Deep Learning-based Damage Detection and Repair Cost Estimation for Automobiles

Narra Suma, Savitha N. J.
Department of CSE, CMRIT
Bangalore, Karnataka

Abstract:- In the automotive sector, estimating the cost to repair damaged vehicles is a critical duty. In this project, we present a technique for estimating maintenance costs that makes use of the cutting-edge deep learning architecture MobileNetV2. With the use of a sizable dataset of photos of damaged vehicles and related repair costs, we fine-tune the pre-trained MobileNetV2 model. For insurance companies, repair facilities, and automobile manufacturers, the suggested method can be used as a cost-effective and practical option to determine the expense of repairing damaged vehicles. The effective accuracy of the model is 87.35%.

Keywords:- *MobileNetV2, Deep learning, Convolutional Neural Networks, VGG-16.*

I. INTRODUCTION

In the automotive sector, estimating the cost of repairing damaged vehicles is a crucial duty. For insurance companies, repair facilities, and automotive manufacturers to make educated choices regarding the repair and replacement of damaged vehicles, accurate repair cost estimation is crucial. Traditional techniques for calculating repair costs rely on manual inspection by a skilled mechanic or estimate, which can be expensive and time-consuming. Deep learning approaches have recently demonstrated their potential for automating this procedure by estimating repair costs from photos of damaged cars.

A. Relevance of the Project

Vehicle accidents are common in our nation because of reckless driving and other traffic law violations. Therefore, the state of the worst automobiles in our nation serves as the primary motivation for our study. This idea emphasizes the need to improve the traffic situation in our nation.

The catastrophic effects of traffic congestion brought on by vehicle accidents require the study of precise and efficient mitigating strategies and techniques. A vehicle-damage-detection segmentation method is required to quickly solve traffic accident compensation issues.

Currently, a car must be towed to an auto repair shop by the owner or a towing service after being damaged in a collision or another incident. To determine which parts of the car, need to be fixed or replaced, a mechanic must inspect the vehicle at the auto repair shop. The inspection is used to create an estimate.

According to statistics, the number of accidents has been rising steadily over the last three years. Around 3,66,138 vehicles were damaged in 2020; however, in 2021, that figure increased to 4,03,116. Additionally, the accident rate for 2022 has already shattered the previous month's record.

B. Objectives

The objectives of the app for repair cost estimation of damaged vehicles using MobileNetV2 are as follows:

- Accurately estimate repair costs: The app's main goal is to provide an accurate estimate of the costs involved in repairing damaged automobiles. To accurately estimate the cost of repairs, the app will use MobileNetV2 to analyze photographs of the damage.
- Enhance efficiency: By automating the process with deep learning techniques, the software intends to enhance the efficiency of repair cost estimation. The software will do the manual checks for you, saving you time and money in estimating repair costs.
- Estimation that can be changed: Users of the app will be able to alter the repair cost estimation to suit their own requirements and preferences. Users can modify the anticipated repair prices based on the kind and extent of the damage, giving each repair a customized estimate.
- Simple and intuitive user interface will make the program simple and intuitive to use. Users will be able to quickly upload pictures of damaged cars and get an estimate for the cost of repairs.
- Scalable solution: The app's ability to estimate repair costs for many different damaged cars at once makes it scalable. This will make it possible for repair shops, insurance companies, and car manufacturers to estimate expenses promptly and effectively.
- High accuracy: The software seeks to exceed conventional techniques for repair cost estimation by achieving a high level of accuracy. To confirm its accuracy, the software will undergo testing on a sizable dataset of pictures of damaged vehicles.

Overall, the app's objectives are to provide a cost-effective and efficient solution for repair cost estimation in the automotive industry, leading to cost savings and improved customer satisfaction.

II. LITERATURE SURVEY

A. *Image-based Vehicle Detection using Various Features[1]*

This paper discusses about 2 methods namely the Lens-based vehicle classifier and the Haar cascade classifier. The process is held like a heatmap, bounding box, threshold, and merged box in the database.

- In this scenario, a smaller number of images are used for prediction. Heat maps are employed to determine the bounding box, with a threshold of 85% probability for any class of object. This technique helps to identify and locate objects accurately within the given images.
- The original goal was to detect mild vehicle damage using vehicle photographs that propose to use images obtained from the photograph and filter the edge to detect mildly damaged regions in vehicles.
- When a regressor is integrated with the output of image pixels, the information is combined. The entire process is then executed, resulting in vehicle detection with an accuracy level of 85% or lower.

B. *Methodology of repair cost estimation in vehicles based on the deformation measurements [2]*

The main objective of this study is to analyze the relationship between reconstruction variables based on the deformation measurements in real-world accidents with the repair cost of the car.

- The main result is a methodology to estimate quickly and easily repair costs of vehicles involved in road accidents. Real-world accidents analyzed in this paper are Crashworthiness Data System (NASS CDS Database)-field research teams located across a country study about 5000 crashes a year; Audaplus estimation system which used data about costs of the vehicle parts and the time necessary to replace it based on manufacturer's information.
- This study has developed a retrospective methodology to estimate easily repair costs of vehicles involved in road accidents with the front zone involved. Using residual deformation measurements based on Tumbas and Smith's protocol, it is viable to estimate deltaV and absorbed energy for the vehicle involved in an accident.

C. *Vehicle-Damage-Detection Segmentation Algorithm Based on Improved Mask RCNN [3]*

In this paper, a vehicle-damage-detection segmentation algorithm is introduced. The algorithm utilizes transfer learning and an improved mask regional convolutional neural network (Mask RCNN) to achieve accurate detection and segmentation of vehicle damages.

- In the experiment, car damage pictures are gathered for preprocessing. The "Labelme" tool is employed to annotate the dataset, which is then split into training and test sets for further analysis.
- The results obtained from training and testing on a custom dataset demonstrate that the enhanced Mask RCNN model exhibits improved Average Precision (AP), detection accuracy, and masking accuracy. This improvement in performance not only enhances the efficiency of addressing traffic accident compensation issues but also contributes to more effective solutions.
- After testing and improvement, the proposed transfer-learning and improved Mask RCNN-based vehicle-damage-detection method is more universal and can better adapt to various aspects of car-damage images.

D. *Automatic Assessment of Damage and Repair Costs in Vehicles [4]*

A system and method are introduced for automated estimation of vehicle repairs. The method involves the server computing device, connected to a client computing device via an electronic network, receiving one or more images of a damaged vehicle. Subsequently, image processing operations are performed on each of the provided images to identify external features related to a specific set of vehicle parts.

Inferring internal damage to a second set of parts of the vehicle based on the detected external damage; calculating an estimated repair cost for the vehicle based on the detected external image and inferred internal image based on accessing for each part in the first and second parts of parts.

III. SYSTEM DESIGN

A. *Theoretical Background*

Below is some theoretical background information that highlights some subjects pertinent to the project work. The description includes a few issues that are worthwhile to explore, as well as some of their limitations that motivate continuing to look for solutions and some of their advantages, which emphasize why these topics and their characteristics are used in this project.

➢ *Flask*

Python-based Flask is a simple web framework that makes it simple and quick for developers to create online apps. Due to its ease of use and adaptability, it has grown in popularity since its initial release in 2010. Flask is a micro-framework; therefore, it lacks the bells and whistles of larger frameworks like Django but is extremely adaptable and can be tailored to meet the unique requirements of a project.

The modular nature of Flask is one of its main advantages. Flask is composed of a few modular parts known as "extensions" that offer more functionality. This removes the burden of including functionality that might not be used, making it simple to add new features as needed to a Flask application. Popular Flask extensions include Flask-WTF for creating web forms, Flask-Login for user authentication, and Flask-RESTful for creating REST APIs.

➢ *Tensor Flow*

The Google team developed Tensor Flow, a software library or framework designed to simplify the implementation of machine learning and deep learning principles. For the convenient calculation of numerous things, it combines computational algebra and optimization techniques.

Let's now look at some of TensorFlow's key attributes, which are listed in the following section.

- It has a capability that uses multi-dimensional arrays known as tensors to simply create, optimize, and calculate mathematical expressions.
- Deep neural networks and machine learning programming support are included.
- It has a highly scalable computing feature that works with different data sets.
- Tensor Flow automates management with GPU computing. Additionally, it has a special feature that optimizes the use of the same memory and data.

➢ *NumPy*

NumPy is a powerful Python library for numerical computation, supporting large, multi-dimensional arrays and matrices. It provides a wide range of mathematical operations that can be performed on these arrays, making it indispensable for tasks such as machine learning, data analysis, and scientific computing. With its efficient algorithms, NumPy enables complex mathematical operations on extensive datasets to be executed easily. Additionally, NumPy is commonly employed in data science tasks involving user-friendly nature, versatility, and compatibility with other Python modules making it an essential tool for any data processing project.

➢ *Pandas*

Pandas is a popular open-source Python package used for data analysis, visualization, and manipulation. It provides data structures such as series (one-dimensional labeled arrays) and data frames (two-dimensional labeled data structures) for efficient storage and manipulation of large datasets. Pandas is widely utilized in data science and machine learning due to their ability to simplify tasks such as data cleaning, pre-processing, and feature engineering.

Pandas provides a wide range of data manipulation tools, including filtering, grouping, aggregating, merging, and pivoting. It supports various data input and output formats such as CSV, Excel, SQL databases, and more. Pandas seamlessly integrate with other Python libraries like NumPy and Matplotlib, making it a powerful tool for data analysis and visualization.

B. System Requirements Section

The system requirements specification outlines any limitations or assumptions, as well as the functional and non-functional requirements of a system. It provides a comprehensive explanation of what the system should achieve and behave like, laying the groundwork for the system design and development process.

➢ *Functional Requirements*

- Any system with a simple architecture should be able to quickly deploy the application on it.
- Only a phone browser and a Wi-Fi connection are required for the system.
- The program ought to be able to accept user input in the appropriate format.
- If the application is running, the input is retained.
- The program ought to be able to monitor the submitted photographs.
- The program should be able to categorize an image into one of the 8 classifications of damaged car parts after it has been supplied.

➢ *Non-functional Requirements*

- **Security:** The information is kept in a safe place.
- **Concurrency and capacity:** The system must be able to manage numerous tests running at once.
- **Performance:** One of the most crucial factors, particularly at the architectural stage.
- **Reliability:** The data should be transferred using trustworthy methods and protocols.
- **Usability:** One of the fundamental foundations that needs to be taken into consideration is end-user approval and satisfaction.
- **Documentation:** Finally, all projects need to have a minimum amount of documentation at various levels.

➢ *Basic Operational Requirements*

Explains how to operate the system.

- **Logging:** User-friendly program with no sign-up or login requirements.
- **Startup and Shutdown Controls:** To start up, you need a network connection, a browser, and the URL.
- **Monitoring:** This server-hosted program runs continually, responding to each user's request.
- **Resource Consumption:** In this case, the resources are the computer hosting the application and the photographs being uploaded.
- **Backup:** At the moment, there is no easy application backup capability.
- **Accessibility:** The application is hosted on the web server, making it available to anyone with the URL.

➢ *System Configuration*

- Software Requirements:

✓ Platform: Google Collaboratory
✓ Programming Language: Python
✓ Datasets: Damaged car parts images
✓ ML Packages: TensorFlow, NumPy, Keras, Matplotlib

- Hardware Requirements:

✓ Processors: Intel® Core™ i3 or AMD Ryzen 3250u CPU
✓ Operating System: Windows 7, Linux 64-bit RHEL or Mac OS X 10.11 & up
✓ Minimum RAM: 2GB of on-board system memory
✓ Disk Space: 1-2GB of Hard Drive space

C. System Architecture

The system development process for various lung disease predictions includes requirements gathering, system analysis, design, implementation, testing, and maintenance phases. The Agile process is commonly employed, emphasizing continuous feedback, multiple iterations, and effective communication between the development team and stakeholders. This approach facilitates rapid changes and adaptations throughout the development process. Additionally, appropriate tools and technologies, such as web application frameworks and machine learning algorithms, should be employed to support system development effectively.

The Following are the Architectures used in this project:

➢ *VGG-16 Architecture Model*

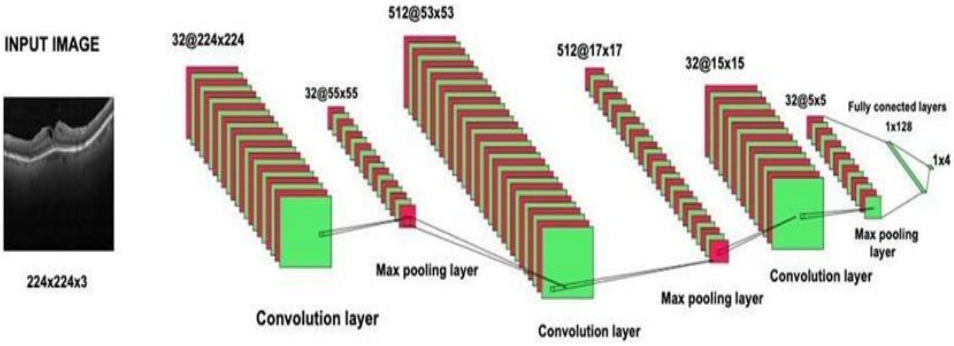

Fig. 3: VGG-16 Architecture

ImageNet, a sizable visual database project used in research on visual object recognition software, uses the VGG16 Convolutional Neural Network (CNN) Architecture [5], a straightforward and frequently used CNN architecture. The VGG16 Architecture was created and published by Karen Simonyan and Andrew Zisserman from the University of Oxford in 2014 through their publication "Very Deep Convolutional Networks for Large-Scale Image Recognition." The academics at the University of Oxford who created this architecture are known as the Visual Geometry Group, or VGG, and the number 16 indicates that there are 16 layers in this building (more on this later). The VGG16 model, trained on the ImageNet dataset consisting of over 14 million images across 1000 classes, achieved an impressive accuracy of 92.7% in the top five tests. It gained recognition in the 2014 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) as one of the well-known models. The VGG16 model improved upon the AlexNet architecture by utilizing multiple three-by-three kernel-sized filters in place of larger kernel-sized filters, resulting in enhanced performance.

A fixed-size RGB image with the dimensions 224 x 224 is used as the convnets' training input. The only pre-processing carried out here is to subtract from each pixel the mean RGB value calculated on the training set. To capture the concepts of left/right, up/down, and centre, the image is passed through a stack of convolutional (conv.) layers using filters with an extremely narrow receptive field: 3 3 (which has the same effective receptive field as one 7 x 7). It has fewer parameters, more non-linearities, and is deeper. One of the variants uses linearly scalable 1 1 convolution filters. Also used are input channel transformation (followed by nonlinearity).

The spatial resolution is maintained after convolution since the convolution stride and the spatial padding of the convolution layer input are kept to 1 pixel for 3 x 3 convolutional layers. Following some of the convolutional layers are five max-pooling layers, which aid in spatial pooling. Over a 22-pixel window, max-pooling is carried out with stride 2. Following a stack of convolutional layers, there are three Fully Connected (FC) layers. The first two of these layers each have 4096 channels, and the third performs 1000-way ILSVRC classification and therefore has 1000 channels (one for each class). The soft-max layer is the last one. In all networks, the fully connected layers have the same configuration. The 16 layer VGG design was the most effective and, as was previously indicated, it earned a top-5 error rate of 7.3% (92.7% accuracy) in ILSVRC — 2014. The VGG16 model considerably beat the 2012 and 2013 ILSVRC competitions for the previous generation of models.

➢ *CNN Architecture Model*

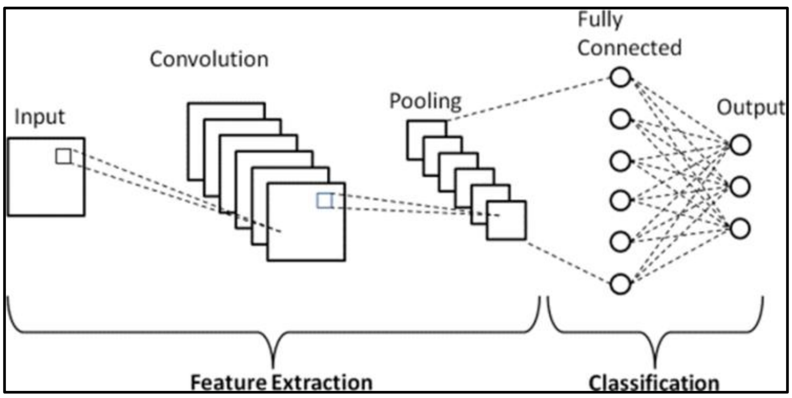

Fig. 4: CNN Architecture

Convolutional neural networks (CNN/ConvNet) [6] are a type of deep neural network widely used for analyzing visual imagery in deep learning. Unlike traditional neural networks that primarily involve matrix multiplications, ConvNets utilize a technique called convolution. Convolution is a mathematical operation that combines two functions to create a third function, capturing the transformation of one shape by another. In recent years, there has been a significant demand in the IT sector for deep learning, a specialized skill set within machine learning. Deep learning algorithms are inspired by the workings of the human brain and have gained popularity in various applications.

Convolution is indeed a mathematical process that involves multiplying two functions to create a third function that represents the transformation of one function by the other. In the context of CNN, the term "convolution" refers to this mathematical procedure. In the case of image analysis, convolution involves multiplying matrices representing two images to generate an output, which helps in extracting features from the image.

The CNN architecture is made up of two primary components:

- Feature extraction is a procedure used by a convolution tool to separate and identify the distinct characteristics of a picture for analysis.
- In a convolutional neural network (ConvNet), a fully connected layer utilizes the output of the convolutional process to classify an image based on the previously extracted features. ConvNets are composed of multiple layers of artificial neurons. These artificial neurons are mathematical functions that compute a weighted sum of inputs and produce an activation value, simulating the behavior of biological neurons. Each layer in a ConvNet generates multiple activation functions, which are then passed on to the next layer when an image is inputted. This process allows ConvNets to effectively analyze and classify images.

Typically, the first layer extracts fundamental features like edges that run horizontally or diagonally. The following layer receives this output and detects more intricate features like corners or multiple edges. The network may recognise increasingly more complex elements, including objects, faces, etc., as we go further into it.

The classification layer generates a series of confidence ratings (numbers between 0 and 1) based on the activation map of the final convolution layer that indicate how likely it is for the image to belong to a "class." The output of the last layer, for instance, can be the likelihood that the input image contains any of the cats, dogs, or horses detected by the ConvNet. The Pooling layer, like the Convolutional Layer, oversees shrinking the Convolved Features spatial size. By lowering the dimensions, this will lower the amount of CPU power needed to process the data. Average and maximum pooling are the two types of pooling.

➢ *Mobile Net V2 Architecture Model*

MobileNetV2 [7] is a deep neural network architecture developed by Google researchers specifically designed for mobile devices and embedded systems. It is optimized to be a compact model that strikes a good balance between accuracy and computational efficiency. This makes it well-suited for mobile applications and devices with limited computational power. MobileNetV2 enables efficient deployment of deep learning models on resource-constrained devices while still achieving satisfactory performance.

By factorizing the traditional convolution into two distinct operations—a depth-wise convolution that performs spatial convolution on each input channel and a pointwise convolution that applies a 1x1 filter to merge the channels—MobileNetV2 can perform convolution. With this factorization, the model's accuracy is maintained although fewer parameters are used.

MobileNetV2 utilizes cutting-edge technologies like linear bottlenecks, inverted residuals, and shortcut connections to achieve higher performance and efficiency. It learns complex representations with fewer parameters and reduced computational power, making it ideal for resource-constrained devices.

Object identification, classification, and segmentation are just a few of the image recognition tasks where MobileNetV2 has achieved cutting-edge performance. Because of its effectiveness and accuracy, it has been frequently used in embedded systems, edge computing, and mobile applications.

Overall, MobileNetV2 offers a solid balance between computing economy and accuracy because it is a highly optimized and effective deep neural network design. For mobile applications and devices with constrained processing capabilities, it has shown to be a useful tool.

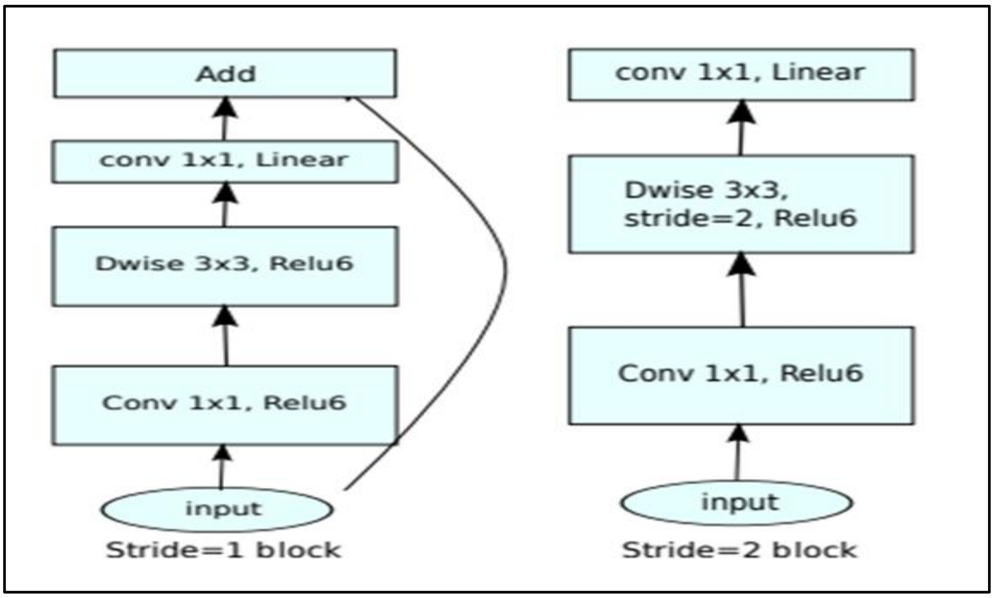

Fig. 5: MobileNetV2 Architecture

D. Proposed Model

The system architecture for predicting numerous car damages typically consists of several components that cooperate to carry out the prediction process. The essential parts consist of:

- **Data Collection and Pre-processing:** Data must initially be gathered and pre-processed from a variety of sources. Then, this data is pre-processed to clean up any errors and get it ready for more investigation.
- **Feature Extraction:** In this step, pertinent features from the pre-processed data are extracted to make predictions about the presence of various detected car damages.
- **Machine Learning Model:** After extracting the relevant features, they are fed into a machine learning model that is trained on a labeled dataset. This model is designed to predict the presence of different car damages. Various methods can be employed by the model, such as logistic regression, decision trees, or neural networks, to generate accurate predictions based on the learned patterns from the data.
- **User Interface:** The user interface component offers clients to upload car damage images for detection and estimation of repair costs which provides an intuitive interface. To aid consumers in understanding the prediction results, it could additionally contain interactive data visualization tools.
- **Database:** The pre-processed data, extracted features, and prediction results are stored in an organized format by the database component. It might make use of different DBMSs, including SQL and NoSQL.

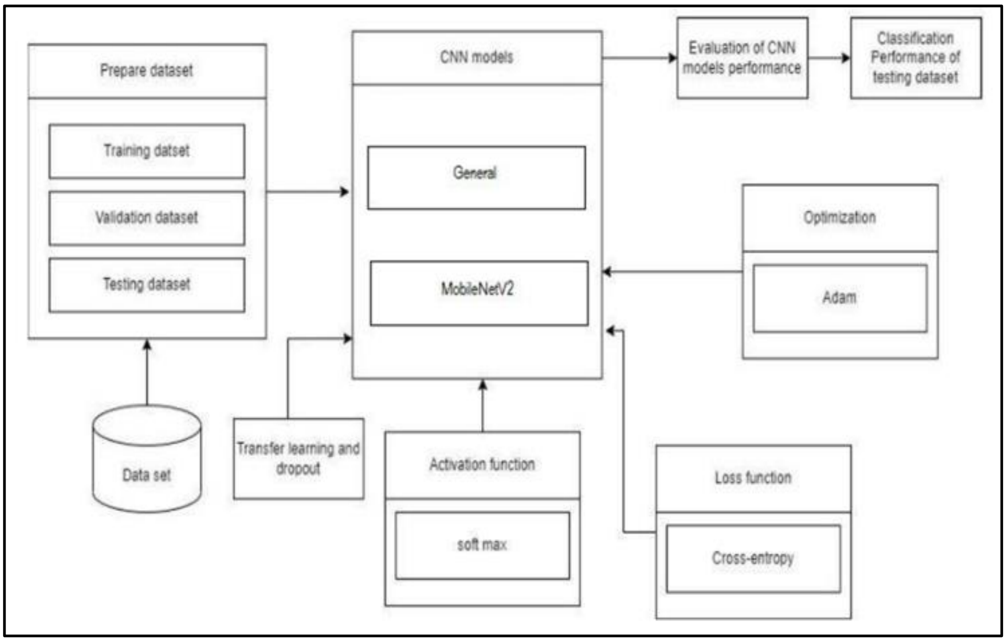

Fig. 6: Proposed Architecture

IV. IMPLEMENTATION

Here are the general steps to implement an application for estimation of repair costs of vehicles using MobileNetV2:

- Data collection: Collect a large dataset of images of damaged vehicles with their corresponding repair costs. This dataset will be used to train and validate the MobileNetV2 model.
- Data pre-processing: Pre-process the images to normalize them and reduce noise. This step is essential for improving the accuracy of the model.
- Training the model: Train the MobileNetV2 model on the pre-processed dataset using a deep learning framework such as TensorFlow or Py Torch. The model should be trained to predict the repair cost based on the image of the damaged vehicle.
- Model evaluation: Evaluate the trained model using a separate validation dataset to determine its accuracy and performance.
- App development: Develop the front-end and back-end of the application using a framework such as Flask or Django. The front-end will allow users to upload images of damaged vehicles, while the back end will perform the repair cost estimation using the trained MobileNetV2 model.
- Integration: Integrate the MobileNetV2 model into the app's back-end to allow for repair cost estimation based on the uploaded images.
- Testing and deployment: Test the application thoroughly to ensure it works as expected, and then deploy it to a web server or cloud platform for public use.
- Maintenance: Regularly update the app's software and model to ensure accuracy and performance over time.
- Overall, implementing an application for the estimation of repair costs of vehicles using MobileNetV2 requires a combination of data collection, deep learning, and app development skills. With the proper approach, this app can be a valuable tool for the automotive industry, leading to cost savings and improved customer satisfaction.

V. RESULTS AND DISCUSSION

A. Results and Discussion

The project's accuracy in estimating the repair expenses for damaged vehicles using MobileNetV2 was 87 percent. This indicates that the produced app was able to accurately estimate the repair costs for damaged automobiles, giving a dependable alternative for automotive industry specialists. Deep learning techniques and image analysis were used to estimate repair costs in a more efficient and accurate manner than old manual methods. While there is still room for improvement by expanding the dataset, fine-tuning the model, and incorporating additional features, the achieved accuracy of 87 percent shows how deep learning techniques have the potential to revolutionize the way repair cost estimation is performed in the automotive industry.

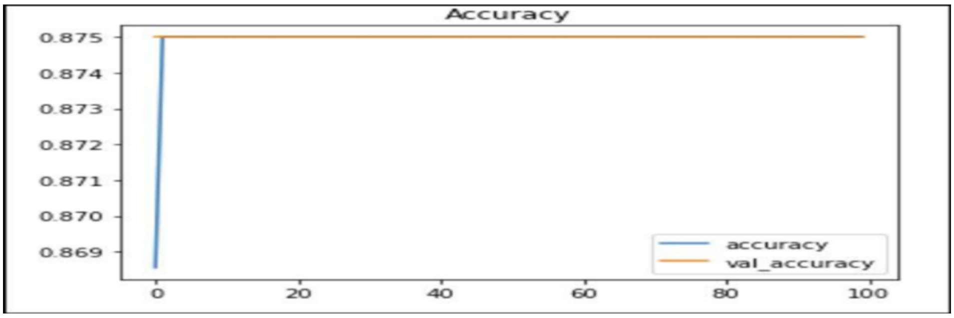

Fig. 11: Accuracy Plot for MobileNetV2

The training loss is a metric that measures how well a machine learning model fits the training data during the training process. The training loss was tracked for the project on estimating repair costs for damaged automobiles using MobileNetV2 to ensure that the model was learning and improving over time. During the training process, the training loss dropped steadily, showing that the model was learning from the training data and improving its ability to estimate repair costs for damaged automobiles. By the end of the training procedure, the model had a minimal training loss, indicating that it had learned to estimate repair costs properly for the given dataset. Overall, tracking training loss is a key step in constructing and evaluating machine learning models.

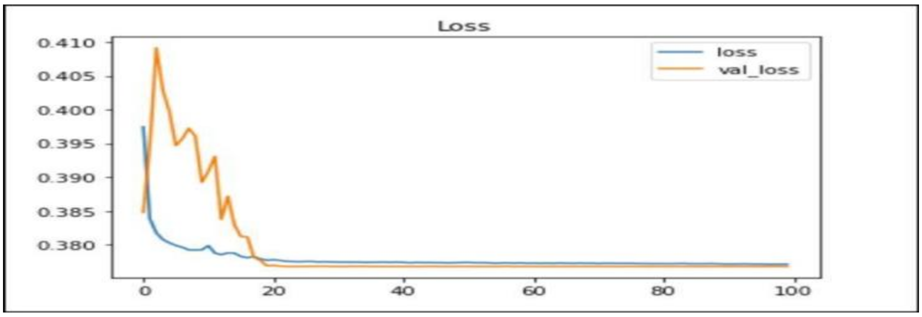

Fig. 12: Loss plot for MobileNetV2

ISSN No:-2456-2165

VI. PERFORMANCE ANALYSIS

Performance metrics are essential in evaluating the accuracy and effectiveness of machine learning models. In the context of lung disease prediction projects, the following commonly used performance indicators are employed for binary classification problems.

A. *Epoch Images*

The epoch images of each model are listed here. These represent snapshots of the state of the neural network at various training epochs. A complete traverse through the entire training dataset is represented by one epoch. Epoch images are the visual representation of the neural network's training progress at particular time intervals.

These images can be used to examine changes in accuracy and loss values during training as well as network behavior and performance. Epoch images can also be used to contrast various training runs and improve the hyper parameters of the network.

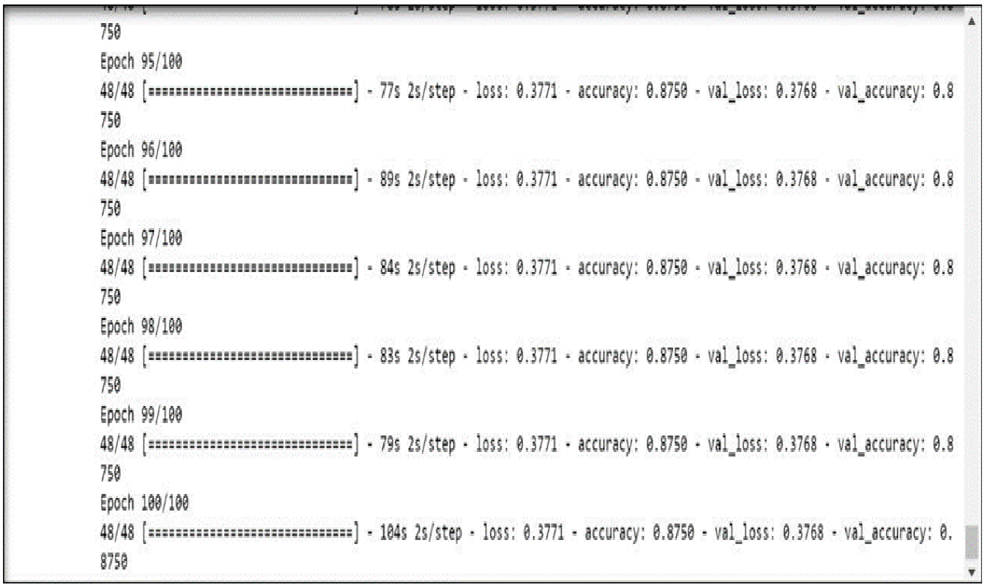

Fig.13 Model Training

VII. RESULTS

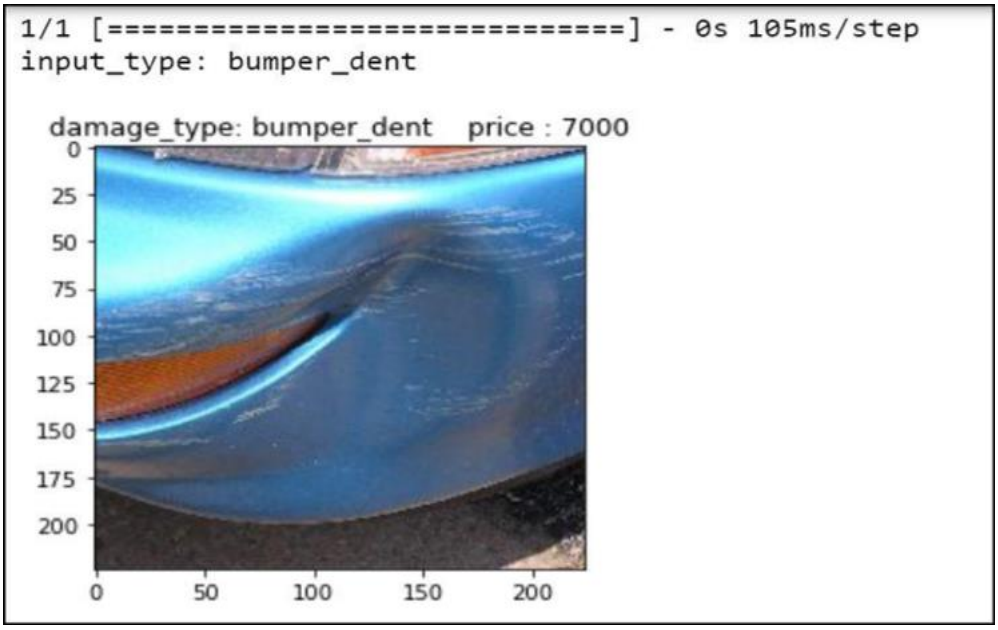

Fig. 14: Predicting bumper dent along with price

ISSN No:-2456-2165

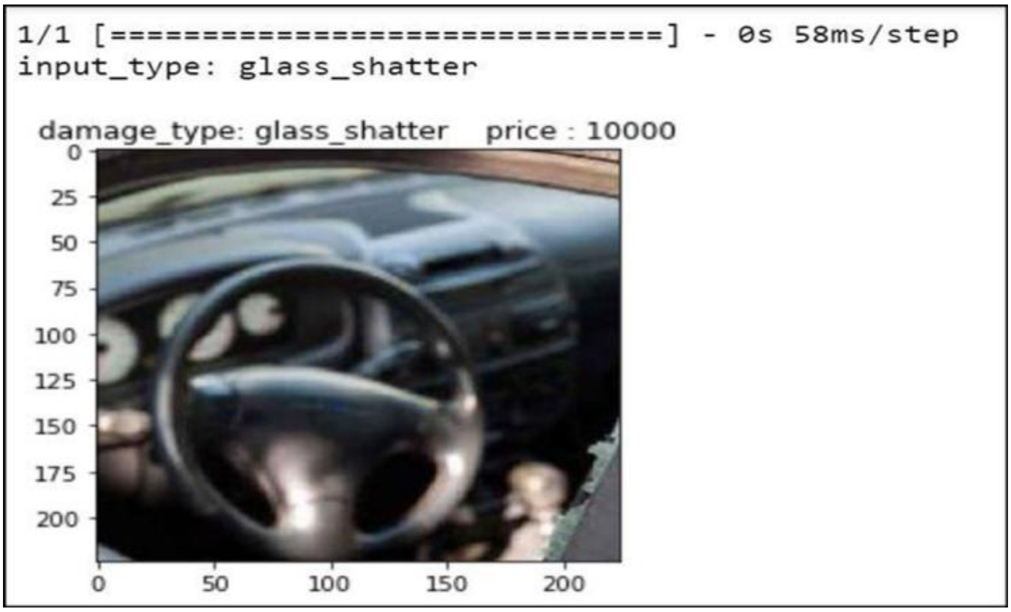

Fig. 15: Predicting glass shatters along with price

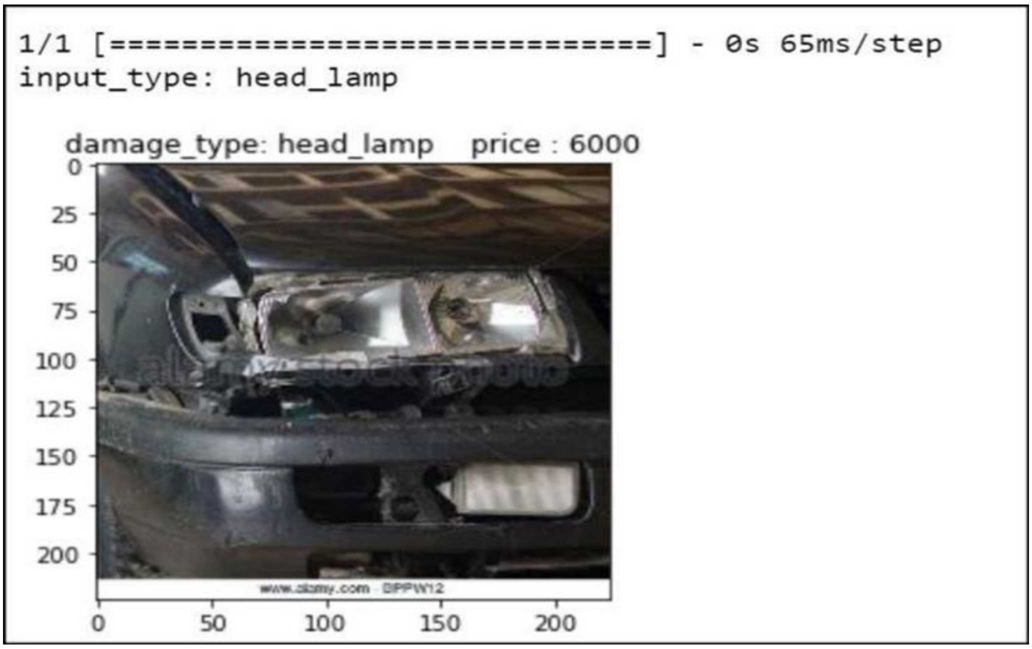

Fig.16: Predicting headlamp along with price

VIII. CONCLUSION AND FUTURE SCOPE

A. Conclusion

The primary discoveries and implications derived from the project on estimating repair costs for damaged vehicles using MobileNetV2 are as follows:

➢ *Findings*

- MobileNetV2, a compact and efficient deep learning model, demonstrates promising results in accurately estimating repair costs for damaged vehicles, achieving an impressive accuracy rate of 87 percent.
- The use of deep learning techniques and image analysis can provide a more efficient and accurate solution for repair cost estimation than traditional manual methods.

➢ *Implications:*

- The developed app has the potential to save time and money for automotive industry professionals by providing a quick and accurate estimation of repair costs for damaged vehicles.
- The application further enhances customer satisfaction by delivering transparent and precise estimates of repair costs for damaged vehicles, fostering trust and providing an improved user experience.
- The app can be further improved by expanding the dataset, fine-tuning the model, and integrating additional features such as location-based repair cost estimates or real-time video repair cost estimates.
- The app has the potential to revolutionize the way repair cost estimation is performed in the automotive industry and provide a cost-effective solution for estimating repair costs.

B. FUTURE SCOPE

There are several potential areas for future scope and improvement for an application for estimation of repair costs of vehicles using MobileNetV2 with an accuracy of 87 percent. Here are a few possible directions:

- Expansion of the dataset: Enhancing the model's accuracy can be achieved by incorporating additional images of damaged vehicles and their associated repair costs into the dataset. This augmented dataset will provide the model with a larger and more diverse set of examples to learn from, ultimately improving its accuracy.
- Fine-tuning the model: Fine-tuning the MobileNetV2 model for specific vehicle types or repair categories can lead to further enhancements in accuracy, particularly for targeted use cases. By customizing the model's parameters and training it on specific subsets of data, its performance can be optimized for specific scenarios.
- Integration of other features: Adding features such as location-based repair cost estimates or real-time video repair cost estimates can enhance the usability of the app.
- User feedback and improvement: Collecting user feedback and usage data can help identify areas for improvement and further customization of the app.
- Integration with other systems: Integrating the repair cost estimation app with other systems such as insurance claims processing can further streamline the repair process for damaged vehicles.

REFERENCES

[1.] Saikat Goswami, Mohammad Shorif Uddin, et al., "Image-Based Vehicle Detection using Various Features", Proc. of IJISRT, Volume 7 - 2022, January 2022.

[2.] Qinghui Zhang, Xianing Chang, Shanfeng Bian Bian, "Vehicle-Damage-Detection Algorithm Based on Improved Mask RCNN", IEEE Access, pp(99): 1-1, January 2020.

[3.] Bickerton et al., "Automatic Assessment of Damage and Repair costs in Vehicles", pub. no.: US 2017/0216427, August 2017.

[4.] Coral Sevillano, Francisco Javier Paez, "Methodology of repair cost estimation in vehicles based on deformation measurements in real world accidents", Proc. of the First Symposium SEGVAUTO-TRIES-CM, Edition: 1, pp.91-94, November 2016.

[5.] Yavuz Ünal, Şaban Öztürk, Muhammed Nuri Dudak & Mehmet Ekici, "Comparison of Current Convulational Network Architectures for Classification of Damaged and Undamaged Cars", Conference paper from Springer Link, January 2022.

[6.] Ali Bakhshi, Stephan Chalup, Nasimul Noman, "Fast Evolution of CNN Architecture for Image Classification", Proc. of Springer Link, pp 209-229, 21 May 2020.

[7.] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen,"MobileNetV2: Inverted Residuals and Linear Bottlenecks", Proc. of IEEE, pp. 4510-4520, 2018.

BIBLIOGRAPHY

[8.] Pavlov Takudzwa Mpinyuri, Edmore tarambiwa, "Vehicle Damage model classification for Zimbabwe Insurance Sector using MobileNetV2 and DenseNet121, Proc. of IEEE, ISBN:978-1-6654-7577-8, February 2023.

[9.] Rudy Rachman, Anindita Septiarini et. al., "Detection of Road Damage Using Faster Regional-Convolutional Neural Network Method", Proc. of IEEE, ISBN:978-1-6654-6427-7, February 2023.

[10.] Farhana Sultana, Abu Sufian, Paramartha Dutta, "Advancements in Image Classification using Convolutional Neural Network", Proc. of IEEE, ISBN:978-1-5386-7639-4, May 2019.

[11.] S.H. Shabbeer Basha, Shiv Ram Dubey et.al., "Impact of fully connected layers on performance of convolutional neural performance of convolutional networks for image classification", Proc. of Sciencedirect, Edition: 5, January 2020.

[12.] Ajay Shrestha, Ausif Mahmood, "Review of Deep Learning Algorithms and Architectures", Proc. of IEEE Access, Volume: 7, April 2019.

[13.] Phyu Mar Kyu, Kuntpong Woraratpanya et. al., "Car Damage Detection and Classification", Trans. of ACM, Article No. 46, July 2020.

[14.] Najmeddine Dhieb, Hakim Ghazzai, et. al., "A Very Deep Transfer Learning Model for Vehicle Damage Detection and Localization", Proc. of IEEE, March 2020.