

INTELLIGENT VEHICLE DAMAGE ASSESSMENT AND COST ESTIMATOR FOR INSURANCE COMPANIES

Category: **Artificial Intelligence**

A PROJECT REPORT

Submitted by

VETRIVEL M

VIGNESH M

SYEDASHIF M

SUBASH CHANDRA BOSE S

FROM

MAHENRA COLLEGE OF ENGINEERING, SALEM

In fulfillment of project in IBM-NALAIYATHIRAN 2022

Team Id: PNT2022TMID31282

PROJECT GUIDES

Industry Mentor: **Swath**

Faculty Mentor: **SOUMYA P**

--	--

INDEX

1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

2. LITERATURE SURVEY

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

4.2 Non-Functional requirements

5. PROJECT DESIGN

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

7.2 Feature 2

7.3 Database Schema (if Applicable)

8. TESTING

8.1 Test Cases

8.2 User Acceptance Testing

9. RESULTS

9.1 Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

Source Code

GitHub & Project Demo Link

1. INTRODUCTION

Introduction According to August 22, 2018, 'China Banking and Insurance Regulatory Commission Office on the Monitoring of Small Claims Insurance Services in 2017' data show that: In 2017, 55.4113 million automobile insurance claims were settled normally. Among them, there are 40.128 million small-scale cases, accounting for 72.22%. The average insurance payment period for small-scale automobile insurance cases is 11.8 days, while the claim period for investigation, damage assessment and claim collection accounts for 9.94 days. These data have triggered several reflections on small-scale cases: Firstly, for insurance companies, 72.22% of small cases require the presence of damage fixers, which leads to high cost of risk investigation, and the leakage problem in the process of damage fixing is difficult to control. Secondly, for the accident party, the long waiting time at the accident site, the slow payment

process, the unreasonable fixed price and other issues, to a certain extent, reduce customer satisfaction with the insurance company. In addition, the potential dangers of traffic congestion and secondary accidents caused by small-scale cases also bring a great pressure to the traffic control department. In the claims industry under the new generation of AI development plan, how can insurance companies move towards a new business model of 'Artificial Intelligence + Scene Application'? Deep convolutional neural networks [1, 2] have led to a series of breakthroughs for image classification [3]. With the development of deep learning [4], the process of computer vision has been greatly accelerated. Research on visual recognition is undergoing a transition from feature engineering to network engineering [5]. With the continuous innovation of AI algorithms and the increasing level of learning, open source deep learning framework and platform have become an important driving force for the development of AI. The improvement of hardware computing power ensures the rapid development of AI. At the same time, the massive data of the automobile insurance industry provides abundant raw materials for network model training. Therefore, we have enough data support and algorithm model to explore the new model of 'Artificial Intelligence + Vehicle Insurance' and build an intelligent damage determination system. Such a new model can not only effectively control the cost expenditure of automobile insurance companies, but also improve the owners' compensation experience. At the same time, it can effectively alleviate road

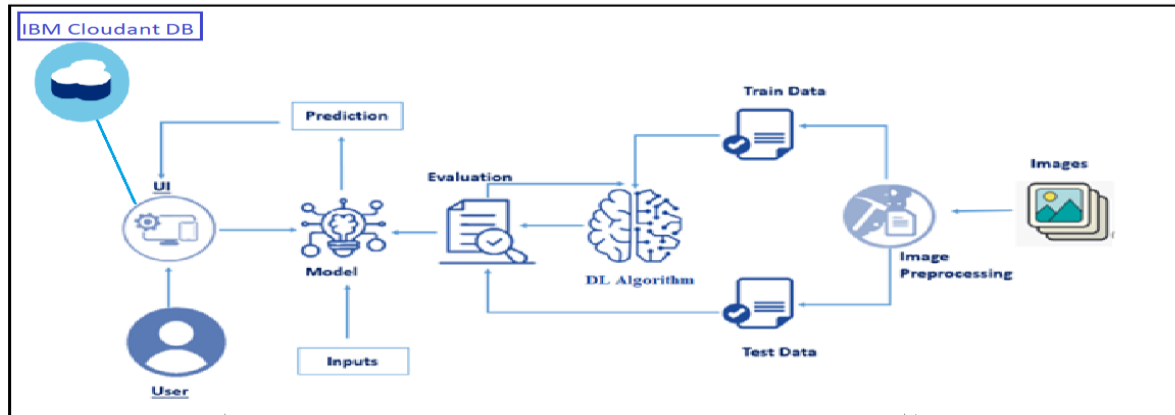
1.1 Project Overview

Nowadays, a lot of money is being wasted in the car insurance business due to leakage claims. Claims leakage Underwriting leakage is characterized as the discrepancy between the actual payment of claims made and the sum that should have been paid if all of the industry's leading practices were applied. Visual examination and testing have been used to may these results. However, they impose delays in the processing of claims.

1.2 Purpose

The aim of this project is to build a VGG16 model that can detect the area of damage on a car. The rationale for such a model is that it can be used by insurance companies for faster processing of claims if users can upload pics and the model can assess damage(be it dent scratch from and estimates the cost of damage.

Technical Intelligence:



2. LITERATURE SURVEY

Kalpesh Patil, et. al.[1] Have used the concept of deep learning in order to classify car damage. The model used is trained on CNN directly. The preprocessing includes the steps of domain-specific pre-training followed by fine-tuning. The paper has conducted a combined and separate study of Transfer Learning and Ensemble Learning. The research has a setback of unavailability of a proper dataset which has resulted in creation of dataset by annotating images. The use of Convolution Auto encoder based pre-training followed by supervised fine-tuning and transfer learning is a novelty factor of this research. Deep learning methodology can also be used for detecting presence or absence of damage and conducting further analysis. The researchers in [2] have applied this in the field of automotives. In this paper, CNN is used for object recognition. The task of classification has been performed on Damaged Vehicle dataset. Mask RCNN is used for segmenting, decomposing and sub-dividing the various instances of Machine Learning. The scope of research is limited to a particular dataset. Extensive research on new data can be performed for testing the quality of the model. Yet the fact that it is an automated system that can classify the damaged vehicle and predict how the damage has occurred remains a unique factor of this research. The concept of faster R-CNN can be helpful for real-time object detection with Region Proposal Networks. This concept is implemented in [3] RPN (Region Proposal Network) is trained end-to-end to generate high-quality region proposals, which are used by Fast R-CNN for detection. RPN and Fast RCNN are merged into a single network by sharing their convolutional features using neural networks with attention mechanisms. The RPN component is essentially used for the unified network to focus on a particular object. The research does not include exploitation and preprocessing on the data.

This process could have been used to improve results. The research has built a unified, deep learning based object detection system to run at near real-time frame rates. Computer Vision Technology can be used for assessment of damage to an object. Xianglei Zhu, et. al. in [4] have developed an unified intelligent framework based on this concept. This paper uses RetinaNet algorithm to identify damaged parts. The accuracy with this algorithm is improved. Mask R-CNN is adopted for the identification of vehicle parts, the damaged parts are determined by the method of sampling, and the time complexity is greatly reduced. The accuracy achieved in this research can be improved in order to get better results. A combination of characteristics of vehicle damage data and suitable data can further strengthen this system. The research has successfully reduced time complexity in damage detection and the use of RetinaNet gives good accuracy in damage detection. The use of Improved Mask RCNN can be used for vehicle damage detection. In [5] this approach is followed using Segmentation algorithm. A deep learning approach is used to detect vehicle-damage for compensation problem in traffic accidents. The algorithm has achieved good detection results in different scenarios. Regardless of the strength of the light, the damaged area of multiple cars, or a scene with an overly high exposure, the fitting effect is better and the robustness is strong. The limitation of this research lies in the mask instance segmentation. In many cases the obvious damage is not considered and segmented leading to inaccurate results. This research contributes to detection of damage of vehicles in a more efficient method through improved Mask algorithm. Convolutional Neural Network (CNN) is a widely used algorithm for the purpose of classification problems. This method is used by Jeffrey de Deijn in [6] The research was able to detect car damage with fairly accurate results. The type, location and size of damage is detected with moderate accuracy. The addition of Ensemble learning could have further improved the results from this research. The use of ConvNets to detect car damage detection and transfer learning are the novelty factors of this research.

2.1 References

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

[2] Rakshata P, Padma H V, Pooja M, Yashaswini H V, Karthik V, "Car Damage Detection and Analysis Using Deep Learning Algorithm For Automotive", Vol 5, Issue 6, International Journal of Scientific Research Engineering Trends (IJSRET), Nov-Dec 2019, ISSN (Online): 2395566X

[3] Ren, Shaoqing He, Kaiming Girshick, Ross Sun, Jian. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence. 39. 10.1109/TPAMI.2016.2577031.

[4] X. Zhu, S. Liu, P. Zhang and Y. Duan, "A Unified Framework of Intelligent Vehicle Damage Assessment based on Computer Vision

Technology," 2019 IEEE 2nd International Conference on Automation, Electronics and Electrical Engineering (AUTEEE), 2019, pp. 124-128, doi: 10.1109/AUTEEE48671.2019.9033150.

[5] Q. Zhang, X. Chang and S. B. Bian, "Vehicle-Damage-Detection Segmentation Algorithm Based on Improved Mask RCNN," in IEEE Access, vol. 8, pp. 6997-7004, 2020, doi: 10.1109/ACCESS.2020.2964055

[1] LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D. Backpropagation applied to handwritten zip code recognition. Neural computation, 1989, pp. 541-551.

[2] Krizhevsky, A., Sutskever, I., Hinton, G. Imagenet classification with deep convolutional neural networks. In NIPS, 2012, pp. 1097-1105.

[3] Zeiler, M. D., Fergus, R. Visualizing and understanding convolutional neural networks. In ECCV, 2014, pp. 818-833.

[4] LeCun, Y., Bengio, Y., Hinton, G. Deep learning. Nature, 2015(521), pp. 436-444.

[5] Simonyan, K., Zisserman, A. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015, pp. 1409.1556.

[6] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In NIPS, 2015, pp. 91-99.

[7] Kaiming He, Georgia Gkioxari, Piotr Dollar, Ross Girshick. Mask R-CNN. In ICCV, 2017, pp. 2989

2.2 Existing problem

During the last few years, a lot of frameworks for fast development and prototyping of NNs have emerged. Most of the available software is open source. These frameworks often differ from one another in their interfaces and support for pre-trained models. Caffe [caf] is a framework written in C++ and Python that has Python, command line, Matlab and C++ interfaces, making it a very attractive candidate for fast prototyping of CNNs. It supports CNNs and has support for CUDA and cuDNN technology. Caffe is very popular among researchers and has a big community. Another important framework is Theano [The]. It is written in Python and has Python interface. It supports CNNs and makes use of CUDA technology. It lacks a command line interface to train or fine-tune models. It also has support for pretrained models. Torch [tor] is another option. It supports CNNs and CUDA. It has it's own scripting interface, a C/C++ interface and a command line interface too. Uses LuaJIT as the preferred scripting language. This last feature might be a drawback since I'm not familiar with Lua. Deeplearning4j [dee] is a framework for NNs implemented in Java that has interfaces for various languages that run on the JVM such as Java, Clojure and Scala. It has no command line interface. It has CUDA support and has pre-trained models. The lack of command line interface and the fact that it isn't a mature framework are drawbacks. Tensorflow [tf] is another major open source framework for NNs. It has a Python interface but lacks a CLI interface. It has some support for pre-trained models, but the support for these models is not as good as Caffe's. Keras [ker] is a framework that relies on a Theano or Tensorflow backend. It exposes a slightly friendlier interface API than Theano or Tensorflow, and also offers a slightly better support for pretrained models. It lacks a CLI. CNTK [cnt] is another framework for working with CNN's. It has a Python and C++ interface and also a CLI. It has great support for Windows OS but a slightly worse support for Linux based systems, which is a drawback. Deep CNNs often require a lot of examples for training. This constraint is especially important in cases where labelled data is not very abundant as is the case of the problem addressed here. One way of mitigating this problem is by using pre-trained nets, that require less data for training, although their performance might be slightly worse [VGBP15]. Since this is a very important issue to be taken into account during this work, frameworks not having pre-trained models were omitted from this discussion. Some of this data discussed here is summarised in Table 2.1

Framework	Language	Command Line Interface	CUDA Support
Caffe	C++ and Python	Yes	Yes
Theano	Python	No	Yes
Torch	C/C++ and Lua	Yes	Yes
Deeplearning4j	Java, Scala and Clojure	No	Yes
Tensorflow	C++ and Python	No	Yes
Keras	Python	No	Yes
CNTK	C++, Python and BrainScript	Yes	Yes

Taking into account the amount of

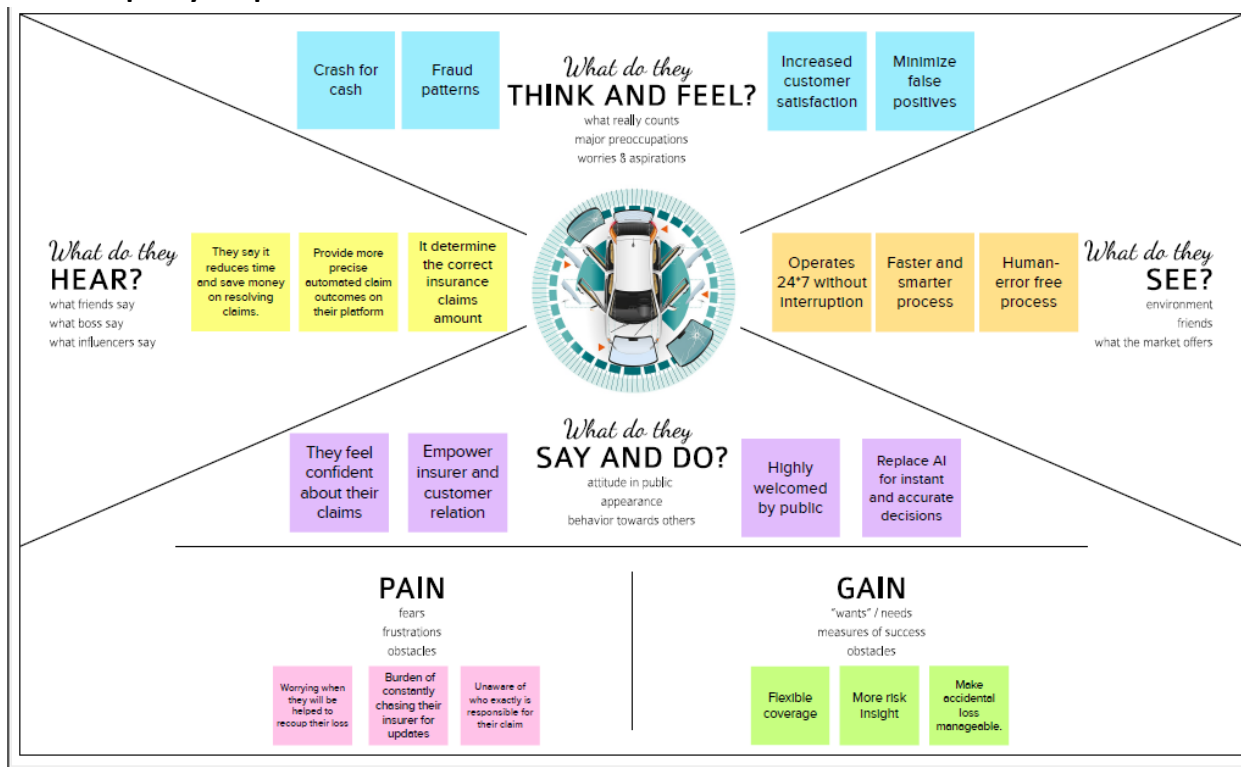
evidence gathered and discussed here, I believe that Caffe is the best option to be used for the system to be developed. Caffe's major benefits include the fact that it has great support for pre-trained CNN models, as well as a big community.

2.3 Problem Statement Definition

The goal of this dissertation is to develop a system capable of automatically identifying and locating damages in images of cars. Automatic detection of car damages in images is a task that may be tackled using a wide variety of approaches. The proposed approach splits the general problem of identifying damages in two different problems. The first problem, consists of distinguishing different types of damages based on their severity. It is important to notice that distinguishing different kinds of damages requires the system to first be able to detect its presence. The second task would be to locate the damages in the car's external surface by saying which areas are damaged. This last problem involves detecting the damage and being able to locate it in the car. Both are multi-class classification tasks with a different number of classes. These two distinct tasks, if successfully performed, may lay ground for more complex tasks such as car repair cost estimation, or other higher level tasks that may require a severity and damage locations estimation to be performed. There are several available location and severity scales that might be used to distinguish different kind of damages. A slight modification of framework proposed by the NSC will be used [Cou83], adapting it to the necessities of the system to be developed here. The NSC is a forum whose function is to advise and assist the US president on national security and foreign policies. It proposes, among other things, policies regarding traffic laws and car safety. The mentioned framework sets parameters for assessing damage sustained by motor vehicles in traffic crashes. It establishes a scale for type of impact (direction) and severity of damages. The document describes sixteen different impact types, and seven different severity levels.

3. IDEATION & PROPOSED SOLUTION

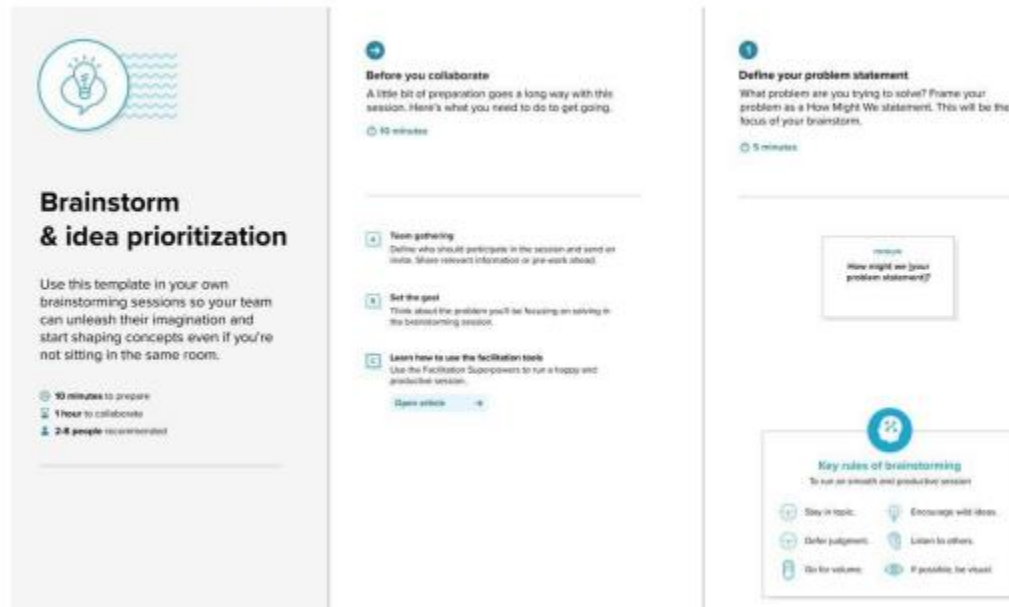
3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming

Brainstorm & Idea Prioritization Template: Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions. Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Step-1: Team Gathering, Collaboration and Select the Problem Statement



3.3 Proposed Solution

parts of the vehicle by image. The system has been experimented many times in the development of intelligent image damage algorithm. Finally, it divides the problem into three parts:

the recognition of appearance parts by image, the recognition of damage parts by image, and the

determination of damage parts by relative position relationship.

3.4 Problem Solution fit

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) CS <ul style="list-style-type: none"> Commercial working people travelling from one point to another Basically belonging to 18+ years old Person who's vehicle experienced some accident or damage in the vehicle A customer with valid insurance policy to claim 	6. CUSTOMER CONSTRAINTS CC <ul style="list-style-type: none"> Troubled network connection might lead to inaccessible of certain features Improper images or blurred images might affect the accurate performance of the application 	5. AVAILABLE SOLUTIONS AS <ul style="list-style-type: none"> Approaching 3rd person for cost estimation Cost estimation done by manual calculations Using slow processing algorithms to detect the damage Pros <ul style="list-style-type: none"> The estimated values stays within the customer and bank agent Cons <ul style="list-style-type: none"> Estimated cost varies frequently The time taken for estimation is very high leading to lots of loss and mental issues 	Explore AS, differentiate
------------------------	--	---	---	---------------------------

2. JOBS-TO-BE-DONE / PROBLEMS J&P <ul style="list-style-type: none"> The main problem will be time consumption in assessing the damage cost and damage percentage To address such an issue it is very important to provide accurate damage percentage and unified cost for that damage Failed to provide perfect value for damage by the Insurance companies 	9. PROBLEM ROOT CAUSE RC <ul style="list-style-type: none"> Deviation or variation from the company calculated cost and the actual cost Rapid development in the AI field paved way to many advance methodologies of estimation customers have to do it because of the change in regulations. 	7. BEHAVIOUR BE <ul style="list-style-type: none"> The customer has to upload the images of the car after an accident. The application will instantly evaluate the damages and displays the claim amount to the customer.
---	--	---

Identify strong TR & EM	3. TRIGGERS TR <ul style="list-style-type: none"> Technological advancement in the field of predictions and estimation colleagues and society demanding instant insurance claim customer wanting to be independent without falling into false traps 	10. YOUR SOLUTION SL <ul style="list-style-type: none"> Accurately estimate the damage percentage Predict the region of damage with respect to the vehicle use fast processing algorithm for functionality interactive and user-friendly solution to make it easily accessible for the user The functionality of the existing solution is slow eliminating human error while estimation 	8. CHANNELS of BEHAVIOR CH	Identify strong TR & EM
	4. EMOTIONS: BEFORE / AFTER EM <p>Before:</p> <ul style="list-style-type: none"> Delay in insurance claim Unable to claim an accurate amount for vehicle damage <p>After:</p> <ul style="list-style-type: none"> Customers felt independent Received their insurance claims at an instant Were able to evaluate an unified insurance claim for their vehicle damages 		8.1 ONLINE <ul style="list-style-type: none"> webpage can be accessed to estimate damage using input image quick access of the artificial intelligence based algorithm for damage assessment 8.2 OFFLINE <ul style="list-style-type: none"> Reach out to the respect insurance agent or the corresponding bank to proceed further with the insurance payment protocols validate the estimate cost with the cost provided by the firm 	

1 Functional requirement

The Functions of Intelligent Damage Assessment System

Intelligent damage determination system can be used to determine the appearance damage of vehicles

in small cases. The system completes the whole process of survey and damage determination through

four functions. They are:

(1) Accident investigation: Photographs of target vehicles and multiple trio vehicles were taken and

uploaded, intelligent recognition, information input, intelligent recognition and event finalization are

completed in accident investigation.

(2) Intelligent image damage assessment: image damage assessment is achieved by intelligent

component recognition and intelligent damage recognition.

(3) Damage result output: Damage results including maintenance scheme recommendation and

maintenance price recommendation are automatically given according to damage recognition results.

(4) Vehicle insurance anti-fraud: In the process of fixing the damage, the anti-fraud screening of

vehicle insurance is completed by means of image fraud recognition and logical detection.

Intelligent damage assessment system can assist the damage locator in the front-end damage

detection process. The operator only needs to take several photos to upload according to the

requirements, and the system can automatically identify the damage degree of the damaged parts and

components. The system in the back-end nuclear damage link can provide auxiliary nuclear damage

and anti-fraud services. It can identify the cases of fixed-loss errors through the logical recognition of

vehicle parts, image fraud recognition, fixed-loss logic recognition, etc. At the same time, it can also

meet the demands of anti-fraud and leakage prevention.

At present, the intelligent damage assessment system can realize the appearance damage of

passenger cars, including CAR, SUV, MPV and VAN. The applicable damage range covers all types

of damage of vehicle exterior parts; the applicable environment range covers rain and snow

environment, dark environment (vehicle can be seen by human eyes), strong light environment and

other scenarios

4.2 Non-Functional requirements

(1) Using smartphones to shoot pictures with no less than 2 million pixels.

(2) For the photography of vehicle damage, it is necessary to shoot the vehicle damage head-on so

that the damage location is as far as possible in the center of the picture. The shooting distance is about

1 meter, and it is suitable to shoot clearly.

(3) Multiple damage or cross-component vehicle appearance damage, if the damage distance is

relatively close, then a photo can be taken, if the damage distance is relatively far, can not take a photo,

then need to be taken separately.

In addition, the intellectualization of photography is also reflected in the following aspects:

When

taking photographs, it automatically identifies whether it is a document photo, a person-car photo, etc.

If the photograph does not meet the requirements is not approved, it needs to be re-taken. At the same

time, it is not mandatory to satisfy what angle of shooting can be taken, which is easy to operate and

makes it easier for the damage fixer or other users to use.

3.1.2. Intelligent ID Recognition. For the photos of the uploaded driving license (front and side pages),

driving license (front and side pages) and other documents, the intelligent damage determination

system embedded OCR recognition technology. The VIN code, license plate number, engine number,

driver's name and other information of the uploaded driving license and driver's license can be

intelligently recognized and filled in.

At present, the embedded OCR technology can recognize Chinese characters, English upper and

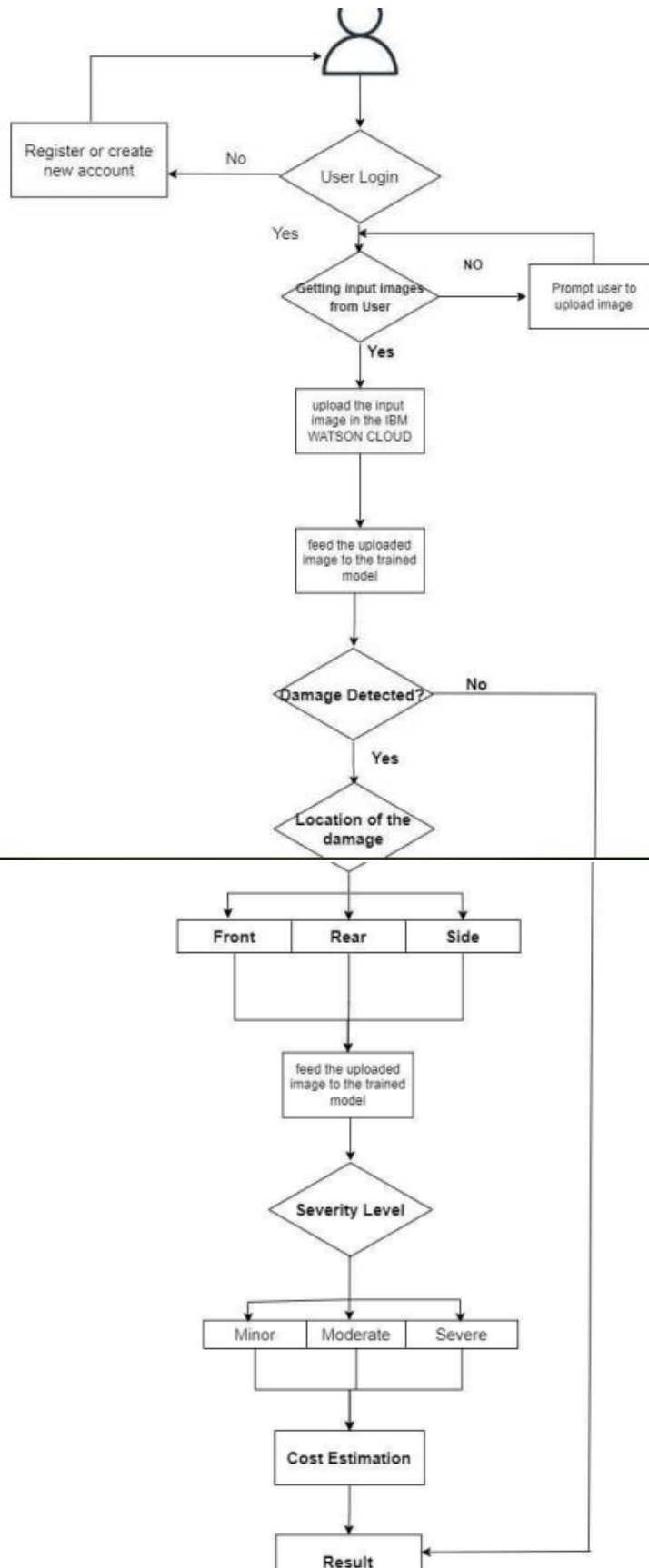
lower case letters, numbers and other information, and the recognition accuracy is 98.5%.

Aiming

5. PROJECT DESIGN

5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

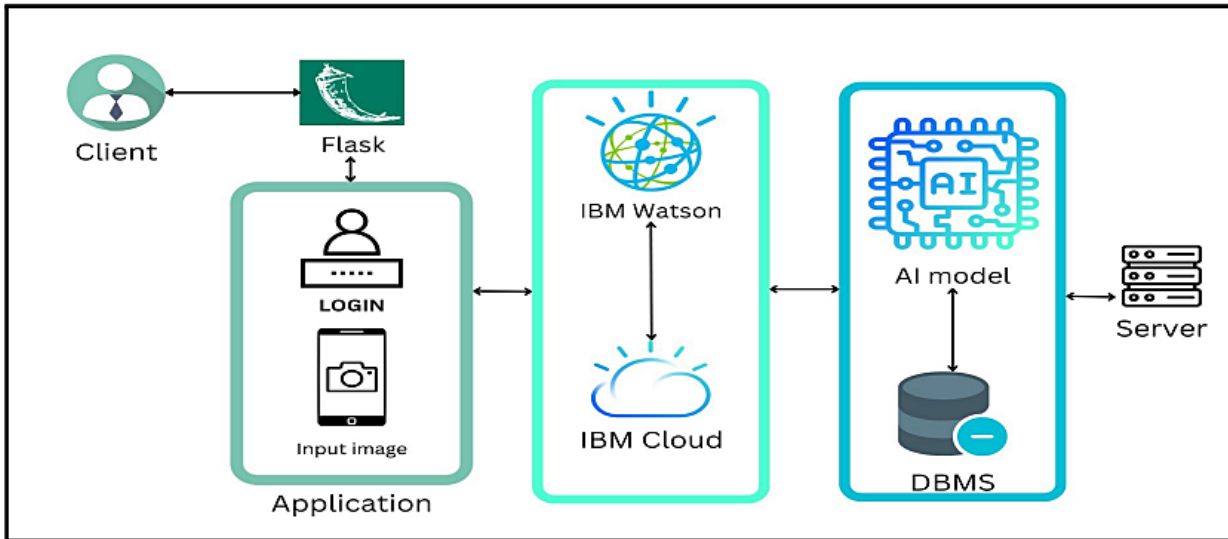


5.2 Solution & Technical Architecture

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Interface	User friendly and simple website
FR-4	Collection of datasets	Information about the user and their vehicle. Information about Insurance plans.
FR-5	Results	Model should be trained with high accuracy. Results obtained from the model should be displayed to the user with easy interpretability.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Intelligent model to assess the damage in the vehicle and estimate the cost to be provided by the insurance company.
NFR-2	Security	The authenticity of the user and the confidentiality of the user details about their vehicle should be maintained.
NFR-3	Reliability	This project should be able to achieve good accuracy in damaging assessment as well in cost estimation so that the user is provided with the accurate and unbiased insurance amount.
NFR-4	Performance	The real time images should be captured and uploaded into the website where the proposed model will carry out the damage assessment and give the cost of insurance accordingly.
NFR-5	Availability	The webpage should be compatible for the web browsers in both mobile phones and computers.
NFR-6	Scalability	The proposed solution will be scalable in future because of the efficient and quicker analysis and exact cost prediction

Technical Architecture:



S.No	Component	Description	Technology
1.	User Interface	The user interacts with the web UI application	HTML, CSS, Python
2.	Application Logic-1	Getting user input image	Python
3.	Application Logic-2	Getting model output for damage prediction	IBM Watson, Python
4.	Application Logic-3	Getting model output for cost estimation	IBM Watson, Python
5.	Database	Data Type – Images and user inputs details are stored	MySQL, Js, IBM DB2
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	Received user details and received user input images of the vehicle is stored in cloud	IBM Block Storage, IBM cloud
8.	Machine Learning Model	Purpose of the AI Model is for estimating the cost of the damaged vehicle.	Object Recognition Model, and CNN based model for damage estimation
9.	Infrastructure (Server / Cloud)	On cloud server we will be deploying the AI Model using flask in the web page	Python Flask

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Open-source frameworks used is IBM Watson	Technology of Open Source framework- IBM Watson
2.	Security Implementations	IBM Cloud	Certified Watson assistant for Encrypted file systems, Encrypted storage systems, Key management systems.
3.	Scalable Architecture	Web server - static and dynamic website content present in the website will be update based upon user demands and suggestion Application server - updation of the basic functionality of the website and integration of new logic within the website can be done Database server - based upon the varying inputs given by the user the database will be modified constantly	IBM Watson Assistant, Python, MySQL
4.	Availability	The AI model is made available instantly to user at any point of time	IBM Watson Cloud assistance
5.	Performance	IBM Watson –automate processes, The deep learning model is trained using IBM Watson studio for better performance and quick accessibility .	IBM Watson Assistant

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, and password, and confirming my password.	I can access my account/dashboard by entering valid credentials	High	Sprint-1
Customer Details	Login	USN-2	As a user, I will receive a confirmation email once I have registered for the application	I can receive a confirmation email & click confirm	High	Sprint-1
Customer Uses	Dashboard	USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-4
Customer Options	Details about insurance companies	USN-4	As a user, I can register for the application through Gmail	I can register & access the dashboard with Facebook Gmail	Medium	Sprint-1
Customer usage	Login and repeated usage	USN-5	As a user, I can log into the application by entering email & password	I can log in and view my dashboard at my demand on any time	High	Sprint-1
Customer needs to do	web page	USN-6	As a user I must capture images of my vehicle and upload it into the web portal.	I can capture the entire vehicle and upload	High	Sprint-2
Customer (Web user) value	Details about estimated cost based on damage	USN-7	As a user I must receive a detailed report of the damages present in the vehicle and the cost estimated	I can get the estimated insurance cost	High	Sprint-3
Customer Care Executive	Provide friendly and efficient customer support and sort out the queries	USN-8	As a user, I need to get support from developers in case of queries and failure of service provided	I can have smooth user experiences and all the issues raised is sorted	Medium	Sprint-4
Administrator	Overview the entire process and act as a bridge between user and developers	USN-9	We need to satisfy the customer needs in an efficient way and make sure any sort of errors are fixed	I can finish the work without any problems	High	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user ,I can resister for the application by entering my email,password, and confirming my password.	2	High	VETRIVEL M
Sprint-1	Registration	USN-2	As a user, I will receive confirmation email once I have Registered for the Application	1	High	VETRIVEL M
Sprint-1	Registration	USN-3	As a user ,I can register for the application Gmail.	2	Low	
Sprint-1	Login	USN-4	As a user ,I can Login to the application by entering email & password .	1	Medium	VETRIVEL M
Sprint-2	Dashboard	USN-5	As a user ,I can view all the plans and methods in the Dashboard.	1	High	SYED
Sprint-3	Storage	USN-1	As a user, I can Register for claim my insurance.	2	High	VIGNESH M
Sprint-3		USN-2	As a user, I can make a call to support line to get help with a product or service	2	High	SUBASHCHANDHARA BOSE S
Sprint-4		USN-3	As a user, I can claim my insurance After getting from the administrator.	1	Medium	VETRIVEL M

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022		
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022		
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022		

Velocity:

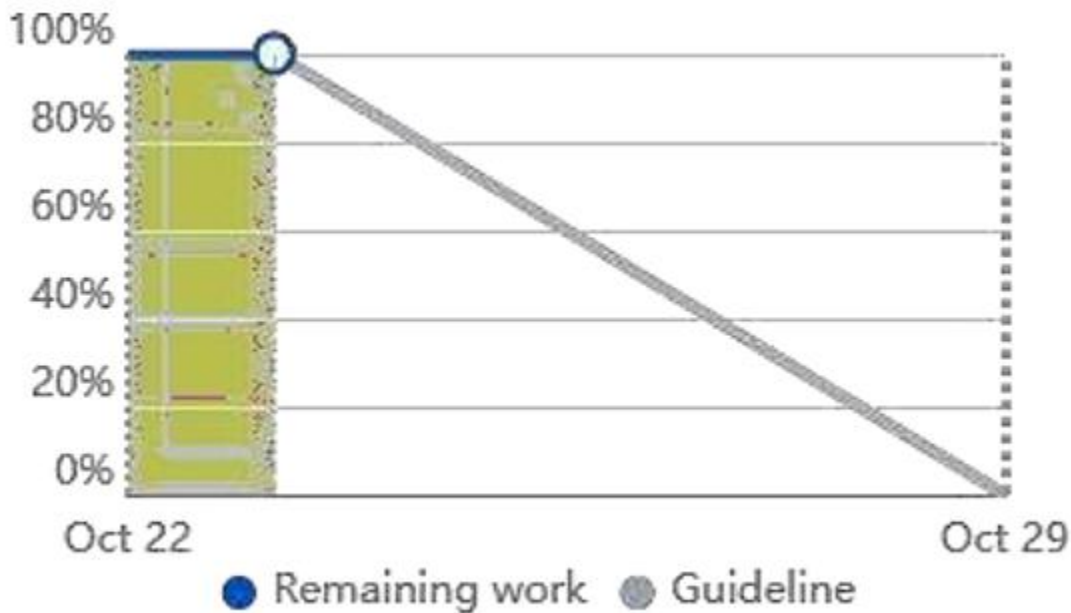
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{\text{sprint duration}}{\text{velocity}} = \frac{20}{10} = 2$$

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

6.3 Reports from JIRA



Sprint	Milestone
Sprint 1	<ol style="list-style-type: none"> 1. User can create new account by providing user information like name , email id and mobile number. 2. The user will receive a confirmation message via email. 3. Once confirmed, new password generation and account creation is done. 4. A user login portal will be created. 5. User can log in and out whenever required.
Sprint 2	<ol style="list-style-type: none"> 1. Users can access their dashboard and update other details 2. User uploads pictures of their vehicle and other relevant details
Sprint 3	<ol style="list-style-type: none"> 1. Developing a Model and verifying the accuracy. 2. Creating a secure database to store and access user information and images. 3. User will get the detailed assessment of their damaged vehicle and estimated cost for insurance will be generated.
Sprint 4	<ol style="list-style-type: none"> 1. Collection user feedback and queries are done. 2. Constant updation of the portal and its respective backend work is done. 3. Additional login features will be implemented.

7. CODING & SOLUTIONING

7.1 Feature 1

```
{  
  "cells": [  
    {  
      "cell_type": "markdown",  
      "metadata": {},  
      "source": [  
        "Importing the Image Data Generator Libraries "  
      ]  
    },  
    {  
      "cell_type": "code",  
      "execution_count": 2,  
      "metadata": {},  
      "outputs": [],  
      "source": [  
        "import tensorflow as tf \n",  
        "import keras \n",  
        "from keras.preprocessing.image import ImageDataGenerator"  
      ]  
    },  
    {  
      "cell_type": "markdown",  
      "metadata": {},
```

```

"source": [
  "Configuring The Image Data Generator "
]
},
{
  "cell_type": "code",
  "execution_count": 3,
  "metadata": {},
  "outputs": [],
  "source": [
    "#training images \n",
    "train_datagen = ImageDataGenerator(\n",
    "    rescale=1./255,\n",
    "    shear_range=0.2,\n",
    "    zoom_range=0.2,\n",
    "    horizontal_flip=True)\n",
    "\n",
    "#val images \n",
    "val_datagen = ImageDataGenerator(rescale=1./255)"
  ]
},
{
  "cell_type": "markdown",
  "metadata": {},
  "source": [
    "Applying the image generator to the body "
  ]
}

```



```

},
{
  "cell_type": "code",
  "execution_count": 5,
  "metadata": {},
  "outputs": [
    {
      "name": "stdout",
      "output_type": "stream",
      "text": [
        "Found 979 images belonging to 3 classes.\n"
      ]
    }
  ],
  "source": [
    "body_train_generator = train_datagen.flow_from_directory(\n",
    "    'V:\\\\Workspace\\\\IBM-Project-23426-1659882722\\\\Dataset\\\\Car",
    "    damage\\\\body\\\\training',\n",
    "    target_size=(150, 150),\n",
    "    batch_size=32,\n",
    "    class_mode='categorical')\n",
    "],
},
{
  "cell_type": "code",
  "execution_count": 6,
  "metadata": {},

```

```
"outputs": [
  {
    "name": "stdout",
    "output_type": "stream",
    "text": [
      "Found 171 images belonging to 3 classes.\n"
    ]
  }
],
"source": [
  "body_val_generator = val_datagen.flow_from_directory(\n",
  "    'V:\\\\Workspace\\\\IBM-Project-23426-1659882722\\\\Dataset\\\\Car",
  "damage\\\\body\\\\validation',\n",
  "    target_size=(150, 150),\n",
  "    batch_size=32,\n",
  "    class_mode='categorical')\n",
  },
  {
    "cell_type": "markdown",
    "metadata": {},
    "source": [
      "Applying the image generator to the level "
    ]
  },
  {
    "cell_type": "code",
```

```
"execution_count": 7,
"metadata": {},
"outputs": [
  {
    "name": "stdout",
    "output_type": "stream",
    "text": [
      "Found 979 images belonging to 3 classes.\n"
    ]
  }
],
"source": [
  "level_train_generator = train_datagen.flow_from_directory(\n",
  "    'V:\\\\Workspace\\\\\\\\IBM-Project-23426-1659882722\\\\\\\\Dataset\\\\\\\\Car",
  "damage\\\\\\\\level\\\\\\\\training',\n",
  "    target_size=(150, 150),\n",
  "    batch_size=32,\n",
  "    class_mode='categorical')\n",
  ],
},
{
  "cell_type": "code",
  "execution_count": 8,
  "metadata": {},
  "outputs": [
    {
      "name": "stdout",
```

```

"output_type": "stream",
"text": [
  "Found 171 images belonging to 3 classes.\n"
]
},
],
"source": [
  "level_val_generator = val_datagen.flow_from_directory(\n",
  "    'V:\\\\Workspace\\\\IBM-Project-23426-1659882722\\\\Dataset\\\\Car",
  "damage\\\\level\\\\validation',\n",
  "    target_size=(150, 150),\n",
  "    batch_size=32,\n",
  "    class_mode='categorical')\n",
  ],
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {},
  "outputs": [],
  "source": []
}
],
"metadata": {
  "kernel_spec": {
    "display_name": "Python 3.7.9 64-bit (microsoft store)",
    "language": "python",

```

```
"name": "python3"
},
"language_info": {
  "codemirror_mode": {
    "name": "ipython",
    "version": 3
  },
  "file_extension": ".py",
  "mimetype": "text/x-python",
  "name": "python",
  "nbconvert_exporter": "python",
  "pygments_lexer": "ipython3",
  "version": "3.7.9"
},
"orig_nbformat": 4,
"vscode": {
  "interpreter": {
    "hash": "b81d285a39ce6bce5fc3d0e228a6124a883049b7dbc9a3a4527a4b38646ff66a"
  }
}
},
"nbformat": 4,
"nbformat_minor": 2
}
```

7.2 Feature 2

```
from flask import Flask, render_template, flash, request, session
```

```
from cloudant.client import Cloudant
```

```
import cv2
```

```
client = Cloudant.iam("eb55a2b7-ae45-4df8-8d1c-69c5229ffdbe-  
bluemix", "YzG5FZg9Vs_HScOBZaWyVXm7PpNjbPrmPaPMfHx7w3X9", connect=True)
```

```
my_database = client.create_database("database-dharan")
```

```
app = Flask(__name__)
```

```
app.config.from_object(__name__)
```

```
app.config['SECRET_KEY'] = '7d441f27d441f27567d441f2b6176a'
```

```
@app.route("/")
```

```
def homepage():
```

```
    return render_template('index.html')
```

```
@app.route("/userhome")
```

```
def userhome():
```

```
    return render_template('userhome.html')
```

```
@app.route("/addamount")
```

```
@app.route("/NewUser")
```

```
def NewUser():
```

```
    return render_template('NewUser.html')
```

```
@app.route("/user")
```

```
def user():
```

```
    return render_template('user.html')
```

```
@app.route("/newuse",methods=['GET','POST'])
```

```
def newuse():
```

```
    if request.method == 'POST':#
```

```
        x = [x for x in request.form.values()]
```

```
        print(x)
```

```
        data = {
```

```
            '_id': x[1],
```

```
            'name': x[0],
```

```

        'psw': x[2]
    }
    print(data)
    query = {'_id': {'Seq': data['_id']}}
    docs = my_database.get_query_result(query)
    print(docs)
    print(len(docs.all()))
    if (len(docs.all()) == 0):
        url = my_database.create_document(data)
        return render_template('goback.html', data="Register, please login using your details")
    else:
        return render_template('goback.html', data="You are already a member, please login
using your details")

@app.route("/userlog", methods=['GET', 'POST'])
def userlog():
    if request.method == 'POST':

        user = request.form['_id']
        passw = request.form['psw']
        print(user, passw)

        query = {'_id': {'$eq': user}}
        docs = my_database.get_query_result(query)
        print(docs)
        print(len(docs.all()))
        if (len(docs.all()) == 0):
            return render_template('goback.html', pred="The username is not found.")

```



```
else:
    if ((user == docs[0][0]['_id'] and passw == docs[0][0]['psw'])):

        return render_template("userhome.html")
    else:
        return render_template('goback.html',data="user name and password incorrect")
```

```
@app.route("/predict", methods=['GET', 'POST'])
```

```
def predict():
```

```
    if request.method == 'POST':
```

```
        file = request.files['fileupload']
```

```
        file.save('static/Out/Test.jpg')
```

```
import warnings
```

```
warnings.filterwarnings('ignore')
```

```
import tensorflow as tf
```

```
classifierLoad = tf.keras.models.load_model('body.h5')
```

```
import numpy as np
```

```
from keras.preprocessing import image
```

```
test_image = image.load_img('static/Out/Test.jpg', target_size=(200, 200))
img1 = cv2.imread('static/Out/Test.jpg')
# test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image, axis=0)
result = classifierLoad.predict(test_image)

result1 = ""

if result[0][0] == 1:

    result1 = "front"

elif result[0][1] == 1:

    result1 = "rear"

elif result[0][2] == 1:

    result1 = "side"


file = request.files['fileupload1']
file.save('static/Out/Test1.jpg')

import warnings
warnings.filterwarnings('ignore')
```

```
import tensorflow as tf
classifierLoad = tf.keras.models.load_model('level.h5')

import numpy as np
from keras.preprocessing import image

test_image = image.load_img('static/Out/Test1.jpg', target_size=(200, 200))
img1 = cv2.imread('static/Out/Test1.jpg')
# test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image, axis=0)
result = classifierLoad.predict(test_image)

result2 = ""

if result[0][0] == 1:

    result2 = "minor"

elif result[0][1] == 1:

    result2 = "moderate"

elif result[0][2] == 1:

    result2 = "severe"
```

```
if (result1 == "front" and result2 == "minor"):
    value = "3000 - 5000 INR"
elif (result1 == "front" and result2 == "moderate"):
    value = "6000 8000 INR"
elif (result1 == "front" and result2 == "severe"):
    value = "9000 11000 INR"

elif (result1 == "rear" and result2 == "minor"):
    value = "4000 - 6000 INR"

elif (result1 == "rear" and result2 == "moderate"):
    value = "7000 9000 INR"

elif (result1 == "rear" and result2 == "severe"):
    value = "11000 - 13000 INR"

elif (result1 == "side" and result2 == "minor"):
    value = "6000 - 8000 INR"

elif (result1 == "side" and result2 == "moderate"):
    value = "9000 - 11000 INR"

elif (result1 == "side" and result2 == "severe"):
    value = "12000 - 15000 INR"

else:
    value = "16000 - 50000 INR"
```

```
if __name__ == '__main__':  
    app.run(debug=True, use_reloader=True)
```

8.1 Test Cases

ADD TEST CASE

X

Name

Sample Test Case with a simple string

Difficulty

Easy

Score

10

Input

balloonbA




Expected Output

a

☒ Mark as a sample test case

Explanation

B I U x₂ x² |≡| ≡≡ |≡≡≡| ≡≡ Styles Normal Size A

 Source 

This is a sample test case to help the candidate understand the expected output from the solution.

Save

9. RESULTS

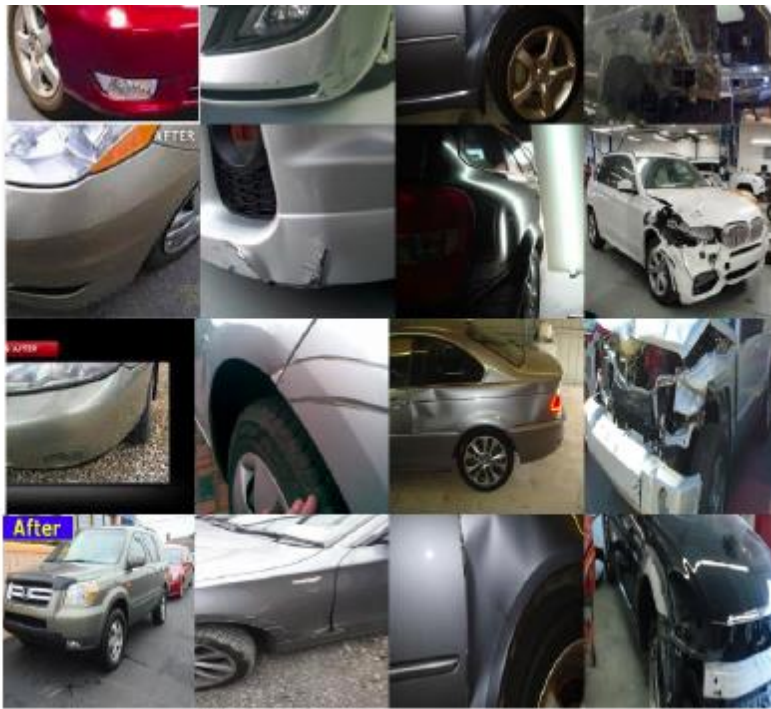
The results obtained using the techniques described in the previous chapter are detailed here. This

chapter starts by briefly giving details of the dataset developed and used for training and testing the

models developed. The results obtained when training the models described in 4, with the required

modifications and additions, employing some of the techniques already mentioned, according to

previously discussed criterions, are then presented and explained.



Category Name	Nr. of Images	Percentage
No Damage	7846	50%
Paint Damage	565	3.6%
Minor Damage	3021	19.6%
Major Damage	4239	27%
Total	15671	100%

Table 5.2: Dataset distribution across damage location scale

Category Name	Nr. of Images	Percentage
No Damage	7062	70.5%
Front	757	7.5%
Back	468	4.7%
Front-Side	733	7.3%
Side	488	4.8%
Back-Side	508	5%
Total	10016	100%

10. ADVANTAGES & DISADVANTAGES

- Estimating damages with over 90% accuracy: Our model for vehicle damage detection is trained with tens of thousands of real images of car accidents. Providing over 90% accuracy for the damage estimation and cost estimation.
- Speeding up the claims process : It is quite evident that using artificial intelligence solutions that can automatically approximate the cost of damages in minutes, would cut down on the extra time taken in the traditional claims process. Hence, insurance holders will not need to wait for weeks for their claim to be approved. This contributes to greatly improved customer experiences.
- Elimination of mistakes: Everything is subject to human error and AI solutions can help prevent inaccuracies in the estimation of damages and costs, which can save a lot of money. There is no need to worry about mistakes being made regarding the estimation of damage caused to a vehicle in an accident because the AI software does all the work in an efficient and reliable manner.
- Versatility: The application can be used for any vehicle, irrespective of type, model, origin, etc. This makes the solution versatile as it can be used on any

vehicle. In an accident, the model and the type of vehicle heavily influence the amount of money needed for damage reparation. AI can be used to estimate these costs in no time, as it can recognize the unique characteristics of each vehicle and act accordingly.

- Severity analysis: It is of paramount importance to know how severe the damages are. A map of the areas affected by the accident is drawn out and the severity is rated. This provides a user-friendly solution to understand how each part of the vehicle has been affected by the accident.

11. CONCLUSION

Computer vision and, in particular, image classification are fields of study where major breakthroughs were achieved in the last few years. These breakthroughs came mainly, but not only, from the usage of Deep Convolutional Neural Networks. Some of these breakthroughs were used in this project. These breakthroughs were applied in damage severity and location estimation in vehicles. These same tasks, and others that depend on them, are nowadays performed manually, without any assistance from computer systems. The results shown in this document lead to the conclusion that systems as the ones studied here, might be used to, at least help humans in these tasks. The two somewhat related tasks were attempted with a distinct level of success, using different methods. The results, shown in Chapter 5 clearly show that DCNN can achieve very interesting accuracy marks on both tasks, even with small datasets and modest computational resources. On the first task, the 76% accuracy mark achieved with a small ensemble, places this model in the same range of accuracy expected of untrained human beings. This is an encouraging mark obtained without a properly sized dataset available. With a more adequate dataset, results would likely be better and place this model on accuracy levels similar to trained human beings. Furthermore, bigger ensembles might bring even better results, but again, for that purpose, a better dataset would have to be gathered and more computational resources would be required. This mark of 76% may seem a little disappointing but a few remarks should be made. The task is very subjective in the sense that it is not always clear, not even to humans, how to categorise.

12. FUTURE SCOPE

The work presented here merely lays ground for further, more detailed work in this field. The conclusions and results obtained allow us to be optimistic about the capabilities of these models in tasks related to car damage detection and classification. Further studies, relying on more computational power and bigger datasets might be desirable. This would make it possible to use even more powerful

models to surpass human performance in related tasks. But the usage of more powerful models would be of no avail without access to a bigger, better dataset. Even with the relatively small models used here, a bigger, more diversified dataset, would likely allow the models to perform best. This work also lays ground for the attempt of more complex tasks related to damage.

13. APPENDIX

Source Code

```
{
"cells": [
  {
    "cell_type": "markdown",
    "metadata": {},
    "source": [
      "Importing the Image Data Generator Libraries "
    ]
  },
  {
    "cell_type": "code",
    "execution_count": 2,
    "metadata": {},
    "outputs": [],
    "source": [
      "import tensorflow as tf\n",
      "import keras\n",
      "from keras.preprocessing.image import ImageDataGenerator"
    ]
  }
]
```

```
},  
{  
  "cell_type": "markdown",  
  "metadata": {},  
  "source": [  
    "Configuring The Image Data Generator "  
  ]  
},  
{  
  "cell_type": "code",  
  "execution_count": 3,  
  "metadata": {},  
  "outputs": [],  
  "source": [  
    "#training images \n",  
    "train_datagen = ImageDataGenerator(\n",  
    "    rescale=1./255,\n",  
    "    shear_range=0.2,\n",  
    "    zoom_range=0.2,\n",  
    "    horizontal_flip=True)\n",  
    "\n",  
    "#val images \n",  
    "val_datagen = ImageDataGenerator(rescale=1./255)"  
  ]  
},
```

```

{
  "cell_type": "markdown",
  "metadata": {},
  "source": [
    "Applying the image generator to the body "
  ]
},
{
  "cell_type": "code",
  "execution_count": 5,
  "metadata": {},
  "outputs": [
    {
      "name": "stdout",
      "output_type": "stream",
      "text": [
        "Found 979 images belonging to 3 classes.\n"
      ]
    }
  ],
  "source": [
    "body_train_generator = train_datagen.flow_from_directory(\n",
    "    'V:\\\\WorkSpace\\\\IBM-Project-23426-1659882722\\\\Dataset\\\\Car",
    "    damage\\\\body\\\\training',\n",
    "    target_size=(150, 150),\n",

```

```
"    batch_size=32,\n"    class_mode='categorical')"\n]\n},\n{\n  "cell_type": "code",\n  "execution_count": 6,\n  "metadata": {},\n  "outputs": [\n    {\n      "name": "stdout",\n      "output_type": "stream",\n      "text": [\n        "Found 171 images belonging to 3 classes.\n"\n      ]\n    }\n  ],\n  "source": [\n    "body_val_generator = val_datagen.flow_from_directory(\n",\n    "    'V:\\\\WorkSpace\\\\IBM-Project-23426-1659882722\\\\Dataset\\\\Car\n",\n    "    damage\\\\body\\\\validation',\n    "    target_size=(150, 150),\n    "    batch_size=32,\n    "    class_mode='categorical')\n  ]\n}
```

```

},
{
  "cell_type": "markdown",
  "metadata": {},
  "source": [
    "Applying the image generator to the level "
  ]
},
{
  "cell_type": "code",
  "execution_count": 7,
  "metadata": {},
  "outputs": [
    {
      "name": "stdout",
      "output_type": "stream",
      "text": [
        "Found 979 images belonging to 3 classes.\n"
      ]
    }
  ],
  "source": [
    "level_train_generator = train_datagen.flow_from_directory(\n",
    "    'V:\\\\Workspace\\\\IBM-Project-23426-1659882722\\\\Dataset\\\\Car",
    "    damage\\\\level\\\\training',\n"
  ]

```

```

"    target_size=(150, 150),\n",
"    batch_size=32,\n",
"    class_mode='categorical')"
]
},
{
"cell_type": "code",
"execution_count": 8,
"metadata": {},
"outputs": [
{
"name": "stdout",
"output_type": "stream",
"text": [
"Found 171 images belonging to 3 classes.\n"
]
}
],
"source": [
"level_val_generator = val_datagen.flow_from_directory(\n",
"    'V:\\\\WorkSpace\\\\IBM-Project-23426-1659882722\\\\Dataset\\\\Car",
"damage\\\\level\\\\validation',\n",
"    target_size=(150, 150),\n",
"    batch_size=32,\n",
"    class_mode='categorical')"
```

```
]
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {},
  "outputs": [],
  "source": []
}
],
"metadata": {
  "kernelspec": {
    "display_name": "Python 3.7.9 64-bit (microsoft store)",
    "language": "python",
    "name": "python3"
  },
  "language_info": {
    "codemirror_mode": {
      "name": "ipython",
      "version": 3
    },
    "file_extension": ".py",
    "mimetype": "text/x-python",
    "name": "python",
    "nbconvert_exporter": "python",
```

```
"pygments_lexer": "ipython3",
"version": "3.7.9"
},
"orig_nbformat": 4,
"vscode": {
  "interpreter": {
    "hash":
    "b81d285a39ce6bce5fc3d0e228a6124a883049b7dbc9a3a4527a4b38646ff66a"
  }
}
},
"nbformat": 4,
"nbformat_minor": 2
}
```

GitHub & Project Demo Link:

https://drive.google.com/file/d/1fequwlW0TA-RRCXypBPjRpPPU_YoyYIJ/view

[IBM-EPBL/IBM-Project-5988-1658821738](#)