

ASSESSING CAR DAMAGE WITH CONVOLUTIONAL NEURAL NETWORKS

A PROJECT REPORT PHASE-II

Submitted in partial fulfilment of requirements for the award of

Bachelor of Engineering

in

Electronics and Communication Engineering

By

M.SRIKARGANESH(39130269)

P.KALYAN(39130333)



DEPERTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SCHOOL OF ELECTRONICS

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI – 600119

APRIL – 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICAITON ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Final Year Project Report is the bonafide work of M.Srikarganesh(39130269)and P kalyan(39130333) who have done the project work as an individual who carried out the project entitled "**ASSESSING CAR DAMAGE WITH CONVOLUTION NEURAL NETWORKS**" under our supervision from JAN 2023 to APRIL 2023.

Internal Guide

Dr.T.VINO

Department of ECE

Head of the Department

Dr. T. RAVI, M.E., Ph.D.,

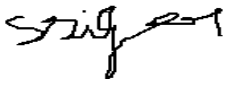
Submitted for Viva voce Examination held on 23.4.2023

Internal Examiner

External Examiner

DECLARATION

We, M.Srikarganesh(39130269) and P kalyan(39130333) here by declare that the Project Report entitled **“CAR DAMAGE ASSESSING USING CONVOLUTION NEURAL NETWORK”** done by us under the guidance of **Dr.T.Vino**, at Sathyabama Institute of Science and technology is submitted in partial fulfilment of the requirements of the award of Bachelor of Engineering degree in **Electronics and Communication Engineering**.

1. 

2. 

DATE: 23.4.2023

PLACE: Chennai

SIGNATURE OF THE CANDIDATES

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing its successfully. We are grateful to them.

We convey our thanks to **Dr. N.M. NANDHITHA M.E., Ph.D.**, Dean, School of Electronics and Communication Engineering and **Dr. T. RAVI M.E., Ph.D.** Head of the Department, Department of Electronics and Communication Engineering for providing me necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Dr.T.Vino**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of our phase-2 project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the **Department of Electronics and Communication Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Visual image classification is a research area that involves both computer vision and machine learning. The task of visually classifying an object consists in assigning an object to a category, or set of categories the object belongs to. Traditionally, visual classification tasks are performed using a two layered system, made up of a first layer featuring an out-of-the-shelf feature extractor and detector, and a second classifier layer. In most recent years, convolutional neural networks have been shown to outperform such previously used systems. Cars have a paramount role in today's world, and being able to automatically classify damages in cars is of great interest specially to the car insurance industry. Car insurance companies deal with car inspections on a daily basis. Such inspections are a manual, lengthy and sometimes faulty processes. Processes that bring costs and inconveniences to costumers and insurance companies alike. Even though the total replacement of such manual inspection processes might still be far away, developing systems to aid, accelerate or enhance the process might be possible with today's technology.

Nowadays, the proliferation of automobile industries is directly related to the number of claims in insurance companies. Those companies are facing many simultaneous claims and solving claims leakage. In Advanced Artificial Intelligence (AI), machine learning and deep learning algorithms can help to solve these kinds of problems for insurance industries.

- Manual estimation of damages in fields like construction, vehicular accidents has been the mainstay of the insurance business.
- However, such methods are replete with biases and inaccurate estimations.
- This project deals with estimating car damage, primarily with auto insurers as our key potential customers.
- For this purpose, three distinct Transfer Learning approaches are used which detect the presence of damage, location, and severity of the damage.
- The basis for algorithms used lies in Convolutional Neural Networks, customized to optimize accuracy.

This research fine-tunes a number of existing approaches and opens doors for collaboration in image recognition, particularly for the car insurance domain

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF FIGURES AND LIST OF TABLES	vii
	LIST OF ABBREVIATIONS	vii
1.	INTRODUCTION	1
	1.1INTRODUCTION	1
2.	LITERATURE SURVEY	15
3.	SYSTEM ANALYSIS	
	3.1 EXISTING SYSTEM	19
	3.2 PROPOSED SYSTEM	20
	3.3 FEASIBILITY STUDY	20
	3.4 REQUIREMENT SPECIFICATION	22
	3.5LANGUAGE SPECIFICATION–PYTHON	23
4.	SYSTEM DESIGN	
	4.1SYSTEM ARCHITECTURE	24
	4.2 DATA FLOW DIAGRAM	24
	4.3 ENTITY RELATIONSHIP DIAGRAM	26
	4.4USE-CASE DIAGRAM	28
	4.5ACTIVITY DIAGRAM	28
	4.6 SEQUENCE DIAGRAM	29
	4.7 CLASS DIAGRAM	29
	4.8 ER DIAGRAM	30
5.	MODULE DESCRIPTION	
	5.1 MODULE 1	32
	5.2 MODULE 2	35
	5.3 MODULE 3	36
6.	TESTING	
	6.1 TYPES OF TESTING	38
	6.2. TESTING TECHNIQUES	40
7.	CONCLUSION	
	7.1 CONCLUSION	42
	APPENDIX 1-CODE	43
	APPENDIX 2-OUTPUT	45
	REFERENCES	48

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
1.1	SCRATCH	2
1.2	MISCELLANEOUS DAMAGE	4
1.4	FLOW CHART OF PROCESS	10
1.5	CNN OVERVIEW	11
1.6	FLOW CHART OF VGG16 MODEL	12
4.1	SYSTEM ARCHITECTURE	24
4.2	DATA FLOW DIAGRAM 0	26
4.3	DATA FLOW DIAGRAM 1	26
4.4	ER DIAGRAM	27
4.5	USE-CASE DIAGRAM	28
4.6	SEQUENCE DIAGRAM	29
4.7	CLASS DIAGRAM	30
4.8	ER DIAGRAM	31
5.1	IMAGE PRE-PROCESSING	33
5.2	IMAGE PRE-PROCESSING	34
6.2.1	BLACK BOX TESTING	40
6.2.2	WHITE BOX TESTING	41
9.1	OUTPUT SCREENSHOT	47
9.2	OUTPUT SCREENSHOT	47

LIST OF TABLES

CHAPTER NO	TITLE	PAGE NO
1.3	CNN LAYERS USE	9
5.1	DATASET DESCRIPTION	32

LIST OF ACRONYMS AND ABBREVIATIONS

CNN-Convolutional Neural Network

RNN-Recurrent Neural Network

CHAPTER 1

INTRODUCTION

General Background

Insurance is a very old industry and it has been quite resistant to change until recently. Manual validation of large-scale claims is not sufficiently quick in meeting dealing with large numbers of insurance claims. Claim amounts are determined by the type of damage and the affected part of the vehicle and an automated car insurance claim processing system - that efficiently detects and estimates damages - could be of great help. Automatic damage assessment employing analysis of images has proved to be fast and efficient, and will improve further as more and more data is collected. Deep learning enables us to automatically detect scratches, teeth, rust, and breakages, to determine the affected part of the vehicle and to quantify the severity of the damage. After processing, reports can be generated with a list of damages and an estimate of the cost of repair. Ability to detect damages in dataset taken cars is a subset of image classification because, at its most fundamental level, identifying claims in images entails categorising an image into a particular type as well as collection of classes. There has been a great deal of research on image classification, but there haven't been many works on car visual damage detection. Nonetheless, and be capable of detecting car damage automatically is a research topic with numerous practical uses. Automobile insurance companies deal with car accidents on a daily basis. Cars are frequently required to be serviced for damage, which is problematic for clients as well as extremely expensive for companies. As a result, it's critical to be able to automate car damage detection, making it easy for customers and affordable.

TYPES OF CAR DAMAGES

The most common car damages after an accident are cosmetic and structural. Some of these damages are obvious, while others may be harder to notice. Regardless of the cause of the damage, it is vital to understand the different types and how they can affect your car. Knowing the various types of car damage can help you analyze them. Through vehicle inspection automation, users can quickly gain insights about the scope and extent of the damages. There are a variety of damages that can occur to a car. The scope and

extent of the damages vary with how the car's body comes into contact with external objects. How the vehicle sustained the damage will also impact its severity and extent. Some damages will be repairable, while some can harm the car part to such an extent that it will have to be replaced by a new part. Car damages can be classified into three primary categories, based on which component they impact. These categories are metal damage, glass damage, and miscellaneous damage.

METAL DAMAGE

Metal damages occur on the metallic parts of the car's body. These parts include the bumper, hood, doors, dickey, and other metal parts. Metal damages can be further classified into the following variations: dent, scratch, and tear.

DENT

Dent is the damage caused when the metal body is pressed inside when it experiences pressure from external objects. This pressure creates a concavity on the metal surface. A considerable number of dents can also occur when a car crashes.

SCARTCH

Scratches are the most common damages that occur on metal surfaces. When a metallic car part moves or rubs against another hard or sharp object's surface, scratches form. Scratches can also form if a hard or sharp object runs over the car's metallic surface. The following set of images depicts scratch damage.



Fig 1.1:Scratch

TEAR:

When a car sustains tear damage, the metal part is split into pieces. This tear can occur either at the edge of the car part or inside it.

GLASS DAMAGE:

Like metal damage on metal parts, glass damages occur on the glass part of the car body. The components it impacts include the windshield, back glass, car windows, headlight, and taillight. Glass damage can be further categorized into the following variations: crack, chip, spider crack, and large range glass damage.

CRACK:

Crack damage occurs on the windshield of the car.

MISCELLANEOUS DAMAGE:

Damages that do not fall under either metal or glass damage are classified as miscellaneous damages. Dislocation of car parts also falls in this category.

Automatically detecting damages in images containing cars is a special case of an image classification task because, at its most basic level, detecting damages in images consists of assigning an image to a particular category or set of categories. There is a lot of research done in image classification but, there aren't many works, that I am aware of, in car visual damage detection. Nevertheless, being able to automatically detect damages in cars is a research topic that has many possible real-world applications. Car insurance companies and car rentals have to deal with car damages on a daily basis. It often happens that cars have to be inspected for damages, often in circumstances that are inconvenient for customers, and costly to the companies themselves. It is therefore important to be able to automate car damage detection, making it both more convenient and cheaper. This report starts by giving a broad overview of the current existing solutions for the problem of automatically identifying car damages in images. Solutions and strategies to solve problems that are only similar to the ones addressed here, will often be mentioned due to the lack of more specific research on the field of automated car damage detection. Some of these solutions, mainly the more promising ones, are later applied to the problem of automatically identifying damages in car images. A comparison of results with the ones obtained by previous works on the same problem will be given



Fig 1.2: Miscellaneous Damage

whenever possible. The methodology used throughout this project consists of adapting solutions that are known to work in a variety of image classification problems, to the particular problem of visually identifying damages in cars and then evaluating their performance relatively to other existing solutions. Several classification systems will be tested, favouring the ones that can be used in practice. The dataset used to validate and evaluate performance of the developed solutions will be composed of images previously gathered from search engines, labelled for training and testing purposes. Do you know that the insurance industry is one of the first industry which invested in innovation, the latest technology and AI? In today's world, when the rate of car accidents are increasing depending on expanding car industries, car insurance companies waste millions of dollars annually, due to claims leakage. The sense of AI technology based on machine and deep learning can solve problems such as analyzing and processing data, detecting frauds, lessening risks and automating claim process in insurance industries. So, insurance organizations are looking for faster damage assessment and agreement of claims. The manual inspection of vehicles for external damages is a process known to most and particularly prevalent in the vehicle rental industry, where the accuracy of damage detection is of high importance. Not only are such visual inspections time consuming for lessee as well as a rental company, the high level of complaints often due to unexpected supplementary charges for subjectively assessed vehicles damages is leading to negative customer experiences as well as an additional cost for all parties involved for further investigation and formal responses before an acceptable resolution can be

achieved. These shortcomings call for an objective solution. With the significant progress made in object detection and instance segmentation, this field of computer vision provides a promising solution. With the possibility to replace manual inspections with visual automated ones, the dependency on the expertise of the assessor is removed. Furthermore, artificial damage recognition reduces the time and effort required for an inspection while also providing a reproducible and objective assessment for all parties involved. While fully supervised models, such as Mask R-CNN have proven to achieve high accuracy rates for the detection of objects the required effort to annotate a large number of images to achieve these results requires high levels of effort and time.

Currently, AI is advancing at a great pace and deep learning is one of the contributors to that. It is good to understand the basics of deep learning as they are changing the world we live. Deep learning is a sub-field of machine learning dealing with algorithms inspired by the structure and function of the brain called artificial neural networks. In other words, It mirrors the functioning of our brains Deep learning algorithms are similar to how nervous system structured where each neuron connected each other and passing information. One of the differences between machine learning and deep learning model is on the feature extraction area. Feature extraction is done by human in machine learning whereas deep learning model figures out by itself Today, in the car insurance industry, a lot of money is wasted due to claims leakage. Money lost through claims management inefficiencies that ultimately result from failures in existing processes (manual and automated). In other words, it's the difference between what you did spend and what you should have spent on a claim. The cause can be procedural, such as from inefficient claim processing or improper/errant payments, or from human error, such as poor decision-making, customer service, or even fraud. Claim Leakage is often discovered through an audit of closed claim files.

DEEP LEARNING

Deep-learning networks are distinguished from the more commonplace single-hidden layer neural networks by their depth; that is, the number of node layers through which data must pass in a multistep process of pattern recognition. Earlier versions of neural networks such as the first perceptron were shallow, composed of one input and one output layer, and at most one hidden layer in between. More than three layers (including

input and output) qualify as “deep” learning. So deep is not just a buzzword to make algorithms seem like they read Sartre and listen to bands you haven’t heard of yet. It is a strictly defined term that means more than one hidden layer. In deep-learning networks, each layer of nodes trains on a distinct set of features based on the previous layer’s output. The further you advance into the neural net, the more complex the features your nodes can recognize since they aggregate and recombine features from the previous layer. This is known as feature hierarchy, and it is a hierarchy of increasing complexity and abstraction. It makes deep-learning networks capable of handling very large, high-dimensional data sets with billions of parameters that pass through nonlinear functions. Above all, these neural nets are capable of discovering latent structures within unlabeled, unstructured data, which is the vast majority of data in the world. Another word for unstructured data is raw media; i.e., pictures, texts, video, and audio recordings. Therefore, one of the problems deep learning solves best is in processing and clustering the world’s raw, unlabeled media, discerning similarities and anomalies in data that no human has organized in a relational database or ever put a name to. For example, deep learning can take a million images, and cluster them according to their similarities: cats in one corner, ice breakers in another, and in a third all the photos of your grandmother. This is the basis of so-called smart photo albums. Deep-learning networks perform automatic feature extraction without human intervention, unlike most traditional machine-learning algorithms.

Given that feature extraction is a task that can take teams of data scientists years to accomplish, deep learning is a way to circumvent the chokepoint of limited experts. It augments the powers of small data science teams, which by their nature do not scale. When training on unlabeled data, each node layer in a deep network learns features automatically by repeatedly trying to reconstruct the input from which it draws its samples, attempting to minimize the difference between the network’s guesses and the probability distribution of the input data itself. Restricted Boltzmann machines, for example, create so-called reconstructions in this manner. In the process, these neural networks learn to recognize correlations between certain relevant features and optimal results – they draw connections between feature signals and what those features represent, whether it be a full reconstruction, or with labeled data. A deep-learning network trained on labeled data

can then be applied to unstructured data, giving it access to much more input than machine-learning nets.

NEURAL NETWORKS

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria. The concept of neural networks, which has its roots in artificial intelligence, is swiftly gaining popularity in the development of trading systems. A neural network works similarly to the human brain's neural network. A "neuron" in a neural network is a mathematical function that collects and classifies information according to a specific architecture. The network bears a strong resemblance to statistical methods such as curve fitting and regression analysis. A neural network contains layers of interconnected nodes. Each node is a perceptron and is similar to multiple linear regression. The perceptron feeds the signal produced by a multiple linear regression into an activation function that may be nonlinear. 14 In a multi-layered perceptron (MLP), perceptron are arranged in interconnected layers. The input layer collects input patterns. The output layer has classifications or output signals to which input patterns may map. Hidden layers fine-tune the input weightings until the neural network's margin of error is minimal. It is hypothesized that hidden layers extrapolate salient features in the input data that have predictive power regarding the outputs. This describes feature extraction, which accomplishes a utility similar to statistical techniques such as principal component analysis.

CNN

Convolutional Neural Network is an algorithm of Deep Learning. That is used for Image Recognition and in Natural Language Processing. Convolutional Neural Network (CNN) takes an image to identify its features and predict it. It captures the spatial and temporal dependencies of the image. Each CNN layer learns filters of increasing complexity. The first layers learn basic feature detection filters like edges, corners. The middle layer learns filters that detect parts of objects. The last layers have higher representations they learn to recognize full objects in different shapes and positions. Suppose, when you see some

image of a Dog, your brain focuses on certain features of the dog to identify. These features may be the dog's ears, eyes, or it may be anything else. Based on these features your brain gives you a signal that this is a dog. Similarly, Convolutional Neural Network processes the image and identifies it based on certain features. Convolutional Neural Network is gaining so much popularity over artificial neural networks.

Steps in Convolutional Neural Network In a Convolutional Neural Network, there are basically the following steps -

1. Convolution Operation.
2. ReLU Layer.
3. Pooling.
4. Flattening.
5. Full Connection.

Generally, a convolutional neural network architecture generally has several components:

- A convolution layer - you can think of this layer as “what relevant features are we picking up in an image?” In a convolutional neural network, we have multiple convolutional layers that extract low to high-level features depending on what specific layer we are focusing on. To give an (over-simplified) intuition, earlier convolutional layers pick up lower-level features (i.e. like lines and edges) while later convolutional layers pick up higher-level features based on inputs from lower-level features (i.e. shapes, structures) - analogous to how vision works in the human brain.

- A pooling layer - convolutional neural networks are typically used for image classification. However, images are high-dimensional data - so we would prefer to reduce the dimensionality to minimize the possibility of overfitting. Pooling essentially reduces the spatial dimensions of the image based on certain mathematical operations such as average or max-pooling (there's a nice graphic here). We generally incorporate pooling since it

(1) generally, acts as a noise suppressant

(2) makes it invariant to translation movement for image classification and

(3) helps capture essential structural features of the represented images without being bogged down by the fine details.

- Fully-connected layer - You can think of a series of convolution and pooling operations

as dimensionality reduction steps prior to passing this information over to the fully connected (Dense) layer. Essentially, what the fully connected layer does is that it takes the "compressed" representation of the image and it tries to fit a basic NN (multi-layer perceptron) when doing classification.

Table 1.3 CNN layers used

S.NO	CNN Layers
1.	Convolutional layer
2.	Pooling Layer
3.	Fully connected Layer

The change in paradigm that the usage of NNs encompasses is a very important one. Prior to the usage of NNs in image classification, the practitioner had to use explicitly coded algorithms for detecting features. This was a job that often involved a lot of work. Machine Learning was not used throughout the whole system, only the last layer of the system, the classifier, had the capacity to learn and adapt itself to the specific problem. NNs, and especially CNNs, dramatically change that. CNN's extract features from images and learns how to do it. The practitioner doesn't need to craft complicated hard-coded algorithms to extract those features. Since the introduction of the first CNNs in 2012, all the winning teams for the classification task have used different types of CNNs. CNN's systems now clearly outperform other image classification systems such as the ones described. State-of-the-art CNNs have even crossed the boundary of what is considered to be the error rate for human beings in image classification tasks. Ever since its introduction in 2012, CNNs have been making steady progress in decreasing the error rate in both tasks. This is due to the introduction of new strategies that further improve the convergence and training speed of CNNs. CNNs are a particular kind of NNs that give their name to the fact that they make extensive use of Convolutional Layers.

Convolutional Layers consist of a learnable filter of fixed size (kernel) to be applied to images. The name comes from the fact that at each forward pass, the learned filter is

convolved with the image resulting on a 2-dimentional map. Another type of layer widely used in CNNs are the pooling layers. These layers perform a type of down-sampling on the input signal. There are various types of Pooling layers, max-pooling being the most common. ReLUs or Rectified Linear Units, are also widely used in CNNs. ReLUs are units that apply a function similar to Equation 2.1. There are often alternatives for these types of units, Equation 2.2 and 2.3 show two of them.

$$\text{ReLU} : f(x) = \max(0, x) \quad (2.1)$$

$$\text{Tanh} : f(x) = \tanh(x) \quad (2.2)$$

$$\text{Sigm} : f(x) = (1 + e^{-x})^{-1} \quad (2.3)$$

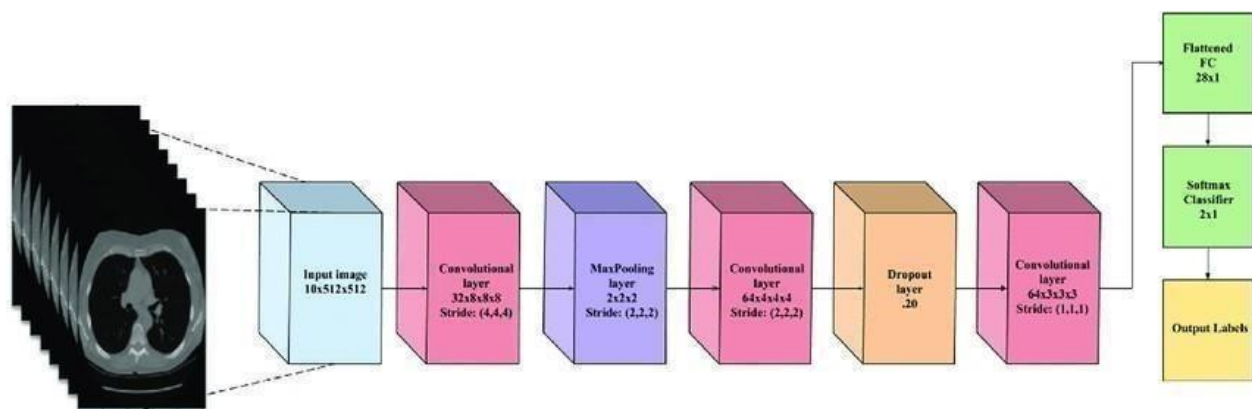


Fig 1.4:FlowChart Of Process

Fully Connected Layers are used too, as in more standard NNs. These layers consist of an array of neurons, each of which is connected to every single output of the previous layer. Loss Layers are typically used at the end of the CNN to figure out the penalty for a given output and to provide feedback used by the CNN to learn. In addition to these more standard set of layers, modern CNNs make use of a very extensive set of techniques and strategies that greatly enhance their performance. Dropout is one of such techniques. It consists on temporarily deactivating some neurons within the net, preventing it from over-fitting. This temporary deactivation of neurons is typically applied only to 8 Fully-connected layers. CNNs nowadays also make extensive use of GPU processing power that allows for a faster training, an idea pioneered in 2012 that is now a standard. Evidence shows that one way of improving a CNN performance is by stacking more layers and making the network effectively deeper. The problem that often arises is that as more layers are stacked, the performance saturates and eventually starts to decrease at some point. To address this issue, the idea of deep residual networks has been proposed.

Residual Networks try, with minor changes to the architecture, and with virtually no impact on performance, to approximate residual functions instead of standard functions.

If we have a CNN with input x and we want the output to approximate a function $O(x)$, we can instead, approximate $F(x) = O(x) - x$ and then compute $O(x)$ from $F(x)$ since $O(x) = F(x) + x$ as shown in Figure 2.2. Evidence shows that CNNs train better and converge faster when the approximated function is in fact $F(x)$, a residual function. This approach opens door for usage of even deeper and more powerful models. Some pre-processing techniques are also used to improve the data fed to the CNN. These techniques include normalization and zero-centering of the data. These techniques are mainly used in order to overcome the vanishing and exploding gradient problem, that often affects NN systems' training. Both problems arise when models are trained using gradient-based methods. Gradients are often computed using the chain rule. In case of near-zero gradients, this lead to several very small numbers being multiplied together in order to calculate the changes in the last layers. This causes the last layers to suffer virtually no change at all in its outputs as the training progresses.

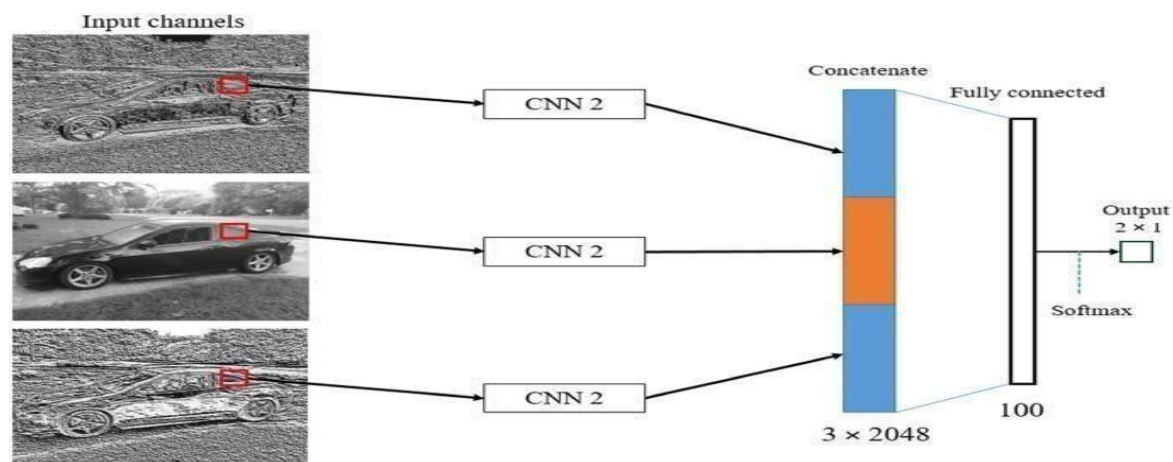


Fig 1.5: CNN Overview

The opposite happens when activation functions have high derivative values, leading 9 Related Work to the uncontrolled increase of the gradient of the last layers. As NN systems tend to require a lot of examples, better results are obtained when data augmentation techniques are used. CNN frameworks often incorporate mechanisms to augment data. Cropping is a very simple technique where several crops of a single image

are fed to the NN, augmenting the dataset by a factor equal to the number of crops. Rotations [KSH12] of the image are also used, augmenting the dataset by a factor equal to the number of rotations done. The techniques summarized here, and others, together with a very complete set of open-source tools, foster the usage of CNNs, allowing for the fast creation of systems that perform very well in noisy datasets, clearly outperforming older image classification systems.

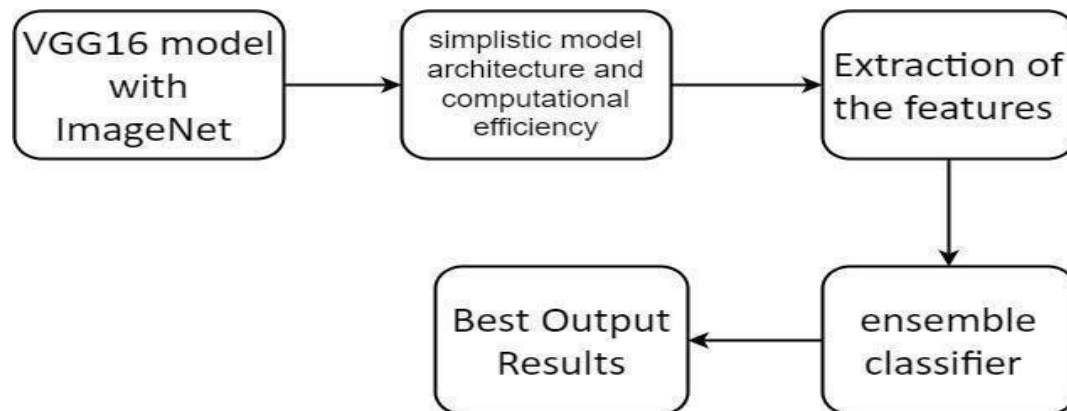


Fig 1.6:Flowchart Of VGG16 Model

Problem Statement

In this document, we'll see the development of a prototype of a system, capable of identifying, and possibly locating damages in car images and evaluate its performance by comparing it to existing systems. A dataset will be gathered for the purpose of training the system and evaluating its performance. This dataset will be built by gathering images from search engines and other publicly available datasets.

Objective

Cars have a central role in today's world. Many people use cars every single day. There are businesses that depend on cars, and some of them base their activity in dealing with car damages and accidents. Non automated visual inspection of cars is a common task in some businesses. This is mainly the case of insurance companies. Cars are inspected both when a new coverage is bought and when an issue is reported to the insurance company. In both situations, inspections are only done by sampling. This happens because the cost of having an expert drive down to an often damaged vehicle, in order to

inspect it is too high. It also causes delays for customer and company alike. This is especially important in cases where the insurance is not bought in one of the insurance companies dealerships, e.g. car insurance bought through the Internet.

If the car for which the insurance was bought had to be inspected by an insurance company, the whole purpose of selling insurance through the Internet would be defeated; The customer would no longer have the convenience of quickly buying an insurance, since he would have to wait for the inspection to be performed before the effectiveness of the insurance was granted. On the other hand, the insurance company would no longer have the benefit of reduced costs in selling its products through these channels, because an expert would have to drive down to the insured car location, possibly a remote location, in order to perform the inspection. Both these inconveniences would be mitigated if the client uploaded some photos of the car to be automatically inspected by a system that could assert if the car has damages or not. Insurance companies often mitigate the problem by reducing the number of inspections they make. This is achieved either by not making them at all, or by making them by sampling, in an attempt to reduce fraud while not spending too much money inspecting it.

Although an automated inspection might not be as effective as an inspection performed by an expert, it makes it possible to inspect all the vehicles, instead of just a few. On the other end, performing such inspections alongside with expert inspections might contribute to detect systematic biases, making it easier to discover potential fraudulent inspection processes or mistakes. The possibility of classifying and distinguishing between different kinds of damages opens up the door for other more complex car damage related problems to be solved in an automated way. Other regression and classification problems that might include repair cost estimation or damaged parts estimation, problems of the uttermost importance for car insurance and car repair companies. These processes are also susceptible of fraud, and are also not automated nowadays.

Even tough cars are very important to many people, they have been somehow neglected by the computer vision research community. Cars are very challenging to work with in computer vision. They present great variation in shape and form by slight variations of viewpoint. This, combined with the fact that cars usually have highly reflective metal bodies that make them very susceptible to inter object reflection, makes the problem of

automatically of detecting damages in car images a very challenging one. This is especially the case if the damages are relatively small, when they are easily mistaken by reflections. There is some research done in the field of automatic damage detection in vehicles. The approach taken often involves the usage of CAD models to estimate image perspective and assert differences between what's in the image and what the image is expected to have . On the other hand, there is the research done in the broader field of image classification. This research often takes a different strategy in solving image classification problems. Image classification technology has endured tremendous changes in the last decade. We now have image classification systems that can allegedly outperform human beings in some tasks, a feat that could not be achieved by decades long research with traditional approaches. While the non traditional approach followed made perfect sense when the state of the art image classification systems yielded results that were not good enough for this tasks, it now makes complete sense to apply the more performant and up to date state of the art, technology to tackle the same problems, hopefully with better results. Automated car damage detection using images is, therefore, an interesting research topic both due to the lack of work done in the field and due to the hope that new state-of-the-art methods might bring serious performance improvements. Also, due to the importance cars have in many places and the fact that cars present interesting unsolved challenges that state-of-the-art image classification technologies might now be able to overcome, this make this an even more interesting research topic.

CHAPTER 2

LITERATURE REVIEWS

[1] Maleika Heenaye–Mamode Khan, Mohammad Zafir Hussein Sk Heerah, Zuhairah Basgeeth, “Automated Detection of Multi-class Vehicle Exterior Damages using Deep Learning”, International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), 2021

Maleika Heenaye has deployed an application in this paper for the automatic detection and classification of vehicle damages, which can be used by insurance companies to process claims or by the police department to record accidents. Manually identifying the types and severity of vehicle damage after an accident can be time-consuming. An automated damage detection application can help with insurance claims. Convolutional Neural Networks (CNN) have had great success in object classification. However, CNN has not been thoroughly investigated or applied for multiclass classifications of vehicle damages. In this paper, pre-trained CNN models, MobileNet and VGG19 are adapted and used in transfer learning on the large built dataset. This application achieved a MobileNet accuracy of 70% and a VGG19 accuracy of 50%.

[2] Najmeddine Dhieb, Hakim Ghazzai, Hichem Besbes, Yehia Massoud, “A Very Deep Transfer Learning Model for Vehicle Damage Detection and Localization”, 31st International Conference on Microelectronics (ICM), 2020

Najmeddine Dhieb proposes efficient and streamlined deep learning-based architectures for vehicle damage identification and localization in this paper. For feature extraction and damage identification, the proposed solution incorporates deep learning, instance segmentation, and transfer learning techniques. Its goal is to automatically detect vehicle damage, locate it, classify its severity levels, and visualise it by contouring its exact location. Numerical results show that our proposed transfer learning solution, which is based on an Inception-ResnetV2 pre-trained model followed by a fully linked neural network, outperforms another pre-trained model, VGG16, in feature extraction and damage detection/localization.

[3] Atharva Shirode, Tejas Rathod, Parth Wanjari, Aparna Halbe, “Car Damage Detection and Assessment Using CNN”, IEEE Delhi Section Conference (DELCON), 2022

If the car is insured, a representative from the insurance industry comes to the customer's home to conduct a survey and prepare a report. Manual verification is a time-consuming process. However, with significant advancements in the field of deep learning algorithms, it can be used to solve these problems in the insurance industry. Two CNN models are used in the proposed

solution. VGG16 is used to detect vehicle damage, as well as the location and severity of the damage. Mask RCNN is used to isolate the damaged region. Both models provide a reasonable estimate of the vehicle's damage, allowing insurance companies to proceed with insurance claims without wasting resources and time on manual verification.

[4] Sruthy C M, Sandra Kunjumon, Nandakumar R, “Car Damage Identification and Categorization Using Various Transfer Learning Models”, 5th International Conference on Trends in Electronics and Informatics (ICOEI), 2021

This study used the transfer learning-based models Inception V3, Xception, VGG16, VGG19, ResNet50, and MobileNet from Keras's library to train our model to predict damage and compare their efficacy. The proposed dataset is trained with these pre-trained models to achieve maximum accuracy and speed with a minimal loss so that the model can be used to predict claims in real-life. When compared to other models, MobileNet is more accurate and has a faster training time. The accuracy in forecasting damage and categorizing it into different types was 97.28%, which is significantly better than previous results in a similar test set.

[5] Umer Waqas, Nimra Akram, Soohwa Kim, Donghun Lee, Jihoon Jeon, “Vehicle Damage Classification and Fraudulent Image Detection Including Moiré Effect Using Deep Learning”, IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), 2020

Umer Waqas considers the problem of car damage classification in this paper, where classifications include medium damage, huge damage, and no damage. For classification, the MobileNet model is proposed using deep learning techniques and transfer learning. Furthermore, moving toward automation comes with a variety of challenges; users can upload bogus images such as screenshots or take screenshots of computer screens, for example. To address this issue, a hybrid approach is proposed in which only authentic images are provided as input to an algorithm for damage classification. To detect fraudulent images, moiré effect detection and metadata analysis are used. Damage classification accuracy is 95%, and moiré effect detection accuracy is 99%.

[6] Qinghui Zhang, Xianing Chang, Shanfeng Bian, “Vehicle-Damage-Detection Segmentation Algorithm Based on Improved Mask RCNN”, IEEE Access (Volume: 8), 2020

This paper proposes a vehicle-damage-detection segmentation algorithm based on transfer learning and an improved mask regional convolutional neural network (Mask RCNN) to solve traffic accident compensation problems quickly. The experiment begins by collecting car damage

images for preprocessing before using Labelme to create data set labels that are divided into training and test sets. The residual network (ResNet) is optimized, and feature extraction is carried out with the help of the Feature Pyramid Network (FPN). The spatial information of the feature map is then preserved in ROIAlign via bilinear interpolation, and different weights are presented in the loss function for different-scale targets. Finally, the results of training and testing on a self-created dedicated dataset show that the improved Mask RCNN has a higher Average Precision (AP) value, masking accuracy, and detection accuracy.

[7] Xianglei Zhu, Sen Liu, Peng Zhang, Yihai Duan, “A Unified Framework of Intelligent Vehicle Damage Assessment based on Computer Vision Technology”, IEEE 2nd International Conference on Automation, Electronics and Electrical Engineering (AUTEEE), 2020

Currently, the vehicle damage assessment industry requires a large amount of workforce, and new automatic intelligent damage identification technology can significantly reduce industrial costs. In this paper, Xianglei Zhu proposes a framework for intelligent vehicle damage assessment algorithms based on object detection and image classification technology. This algorithm can automatically identify the damage position, type, and degree based on photos provided by users, allowing it to offer appropriate maintenance pricing while meeting actual application requirements.

[8] W.A. Rukshala Harshani, Kaneeka Vidanage , “Image processing based severity and cost prediction of damages in the vehicle body: A computational intelligence approach”, National Information Technology Conference (NITC), 2018

This paper describes a novel method for predicting the severity and cost of vehicle body damage using 2D images. Thus, if the vehicle body sustains damage, the driver does not have to wait for the insurance company to calculate the appraisal; instead, he can get a rough estimate of how much it will cost to repair the damage. When an image is uploaded, the system processes it to identify the dent. The vehicle is then classified into the appropriate severity class, taking into account features such as the make, model, and year of manufacture. Following that, the severity generated by the damage image is mapped with the cost rules, which are built based on the vehicle's features such as the make, model, and manufactured year. Finally, the user is notified of the damage severity class and the average cost of repairing the damage.

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

Automobile insurance claims in Indonesia might take a significant amount of time. Consequently, the purpose of this study is to develop a unified deep-learning system to expedite the handling of auto insurance claims. Damage to a vehicle may be segmented, detected, and categorized according to severity thanks to three key duties. There are three distinct datasets included here: one for damaged segmentation, another for segmenting automobile components, and a third for classifying car damage. In this work, deep transfer learning techniques are employed to efficiently train the models used in the research. In the segmentation task, we use the Mask R-CNN model, whereas in the classification task, we evaluate EfficientNet and MobileNetV2.

[10] Yunxi Wu, Yuehong Zhou and Xiaolong Peng, "Measurement and repair of structural body damage to cars in collisions," 2011 Second International Conference on Mechanic Automation and Control Engineering, Inner Mongolia, China, 2011, pp. 1381-1385, doi: 10.1109/MACE.2011.5987202.

The structural dimensions of cars that have been in collisions and have sustained structural damage must be recovered in order to repair these vehicles safely and reliably. This paper analyzes the ideal characteristics and the types of deformation on car body structures during frontal and side impacts. Using a multi-purpose pulling device, accurate 3D measurement system for car bodies, and multi-purpose fixtures and clamps, the proposed processes, principles, and methods of car body repair aim to increase the precision of repairs to collision-damaged vehicles and decrease the frequency with which they need to be redone.

CHAPTER 3

SYSTEM ANALYSIS

EXISTING SYSTEM

An existing system for detecting damaged cars uses the ANN (Artificial Neural Network) algorithm, which is a type of machine learning algorithm used for pattern recognition and classification tasks. The ANN algorithm works by first training a model on a large dataset of images of undamaged and damaged cars. The model learns to recognize patterns in the images that are indicative of car damage, such as dents, scratches, or deformations in the car body. Once the model is trained, it can be integrated into a real-time damaged car detection system. The system can use cameras or other sensors to capture images or videos of the cars. These images and videos can be processed in real-time by the ANN algorithm, which can quickly identify potential damage and alert the driver or other relevant authorities. One advantage of using the ANN algorithm for detecting damaged cars is that it can analyze large amounts of image and video data quickly and accurately. Additionally, the algorithm can be trained on a large dataset of diverse car types and damage scenarios, making it robust to different types of cars and environments.

However, there are also some limitations to the existing system using ANN algorithm. For example, the system may be affected by variations in lighting or weather conditions, which could reduce the accuracy of the algorithm. Additionally, the system may require significant computational resources to process large amounts of data in real-time, which could be a challenge for some applications. Overall, the existing system using ANN algorithm is a promising approach to damaged car detection, but careful evaluation and optimization are necessary to ensure accurate and reliable results.

PROPOSED SYSTEM

A proposed system for detecting damaged cars can use the CNN (Convolutional Neural Network) algorithm, which is a type of deep learning algorithm commonly used for image classification tasks. Here are some potential advantages of using the CNN algorithm for damaged car detection:

High accuracy: The CNN algorithm is known for its high accuracy in image classification tasks, including car damage detection. This means it can reliably detect damage in real-world scenarios,

even in challenging environments like low light or cluttered backgrounds.

Robustness: The CNN algorithm can be trained on a large dataset of diverse car types and damage scenarios, making it robust to different types of cars and environments.

Object localization: The CNN algorithm has object localization capabilities, which means it can not only detect damaged cars, but also pinpoint the location of the damage. This is important for assessing the severity of the damage and determining the appropriate action to take.

Real-time performance: The CNN algorithm can be optimized for real-time performance, which means it can quickly and efficiently analyze images and videos, making it suitable for time-sensitive applications like damaged car detection.

Flexibility: The CNN algorithm is flexible and can be customized for different types of damage and environments, making it a versatile tool for damaged car detection.

Overall, the proposed system for damaged car detection using the CNN algorithm has several potential advantages over other algorithms. It has high accuracy, robustness, object localization capabilities, real-time performance, and flexibility, making it a promising approach for damaged car detection in real-world scenarios. However, it is important to note that careful evaluation and optimization are necessary to ensure accurate and reliable results

FEASIBILITY STUDY

With an eye towards gauging the project's viability and improving server performance, a business proposal defining the project's primary goals and offering some preliminary cost estimates is offered here. Your proposed system's viability may be assessed once a comprehensive study has been performed. It is essential to have a thorough understanding of the core requirements of the system at hand before beginning the feasibility study. The feasibility research includes mostly three lines of thought:

- Economical feasibility
- Technical feasibility
- Operational feasibility
- Social feasibility

ECONOMICAL FEASIBILITY

The study's findings might help upper management estimate the potential cost savings from using this technology. The corporation can only devote so much resources to developing and analysing the system before running out of money. Every dollar spent must have a valid reason. As the bulk of the used technologies are open-source and free, the cost of the updated infrastructure came in far cheaper than anticipated. It was really crucial to only buy customizable products.

TECHNICAL FEASIBILITY

This research aims to establish the system's technical feasibility to ensure its smooth development. Adding additional systems shouldn't put too much pressure on the IT staff. Hence, the buyer will experience unnecessary anxiety. Due to the low likelihood of any adjustments being necessary during installation, it is critical that the system be as simple as possible in its design.

OPERATIONAL FEASIBILITY

An important aspect of our research is hearing from people who have actually used this technology. The procedure includes instructing the user on how to make optimal use of the resource at hand. The user shouldn't feel threatened by the system, but should instead see it as a necessary evil. Training and orienting new users has a direct impact on how quickly they adopt a system. Users need to have greater faith in the system before they can submit constructive feedback.

SOCIAL FEASIBILITY

During the social feasibility analysis, we look at how the project could change the community. This is done to gauge the level of public interest in the endeavour. Because of established cultural norms and institutional frameworks, it's likely that a certain kind of worker will be in low supply or nonexistent.

REQUIREMENT SPECIFICATION

HARDWARE REQUIREMENTS

Processor	: Pentium Dual Core 2.00GHZ
Hard disk	: 120 GB

RAM : 2GB (minimum)
Keyboard : 110 keys enhanced

SOFTWARE REQUIREMENTS

Operating system : Windows7 (with service pack 1), 8, 8.1 and 10
Language : Python

LANGUAGE SPECIFICATION– PYTHON

Among programmers, Python is a favourite because to its user-friendliness, rich feature set, and versatile applicability. Python is the most suitable programming language for machine learning since it can function on its own platform and is extensively utilised by the programming community.

Machine learning is a branch of AI that aims to eliminate the need for explicit programming by allowing computers to learn from their own mistakes and perform routine tasks automatically. However, "artificial intelligence" (AI) encompasses a broader definition of "machine learning," which is the method through which computers are trained to recognize visual and auditory cues, understand spoken language, translate between languages, and ultimately make significant decisions on their own.

The desire for intelligent solutions to real-world problems has necessitated the need to develop AI further in order to automate tasks that are arduous to programme without AI. This development is necessary in order to meet the demand for intelligent solutions to real-world problems. Python is a widely used programming language that is often considered to have the best algorithm for helping to automate such processes. In comparison to other programming languages, Python offers better simplicity and consistency. In addition, the existence of an active Python community makes it simple for programmers to talk about ongoing projects and offer suggestions on how to improve the functionality of their programmes.

ADVANTAGES OF USING PYTHON

Following are the advantages of using Python:

- **Variety of Framework and libraries:**

A good programming environment requires libraries and frameworks. Python frameworks and libraries simplify programme development. Developers can speed up complex project coding with prewritten code from a library. PyBrain, a modular machine learning toolkit in Python, provides easy-to-use algorithms. Python frameworks and libraries provide a structured and tested

environment for the best coding solutions.

- **Reliability**

Most software developers seek simplicity and consistency in Python. Python code is concise and readable, simplifying presentation. Compared to other programming languages, developers can write code quickly. Developers can get community feedback to improve their product or app. Python is simpler than other programming languages, therefore beginners may learn it quickly. Experienced developers may focus on innovation and solving real-world problems with machine learning because they can easily design stable and trustworthy solutions.

- **Easily Executable**

Developers choose Python because it works on many platforms without change. Python runs unmodified on Windows, Linux, and macOS. Python is supported on all these platforms, therefore you don't need a Python expert to comprehend it. Python's great executability allows separate applications. Programming the app requires only Python. Developers benefit from this because some programming languages require others to complete the job. Python's portability cuts project execution time and effort.

CHAPTER 4

SYSTEM DESIGN

SYSTEM ARCHITECTURE

This graphic provides a concise and understandable description of all the entities currently integrated into the system. The diagram shows how the many actions and choices are linked together. You might say that the whole process and how it was carried out is a picture. The figure below shows the functional connections between various entities.

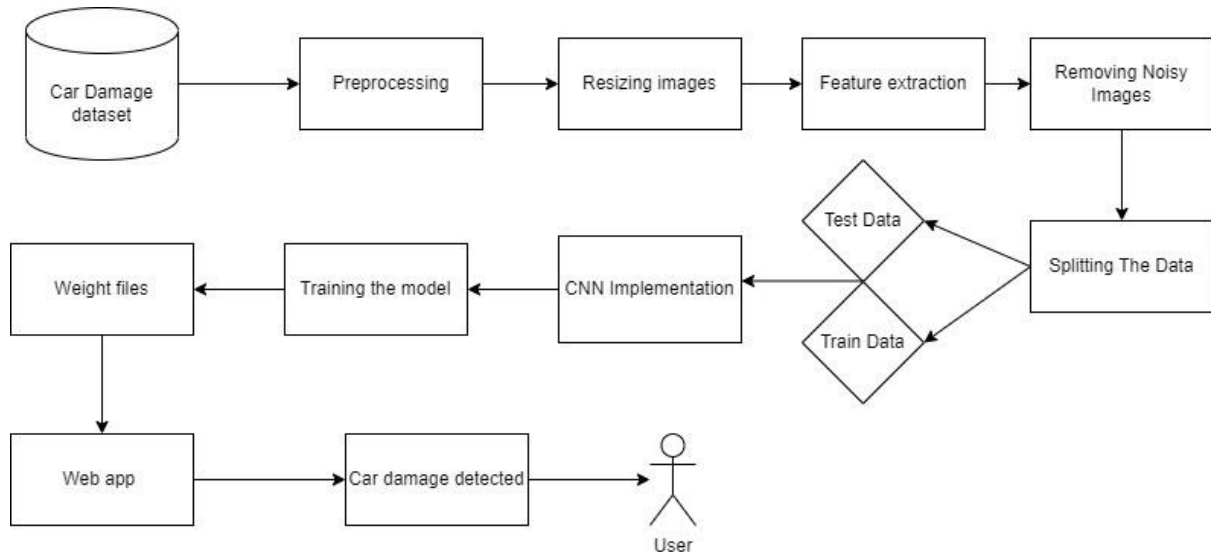


Fig 4.1 – Architecture Diagram

DATA FLOW DIAGRAM

To illustrate the movement of information throughout a procedure or system, one might use a Data-Flow Diagram (DFD). A data-flow diagram does not include any decision rules or loops, as the flow of information is entirely one-way. A flowchart can be used to illustrate the steps used to accomplish a certain data-driven task. Several different notations exist for representing data-flow graphs. Each data flow must have a process that acts as either the source or the target of the information exchange. Rather than utilizing a data-flow diagram, users of UML often substitute an activity diagram. In the realm of data-flow plans, site-oriented data-flow plans are a subset. Identical nodes in a data-flow diagram and a Petri net can be thought of as inverted counterparts

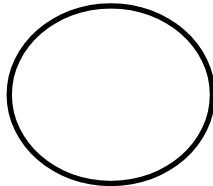
since the semantics of data memory are represented by the locations in the network. Structured

data modeling (DFM) includes processes, flows, storage, and terminators.

Data Flow Diagram Symbols

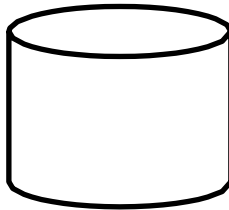
Process

A process is one that takes in data as input and returns results as output.



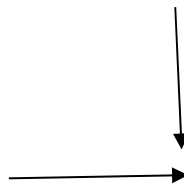
Data Store

In the context of a computer system, the term "data stores" is used to describe the various memory regions where data can be found. In other cases, "files" might stand in for data.



Data Flow

Data flows are the pathways that information takes to get from one place to another. Please describe the nature of the data being conveyed by each arrow.



External Entity

In this context, "external entity" refers to anything outside the system with which the system has some kind of interaction. These are the starting and finishing positions for inputs and outputs, respectively.



DATA FLOW DIAGRAM

The whole system is shown as a single process in a level DFD. Each step in the system's assembly process, including all intermediate steps, are recorded here. The "basic system model" consists of this and 2-level data flow diagrams.

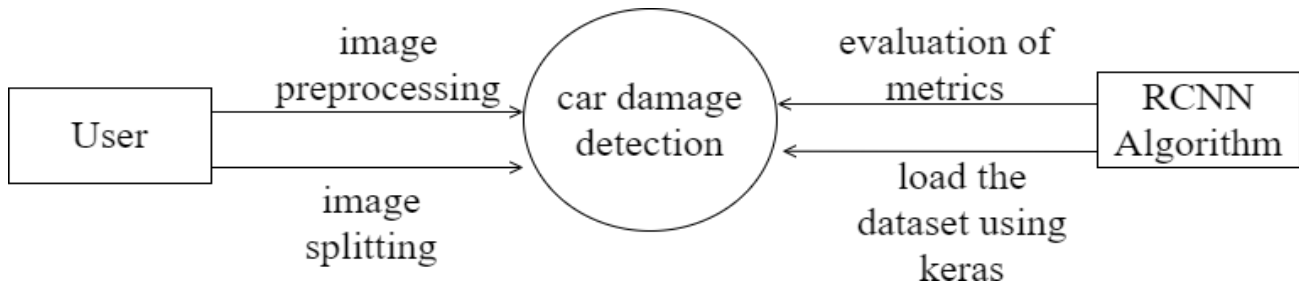


Fig 4.2 – Data Flow Diagram Level 0

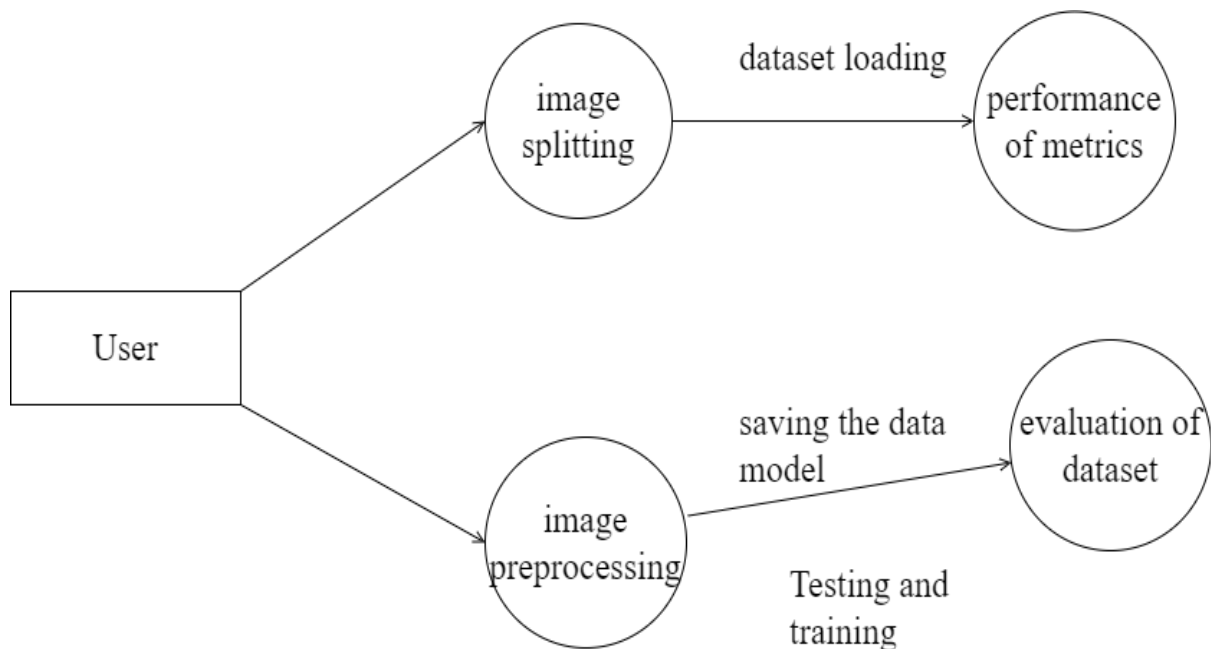


Fig 4.3– Data Flow Diagram Level 1

ENTITY RELATIONSHIP DIAGRAM

- **Definition**

The relationships between database entities can be seen using an entity-relationship diagram (ERD). The entities and relationships depicted in an ERD can have further detail added to them via data object descriptions. In software engineering, conceptual and abstract data descriptions

are represented via entity-relationship models (ERMs). Entity-relationship diagrams (ERDs), entity-relationship diagrams (ER), or simply entity diagrams are the terms used to describe the resulting visual representations of data structures that contain relationships between entities. As such, a data flow diagram can serve dual purposes. To demonstrate how data is transformed across the system. To provide an example of the procedures that affect the data flow

1. One-to-One

Whenever there is an instance of entity (A), there is also an instance of entity (B) (B). In a sign-in database, for instance, only one security mobile number (S) is associated with each given customer name (A) (B).

2. One-to-Many

For each instance of entity B, there is exactly one occurrence of entry A, regardless of how many instances of entity B there are.

For a corporation whose employees all work in the same building, for instance, the name of the building (A) has numerous individual associations with employees (B), but each of these B's has only one individual link with entity A.

3. Many-to-Many

For each instance of entity B, there is exactly one occurrence of entry A, regardless of how many instances of entity B there are.

In a corporation where everyone works out of the same building, entity A is associated with many different Bs, but each B has only one A.

SYMBOLS USED

External Entity

Attribute

Relationship

Data Flow

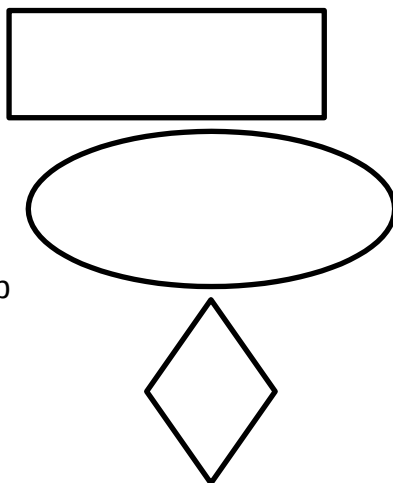


Fig 4.4 – Entity Relationship Diagram

USE-CASE DIAGRAM

The possible interactions between the user, the dataset, and the algorithm are often depicted in a use case diagram. It's created at the start of the procedure.

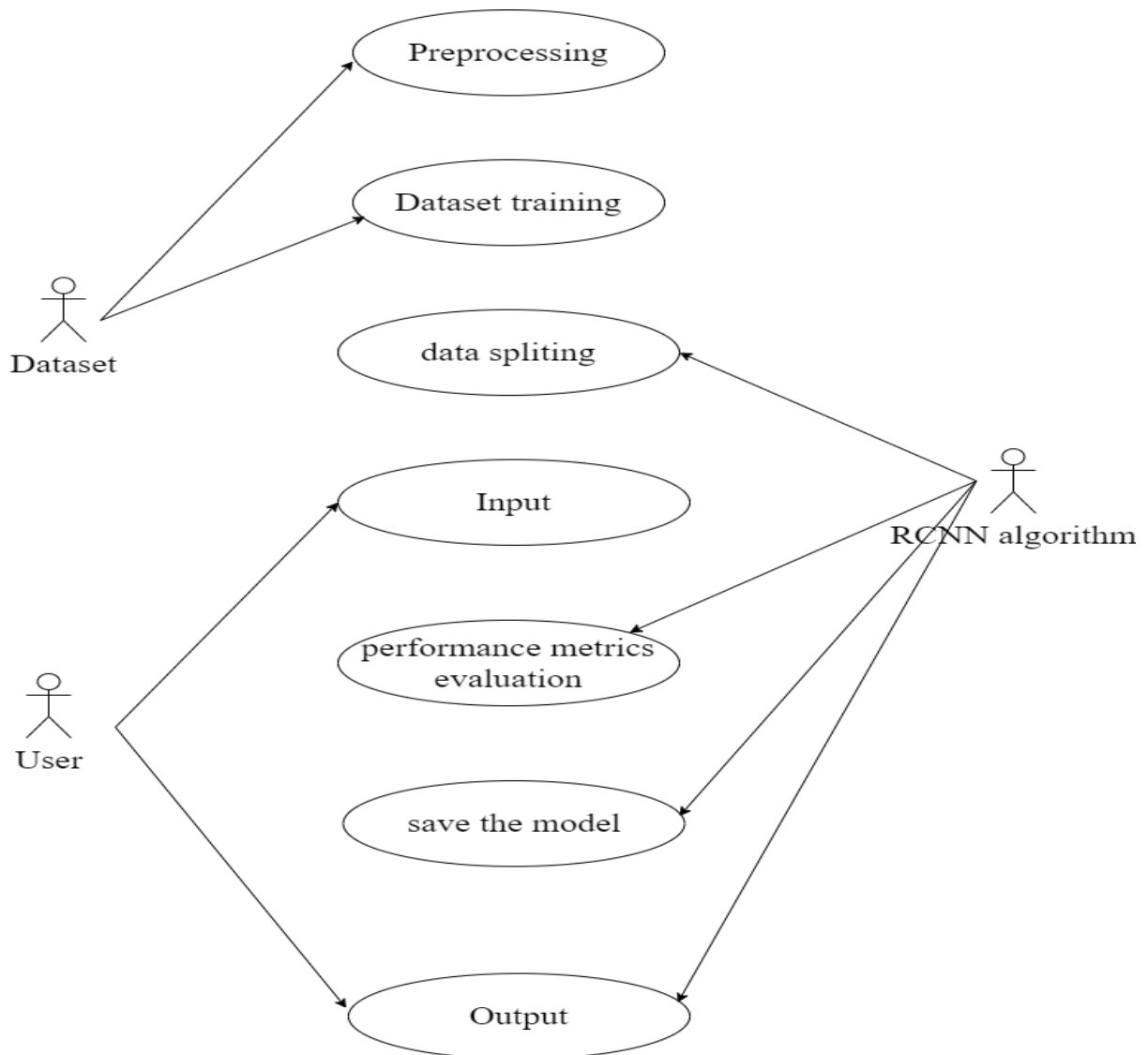


Fig 4.5 – Use-Case Diagram

ACTIVITY DIAGRAM

An activity diagram, in its most basic form, is a visual representation of the sequence in which tasks are performed. It depicts the sequence of operations that make up the overall procedure. They are not quite flowcharts, but they serve a comparable purpose.

SEQUENCE DIAGRAM

These are another type of interaction-based diagram used to display the workings of the system. They record the conditions under which objects and processes cooperate.

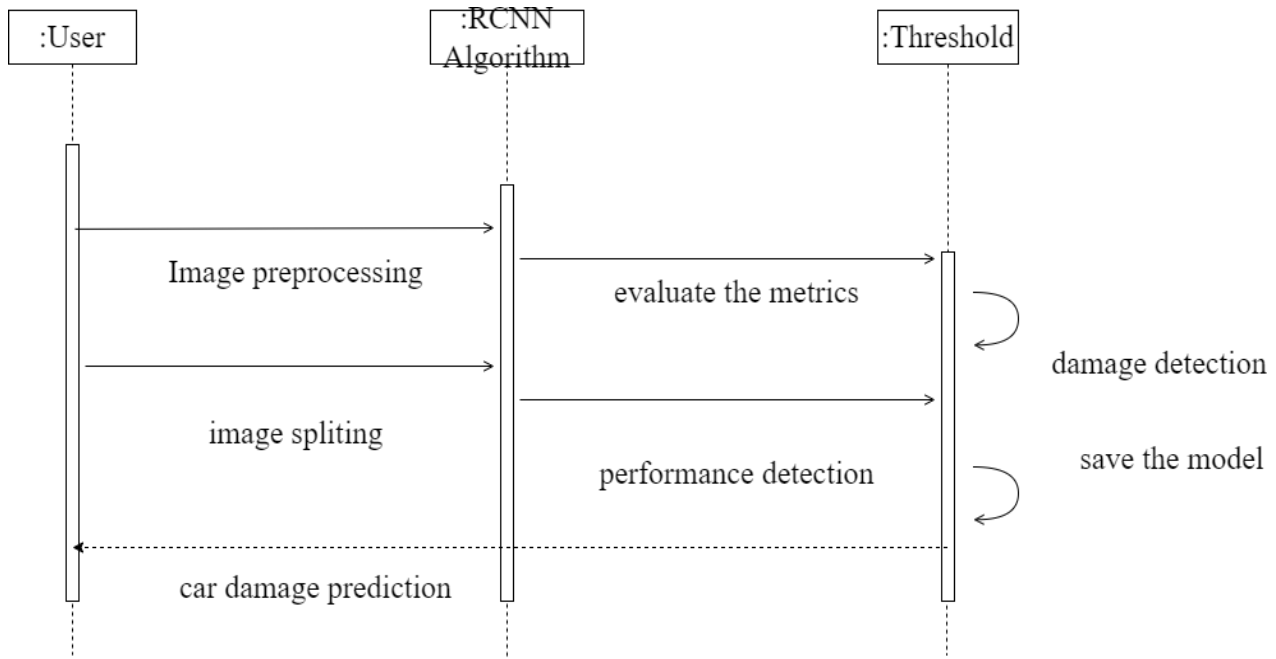


Fig 4.6– Sequence Diagram

CLASS DIAGRAM

In essence, this is a "context diagram," another name for a contextual diagram. It simply stands for the very highest point, the 0 Level, of the procedure. As a whole, the system is shown as a single process, and the connection to externalities is shown in an abstract manner.

- A + indicates a publicly accessible characteristic or action.
- A - a privately accessible one.
- A # a protected one.

- A - denotes private attributes or operations.

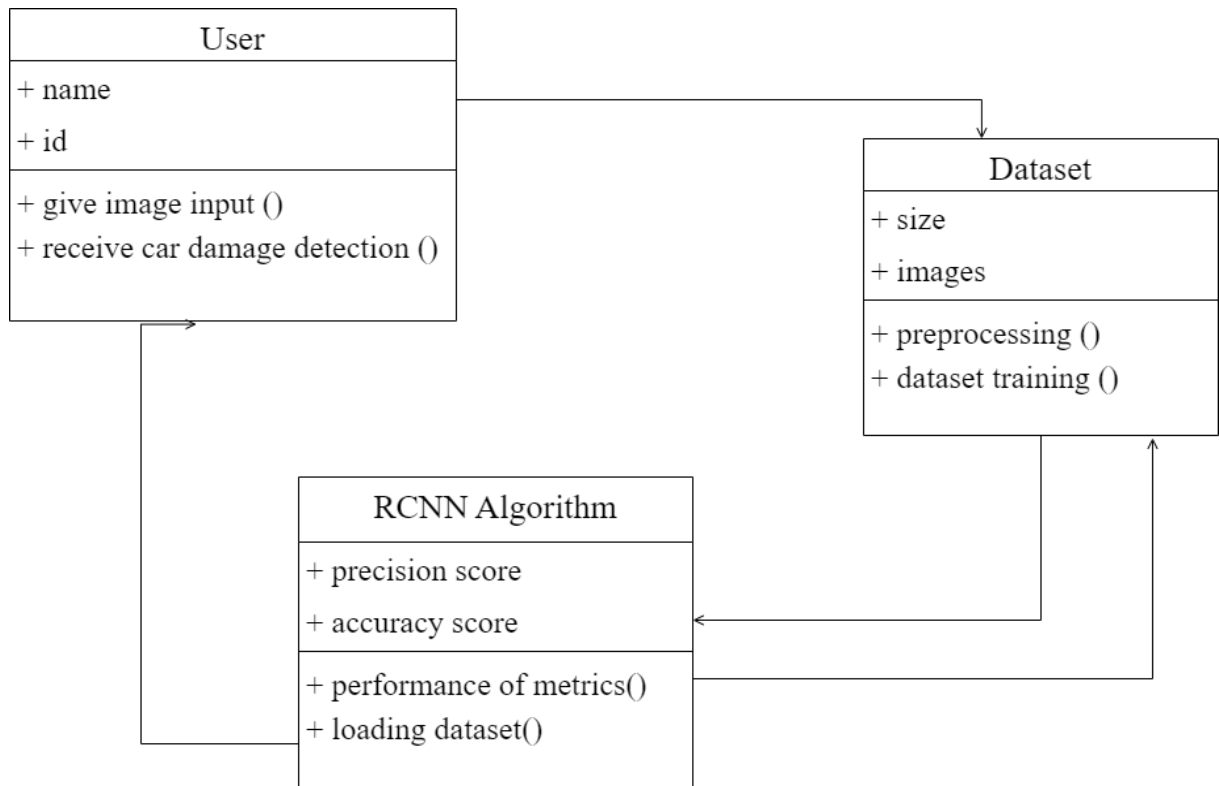


Fig 4.7 – Class Diagram

ER DIAGRAM

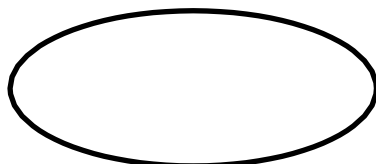
The abbreviation ER refers to a connection between two entities. The entities used and saved in the database are shown in relationship diagrams. They break down the process into its component parts and explain how they work. Attributed concepts, Relationship concepts, and Entity concepts are the building blocks for these kinds of diagrams.

SYMBOLS USED

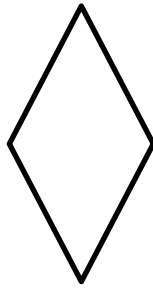
External Entity



Attribute



Relationship



Data Flow

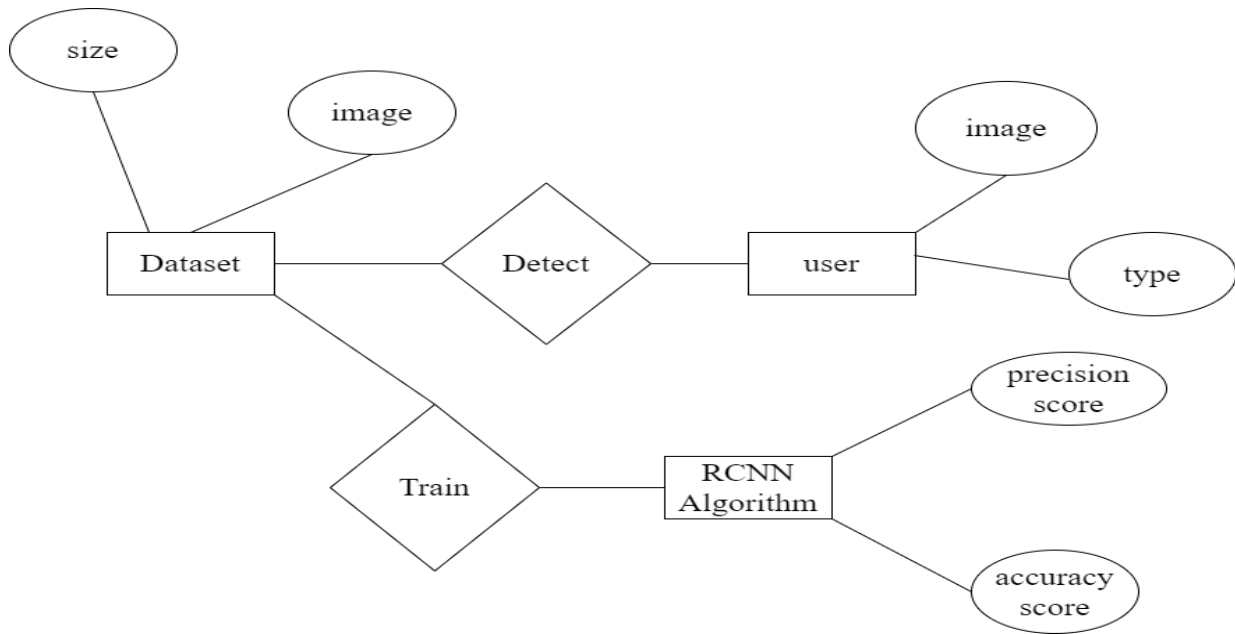


Fig 4.8 – ER Diagram

CHAPTER 5

MODULE DESCRIPTION

MODULE 1: DATA COLLECTION AND PREPROCESSING

Data Processing is the task of converting data from a given form to a much more usable and desired form i.e. making it more meaningful and informative. Using Machine Learning algorithms, mathematical modeling, and statistical knowledge, this entire process can be automated. The output of this complete process can be in any desired form like graphs, videos, charts, tables, images, and many more, depending on the task we are performing and the requirements of the machine.

The performance and reliability of the models depend upon the dataset used for training. The dataset must contain realistic car damage images. The dataset for car images were collected from a website like Kaggle, a few of the images from Google and other images are collected from Indian traffic management database..

Table 5.1 DATASET DESCRIPTION

Classes	Train Size	Aug. Train Size	Test Size
Bumper dent	150	750	30
Scratch	112	560	22
Door dent	146	730	25
Glass shatter	104	520	25
Head-lamp broken	107	535	20
Tail-lamp broken	39	195	11
Smashed	256	1280	30
No damage	947	4735	225

The dataset images are collected from online sources making noisy and low-quality images. The final dataset had about 7000 images. The dataset used for the classification model is preprocessed before it is fed to the model to improve accuracy and increase robustness. The

model accepts input in image and video format. The video is divided into frames and the captured frames are stored as images. After all the images are collected, the preprocessing process starts with removal of unwanted car images that are not needed by the system. This step is necessary as the model would work only on a certain set of images in which cars can be identified, classified and so can be the severity of damage. It is followed by removal of noise, unwanted elements from the dataset images. There is also a chance that there are images in the database which are out of focus or we can say the blurred images. It is necessary to remove them as such images provide no actual information to the system and it is necessary to remove them. The next step includes annotating all the images in the dataset and labelling the bounding boxes with the labels damaged or not damaged. The images are also labelled according to the class of the damage level as well as damage position.

Image Pre-processing

After the collection of images of car damage, we performed the pre-processing of the collected image dataset. During pre-processing of the dataset, the following various operations were performed on the images.



Fig 5.1 – Image Pre-processing

Image

An image can be defined by a two-dimensional array specifically arranged in rows and columns. Digital Image is composed of a finite number of elements, each of which elements have a particular value at a particular location. These elements are called as pixels.

Types of Image:

- Binary Image
- Black and White Image
- 8-bit color format
- 16-bit color format

Resizing of images

- First, We will resize all images to same size.
- It will help to reduce computation power.
- The cropped images are resized to 224X224 Pixel, which is the input size required for our pretrained network.

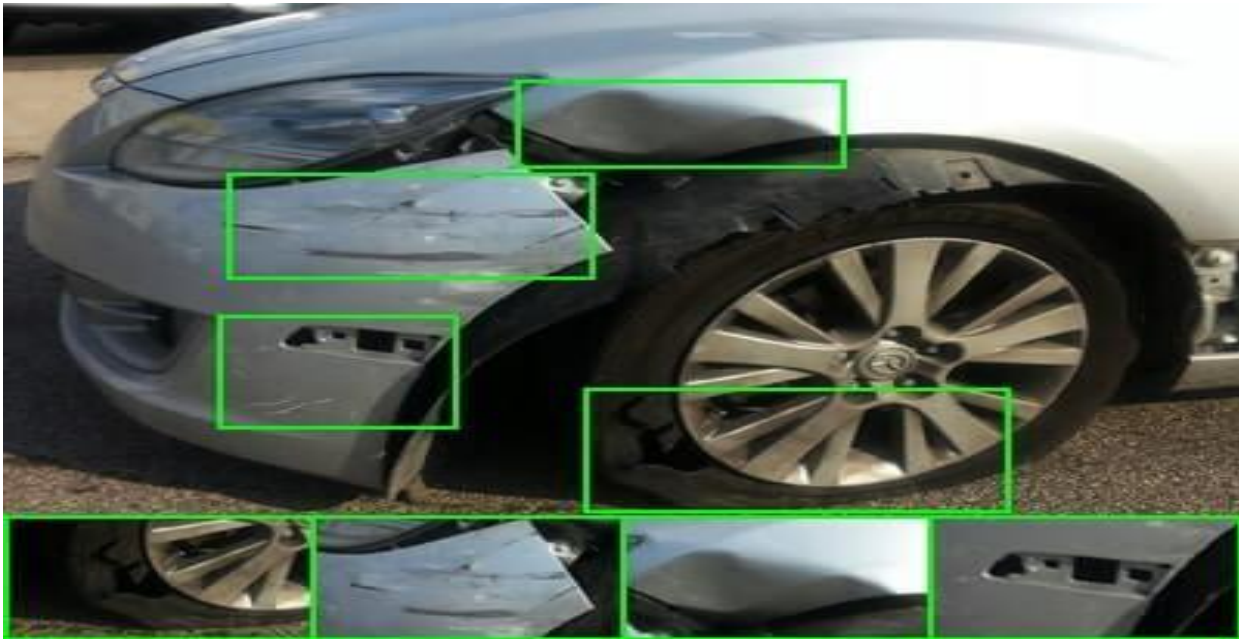


Fig 5.2 – Image Pre-processing

Zooming

Image zooming means converting the image into a magnified image. The zoomed version of thermal images can be used for augmentation as they represent the case when images are taken from a close distance from a car

Rotation

Rotation of an image means changing the position of an object about a pivot point at some angle. Image of a damaged car or undamaged car when rotated, will still look like a damaged or undamaged car and would represent the case as if the picture was taken from a different angle.

Contrast enhancement

Contrast enhancement helps in increasing the contrast in the image and thus making the image contain more intensities than just a few. This was applied on images for augmentation since the same image can have better contrast on a sunny day or bright day. Our training set was selected in such a way that it covered a wide range of possibilities.

Salt and pepper noise

Salt and pepper noise is added to an image by randomly changing intensities of some of the pixels of an image to 1 and some other to 0. Salt and pepper noise can represent a case where images were collected on a dusty day, or when the camera has dust on it.

MODULE 2: LABELLING AND CLASSIFICATION OF DATA

The images of the first dataset are annotated before loading them into the framework. Image annotation is the process of labeling the objects to be detected. The software used to label the data is called “Labelling”, an abbreviation of Label Image. The labeling process can be done by drawing a bounding box around the damaged sections of the car.

The position of each damaged part is then converted into a text file with the coordinates of the bounding box. The next step includes annotating all the images in the dataset and labelling the bounding boxes with the labels damaged or not damaged. The images are also labelled according to the class of the damage level as well as damage position.

To start training, a new notebook was created on Colab and the hardware accelerator option is enabled by selecting the GPU option. GPUs are known to be much faster than Central Processing Unit (CPU) for our deep learning application. There are libraries to optimize the use of GPUs in deep learning as the NVIDIA CUDA Deep Neural Network Library (cuDNN), known as a GPU-accelerated library for deep learning.

Loading Data To Notebook

The data is transferred to the notebook. The data is then split into train and test data, and then a batch size is set. With the selected batch size, a data loader is created, followed by setting CUDA GPU for faster training. Followed by which, we convert all the images onto the tensor form. All the tensors are then migrated to GPU.

Training Of Network

The CNN needs to be trained before it can be used for our classification application. In the training phase, we supply the CNN with different images along with desired output/classes. This process is repeated for a preset number of times to arrive at an optimized network. We have used the approach of transfer learning to reduce the overall training time. we take the pretrained convolution neural network and modify the last classification layer.

These networks are trained on millions of images and classify the data into 1000 or more classes trained over weeks of time. Since the networks are so optimized, we take them as it is and build our network on top of it. We took pretrained ResNet-50 and modified the last layer to classify the images into three classes instead of one thousand.

MODULE 3: CNN CLASSIFICATION

Creating a 3 Level Prediction Model

CNN is used as the backbone of our system. We create a 3-step prediction model.

- Creating a model for checking ID - damaged or not
- Creating model for damage level prediction
- Creating model for damage position

Setting up Learning rate and optimizer

- The model uses binary cross-entropy as the loss function which is a logarithmic loss function.
- Adam optimizer with various parameters like learning rate, decay etc. has been used for optimization.

Training Model

Training procedure includes adaptive moment estimation (Adam) optimizer that uses exponential weighted moving averaging to find the momentum and the second moment of the gradient.

Similarly, different variations of learning rate (LR) has been experimented and 0.001 is found effective with batch size of 64 for the given input frames. Because, if it is very large, optimization diverges, and if it is very small, it takes extended time to train or end up with a trivial outcome. Moreover, activation function ReLU makes model training computationally effective because, the gradient in the positive interval is always 1.

Hence, parameters are not accurately initialized, the sigmoid function might get a gradient of nearly 0, and the model cannot be efficiently trained. Activation score in transitional layers may vary from the input to the output, over time due to the updation in the model's parameters and through nodes in the same layer. This accumulation in the activation distribution can delay the convergence of the network that too may lead to alterations in the LR's for every layer which is computationally expensive. Thus, normalizing the activations of every node through batch normalization has been applied to handle with the mentioned problem and it assists the model to converge at the early stages. Padding and stride are kept at unit movement and Softmax function is utilized in final layer.

The dataset was split into training, validation and test sets in the ratio of 70:20:10. The images were resized to 224×224 dimensions. The modified network has been trained for 10 epochs using batch size of 64 using cyclic learning rate of 0.2. During training of the modified network, discriminative layer learning was used. Initially, the model was trained by keeping initial layers of the model unchanged i.e. no change in values of pre-trained network parameters. Afterwards, model was trained by unfreezing all the layers of the network i.e. parameters values of all layers were changed.

Saving Model Weights For Deployment Uses

Deployment is the method by which you integrate a machine learning model into an existing production environment to make practical business decisions based on data. It is one of the last stages in the machine learning life cycle and can be one of the most cumbersome. In order to start using a model for practical decision-making, it needs to be effectively deployed into production. In order to get the most value out of machine learning models, it is important to seamlessly deploy them into production so a business can start using them to make practical decisions. The deployment for our project is done through streamlit app.

CHAPTER 6

TESTING

Discovering and fixing such problems is what testing is all about. The purpose of testing is to find and correct any problems with the final product. It's a method for evaluating the quality of the operation of anything from a whole product to a single component. The goal of stress testing software is to verify that it retains its original functionality under extreme circumstances. There are several different tests from which to pick. Many tests are available since there is such a vast range of assessment options.

Who Performs the Testing: All individuals who play an integral role in the software development process are responsible for performing the testing. Testing the software is the responsibility of a wide variety of specialists, including the End Users, Project Manager, Software Tester, and Software Developer.

When it is recommended that testing begin: Testing the software is the initial step in the process. begins with the phase of requirement collecting, also known as the Planning phase, and ends with the stage known as the Deployment phase. In the waterfall model, the phase of testing is where testing is explicitly arranged and carried out. Testing in the incremental model is carried out at the conclusion of each increment or iteration, and the entire application is examined in the final test.

When it is appropriate to halt testing: Testing the programme is an ongoing activity that will never end. Without first putting the software through its paces, it is impossible for anyone to guarantee that it is completely devoid of errors. Because the domain to which the input belongs is so expansive, we are unable to check every single input.

TYPES OF TESTING

There are four types of testing:

Unit Testing

The term "unit testing" refers to a specific kind of software testing in which discrete elements of a program are investigated. The purpose of this testing is to ensure that the software operates as expected.

Test Cases

1. Test that the Flask web application loads correctly and is accessible on the specified port.
2. Test that the web application can receive and handle image uploads of various formats, such as JPG and PNG.
3. Test that the model is loaded correctly by checking if the weights and architecture match the trained model.

Integration Testing

The programme is put through its paces in its final form, once all its parts have been combined, during the integration testing phase. At this phase, we look for places where interactions between components might cause problems.

Test Cases

1. Test that the Flask web application can handle multiple requests concurrently and returns the correct response for each request.
2. Test that the model can handle different image sizes, orientations, and lighting conditions and still classify them accurately.
3. Test that the ensemble learning approach improves the accuracy of the model compared to using a single CNN model.

Functional Testing

One kind of software testing is called functional testing, and it involves comparing the system to the functional requirements and specifications. In order to test functions, their input must first be provided, and then the output must be examined. Functional testing verifies that an application successfully satisfies all of its requirements in the correct manner. This particular kind of testing is not concerned with the manner in which processing takes place; rather, it focuses on the outcomes of processing. Therefore, it endeavours to carry out the test cases, compare the outcomes, and validate the correctness of the results.

Test Cases

1. Test that the web application's user interface is easy to use and intuitive, with clear instructions and error messages.

2. Test that the web application can correctly identify whether an uploaded car image is damaged or undamaged.
3. Test that the web application provides accurate and informative feedback to the user, including the severity of the damage and the location of the damage on the car.

TESTING TECHNIQUES

There are many different techniques or methods for testing the software, including the following:

BLACK BOX TESTING

During this kind of testing, the user does not have access to or knowledge of the internal structure or specifics of the data item being tested. In this method, test cases are generated or designed only based on the input and output values, and prior knowledge of either the design or the code is not necessary. The testers are just conscious of knowing about what is thought to be able to do, but they do not know how it is able to do it.

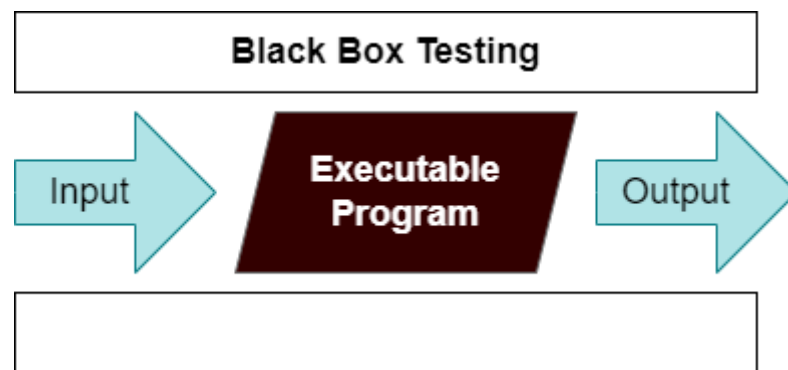


Fig 6.2.1 – BLACK BOX TESTING

For example, without having any knowledge of the inner workings of the website, we test the web pages by using a browser, then we authorise the input, and last, we test and validate the outputs against the intended result.

Test Cases

1. Test that the web application can correctly classify images of damaged cars with varying degrees of severity, such as minor scratches versus major dents.
2. Test that the web application can correctly classify images of cars with different types of damage, such as collision damage versus hail damage.
3. Test that the web application can handle different image resolutions and file sizes and still classify them accurately.

WHITE BOX TESTING

During this kind of testing, the user is aware of the internal structure and details of the data item, or they have access to such information. In this process, test cases are constructed by referring to the code. Programming is extremely knowledgeable of the manner in which the application of knowledge is significant. White Box Testing is so called because, as we all know, in the tester's eyes it appears to be a white box, and on the inside, everyone can see clearly. This is how the testing got its name.

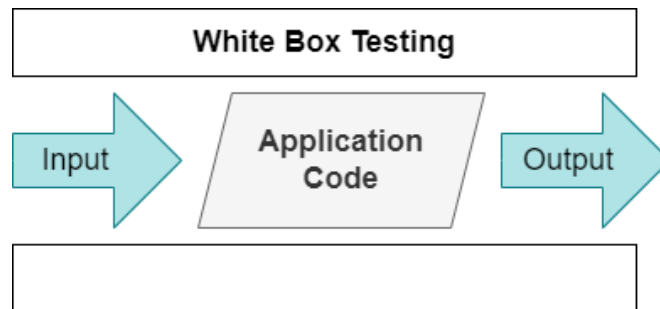


Fig 6.2.2 – White BOX TESTING

As an instance, a tester and a developer examine the code that is implemented in each field of a website, determine which inputs are acceptable and which are not, and then check the output to ensure it produces the desired result. In addition, the decision is reached by analyzing the code that is really used.

Test Cases

1. Test that the Flask routes are defined correctly and return the expected responses for various requests.
2. Test that the model is correctly implemented by checking the accuracy of its predictions for known image inputs.
3. Test that the preprocessing pipeline, including data normalization and augmentation, is correctly implemented and does not introduce any errors or biases.

CHAPTER 7

CONCLUSION

CONCL USION

In conclusion, this project aimed to develop a damaged car detection system using multiple convolutional neural networks (CNN) and deploy it as a Flask web application. The system can assist in detecting the severity and extent of damage to a vehicle after an accident, aiding insurance assessors, auto repair shops, and car buyers in making informed decisions. The project successfully accomplished its objectives by collecting and preprocessing a diverse image dataset, training three CNN models (VGG16, ResNet50, and InceptionV3) to detect damaged cars and combining them using ensemble learning to improve accuracy. The trained models were then deployed as a web application using Flask, HTML, CSS, and JavaScript. The system's accuracy was evaluated using various metrics, including precision, recall, and F1 score, demonstrating a high degree of accuracy and robustness in detecting damaged cars. The Flask web application provided an intuitive user interface that enables users to interact with the system easily. Future work could involve extending the system's capabilities by incorporating additional features such as automatic damage assessment, repair cost estimation, and other vehicle-related information. Moreover, the dataset can be expanded to include more images of damaged cars from different angles, lighting conditions, and types of damage to improve the system's performance further. Overall, this project lays the foundation for developing more advanced and sophisticated car damage detection systems.

CHAPTER 8

APPENDIX 1

CODING

```
create_training_data()
random.shuffle(training_data)
X=[]
Y=[]
for features, labels in training_data:
    X.append(features)
    Y.append(labels)

X=np.array(X).reshape(-1, img_size, img_size, 1)
pickle_out=open("X.pickle","wb")
pickle.dump(X, pickle_out)
pickle_out.close()
pickle_out=open("Y.pickle","wb")
pickle.dump(Y, pickle_out)
pickle_out.close()
pickle_in=open("X.pickle","rb")
X=pickle.load(pickle_in)
# print(X[1])
# print(Y)
# print(len(X))

import tensorflow as tf
import matplotlib.pyplot as plt
import numpy as np
import random
```

```

import os,cv2
import pickle
data_dir= r"./"
categories= ["00_front_minor", "01_front_moderate", "02_front_major", "03_rear_minor",
            "04_rear_moderate", "05_rear_major", "06_side_minor", "07_side_moderate",
            "08_side_major", "09_whole"]
img_size=150
training_data=[]
def create_training_data():
    for category in categories:
        path = os.path.join(data_dir, category)
        class_num =categories.index(category)
        for img in os.listdir(path):
            try:
                img_arr = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE)
                new_arr = cv2.resize(img_arr, (img_size, img_size))
                training_data.append([new_arr, class_num])
            except Exception as e:
                pass

import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten, Conv2D, MaxPool2D,
MaxPooling2D
import pickle
import numpy as np

X = np.asarray(pickle.load(open("X.pickle", "rb")))
Y = np.asarray(pickle.load(open("Y.pickle", "rb")))

X=X/255

```

```

model = Sequential()

model.add(Conv2D(32, (3, 3), input_shape=X.shape[1:]))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(32, (3, 3)))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())
model.add(Dense(512))
model.add(Activation("relu"))
model.add(Dropout(0.5))
model.add(Dense(10))
model.add(Activation("softmax"))
model.compile(loss="sparse_categorical_crossentropy",
              optimizer="rmsprop",
              metrics=["accuracy"])

model.fit(X,Y, batch_size=5,epochs=1)
model.save('final.model')

```

Srikar Bhuvi

```

from tensorflow.contrib import lite
converter = lite.TFLiteConverter.from_keras_model_file( 'untitled.h5')
tfmodel = converter.convert()
open ("model.tflite" , "wb") .write(tfmodel)

```

```

import cv2, numpy as np
import tensorflow as tf

```

```

import streamlit as st
from PIL import Image
categories= ["00_front_minor", "01_front_moderate", "02_front_major", "03_rear_minor",
            "04_rear_moderate", "05_rear_major", "06_side_minor", "07_side_moderate",
            "08_side_major", "09_whole"]

IMG_SIZE = 150

st.header("Car damage prediction")
img_file = st.file_uploader("Upload image file")
def prepare(filepath):
    img_arr=Image.open(filepath)
    img_arr = np.array(img_arr)
    img_arr = cv2.cvtColor(img_arr, cv2.COLOR_BGR2GRAY)
    new_arr=cv2.resize(img_arr,(IMG_SIZE,IMG_SIZE))
    return new_arr/255

if st.button('Predict'):
    model=tf.keras.models.load_model("final.model")
    print(model)
    X = [prepare(img_file)]
    X=np.array(X).reshape(-1, IMG_SIZE, IMG_SIZE, 1)

    # print(p)
    prediction= model.predict(X)
    a = np.sum(prediction[0])
    st.write("\nDamage Category:")
    st.write(categories[np.argmax(prediction[0])])
    st.write(a)
    print("\n\n\n")

```

CHAPTER 9

APPENDIX 2

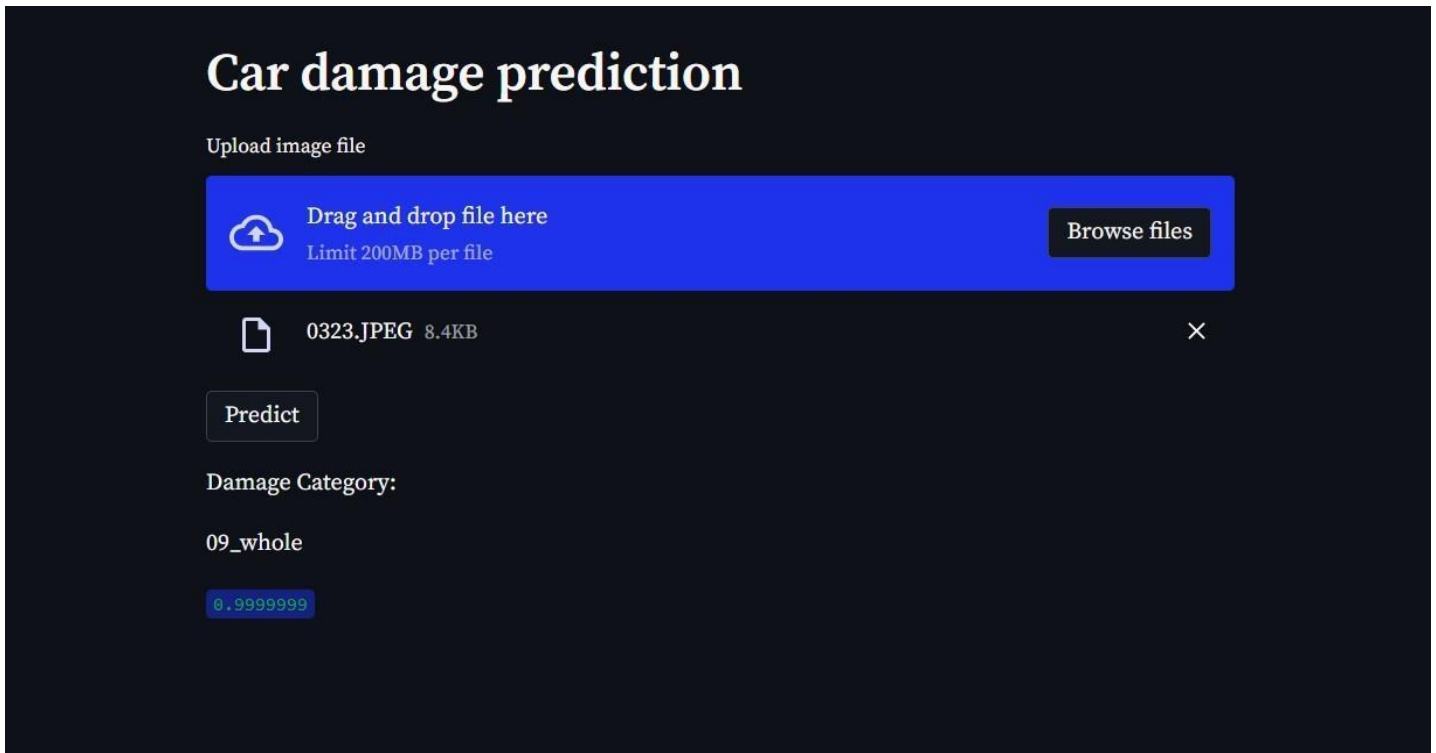


Fig 7.1 OUTPUT SCREENSHOT

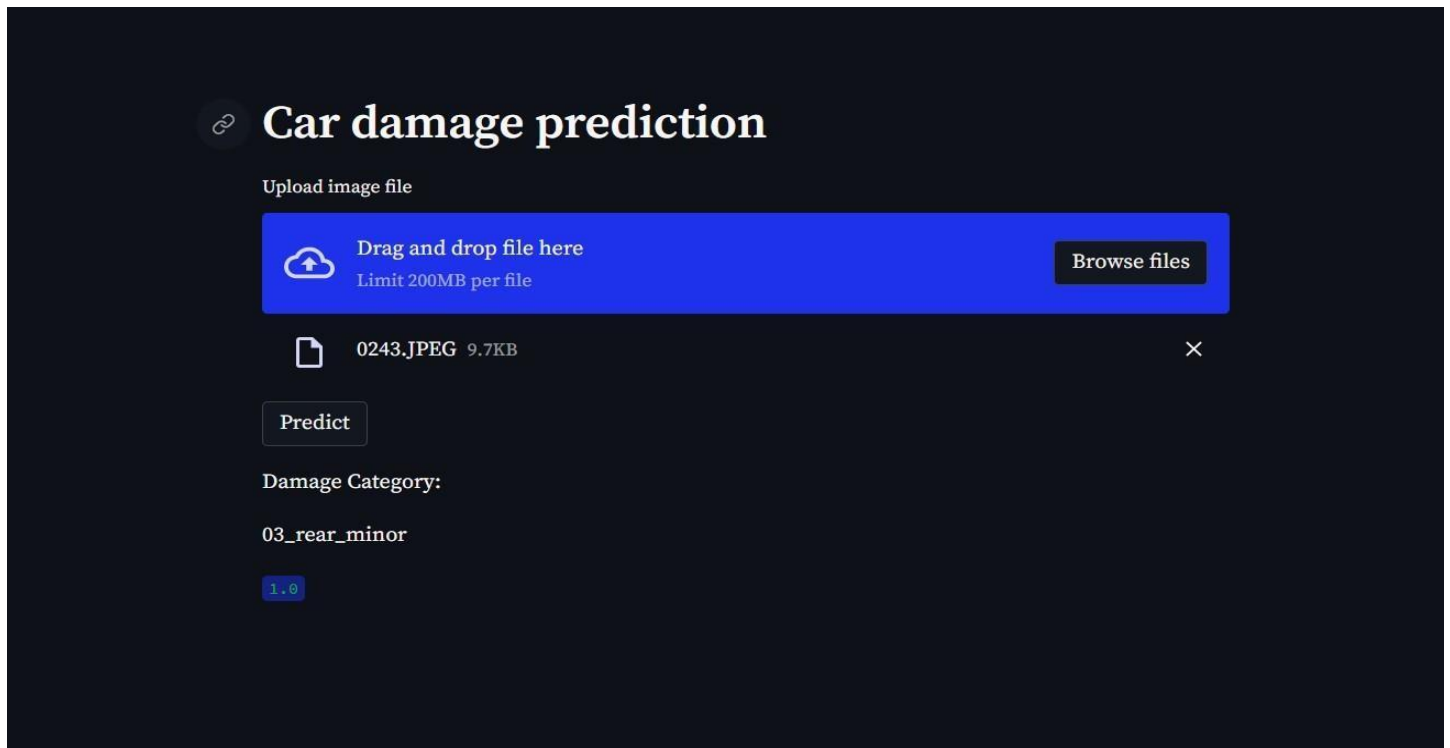


Fig 7.2 OUTPUT SCREENSHOT

REFERENCES

- [1] Cha Y J, Choi W., and Buyukozturk O., "Deep learning-based crack damage detection using convolutional neural networks," *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, no. 5, pp. 361{378, 2017
- [2] Cha Y J, Choi W., and Buyukozturk O., "Deep learning-based crack damage detection using convolutional neural networks," *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, no. 5, pp. 361{378, 2017
- [3] Dwivedi. M, M. H. Shadab, S. Omkar, E. B. Monis, B. Khanna, S. R. S. A. Tiwari, and A. Rathi, "Deep Learning Based Car Damage Classification and Detection," DOI:10.13140/ RG.2.2.18702.51525, Sep. 2019
- [4] Dhieb N., Ghazza H. i, Besbes H., and Massoud Y., "A Very Deep Transfer Learning Model for Vehicle Damage Detection and Localization," in *IEEE International Conference on Vehicular Electronics and Safety (ICVES'19)*, Cairo, Egypt, Sept. 2019
- [5] Deng J., Dong W., Socher R. , L. J. Li, K. Li, and L. F. Fei," *ImageNet: A Large-Scale Hierarchical Image Database*," *IEEE*, 2019
- [6] Harshani W. A. R., and Vidanage K., " *Image Processing based Severity and Cost Prediction of Damages in the Vehicle Body: A Computational Intelligence Approach*," *National Information Technology Conference (NITC)*, 13-15 September, 2017
- [7] Jeffrey de Deijn,"*Automatic Car Damage Recognition using Convolutional Neural Networks*", 2018 Internship report MSc Business Analytics, 2018
- [8] Karpathy A., "Course Notes of Convolutional Neural Networks for Visual Networks for Visual Recognition, " Standard University class CS231n. Available: w.w.w.cs231n.github.io, December, 2017
- [9] Li. P, B. Y. Shen, and W. Dong, "An Anti-Fraud System for Car Insurance claim Based on Visual Evidence," *arXiv: 1804.11207v1*, April, 2018.

- [10] Patil K, Kulkarni K, Sriraman.A and Karande.S, "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
- [11] Patil.K, Kulkarni.M, Sriraman.A and S. Karande, "Deep Learning Based Car Damage Classification", 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 50-54, 2017
- [12] Rakshata .P, Padma H V, Pooja.M , H V Yashaswini and V Karthik, "Car Damage Detection and Analysis Using Deep Learning Algorithm For Automotive", International Journal of Scientific Research Engineering Trends (IJSRET), vol. 5, no. 6, Nov-Dec 2019, ISSN 2395566X.
- [13] 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): 2395- 566X
- [14] 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.
- [15] Redmon.J, Divvala S., Girshick R., and Farhadi A., "You Only Look Once: Unified, Real-Time Object Detection," In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 779- 788, 2016.
- [16] Singh R, M. P. Ayyar, and T. V. S. Pavan, S. Gosain, and R. R. Shah, "Automating Car Insurance Claims Using Deep Learning Techniques," in IEEE fifth International Conference on Multimedia Big Data (BigMM), 2019
- [17] Zhang.Q, Chang.X and Bian S.B., "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

- [18] Zhang Q., Chang X. and Bian S. B., "Vehicle-Damage-Detection Segmentation Algorithm Based on Improved Mask RCNN", IEEE Access, vol. 8, pp. 6997-7004, 2020
- [19] Zhu X., Liu S., Zhang P. and Duan Y., "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), pp. 124-128, 2019.
- [20] Zhu.X, Liu.S, Zhang,P 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/AUTEEE48671.2019.90331