AUTO-CLAIM CAR INSURANCE USING IMAGE PROCESSING

Pavithra Yarrakula
11606138
Email: pavitrayarrakula@my.unt.edu

Sriram Relangi
11556267
Email: sriramrelangi@my.unt.edu

priyanka jampani
11653109
Email: priyankajampani@my.unt.edu

***Abstract*—Dealing with the aftermath of a car accident and then going through the insurance process is quite a hassle. Traditionally the responsibility for detecting and handling the insurance is done by the insurance company where the surveyor visits the scene which has a series of steps like collecting images, submitting documents etc. This is the extensive process which is time consuming and leaves room for potential issues like risk of delays, the surveyors can make errors or engage in dishonest practices. Taking into consideration all these challenges we have embarked on a mission to simplify the insurance claim process. Our project empowers the responsible driver by offering efficient ways to calculate the appraisal and recovery costs. Using image processing, particularly leveraging Convolution Neural Networks, we have created a user-friendly platform. Now instead of waiting for the surveyor at the scene the user can take and upload images onto the website. By simplifying the claim process and putting control back in the hands of the affected parties, we aim to identify the damage to the vehicle, which pinpoints the part of the damage and its severity.**

***Index Terms*—persistencia, políglota, NoSQL, MongoDB, N eo4j.**

I. INTRODUCTION

We understand that the world of insurance claims can be maze, especially when we have into such a situation to deal with. We are about to flip the scenario and make the insurance process smooth and joy ride. We have harnessed the power of image processing delving into the fascinating realm of Convolution Neural Networks. Now there is no need for the hassle of waiting for the surveyor and drowning in the paperwork. Then here the task of the driver responsible is to snap the picture of the damage and let our system work. Here we have crafted a user-friendly platform where it not only identifies the damage but also points out the severity of the damage. This gives lightning-fast and spot-on estimates of those repair costs putting in the driver seat of the whole appraisal process.

II. MOTIVATION AND OBJECTIVES

A. Objective

We are on a mission to make life simpler and smoother when it comes to dealing with the auto insurance claims. We are taking serious savings, super smooth operations and using the data to keep risks in check. Here the goal of bringing up image processing is cut down the on time this takes claiming the insurance process and improves the accuracy of the damage assessment for giving the holders faster and convenient services. By protecting the security of the customer information which is sensitive which reduces the fraudulent claims which enhances customer satisfaction. This will result in cost saving, increased operational efficiency and compliance.

B. Significance

This is process is set to turbocharge the auto claim insurance process. This will improve the fairness, transparency, and efficiency of the auto – claim insurance process which provides benefit to both policy holders and the insurers. This is the crucial step in how the insurance company for better experience and services to the customers.

C. Motivation

This project uses advanced technology for making the insurance process efficient, user-friendly, and accurate. This aims to address longstanding issues in the insurance company, which helps improve the relation between policy holders and insurers. There is no more waiting for the verdict, the driver becomes the detective. This is not just about saving time but also this complete process feels like a chore.

III. DATA COLLECTION

We have collected our own dataset which contains different images representing vehicle damage. Our classification consists of seven commonly observed damage types which includes bump rod, door damage, glass crack, headlamp breakage, tail lamp breakage, scratching and cracking. We have gathered images that are non-abrasive additionally which helps to ensure the robustness of our dataset. These images are collected from the web and customized to satisfy the need of the project. The main aim of these image customizationś is to align them with the project's specific requirements. By personally curating and adapting these images we have tailored our dataset to meet unique challenges and intricacies of automating the insurance claim process which helps to increase the accuracy of the model. Here we have considered various damages that occur in real-world situations which help to train our model effectively. Achieving this diversity requires effective searching and selection of images from various sources.

Striking the right balance between the damaged and non-damaged images is important for the model's accuracy and for identifying the vehicle damage. The process of customizing images to meet the project requirements was time consuming. For addressing the need for the large dataset, we have employed the augmented data techniques such as random rotation and horizontal flipping. Striving for the dataset that accurately represents the real-world scenarios was a continuous effort. This involved ongoing refinement and adjustments to make sure that our dataset remains relevant and reflective of the actual challenges encountered in auto insurance claims. Here we are using the power of computers and neural networks to assess vehicle damage and handle insurance claims more accurately and faster.

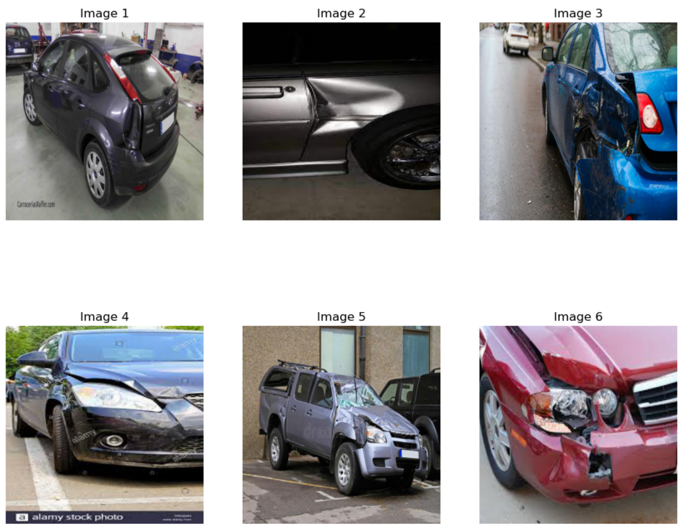


Fig. 1. Car damaged images

IV. Data Preprocessing

Data preprocessing is the crucial step for image classification tasks. In the previous section we collected data and now, before sending it into the model, we must clean it. The data cleaning steps include removing duplicate images, handling missing or corrupted images, cropping images, normalizing the pixels, correcting label errors, balancing the class distribution, removing the outliers and data augmentation.

Firstly, we consider removing the duplicate images which results in biased model accuracy. This is the essential step to reduce the redundancy in our dataset. The next step of data preprocessing includes handling missing or corrupted images where images may be damaged or incomplete which poses a challenge to the learning algorithm.

Normalization of pixels comes to play to standardize the pixel values for all the images. This step ensures consistency in the scale of pixel intensities which prevents images from dominating the learning process because of variations in the pixel range.

Identifying and removing outliers further refines the dataset. Outliers, which are data points significantly different from others, can distort the learning process. By systematically detecting and eliminating these outliers, we enhance the robustness of our dataset.

Finally, data augmentation involves generating new, slightly modified versions of existing images. This step introduces diversity into the dataset, allowing the model to generalize better to various scenarios. Through these augmented images, the model becomes more efficient at recognizing patterns and features, contributing to its overall performance on unseen data. Together, these data cleaning steps form a comprehensive and meticulous process, ensuring that our image dataset is well-prepared for effective machine learning. This multi-step approach ensures the meticulous tracking and handling of duplicate images within our dataset, contributing to the overall cleanliness and reliability of the data we use to train and evaluate machine learning models.

V. Model Parameters

In a way to identify the image recognition capabilities we have created a model which comprises of convolution layer, dense layer, activation layer, Max Pooling layer and Flatten Layer. We have structured the architecture effectively to handle images of size 640*640 pixels.

A. *Convolution Layer*

The initial layer is the convolution layer which scans the input image which detects the features such as edges, textures, and padding, makes sure that the size of the output images is same as the input image and maintains the dimensionality of the images.

B. *Activation Layer (ReLU)*

This layer helps the model to learn the complex patterns. For example, this layer takes an input image and then passes the image to different layers and helps to identify objects in the image. To make this process effective the layer has to understand which data important and which data is not. For this situation we use the activation layer. This ReLU works on simple principle that is if the input is positive, it remains as same, if the input data is negative then it changes the value to zero. This layer helps to identify the complex patterns of the image and distinguish between the damaged vehicles.

C. *Max Pooling Layer*

This max pooling layer helps to reduce the computational load and overfitting by giving abstract features. This helps to focus on the most important part of the features, for example here in our images of vehicle damage. The damaged part is important, so this max pooling layer helps to focus on the damaged part of the image. Another advantage of using this layer is that it prevents overfitting.

D. *Flattening*

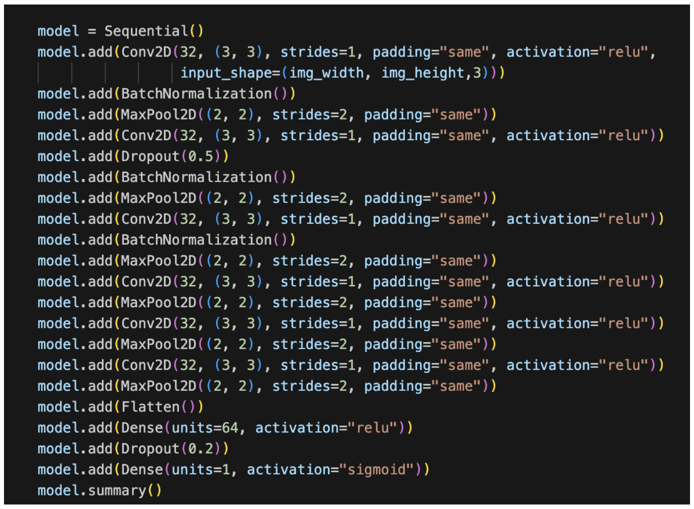
After the convolution layer and maxpooling layer we perform the flattening to convert the image to one dimensional array. This conversion is important to pass the data to the next layer which is the dense layer. We can say that the above layers detect the features, and the detected features can be feed into the below layers for decision making where flattening helps to convert the features to one dimensional array which are fed to the below layers to make the decisions.

E. Dense Layer

The fully connected layer of 64 units is used to interpret the features extracted by the convolution layer. Here we are considering 64 units, this is like the number of people working for a company if the people are more than overfitting or if the people are less than we are not able to handle the work. Similarly, for our model, we are using image processing where the model must identify and classify the prominent features of the image without being overwhelming.

F. Output Layer

Here the output layer is considered as the decisive moment. By using the sigmoid activation function the output layer assigns the probability score between 0 to 1 which gives the decision regarding the image classification. The closer the probability score is 1 the more resolute that image belongs to the specific category.



```
model = Sequential()
model.add(Conv2D(32, (3, 3), strides=1, padding="same", activation="relu",
                input_shape=(img_width, img_height,3)))
model.add(BatchNormalization())
model.add(MaxPool2D((2, 2), strides=2, padding="same"))
model.add(Conv2D(32, (3, 3), strides=1, padding="same", activation="relu"))
model.add(Dropout(0.5))
model.add(BatchNormalization())
model.add(MaxPool2D((2, 2), strides=2, padding="same"))
model.add(Conv2D(32, (3, 3), strides=1, padding="same", activation="relu"))
model.add(BatchNormalization())
model.add(MaxPool2D((2, 2), strides=2, padding="same"))
model.add(Conv2D(32, (3, 3), strides=1, padding="same", activation="relu"))
model.add(MaxPool2D((2, 2), strides=2, padding="same"))
model.add(Conv2D(32, (3, 3), strides=1, padding="same", activation="relu"))
model.add(MaxPool2D((2, 2), strides=2, padding="same"))
model.add(Conv2D(32, (3, 3), strides=1, padding="same", activation="relu"))
model.add(MaxPool2D((2, 2), strides=2, padding="same"))
model.add(Flatten())
model.add(Dense(units=64, activation="relu"))
model.add(Dropout(0.2))
model.add(Dense(units=1, activation="sigmoid"))
model.summary()
```

Fig. 2. Traditional CNN Architecture

VI. Approach Of Model Building

In the code serves as the proclamation purpose which specifies the model quest is to minimize the binary cross entropy and navigate through the data using the adam optimizer and provides the accuracy metrics. During the training, Early Stopping acts as the compass which monitors the model progress. This stops the training process when no substantial changes are detected. Then the ModelCheckpoint is used to document the accomplishments of the model. This makes sure that the important model achievements are not lost, which provides a tangible record of success. The model is being serialized into JSON format. This helps the model to effectively summarize the configurations. The results of the JSON are stored in the model.json file and the concurrent model weights are being saved in the HDF5 format by preserving the learned parameters of the model.

The saved JSON file is read by using the I/O operations and the content is loaded into the variable. By using model_from_json we reconstruct the original model and transform the model JSON representation back to the functional Keras model. Then the next step involves compiling it with the same configuration as the original model. The model is compiled with the categorical entropy loss and adam optimizer and accuracy of metric assessment. The loaded model is given with the name as car_model which is being indicated in the function. The model serialized and learned weights is preserved, ensuring its performance.

A. Final Prediction

After the image is preprocessed appropriately then we move into the prediction phase of the model. The previously trained model is used for image classification to make predictions on the image tensor. The output predictions are stored in pred which are used for the inspection. The values set above 0.6 are predicted as 1 and the values below 0.6 as 0, which indicates the less prediction. We must be attentive to any signs of divergence or fluctuations resulting in the model's overfitting.

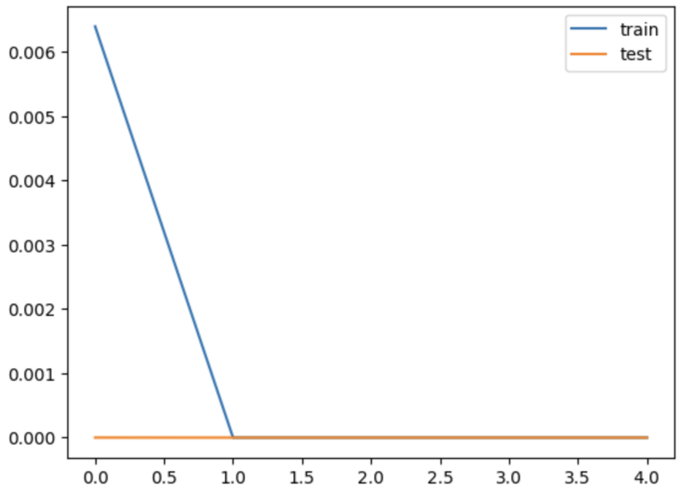


Fig. 3. Validation in train and Test

The important act unfolds through the implementation of the evaluate_generator which focuses on scrutinizing the model on the validation dataset. The model processes the validation data and compares them with the ground truths. The insights generated by this evaluation explain to us the model capabilities and readiness for deployment. This is a visual journey that transcends the predictions which provide the human intuition of identifying the damaged vehicles.

VII. Model Improvement

In our project, the model must identify and recognize the damaged vehicles where convolution neural networks (CNNs) play a remarkable role in performing a wide array of tasks for image recognition for understanding the natural language.

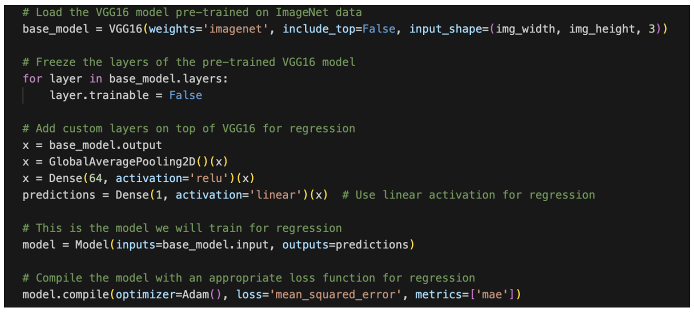
Fig. 4. Prediction of model

CNN helps in particularly spotting the intricate patterns of the images which identifies the damaged part of the vehicle. When we know CNN provides satisfactory results, it acknowledges that transfer learning increases the ability to provide extraordinary outcomes. In our project we are delving into the implementation of advanced techniques such as VGG16. Here we are importing pre-trained models, including weights designed originally for recognizing the images. The final layers are of the pre-trained models and are replaced with the fresh layers which are finely tuned which specifically intricate the categorizing the damaged vehicle images.

Now we are using a predefined model VGG16 to train the dataset and then to find the severity of damaged vehicle.

A. VGG16

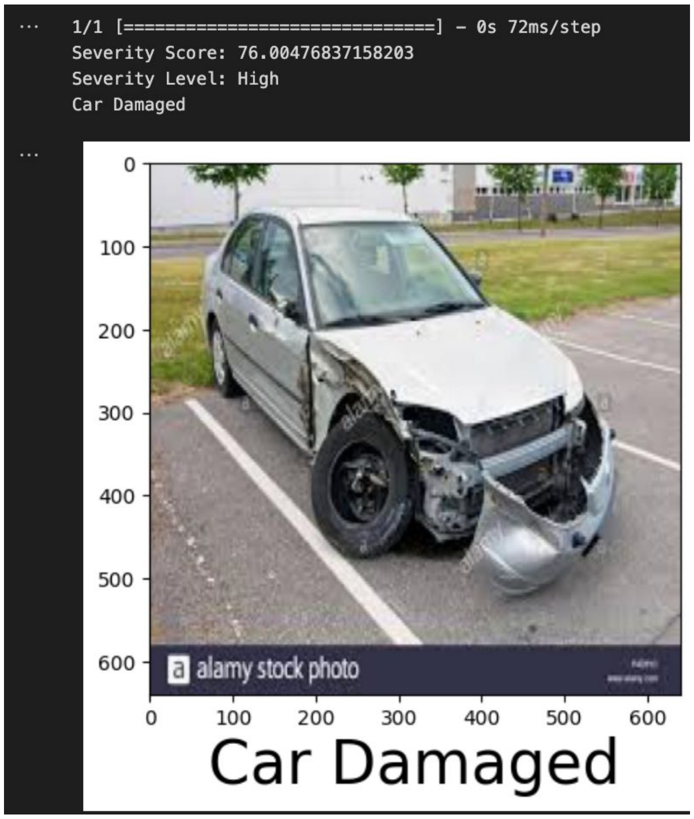
We have implemented a predefined model for assessing the severity of car damage in our dataset. Our project aim is evaluating the car damage and assisting in the case of insurance claim process. Then coming to VGG16 model the architecture contains the linear stack of layers which are sequential. These layers extract the information from the images and perform the max pooling which helps to reduce the spatial dimensions. In our custom model there is overfitting that we faced and now we are trying to prevent a dropout layer from removing a random value during the training. The training history includes the metrics such as loss and mean absolute error are recorded. Then the output is generated, suitable for the regression tasks, representing the predicted severity score. Then the trained model is saved with the name severity_model.h5. The entire process explains the efficiency and accuracy of the insurance claim process.



```python
# Load the VGG16 model pre-trained on ImageNet data
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(img_width, img_height, 3))

# Freeze the layers of the pre-trained VGG16 model
for layer in base_model.layers:
    layer.trainable = False

# Add custom layers on top of VGG16 for regression
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(64, activation='relu')(x)
predictions = Dense(1, activation='linear')(x)  # Use linear activation for regression

# This is the model we will train for regression
model = Model(inputs=base_model.input, outputs=predictions)

# Compile the model with an appropriate loss function for regression
model.compile(optimizer=Adam(), loss='mean_squared_error', metrics=['mae'])
```

Fig. 5. VGG16 model

Then we implement the severity function which categorizes the severities as High, Medium, and Low. We are using the severity score to make the decision. If the severity score is equal to or greater than 0.8 then it is High. If the severity score is between 0.5 and 0.8 it is medium and any severity below 0.5 is considered as Low. This is the crucial step for making the severity assessment. Here the utilization of the categories helps to enhance the model prediction



```
...   1/1 [==============================] - 0s 72ms/step
      Severity Score: 76.00476837158203
      Severity Level: High
      Car Damaged
...
```

Fig. 6. Severity analysis

For example, assuming ’predictions’ represents an array of severity scores that are generated by the model by using our dataset, then we apply the categorize_severity function to each individual severity score in the ’predictions’ array. The resulting 'severity_categories' array stores the corresponding severity categories for each predicted severity score.

VIII. Project Management

A. Links

The following is the github link to our project https://github.com/sriramofficial9/Feature_Engineering_project.git

The following attached is the overleaf link for document in IEEE format https://www.overleaf.com/read/tvwkrjdtgcgy#6b7fd4

The following attached table contains work completion details and contribution of team members based on the assigned responsibilities.

Description	**Responsibility**	**Contribution**
Identifying the severity by using the image processing techniques	Retrieving Severity of the model.	Priyanka Jampani – 40% Pavitra Yarrakula – 60%
Categorizing the severity scores	After getting the severity score, categorize them to High, Medium, Low.	Pavitra Yarrakula – 40% Priyanka Jampani – 60%
Comparing the regression scores with the predefined models	Compare the models from first draft and now and discuss the analysis	Sriram Relangi – 100%
Analysis of the results obtained	Perform the analysis using the results of previous model from first draft.	Sriram Relangi – 100%

TABLE I
Work Completed and Contributions

References

[1] S. C M, S. Kunjumon and N. R, "Car Damage Identification and Categorization Using Various Transfer Learning Models," 2021 5th International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2021, pp. 1097-1101, doi: 10.1109/ICOEI51242.2021.9452846.

[2] D. Mallios, L. Xiaofei, N. McLaughlin, J. M. D. Rincon, C. Galbraith and R. Garland, "Vehicle Damage Severity Estimation for Insurance Operations Using In-The-Wild Mobile Images," in IEEE Access, vol. 11, pp. 78644-78655, 2023, doi: 10.1109/ACCESS.2023.3299223.

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

[4] R. Singh, M. P. Ayyar, T. V. Sri Pavan, S. Gosain and R. R. Shah, "Automating Car Insurance Claims Using Deep Learning Techniques," 2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM), Singapore, 2019, pp. 199-207, doi: 10.1109/BigMM.2019.00-25.