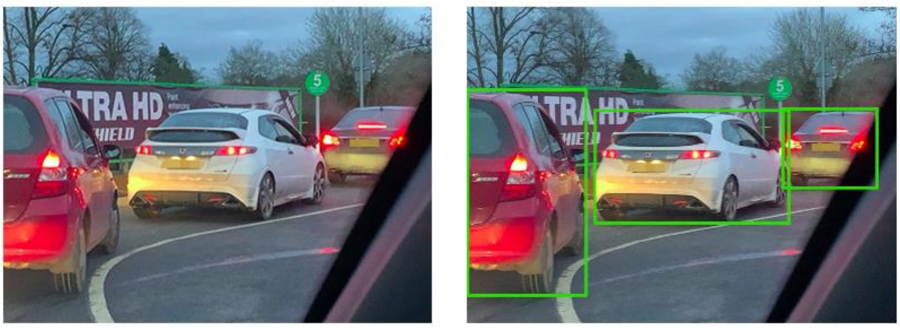

Figure 2: Detecting Bounding Boxes [2].

4.3 Vehicle Make and Model Recognition Using Local Features and Logo Detection

Tafazzoli and Frigiue [3] proposed a two-stage framework to detect the brand and the model of cars. They used Multiple Instance Learning for high accuracy classification. In the first stage, they both detect the brand and the model of the car; in the second stage, they only detect the brand of the car from the logo of the car to improve the accuracy of the first stage.

Multiple Instance Learning is a paradigm for supervised learning that handles ambiguously labeled data in classification and regression issues. Multiple Instance Learning analyzes packets of instances in each packet as opposed to classical supervised learning, which use fixed-length feature vectors as instances.

The work done by Tafazzoli and Frigiue covers what we need to do in the detection of brands and models of the car, not the part where we need to identify the damaged car parts.

4.4 Damage Identification of Selected Car Parts Using Image Classification and Deep Learning

They have proposed [4] a two-level machine learning based system to detect three car parts: front bumper, rear bumper and car wheels. First model is for car part classification, the second one is for damage identification. The accuracy of the first model during training is 94.84%, while accuracy during validation is 81.25%. The second model’s training accuracy is 97.16% and its validation accuracy is 49.28%.

To classify car parts, they used photos from Google as their dataset. They used convolutional neural networks that are made out of 6 layers (excluding poolings) for their models. The 6 layers are the following: Convolution2 - Pooling2 - Convolution2 - Pooling2 - Convolution2 - Pooling2 - Convolution2 - Pooling2 - RELU - Softmax.

Output of the first model is the car part names (Rear Bumper, Front Bumper, Car Wheel). Output of the second model are the damaged and undamaged car part names (Undamaged Front Bumper, Undamaged Rear Bumper, Undamaged Car Wheel, Damaged Front Bumper, Damaged Rear Bumper, Damaged Car Wheel).

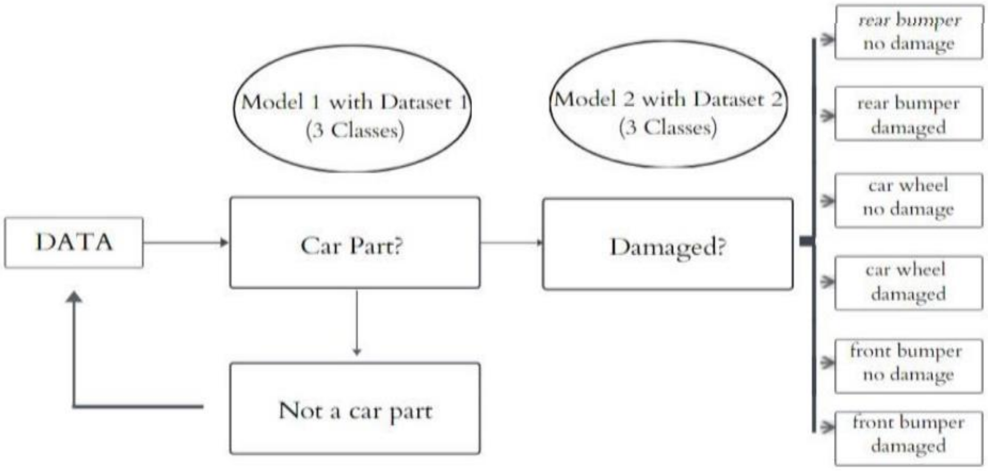

Figure 3: Experimental Flowchart [4].

The work they did covers what we need to do in car part detection, and the damage detection. They did not perform car brand detection, which is needed in our project. Also, the number of car parts they detected are very limited, we aim to detect more car parts.

In contrast, the related works we have found do not fully cover our project, they only cover some part of it. Our project will be a combination of these related works. For better understanding you can look at the chart below.

Name	Brand Detection	Model Detection	Damaged Part Detection	Damage Percentage Detection
4.1 Vehicle Brand Detection Using Deep Learning Algorithms	**Yes**	No	No	No
4.2 CarNet.AI	**Yes**	**Yes**	No	No
4.3 Vehicle Make and Model Recognition Using Local Features and Logo Detection	**Yes**	**Yes**	No	No
4.4 Damage Identification of Selected Car Parts Using Image Classification and Deep Learning	No	No	**Yes**	No
Our Work	**Yes**	**Yes**	**Yes**	**Yes**

Table 1: Related Works Summarized.

5. SCOPE OF THE PROJECT

This project has two important parts to be explained in detail: First aspect is related to drivers (customers). The properties of the project for drivers are detecting car brand and damage assessments with four photographs which are taken by driver after accident or saved photographs in their mobile phones. Photographs must be taken in the form of the left-right side, rear and front of the car. Brand and damage detection will be done with the same photos. Application will try to detect the car brand first. After detection, results will be shown to the driver to control whether the car brand is correct or not. If the car brand is detected correctly by application, damage assessment will be done. Otherwise, drivers must enter the car brand manually. Brand and model are important factors because every part of cars is different according to the brand and model of the car. After brand and damage detection, required parts to be changed will be shown to the driver sorted by cheapest total price of damaged parts. Application will search for the cheapest parts for each part of the damaged car. Total price of damaged parts in mechanic shops will be listed in application. Drivers will see the cheapest total price and cheapest parts one by one in each mechanic shop.

Second aspect of the project is creating a marketplace for mechanic shops. They can add their products with names of the products, prices and descriptions. These mechanic shops will be pinned on the map in application. Closest ones or cheapest ones will be sorted by choosing selection of the sorting in application.

Machine learning and image processing algorithms will be written to detect car brand and damage assessment. These algorithms are for the driver perspective of application. Sorting, GPS and marketplace features will be used from the mechanic shops perspective of application.

Website will be created first for all of these properties, and an Android application will be developed. Android application will be based on a website that is created first. So, web-based android applications will be the last product for both drivers and mechanics shops. Python-Flask API is the core of artificial intelligence part of our application. MVC Dotnet Core 6 is the main development environment for the website of the project.

5.1 Constraints

- Dataset for machine learning and image processing algorithms will be collected from insurance and expert companies. License plates will be censored if there will be a need for it.

- Only automobiles are in the scope of the project. Vehicles except automobiles such as trucks, motorcycles, bikes etc. are not in our concern.
- Data will be labeled manually if there is a lack. Percentage of the damage, damaged parts, brand and model of the car and needed data will be labeled, but our goal is to find a labeled dataset. Trained model's accuracy must be maximized with an accurate dataset.
- Android mobile phones are required to be able to implement our website. These devices must be connected to the internet to access our application. IOS devices are not in our scope.
- Photos of the car must be clear to make a precise detection. We have to deal with garbage data which will come from drivers. Photos will be controlled in our application and needed warnings will be given to drivers.

6. METHODOLOGY AND TECHNICAL APPROACH

Our plan is to combine several Machine Learning and Deep Learning models to achieve our goal. First model which is Faster R-CNN will detect Car Brand and Model, YOLO will detect the damaged parts and DenseNet model of Keras will detect whether the car is damaged or not, percentage of the damaged parts and location of the damaged parts. Detailed information about models and how we will develop our project, and the reasons why we will implement these models are given below.

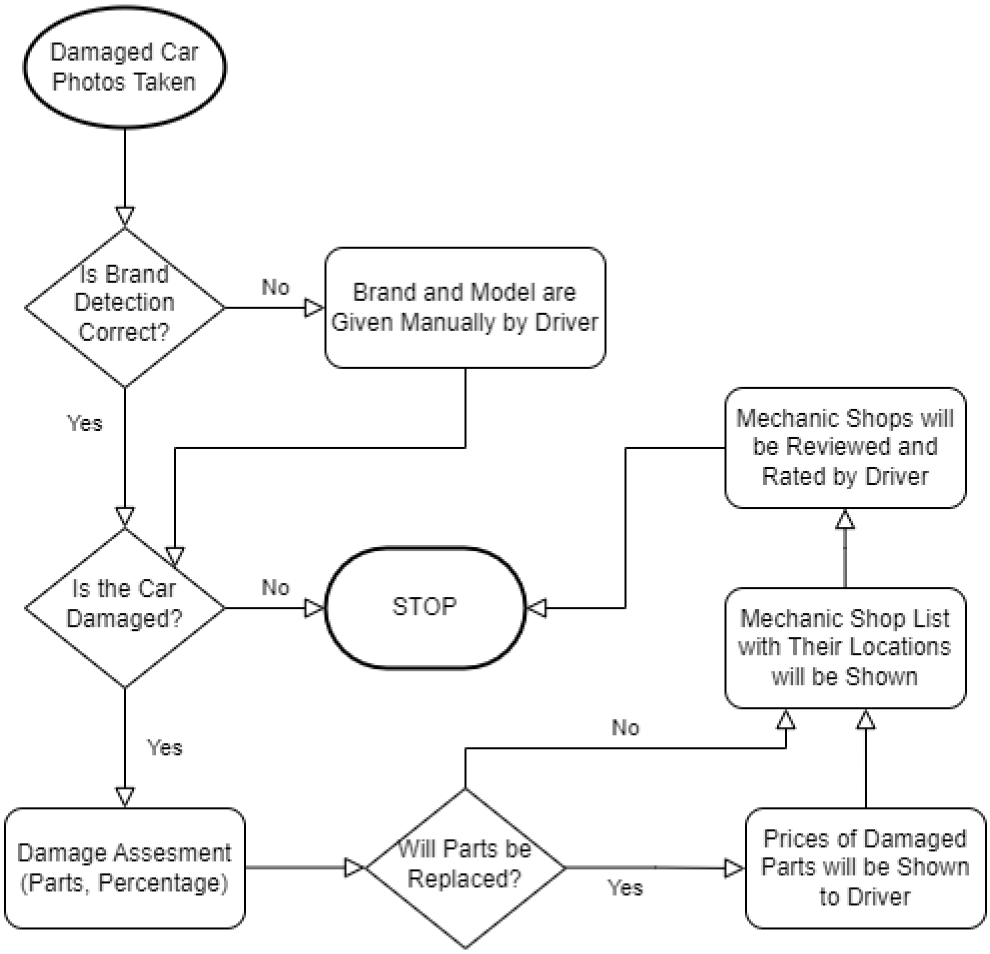

Figure 4: Flowchart of our project.

6.1 Data Collection

We need logo photos of automobile brands. There will be many photos for each photo to train models better. The dataset of logos is available in public websites such as Kaggle and Github. Our plan is to use this data, and search for more data on the internet. In the dataset which is provided on Kaggle has more than 2900 logo photos of the car brands [5]. Also, there are 374 photos of logos on the Github resource [6].

One of the most important keys of our project is a dataset of damaged cars. This dataset must be prepared and labeled carefully. We planned to get photos of damaged cars from insurance companies and public resources on the internet. There are resources on Kaggle similar to the logo of the car brand data. If there is no labeling in the datasets, we will label them properly. One of the datasets from Kaggle has more than 1500 damaged car photos [7]. Another resource from Kaggle has more than 145 damaged car photos [8].

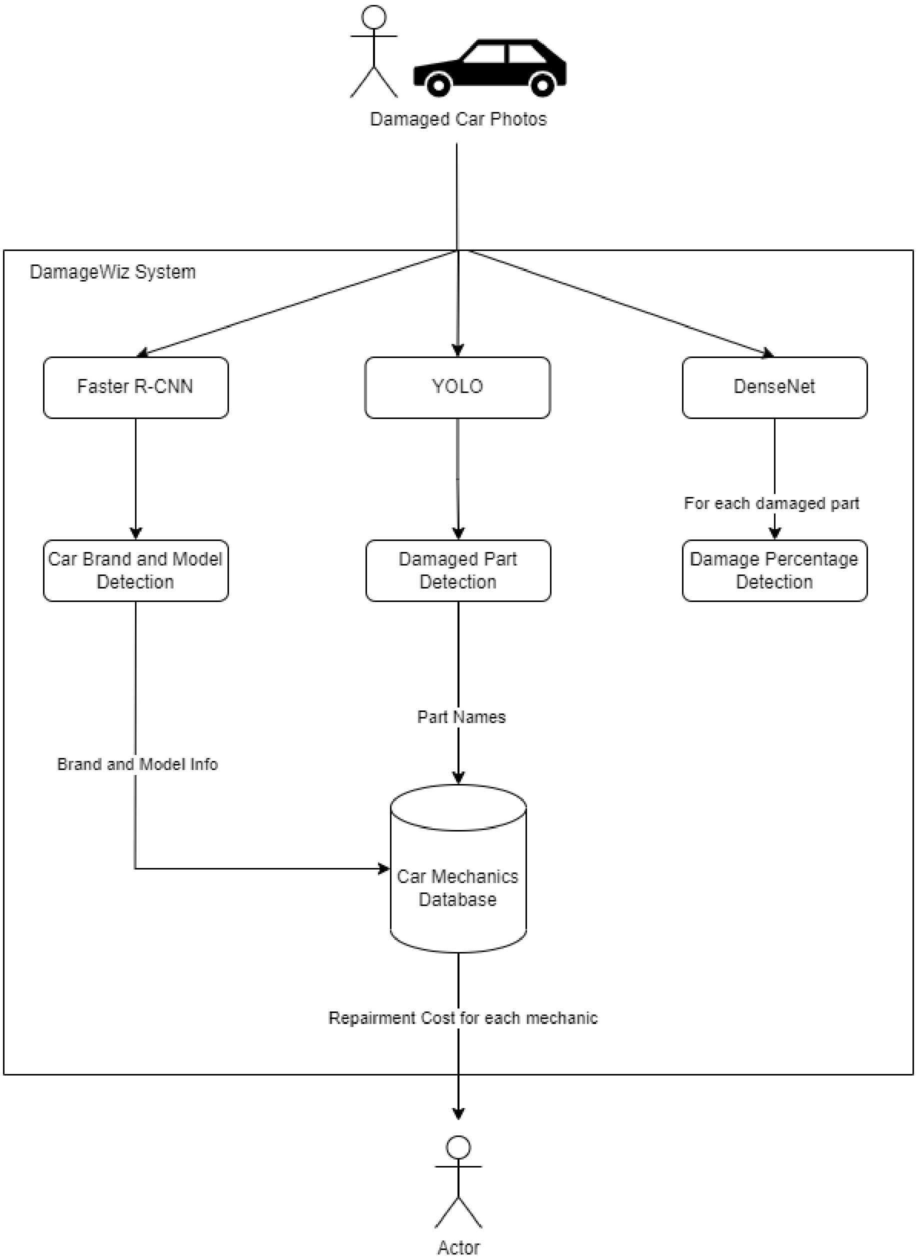

Figure 5: Block Diagram of our project.

6.2 Brand Detection with Faster R-CNN

A popular object identification method known as Faster R-CNN combines the generation of site proposals and their classification into a single pipeline, increasing computational power. For that reason, we will use the Faster R-CNN model to determine the brands and models of the vehicles whose photos are entered. The theoretical information about the Faster R-CNN model is provided below.

Faster R-CNN is an object detection algorithm which is built on top of Fast R-CNN algorithm which is also based on R-CNN algorithm.

R-CNN is an abbreviation of region-based Convolutional Neural Network. It works by extracting region proposals using selective search algorithm. Selective search groups regions hierarchically, based on the pixel intensities. Inventor of the R-CNN algorithm finds about 2k proposals using selective search. After executing the algorithm these region proposals must be labeled. After that, region proposals must be labeled. Region proposals that have more than 0.5 intersection-over union (IOU) are labeled. Proposals with less than 0.3 IOU are taken as backgrounds. Others are not being used.

$$
\begin{aligned}
t_x &= (G_x - P_x)/P_w \\
t_y &= (G_y - P_y)/P_h \\
t_w &= \log(G_w/P_w) \\
t_h &= \log(G_h/P_h).
\end{aligned}
$$

Figure 6: Intervals of the Bounding Box [9].

The mathematical expressions above denote the intervals of the bounding box. x and y are the centers of the box, w and h are the width and height of the box.

Fast R-CNN is a convolutional neural network algorithm. It uses an image and multiple regions of interests as inputs. These inputs are fed into a convolutional network where each of the regions of interests get pooled to a feature map. After that they get mapped to a feature vector. There are two outputs for every region of interest. First one is softmax probabilities and the second one is the points of the detected image [10].

In Fast R-CNN, instead of using proposals as inputs, the input image is directly given into the CNN, where the feature map is generated. From the feature map the region of proposals is being found. Fast R-CNN is faster than R-CNN because you do not find 2,000 region proposals every time, feature map is generated once per image.

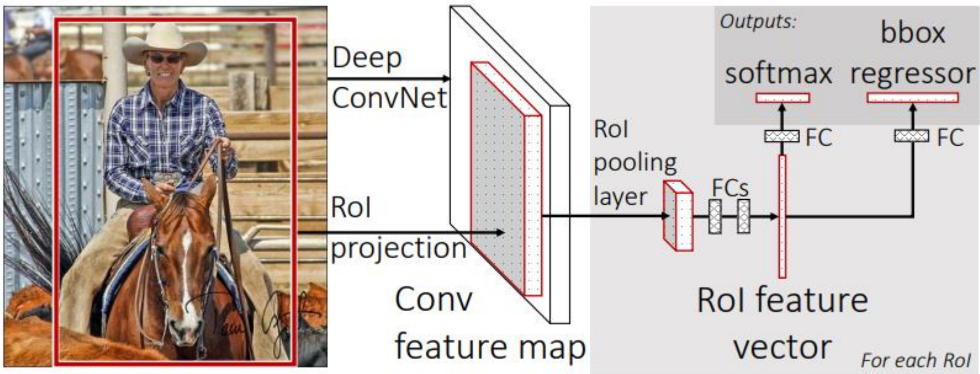

Figure 7: Fast R-CNN Architecture [10].

Faster R-CNN algorithm uses Region Proposal Network (RPN) algorithm to get better results. R-CNN and Fast R-CNN algorithms depend on selective search during finding the region proposals, in Faster R-CNN the model itself learns the region proposals.

RPN is a convolutional network that predicts object points for every position in the image. RPN takes an image as an input, and it returns a set of object locations along with their scores [11].

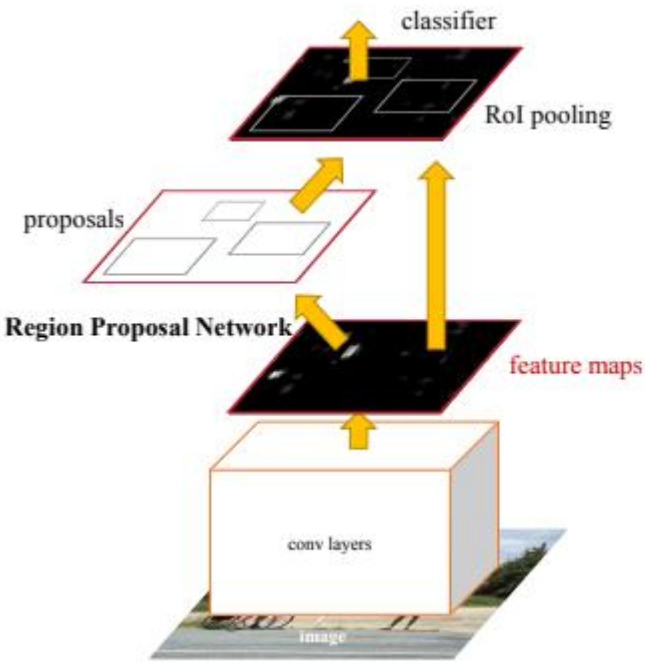

Figure 8: Faster R-CNN Architecture [11].

In a logic similar to Fast R-CNN, the input image is fed into a convolutional network, where the convolutional feature map is found. Instead of applying selective search on the feature map to get region proposals, a new network is utilized to find region proposals. The found region proposals are then pooled using RoI method. RoI pooling is used to categorize the image within the proposed region and find the offset values for the bounding boxes [12].

RoI pooling is used to speed up the learning process and to fix the dimensions of the input. Instead of receiving region suggestions in Faster R-CNN architecture, it is possible to submit region proposals within the network. Speed is gained by doing this.

In Faster R-CNN, the input image is taken and passed through the convolutional neural network to the feature map. At this stage, instead of getting regional proposals, a separate regional proposal network is created. Now regional proposals are made on this created network.

After specifying the network zones, the rest of the process is the same as Fast R-CNN. Both the network giving the proposals and the network with convolution are trained.

6.3 Damaged Part Detection with YOLO

Efficient and accurate completion of object identification tasks is essential in computer vision. The deep learning-based damage detection methods currently in use suffer from complex models and computationally time-consuming problems. That is why we will use the YOLO model to quickly detect where the damage is on the vehicle. Information about the YOLO model can be found below.

YOLO is the acronym for “You Only Look Once”. Real-time object detection is the key characteristic that sets YOLO apart from other techniques. One of the most well-liked object detection methods employing convolutional neural networks is YOLO architecture. The algorithm is capable of quickly and all at once detecting objects. As soon as the algorithm gets going, it can identify objects and their positions in pictures and videos.

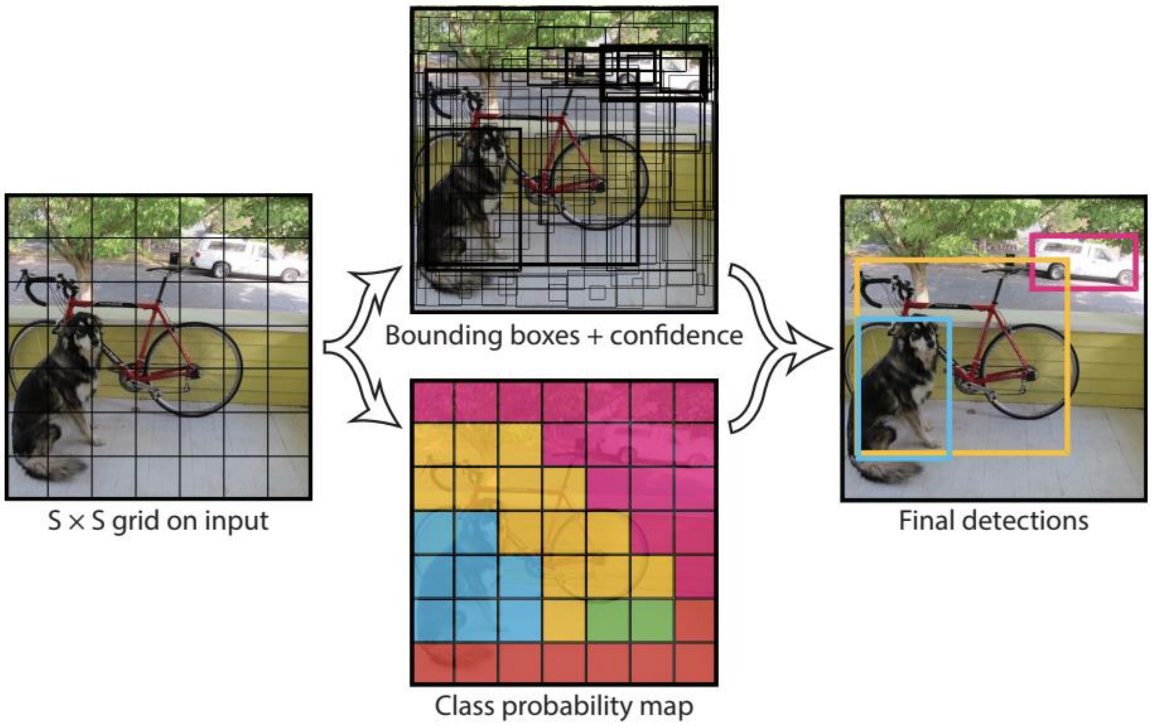

Figure 9: YOLO detection of objects [12].

The YOLO algorithm initially creates a grid of specific-sized sections from the input image (which can be thought of as a grid). The objects in each region are then defined by bounding boxes. A confidence score is computed because an object might be described by more than one bounding box. The bounding box with the highest confidence score, or the best bounding box to represent an object, is chosen and the other bounding boxes are eliminated from the image in accordance with the computed confidence score.

YOLO-v1; Can only detect one object in a grid since only two bounding boxes can be suggested for each grid. Small items are challenging to detect for the algorithm. There is a significant level of localization mistake as compared to Faster R-CNN (the situation where the algorithm detects which class it belongs to and is less successful in drawing a bounding box than other models).

To address the aforementioned issues, Yolo - v2 architecture was developed.

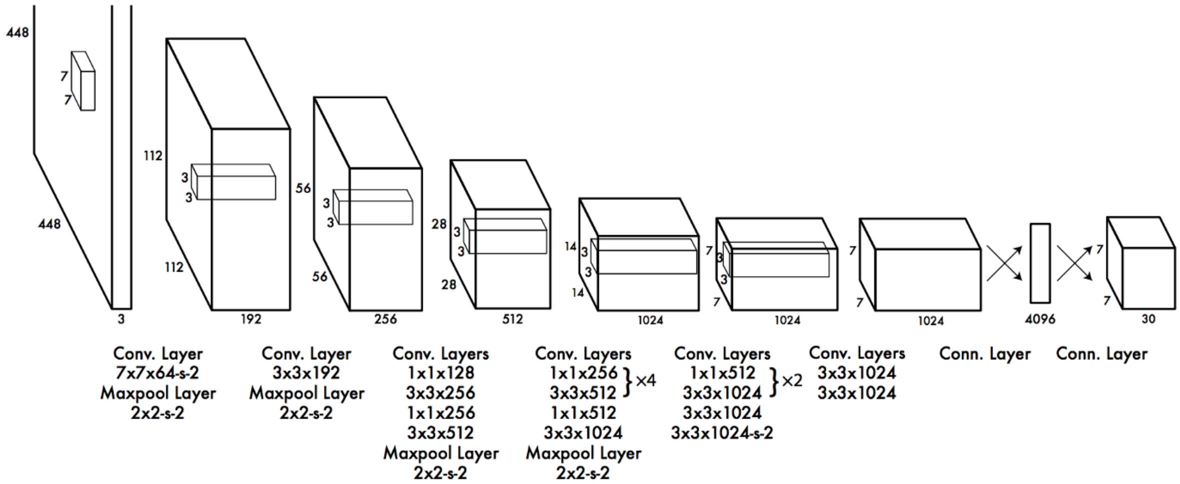

Figure 10: YOLO-v1 layer structure [13].

YOLO-v2 uses anchor boxes to detect multiple objects in a grid. This allows for estimates up to the number of anchor boxes identified for each grid. The YOLO-v2 design's most significant feature, the abundance of anchor boxes and object identification, has solved the greatest drawback of the YOLO-v1 architecture. Another benefit is that while it can recognize very small items that the YOLO-v1 algorithm cannot, the YOLO-v1 drawbacks are solved with this architecture because it can detect huge photographs of the same object.

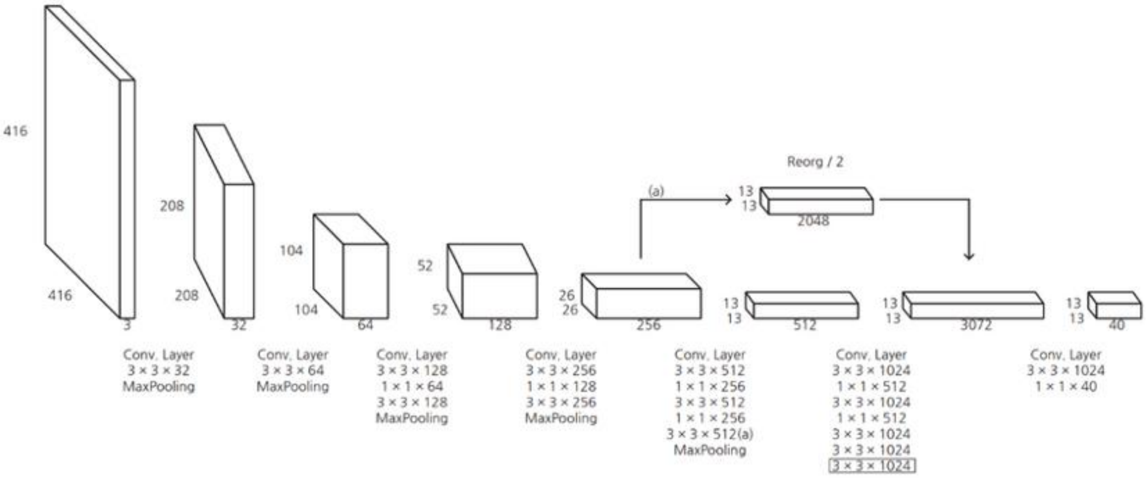

Figure 11: YOLO-v2 layer structure [13].

YOLO-v3 is an incremental upgrade to the prior one. Since there are numerous object detection algorithms and they have all been around for some time, the competition is on how quickly and accurately things can be found. YOLO-v3 includes all the tools we want for accurately detecting things in real-time and classifying them.

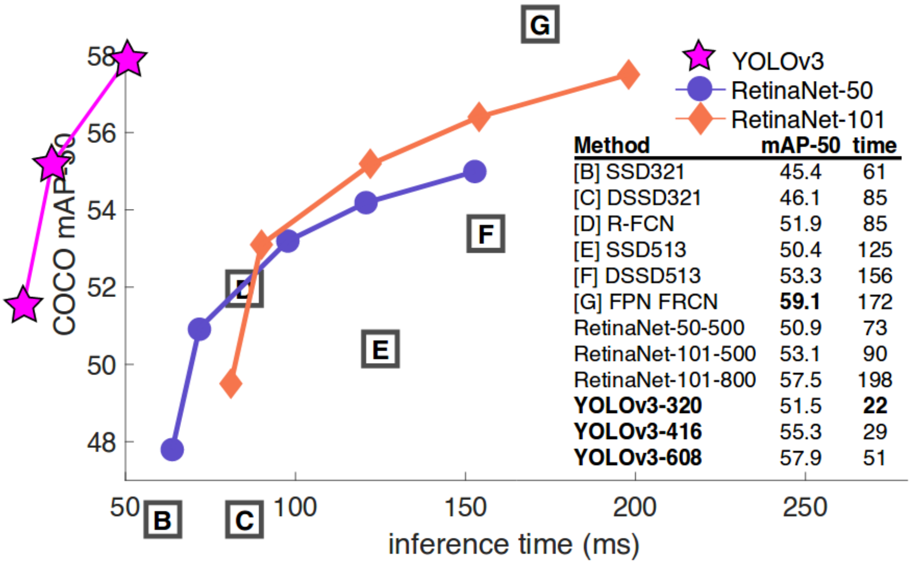

Figure 12: YOLO-v3 [13].

6.4 Damage Percentage Detection with DenseNet

Studies about DenseNet claim that the categorization using DenseNet has demonstrated the accuracy of the final model [14]. Reusing features is the key benefit of DenseNet architecture. Therefore, we will find the percentage of damage using the DenseNet model. Theoretical information of DenseNet can be found below.

DenseNet is a type of convolutional neural network that makes use of dense connections between layers by connecting all layers (with matching feature-map sizes) directly with one another using Dense Blocks [15]. Each layer receives extra inputs from all earlier layers and transmits its own feature-maps to all later layers in order to maintain the feed-forward character of the system.

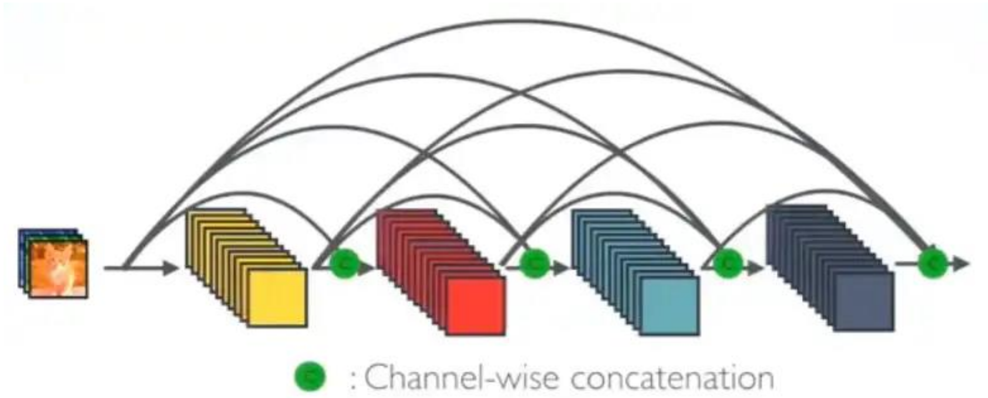

Figure 13: One Dense Block in DenseNet [16].

Each layer in DenseNet receives extra inputs from all levels that came before it and transmits its own feature-maps to all layers that came after it. You utilize concatenation. Each layer receives collective knowledge from the levels that came before it (Figure 11). Each layer receives feature maps from all layers that came before it, allowing for a more compact and thinner network with fewer channels. The extra number of channels for each layer is the growth rate k (Figure 12). Therefore, it has higher memory and processing efficiency.

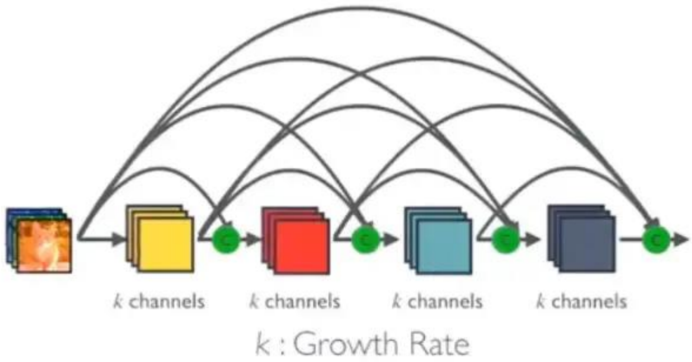

Figure 14: Dense Block in DenseNet with Growth Rate k [16].

7. PROFESSIONAL CONSIDERATIONS

7.1 Methodological considerations/engineering standards

We are going to use Git as our version control system and as a remote git repository we are going to use a private Github repository. We will use Python to train our machine learning models. We will use Tensorflow and Keras libraries along with Python in training. We will also use Flask to develop an API to utilize our trained machine learning models. For development of the DamageWiz application we will use C# along with .NET Core MVC. PostgreSQL will be used as our relational database management system. Java will be used to develop Android mobile applications for our project.

Tool	Explanation	Purpose
git	Version Control System	Version our source codes
GitHub	Remote Git Repository	To store our source codes in a remote place
python™	Programming Language	To implement Machine Learning models
TensorFlow	Machine Learning Library for Python	To implement Machine Learning models
K Keras	Machine Learning Library for Python	To implement Machine Learning models
Flask	Micro Web Framework for Python	To implement Machine Learning models
C#	Programming Language	To implement DamageWiz application
.NET Core	An advanced Web Framework C#	To implement DamageWiz application
PostgreSQL	Relational Database Management System	To implement DamageWiz application
Java	Programming Language	To implement android client of DamageWiz application

Table 2: Technologies used.

7.2 Realistic Constraints

7.2.1 Economical

While training the machine learning models, we might need expensive GPUs or cloud services. To deploy the DamageWiz application we will also need a Linux VPS which will cost some money to us.

On the other hand, our project will minimize the cost of the damage assessment process. Insurance companies will no longer need to pay experts for damage assessment.

We may also charge insurance companies some percentage to profit from the project.

7.2.2 Environmental

We can say that our project will reduce the air pollution a little because experts will not need to travel with their cars for damage assessment.

7.2.3 Ethical

We will take Personal Data Protection Law (KVKK), General Data Protection Regulation (GDPR) and any related regulation into account while training the machine learning models. We will blur all personal information such as car plates, chassis numbers while generating our dataset.

Also, in the DamageWiz application we will try to store as little information as possible and comply with the related regulations.

7.2.4 Health and Safety

Our project is not related to health or safety in any way because we are detecting the damages of cars and the cost of fixing.

7.2.5 Sustainability

As long as people drive cars, our project will survive. Because these cars will always get damaged, and these damages will always be assessed.

7.2.6 Social

Our project will protect the accident victim in a fair way. Damage assessment will be done with machine learning and the cost of repairment will be listed by the mechanics transparently. We are trying to minimize possible misconducts by automatizing the process.

7.3 Legal Considerations

We should be careful not to put any personal information on GitHub because it is a private company located outside of our country. Storing personal information in these kinds of products may not comply with data regulations of our country.

Other development tools we are using are open-sourced software, so complying with their licenses will not be a problem for us. During the generation of the dataset, we will blur the plates of the cars to comply with KVKK.

8. MANAGEMENT PLAN

8.1 Description of Task Phases

Phase 1 - Literature Survey: Literature Survey of Faster R-CNN, TensorFlow YOLO v3, Keras. Duration of phase 1 is five months.

- **1.1:** Deep Learning algorithms for brand detection such as Faster-RCNN and Tensorflow YOLO v3 will be researched.
- **1.2:** Related works and studies will be analyzed and discussed.
- **1.3:** Keras and its applications such as densenet will be studied for damage assessment.

Phase 2 - Data Preparation: Dataset will be collected and designed well. Duration of phase 2 is two months.

- **2.1:** Dataset of damaged cars will be collected from insurance companies and public resources on the internet.
- **2.2:** Dataset of car brand logos will be collected from public resources on the internet.

Phase 3 - Implementation of YOLO: Implementation of Tensorflow YOLO for detection of the brand. Duration of phase 3 is one month.

- **3.1:** Methods of algorithm will be applied.
- **3.2:** Outcome of implementation will be analyzed and tested.

Phase 4 - Implementation of Keras: Implementation of Keras for damage assessment. Duration of phase 4 is two months.

4.1: Selected applications of Keras such as densenet will be implemented.

4.2: Outcome of implementation will be analyzed and tested.

Phase 5 - Data Partitioning: Dataset will be separated for training and testing. They will be applied on models. Models and their efficiency for our projects will be compared to decide fitting models. Duration of phase 5 is one month.

Phase 6 - Web Application Development: Developing web application for the project. Duration of phase 6 is two months.

6.1: Frontend of the website will be developed with HTML CSS and Bootstrap library.

6.2: Backend of the website will be developed with MVC .NET Core and PostgreSQL for database.

6.3: Marketplace for mechanic shops will be created.

Phase 7 - Mobile Application Development: Developing web-based Android mobile application. Duration of phase 7 is two months.

7.1: Web application will be adapted to the android application of our project.

7.2: Java will be used for the mobile application development process.

Phase 8 - Testing and Deployment: Integration of the system parts. Testing and observing all of the outcomes of the project:

- Testing of whether brand detection is working correctly or not.
- Testing of damage assessment and detection of damaged parts.
- Testing of damage percentage of the damaged parts.
- Testing of the product screen that will occur after damaged parts of the car matches with products of mechanic shops. Database queries and different scenarios will be tested.
- Other properties of website and mobile application tests will be done. Duration of phase 8 is eight months.

8.2 Division of Responsibilities and Duties among Team Members

Distribution of work of each phase can be seen below in Table 3.

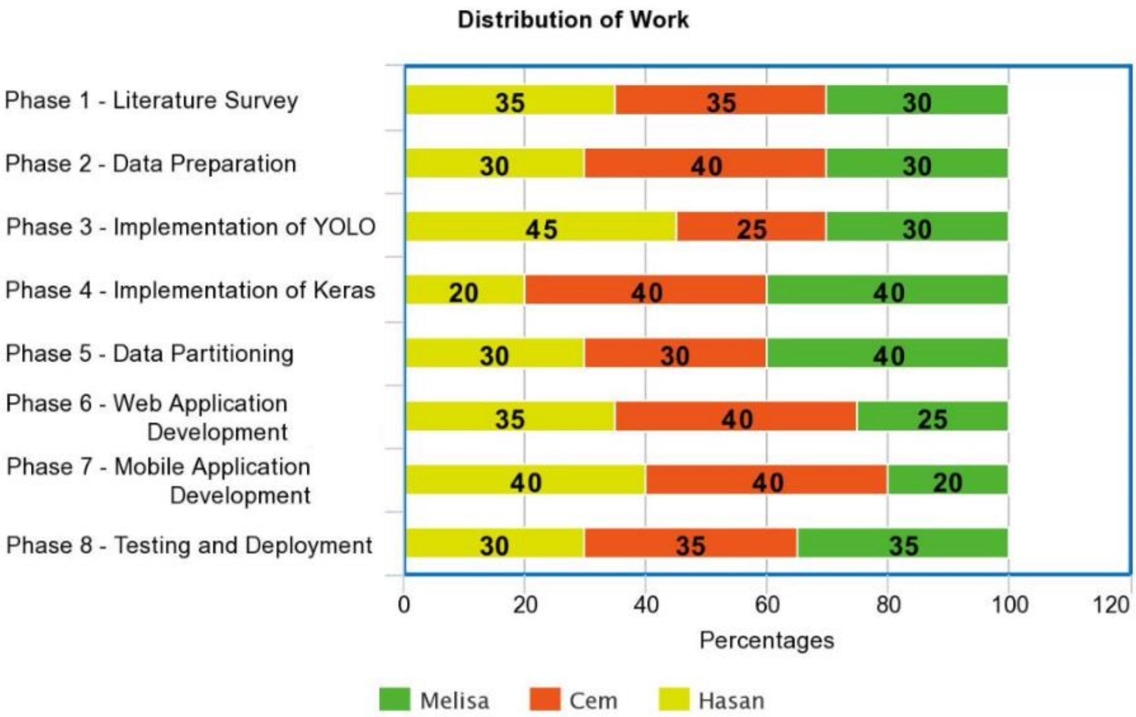

Table 3: Work Distribution of Team Members.

8.3 Timeline

Timeline of the project is shown below in Table 4. The duration of each phase can be seen in terms of months.

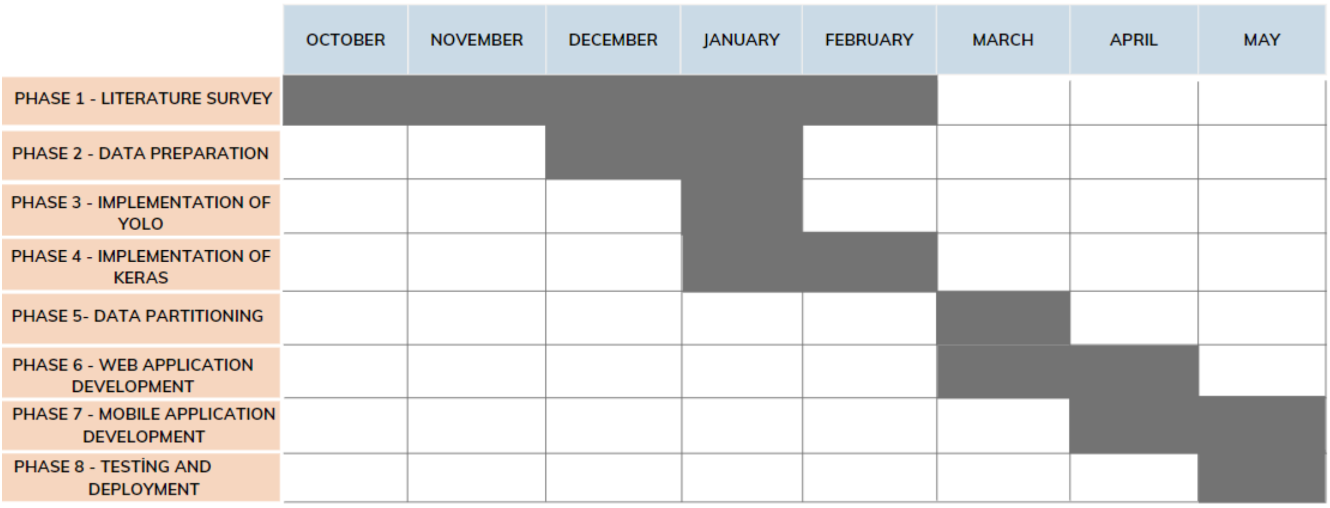

Table 4: Gantt Chart of Timeline of the Project.

9. SUCCESS FACTORS AND RISK MANAGEMENT

9.1 Success Factors

- Need to find out what the brand and model of the vehicle is with an accuracy of 65%. F1-score will be used here to compute the accuracy.
- The location of the damage should be detected with 70% accuracy from the photos taken from the front, rear, left and right sides of the vehicle. F1-score will be used here to compute the accuracy.
- The damaged part must be matched with the model of the car with 70% accuracy. F1-score will be used here to compute the accuracy.
- It will be shown with 100% accuracy in which mechanics the found damaged part is available.
- The mechanics who have the materials will be shown 95% accuracy.
- The application will have a sorting feature. It will sort the mechanics from the cheapest one to the expensive and the nearest one to far with 100% success.

9.2 Risk Management

- We may not be able to find properly labeled data:
  In this case we will obtain the unlabeled data from an experts office and label them ourselves. Also, to obey GDPR or KVKK laws we will need to blur the plates of the cars.
- We may not be able to detect the damaged car's brand:
  If we cannot detect the damaged car's brand, users will select the brand themselves in the application.
- We may not be able to have enough data to train our model:
  In this case we will apply data augmentation to our existing data to generate more data. The new data will be actually the same data from before, but they will be rotated, resized, or cropped.

10. BENEFITS AND IMPACT OF THE PROJECT

The biggest potential benefit of our project is to save users time and money. When our project is completed, our project will be used by people who have had an accident and whose car has been damaged. Our project will also benefit the mechanics. They will be able to find their potential customers through the application.

10.1 Scientific Impact:

The project we are trying to do is not a totally new project. In our research, we have seen projects that focused only on specific issues, such as brand detection or damage detection. Our project will be more comprehensive. Therefore, it will be a continuation of an existing scientific situation. It could be the subject of an article that we may write.

10.2 Economic/Commercial/Social Impact:

The project will not have a major economic impact, but it will save a lot of extra expense personally. Instead of paying money for the expert on the determination of the damage, they will be able to learn from the application for free. Also, it will prevent time wasting.

10.3 Potential Impact on New Projects:

This project is a continuation of existing projects. Similar studies have been done before. Rather than being a pioneer, we can say that it can be a continuation of other projects and give way for the newcomers.

10.4 Impact on National Security:

It will not affect any national security.

REFERENCES

[1] Kunduracı M. F., Kahramanli Örnek, H. "Vehicle Brand Detection Using Deep Learning Algorithms". *International Journal of Applied Mathematics Electronics and Computers 7 (2019)*: 70-74.

[2] CarNet.AI [Date of Access: 27/11/2022] [https://carnet.ai/make-model-recognition-api].

[3] Tafazzoli F., Frigui H. "Vehicle Make and Model Recognition Using Local Features and Logo Detection". *International Symposium on Signal, Image, Video and Communications (ISIVC) (2016)*: 353-358.

[4] Chua A. C. et al. "Damage Identification of Selected Car Parts Using Image Classification and Deep Learning". *IEEE 13th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (*HNICEM*) (2021)*: 1-5.

[5] Volkan Özdemir, “Car Brand Logos” [Date of Access: 27/11/2022] [https://www.kaggle.com/datasets/volkandl/car-brand-logos].

[6] Filip Papranec, “car-logos-dataset” [Date of Access: 27/11/2022] [https://github.com/filippofilip95/car-logos-dataset].

[7] Hamza Manssor, “Car Damage Assessment” [Date of Access: 27/11/2022] [https://www.kaggle.com/datasets/hamzamanssor/car-damage-assessment].

[8] Lalu Prasad Lenka, “Coco Car Damage Detection Dataset” [Date of Access: 27/11/2022] [https://www.kaggle.com/datasets/lplenka/coco-car-damage-detection-dataset].

[9] Girshick R., Donahue J., Darrel T., Malik J. “Rich feature hierarchies for accurate object detection and semantic segmentation”. *IEEE Conference on Computer Vision and Pattern Recognition (2014)*: 580-587.

[10] Girshick R. “Fast R-CNN”. *IEEE International Conference on Computer Vision (ICCV) (2015)*: 1440-1448.

[11] Ren S., He K., Girshick R., Sun J. “Faster R-CNN”. *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1 (2015)*: 91-99.

[12] Gandhi R. “R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms” [Date of Access: 15/11/2022] [https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e].

[13] Karakuş H. V. “Darknet YOLOv3 hızlı bakış” [Date of Access: 27/11/2022] [https://medium.com/@karakus.haciveli/darknet-yolov3-h%C4%B1zl%C4%B1-bak%C4%B1%C5%9F-ddc9cd5582ea].

[14] Pleiss G, Chen D, Huang G, Li T, Van Der Maaten L, Weinberger KQ “Memory-efficient implementation of dense nets” *CoRR (2017).*

[15] Huang G., Liu Z., Maaten L., Weinberger K. Q. “Densely Connected Convolutional Networks”. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017):* 2261-2269.

[16] Tsang S. “Review: DenseNet — Dense Convolutional Network (Image Classification)” [Date of Access: 11/11/2022]. [https://towardsdatascience.com/review-densenet-image-classification-b6631a8ef803]