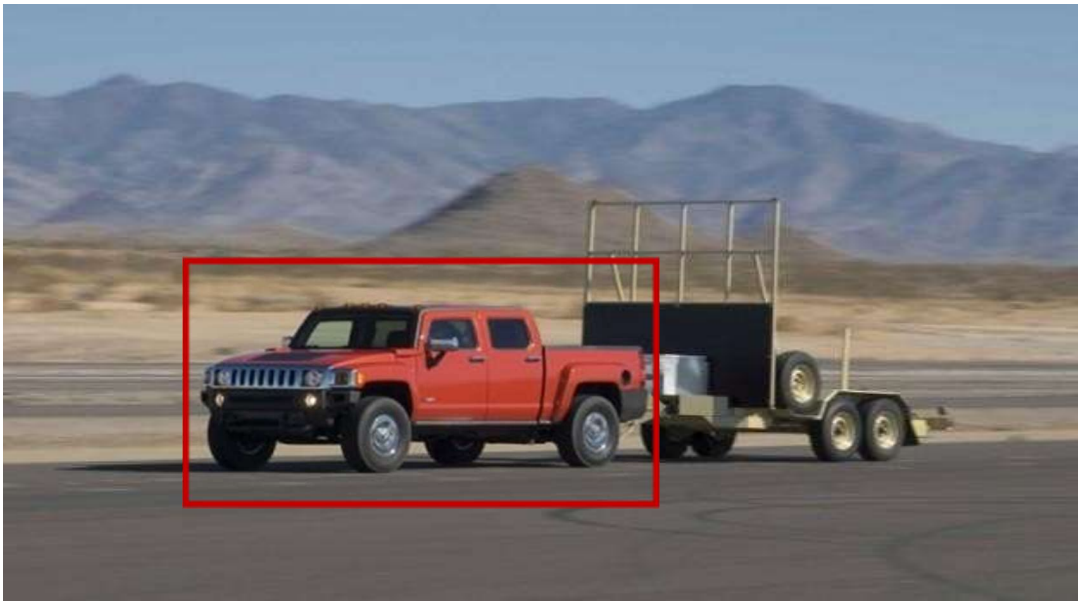
CAPSTONE PROJECT - FINAL REPORT

Submitted by - AIML GROUP1 -Mar, 23

OBJECT DETECTION - CAR

BOUNDING BOX REGRESSION AND MULTICLASS CLASSIFICATION

OF STANFORD CAR IMAGES



Mentor - Mr. Amit Kumar

Members:

Birla Nesan NP,

Garfield Oliveira,

Ashish Murali,

Davinder Kaur,

Arvind Kumar M,

Nikita Satish Salunkhe

Contents

SECTION 1: INTRODUCTION

Summary of the Problem statement

Project Objective

The project aims to showcase a comprehensive understanding and mastery of key concepts and technologies in Deep Learning. Specifically, the objective is to develop a Deep Learning model for car identification using the Stanford Cars dataset, which comprises 16,185 images spanning 196 classes of cars. The dataset is divided into 8,144 training images and 8,041 testing images, with each class evenly split between the training and testing sets.

The problem statement involves multi-class object detection, which consists of two main tasks:
Detection of the location of objects within input images, represented by bounding boxes.
Prediction of the identity of the detected objects, i.e., determining the class to which each object belongs.

By addressing these challenges, the project aims to provide insights into current practices in Deep Learning, as well as the practical considerations and trade-offs inherent in solving real-world problems using this technology.

Dataset Description:

The dataset comprises real images of cars categorized by their make and year, along with annotations for training and testing images. Specifically:

Train Images: Real images of cars, each labeled with the car's make and year. These images serve as the training data for the model.

- Test Images: Like the training set, this contains real images of cars labeled by make and year. These images are reserved for evaluating the model's performance.
- Train Annotation: Annotations specifying bounding box regions for the cars in the training images. Crucial for training object detection models.
- Test Annotation: Like the training set, this provides bounding box annotations for cars in the test images. Used for evaluating the model's ability to detect cars accurately.
- Car Make: A comprehensive list of car makes, including names and years, serving as a reference for understanding the car classes in the dataset.

This dataset facilitates the development of a deep learning-based car identification model, enabling accurate detection of cars and their bounding box regions.

Data and Findings:

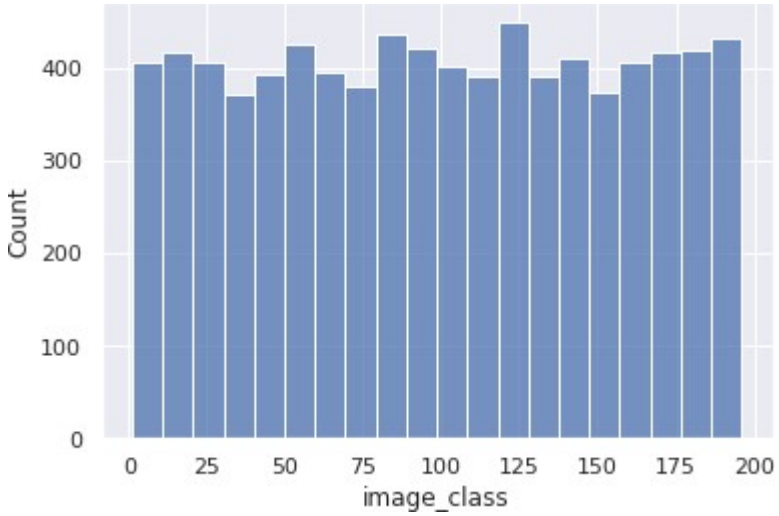
Data:

- Image Folders: The dataset includes two folders, "Train images" and "Test images," containing images categorized by car make, model, and year.
- Car Names and Make CSV: This file provides a list of cars present in the dataset, likely including their make, model, and year.
- Train and Test Annotation CSV Files: These files map images (file names) to output labels, including class names and bounding box coordinates for object detection.

Findings:

- Bounding Box Coordinates Format: Analysis of the bounding box coordinates in the annotation CSV files revealed ambiguity regarding their format (e.g., (xmin, ymin, xmax, ymax) or (x, y, width, height)). Drawing bounding boxes using both formats confirmed that they were in the (xmin, ymin, xmax, ymax) format.
- Consistency Check: The number of unique car makes in the training annotations matches the number of car names in the "Car names and make.csv" file. This verification ensured that none of the folders were inadvertently missed during the training process.
- Balanced Class Distribution: The number of images in each class is evenly balanced, indicating a well-structured dataset with no class imbalance issues. This balance is crucial for training a robust and unbiased model.

These findings provide insights into the dataset's structure and quality, aiding in the effective utilization of the data for training and evaluating car identification models.



Solution Overview:

The problem statement resides within the Automotive domain, focusing on surveillance tasks where computer vision plays a crucial role in automating supervision and event triggering based on images of interest. In response, we adopted a transfer learning approach utilizing pre-built models

available in TensorFlow. Leveraging pre-trained models, we fine-tuned specific layers to achieve high accuracy and precise bounding box predictions.

Approach:

- Transfer Learning: We harnessed the power of transfer learning, enabling us to utilize pre-trained models efficiently. By fine-tuning select layers, we optimized the model for accurate bounding box predictions.

Model Evaluation:

- Through rigorous experimentation, we evaluated multiple pre-trained models, including MobileNet, ResNet50, and EfficientNet.
- Our findings highlighted EfficientNet-B7 as the most effective model, delivering superior results in terms of accuracy and bounding box prediction.

Key Insights:

- EfficientNet-B7 emerged as the optimal choice, showcasing its prowess in handling complex automotive surveillance tasks.
- Transfer learning demonstrated its versatility and adaptability, enabling seamless integration of pre-trained models into our solution.

SECTION 2: PROCESS

Overview of the Process:

In this section, we provide an overview of the process followed in the project, focusing on data preprocessing and preparation for model training. Key steps include loading annotations, resizing images, and scaling bounding box coordinates.

Salient Features of the Data:

The dataset comprises images of cars categorized by make, model, and year, along with corresponding annotations specifying bounding box regions. Each annotation record contains:

- Image Filename
- Starting x-coordinate
- Starting y-coordinate
- Ending x-coordinate
- Ending y-coordinate
- Class label number

Overview of the Placement of the cars in the image:

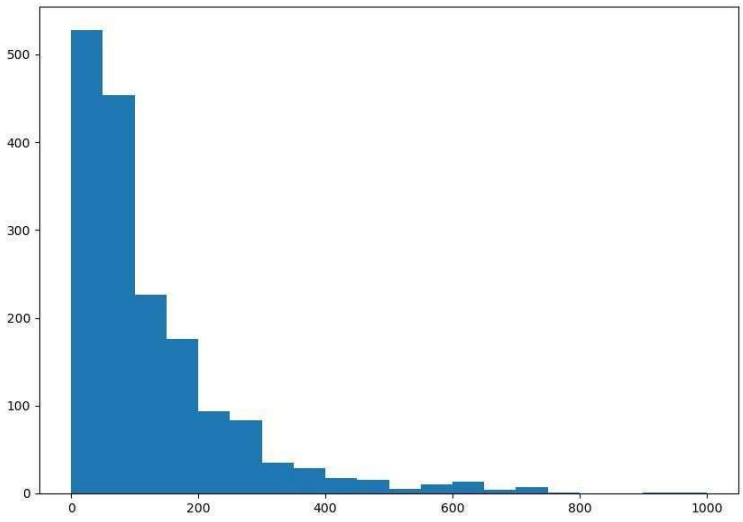


Fig1. Car Placement in the image

Data Pre-processing:

Data pre-processing is a crucial step in preparing the dataset for model training. Below are the steps followed for pre-processing images and annotations:

1. Reading Images from Images Folder:
    - Images are read from the images folder containing subdirectories corresponding to the label names.
    - Each subdirectory contains images related to the respective label.
2. Reading Annotation File:
    - The annotation file is read to extract information about bounding box coordinates and class labels for each image.
3. Extracting Image Height and Width:
    - Image height and width are extracted to facilitate resizing and scaling of images and bounding box coordinates.
4. Merging Image Annotation Dataset:
    - The image annotation dataset is merged with the data obtained from the images folder and annotation file.
    - This combined dataset contains information about images, class labels, and bounding box coordinates.

Creating Dataset from Images Folder:

- The images folder structure is leveraged to create the dataset.
- Car names and class names are loaded into a pandas dictionary object.
- CSV annotation files are processed to extract image information, including labels and bounding box coordinates.
- Images are loaded from disk in TensorFlow format and preprocessed by resizing to

224x224 pixels.

- Label encoding is performed using One-hot encoding techniques with LabelBinarizer.

Input and Output Format:

- Input: 224x224 image matrix
- Output: Class label, 4 bounding box coordinates

The Algorithms:

Convolutional Neural Networks (CNNs) are a class of Deep Neural Networks designed for recognizing and classifying features from images. They are widely used in various applications such as image and video recognition, image classification, medical image analysis, computer vision, and natural language processing.

Main Components of a CNN Architecture:

1. **Convolutional Layers:**
   - Responsible for feature extraction from images.
   - Utilizes convolutional filters to identify various features and patterns within the image.
   - Captures spatial hierarchies of patterns, including edges, textures, shapes, and higher-level representations of objects.
2. **Fully Connected Layers:**
   - Utilizes features extracted by convolutional layers to predict the class of the image.
   - Maps learned features to output classes through weighted connections and activation functions.
   - Performs multi-class classification based on the features extracted in previous stages.

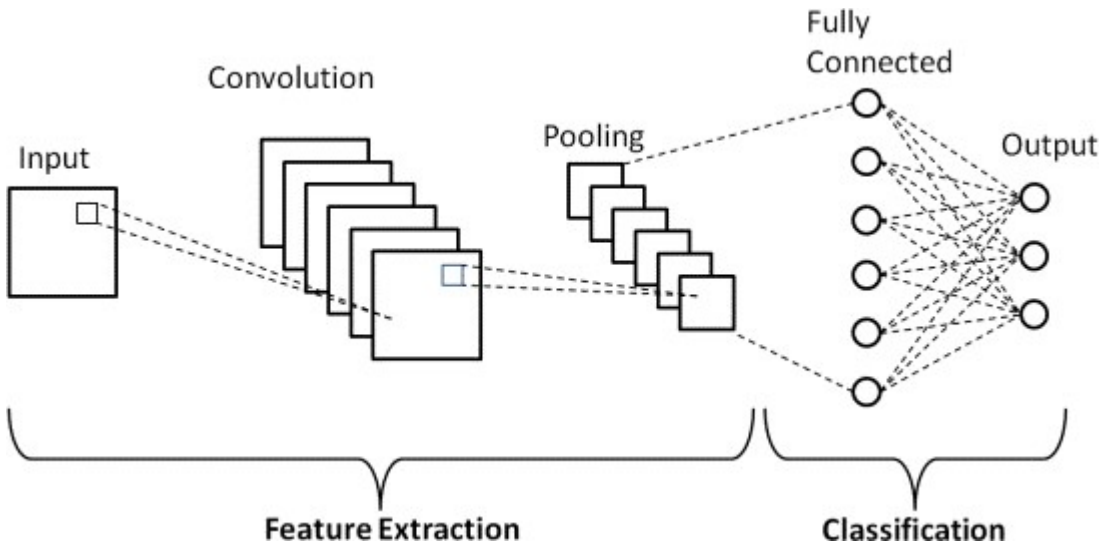


Fig2. CNN architecture overview

Bounding Box Regression:

- Used for object localization in object detection algorithms.
- Aims to predict the location of target objects using rectangular bounding boxes.
- Refines the location of predicted bounding boxes to accurately localize objects within an image.
-

Object Classification and Localization:

- Object classification involves multi-class classification to identify the class of the object depicted in an image.
- Object localization utilizes bounding box regression to predict the precise location of the object within the image.

Experimentation with Pre-trained Models:

- Considered and executed various pre-trained models such as MobileNet, ResNet50, EfficientNet-B5, YOLO, and TFOD.
- EfficientNet-B7 was selected based on its superior performance for object classification and localization tasks.

SECTION 3: SOLUTION

Step 1 as illustrated in Data collection, Data preprocessing and EDA part.

Step 2 involves the iterative process of model building, model evaluation and fine tuning.

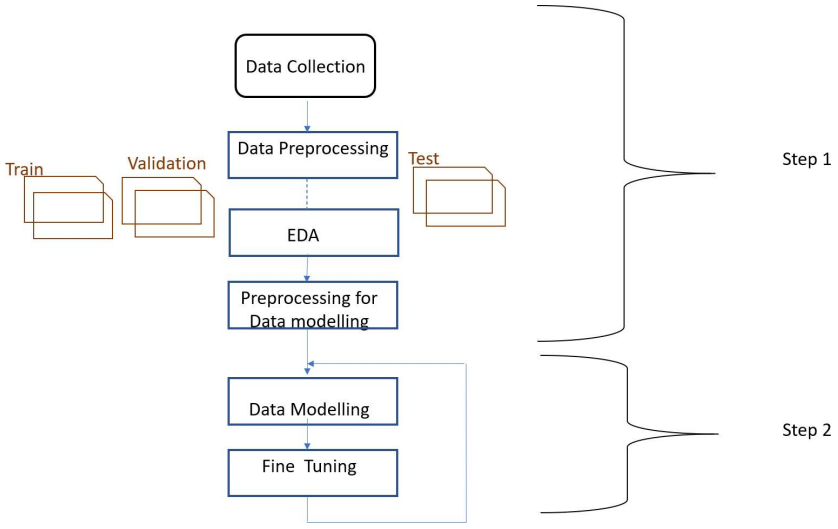


Fig3. Data Modeling Solution Overview

STEP1:

Data Loading:

To load the data and prepare it for model training, the following steps are undertaken:

1. Loading Car Name and Class Name:
    - Car names and class names are loaded into a pandas dictionary object.

2. Looping Over CSV Annotation Files:
    - CSV annotation files containing image information, including filename, coordinates, and class label, are iterated over.

3. Data Processing within the Loop:
    - Each row in the CSV file is unpacked to obtain the filename, coordinates, and class label.
    - For each annotation, new columns for class name and image path name are updated.
    - The image is loaded using the imagePath derived from the configuration, class label, and filename.
    - The spatial dimensions of the image are extracted.
    - Bounding box coordinates are scaled relative to the original image dimensions to the range [0, 1].

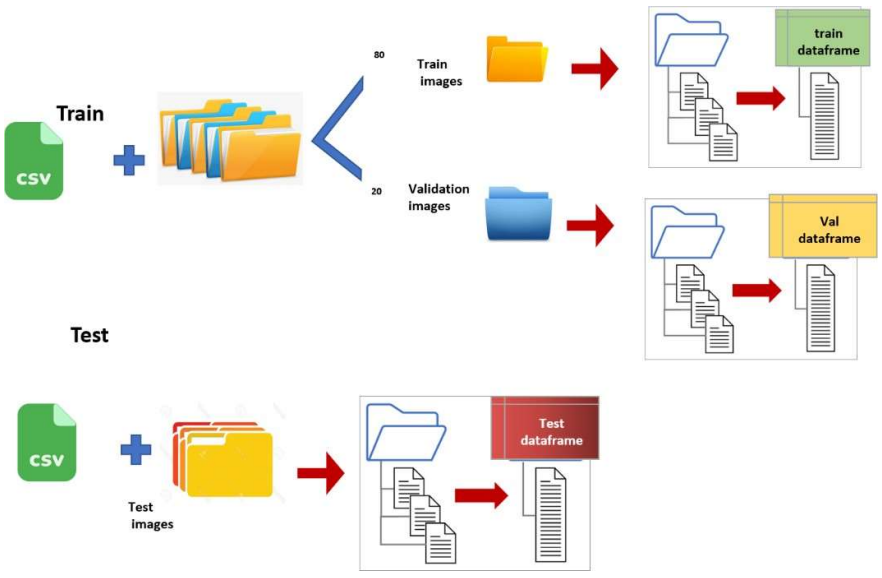


Fig4. Date loading Overview

Data Preprocessing:

Data preprocessing is crucial for preparing raw data to make it suitable for machine learning models. In this project, the following steps are involved in data preprocessing:

1. Extracting Image Information:
    - Image information is extracted and preprocessed to be compatible with the Keras/TensorFlow format.
2. Scaling Images:
    - Images are resized to a suitable dimension of 224x224 to ensure consistency for model input.
3. Output Representation:
    - The output consists of a class label and 4 bounding box coordinates.
    - Class labels are one-hot encoded using LabelBinarizer to represent them numerically for model training.
4. Bounding Box Coordinates Scaling:
    - Bounding box coordinates are scaled relative to the original image dimensions with respect to the 224x224 dimension.
    - This scaling ensures that the bounding box coordinates are proportional to the resized image.

Exploratory data analysis

Class having minimum number of images in train dataset

Below mentioned picture shows 5 classes which are having a smaller number of images in the Train dataset.

[48]:

carName	count
Rolls-Royce Phantom Drophead Coupe Convertible 2012	30
Maybach Landaulet Convertible 2012	29
Chevrolet Express Cargo Van 2007	29
FIAT 500 Abarth 2012	27
Hyundai Accent Sedan 2012	24

Fig5. Table having min number of images in Train dataset

Class having maximum number of images in Train dataset

Below mentioned picture shows 5 classes which are having a greater number of images in the Train dataset.

[44]:

carName	count
GMC Savana Van 2012	68
Chrysler 300 SRT-8 2010	49
Mitsubishi Lancer Sedan 2012	48
Mercedes-Benz 300-Class Convertible 1993	48
Jaguar XK XKR 2012	47

Fig6. Table having min number of images in Test dataset

Class having maximum number of images in test dataset

Below mentioned picture shows 5 classes which are having a greater number of images in the Test dataset.

[47]:

carName	count
GMC Savana Van 2012	68
Mercedes-Benz 300-Class Convertible 1993	48
Chrysler 300 SRT-8 2010	48
Mitsubishi Lancer Sedan 2012	47
Bentley Continental GT Coupe 2007	46

Fig7. Table having max number of images in Train dataset

Class having minimum number of images in Test dataset

Below mentioned picture shows 5 classes which are having a smaller number of images in the Test

dataset.

[48]:

carName	count
Rolls-Royce Phantom Drophead Coupe Convertible 2012	30
Maybach Landaulet Convertible 2012	29
Chevrolet Express Cargo Van 2007	29
FIAT 500 Abarth 2012	27
Hyundai Accent Sedan 2012	24

Fig8. Table having Min number of images in Test dataset

Number of car images based on Year

Below mentioned picture depicts the number of car images based on car model year. There are more images of cars which are of model 2012 and very less images belonging to model 1996.

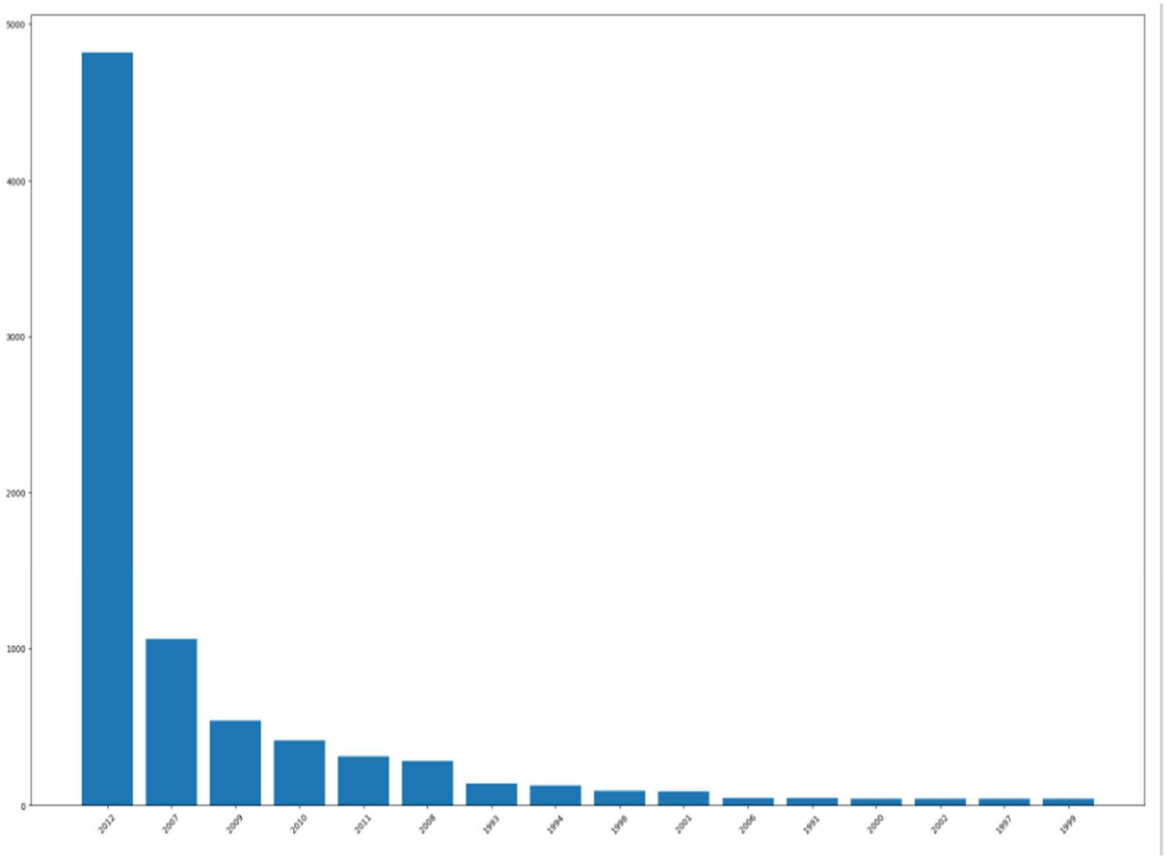
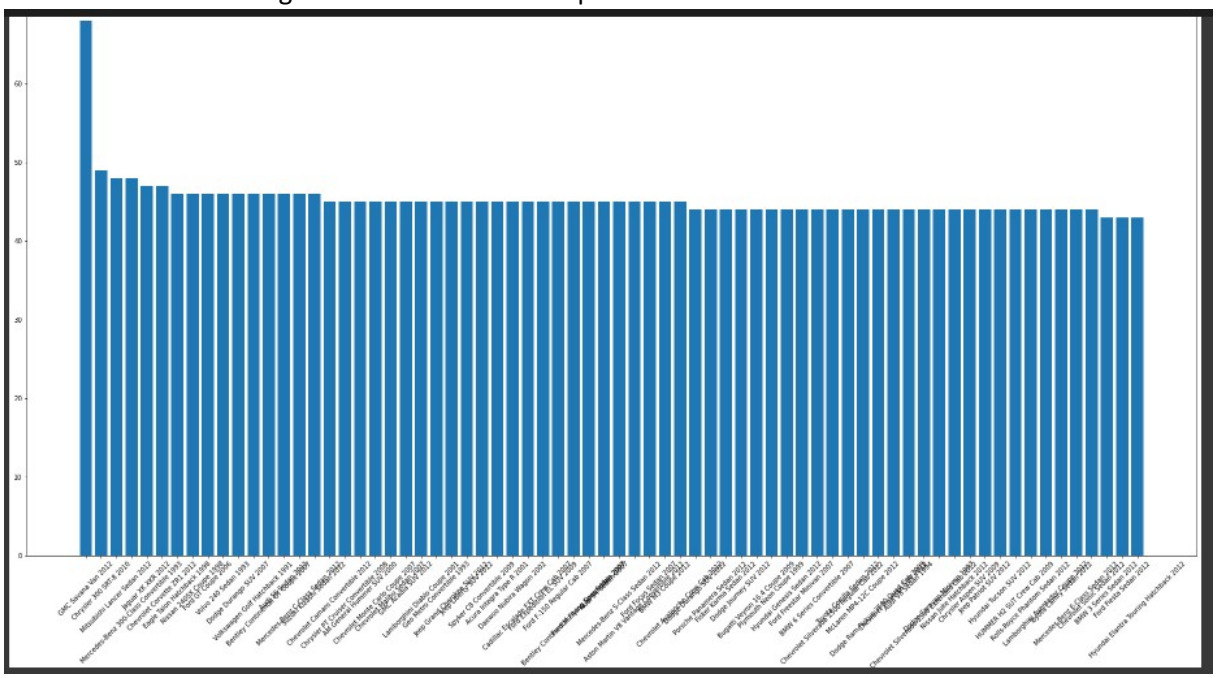
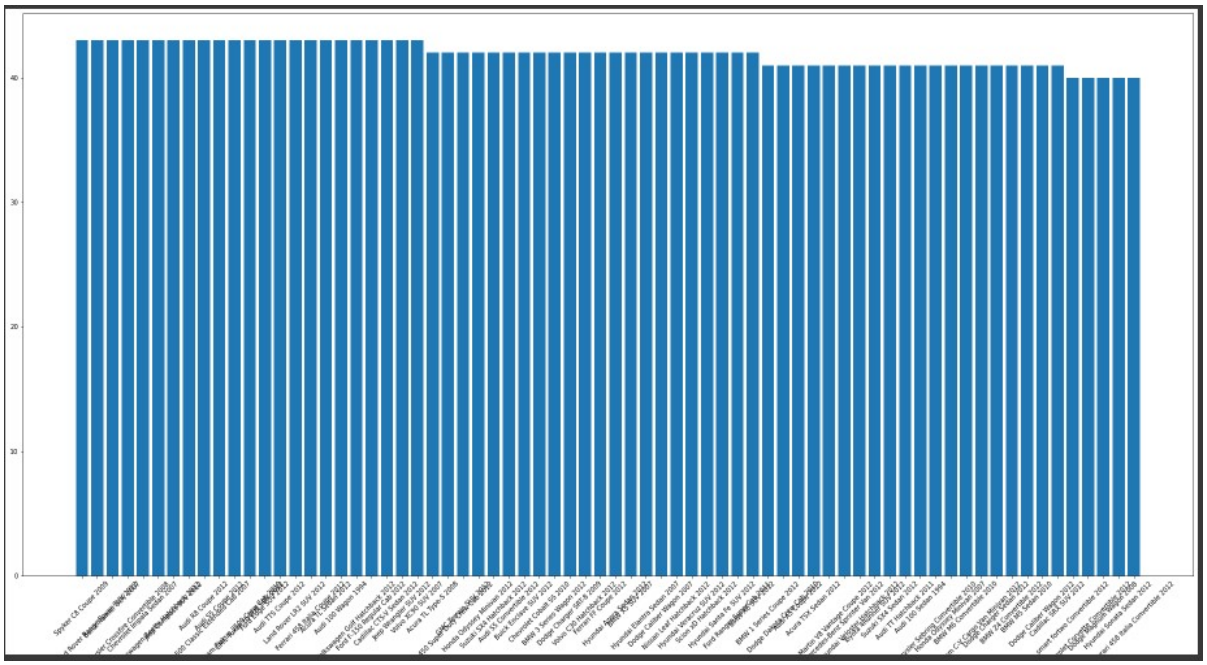


Fig9. Cars Vs year of make

Class distribution

Almost all classes of cars images have an equal number of images which is of count around 40. There is a class named “GMC savana van 2012” which has 60 images and “Hyundai Accent sedan 2012” which has around 25 images. In below mentioned pictures we can view the class distribution.





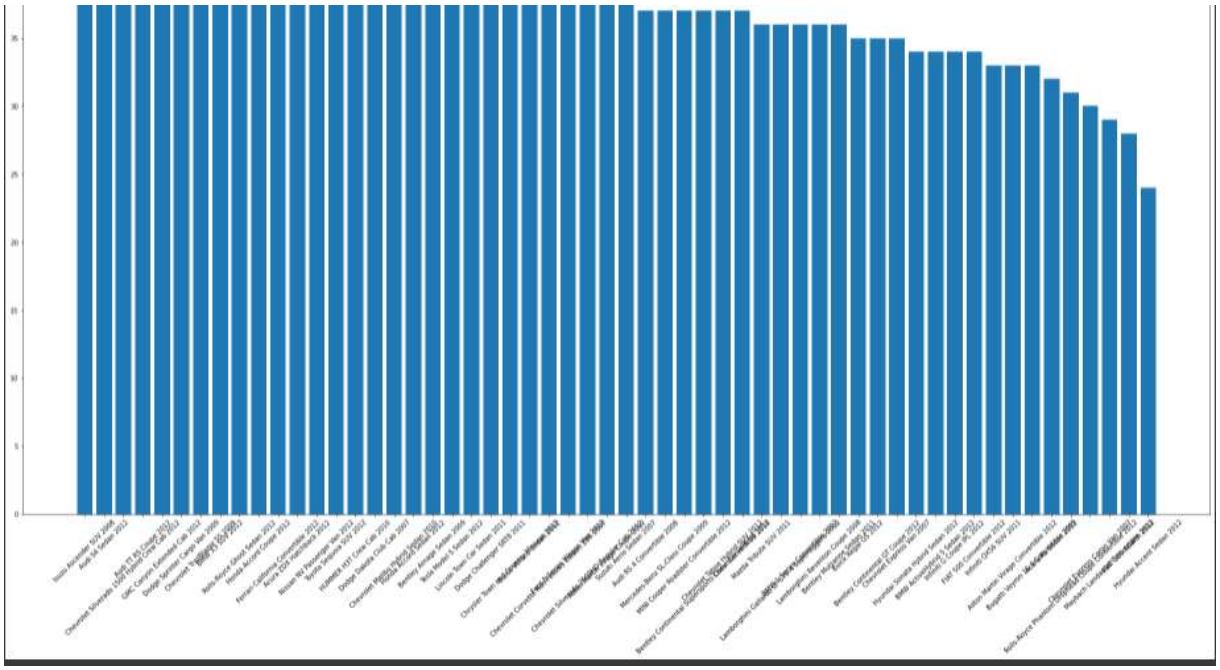


Fig10, 11, 12. Car count histogram (of 196 classes)

Maximum Size of the image available in dataset

Image with name 05945.jpg which belongs to class “Chevrolet Sonic Sedan 2012” is having maximum height and width. The dataset has uneven distribution of the image size and hence preprocessing is required before running the model on data. While preprocessing we have to define a fixed image size and the tensorflow preprocessing function was applied on each images before training the model.

[60]:

	carName	imageName	Height	Width
2942	Chevrolet Sonic Sedan 2012	05945.jpg	3744	5616

Fig13. Maximum Size of the image available in dataset

Minimum Size of the image available in dataset:

Below mentioned pictures show the details which are images which are having very less size.

[65]:

	carName	imageName	Height	Width
2339	Chevrolet Corvette Ron Fellows Edition Z06 2007	00097.jpg	58	78
2373	Chevrolet Corvette Ron Fellows Edition Z06 2007	07469.jpg	58	78

Fig14. Maximum Size of the image available in dataset

STEP2

Design:

Considering the volume of data and available resources, Jupyter Notebook was primarily utilized for data preprocessing and modeling phases.

Given the nature of the problem, it was determined that deep learning techniques would be most effective. Convolutional Neural Networks (CNNs) were identified as a suitable approach due to their high accuracy in tasks such as image classification, localization, and detection. CNNs use a hierarchical model resembling a funnel, culminating in a fully connected layer where all neurons are interconnected for processing. CNNs excel in developing internal representations of images by analyzing subsets of pixels.

In contrast, dense neural networks are not suitable for computer vision tasks in deep learning. Unlike convolutional layers, dense layers require more parameters as they perform a linear operation on each input, resulting in many connections.

Deploy:

For this problem statement, transfer learning was chosen, utilizing a pre-built model in TensorFlow. By training only a few layers of pre-built models, high accuracy and optimal predictions for bounding boxes can be achieved. Transfer learning allows pre-trained models to be employed directly for feature extraction or integrated into new models.

Several pre-trained models were evaluated, including MobileNet, ResNet50, and EfficientNet. Among these, EfficientNet-B7 yielded the best results, demonstrating superior performance for the task at hand.

```
model = tf.keras.applications.efficientnet.EfficientNetB7(include_top=False,
                                        input_shape=(224, 224, 3),
                                        weights='imagenet')
```

Few layers of pre-trained model were unfreezed for more training. The output layer of pretrained model was flattened and tuned for label classification training (for 196 classes) using softmax activation and

bounding box regression training (for 4 outputs) using sigmoid activation dense layer. This non-sequential model output was then trained for categorical crossentropy and userdefined IOU metrics respectively over the Adam optimizer.

Model Pickling

Final model was pickled in h5 format, this file was later used to load the to be used for predict and draw as below

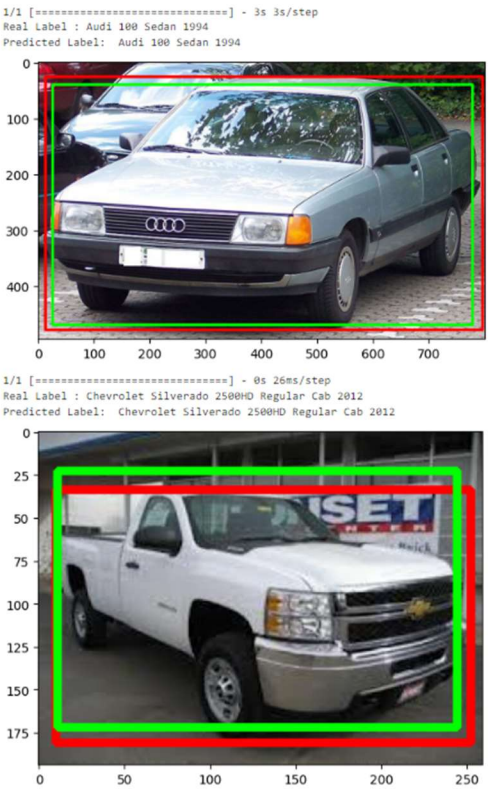


Fig15. Predicted Result

SECTION 4: MODEL EVALUATION

Model evaluation is a critical step in the development process, aimed at identifying the best-performing model for the given task. Various models were experimented with and evaluated before finalizing the best model. The evaluation methods and benchmarks are detailed below:

Classification Parameters:

1. **Categorical Cross Entropy Loss:**
   - Categorical cross entropy is used as the loss function for this multi-class classification task.
   - It quantifies the difference between predicted and actual probability distributions.
   - Mathematically, it is defined as:

$$Loss=-i=1\sum tyi\cdot logy\hat{}i$$

   where:
   - ŷ_i is the predicted probability for class i.
   - y_i is the actual probability for class i.

2. **Classification Report:**
   - A classification report is employed to evaluate the model's performance.
   - It provides precision, recall, and F1-score metrics on a per-class basis.
   - These metrics offer insights into the model's ability to correctly classify each class.
3. **Accuracy:**
   - Accuracy measures the percentage of correct predictions for the test data.
   - It is calculated by dividing the number of correct predictions by the total number of predictions.

Accuracy = (Correct Predictions / Total Predictions)

4. **Precision:**
   - Precision represents the fraction of relevant examples (true positives) among all examples predicted to belong to a certain class.
   - It is calculated as the ratio of true positives to the sum of true positives and false positives.

Precision = (True Positives / (True Positives + False Positives))

5. **Recall:**
   - Recall indicates the fraction of examples predicted to belong to a class relative to all examples that truly belong to the class.
   - It is calculated as the ratio of true positives to the sum of true positives and false negatives.

Recall = (True Positives / (True Positives + False Negatives))

6. **F1 Score:**
   - The F1 score is the weighted harmonic mean of precision and recall, with the best score being 1.0 and the worst being 0.0.
   - It is calculated using the formula:

F1 Score = 2 * (Recall * Precision) / (Recall + Precision)

Object Detection Parameter

1. **Mean Average Precision (mAP):**
    - Mean average precision is a widely-used metric for evaluating the accuracy of object detection models.
    - It measures the average precision of detection across all classes in the dataset.
    - mAP is calculated by computing the average precision (AP) for each class and then taking the mean across all classes.
    - Higher mAP values indicate better performance, with a maximum value of 1.0.
2. **Intersection over Union (IoU):**
    - Intersection over Union is a critical component when calculating mAP.
    - It quantifies the amount of overlap between the predicted bounding box and the ground truth bounding box.
    - IoU is a number ranging from 0 to 1, where:
    - An IoU of 0 indicates no overlap between the bounding boxes.
    - An IoU of 1 indicates complete overlap, meaning the predicted and ground truth bounding boxes perfectly match.
    - IoU is used to determine whether a predicted bounding box is considered a true positive or a false positive during object detection evaluation.

These object detection parameters, including mAP and IoU, provide valuable insights into the accuracy and reliability of the model's object detection capabilities. They are essential for assessing the model's performance in localizing objects within images accurately.

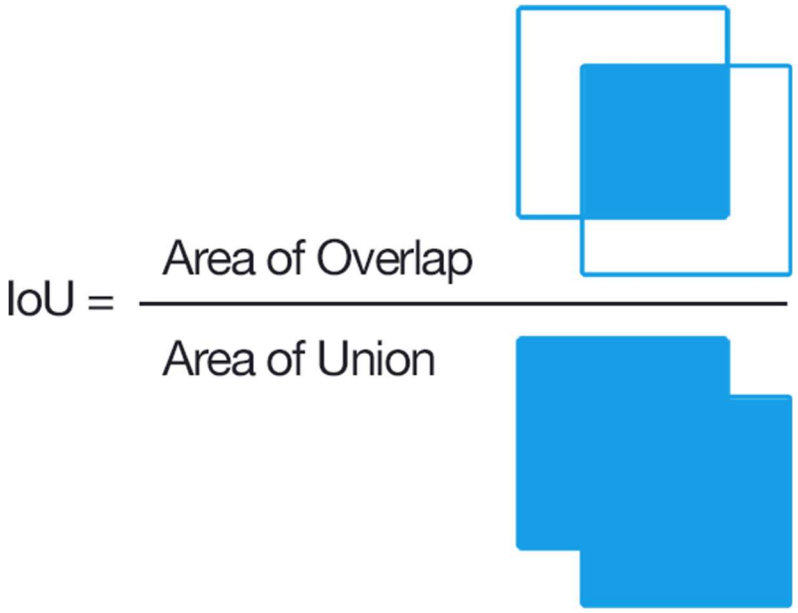


Fig 16. IOU explained

Model Building:

The following are the various models compiled and evaluated on the based on the parameters listed above.

MobileNet

MobileNets represent a lightweight approach to deep neural networks, leveraging depth-wise separable convolutions to achieve efficiency. They introduce two global hyper-parameters that allow for balancing between latency and accuracy, enabling model builders to tailor the model size to their specific application constraints. In this case, MobileNetV2, a specific version of MobileNet, is utilized.

MobileNetV2 shares similarities with the original MobileNet but incorporates inverted residual blocks with bottlenecking features, resulting in significantly fewer parameters compared to its predecessor. MobileNets can accommodate input sizes larger than 32x32, with larger images generally yielding better performance. The network architecture of MobileNetV2 is relatively straightforward, featuring a reduced number of trainable weights. The model is retrained from the input layer, with additional layers such as dense, dropout, and batch normalization layers appended for classification and regression tasks (such as bounding box calculation).

Categorical crossentropy loss is employed for classification, while mean squared error (MSE) is used for regression. The overall loss of the model is computed by summing the regression loss and classification loss. To evaluate the accuracy of bounding boxes on images, an Intersection over Union (IOU) function is defined, serving as an evaluation metric for object detection performance.

The total parameters for the MobileNetV2 model are reported as 4,988,104, with 4,953,352 trainable parameters and 34,752 non-trainable parameters.

Resnet50

ResNet50 is a variant of the ResNet architecture, featuring 48 convolution layers, 1 MaxPool layer, and 1 Average Pool layer, with a total of 3.8 x 10^9 floating point operations. It's a widely used ResNet

model known for its effectiveness in addressing the vanishing gradient problem encountered in deep CNNs. In this instance, the entire ResNet50 model is retrained, boasting approximately 28 million trainable parameters. The final layer of the model consists of a GlobalAveragePooling2D layer, followed by two Dense layers, a dropout layer, and a BatchNormalization layer. For classification tasks, a softmax activation function with 196 classes is utilized, while a sigmoid activation function is employed for regressing the four coordinates of the bounding box.

Similar to the MobileNet approach, categorical crossentropy loss is applied for classification, and mean squared error (MSE) is used for regression. The overall loss of the model is computed by summing the regression loss and classification loss. Furthermore, an Intersection over Union (IOU) function is defined to evaluate the accuracy of bounding boxes on images.

The ResNet50 model comprises a total of 28,937,800 parameters, with 28,883,656 being trainable and 54,144 being non-trainable.

Efficientnet-b5

The EfficientNet-B5 model, pretrained on the ImageNet database, is employed for developing a network for both classification and regression tasks in the given problem statement. EfficientNet models are specifically designed for image classification tasks, and EfficientNet-B5 is one of the variants in the series.

EfficientNet-B5 comprises a total of 576 layers, with the layers starting from number 257 being made trainable for this task. The model contains approximately 28 million trainable parameters. In the final layer of the model, there is a GlobalAveragePooling2D layer, followed by one Dense layer and a BatchNormalization layer. For classification, a softmax activation function with 196 classes is used, while a sigmoid activation function is applied for regressing the four coordinates of the bounding box.
Similar to the approaches with MobileNet and ResNet50, categorical crossentropy loss is utilized for classification, and mean squared error (MSE) is employed for regression. The overall loss of the model is computed by combining the regression loss and classification loss. Additionally, an Intersection over Union (IOU) function is defined to evaluate the accuracy of bounding boxes on images.

The EfficientNet-B5 model consists of a total of 33,127,871 parameters, with 31,496,968 parameters being trainable and 1,630,903 parameters being non-trainable.

Efficientnet-b7

For the given problem statement, an EfficientNet-B7 model with ImageNet weights is utilized for network development, addressing both classification and regression tasks. EfficientNet models, including EfficientNet-B7, are tailored for image classification and come pretrained on the ImageNet image database. EfficientNet-B7 comprises a total of 813 layers, with training initiated from layer number 351. The model features approximately 64 million trainable parameters. In its final layer, the model includes a GlobalAveragePooling2D layer, followed by one Dense layer and a BatchNormalization layer. For classification purposes, a softmax activation function with 196 classes is employed, while a sigmoid activation function is used for regressing the four coordinates of the bounding box.

As with previous models, categorical crossentropy loss is applied for classification, and mean squared error (MSE) is used for regression. The overall loss of the model is computed by combining the regression loss and classification loss. Additionally, an Intersection over Union (IOU) function is defined to assess the accuracy of bounding boxes on images.

The EfficientNet-B7 model encompasses a total of 71,176,287 parameters, with 67,594,112 parameters being trainable and 3,582,175 parameters being non-trainable.

SECTION 5: BENCHMARK

Benchmarking is used to measure performance using a specific indicator resulting in a metric that is then compared to others. In other words, it is the comparison of a given model's inputs and outputs to estimates from alternative internal or external data or models

Test Results Comparison

For the object localization problem, we experimented with MobileNet, ResNet50, EfficientNet-B5, and EfficientNet-B7. Among these models, EfficientNet-B5 yielded the best performance based on observed test accuracy.

Here's a summary of the findings for each model:

1. **MobileNet**: While MobileNet showed good accuracy on the training and validation datasets, it performed poorly on the unseen test dataset. Test results are detailed in the table titled "Test

Result for MobileNet", and the classification report is available in the Tables and Figures section under the title "Classification report for MobileNet".

2. **ResNet50**: Similar to MobileNet, ResNet50 also demonstrated poor performance on the unseen test dataset. Test results are detailed in the table titled "Test Result for ResNet50", and the classification report is available in the Tables and Figures section under the title "Classification report for ResNet50".
3. **EfficientNet-B5**: EfficientNet-B5 performed well on the unseen test dataset, indicating its effectiveness for this problem. Test results are detailed in the table titled "Test Result for EfficientNet-B5", and the classification report is available in the Tables and Figures section under the title "Classification report for EfficientNet-B5".
4. **EfficientNet-B7**: Similar to EfficientNet-B5, EfficientNet-B7 also performed well on the unseen test dataset. Test results are detailed in the table titled "Test Result for EfficientNet-B7", and the classification report is available in the Tables and Figures section under the title "Classification report for EfficientNet-B7".

EfficientNet-B5 and EfficientNet-B7 appear to be the most promising models based on their performance on the unseen test dataset.

Model	loss	Class_op_loss	reg_op_loss	class_op_accuracy	reg_op_IoU
MobileNet	1.3123	1.3064	0.0059	0.7045	0.7503
ResNet50	4.6202	4.6122	0.0080	0.1060	0.7335
EfficientNet-B5	1.1020	1.0989	0.0032	0.7860	0.8151
EfficientNet-B7	1.0924	1.0892	0.0032	0.7782	0.8143

Fig17. Result comparison

EfficientNet-B7 model Numbers

The finalized model EfficientNet-B7 is trained in 2 phases, 1st phase with 10 epochs and batch size as 64 and in 2nd phase 15 more epochs with batch size as 16.

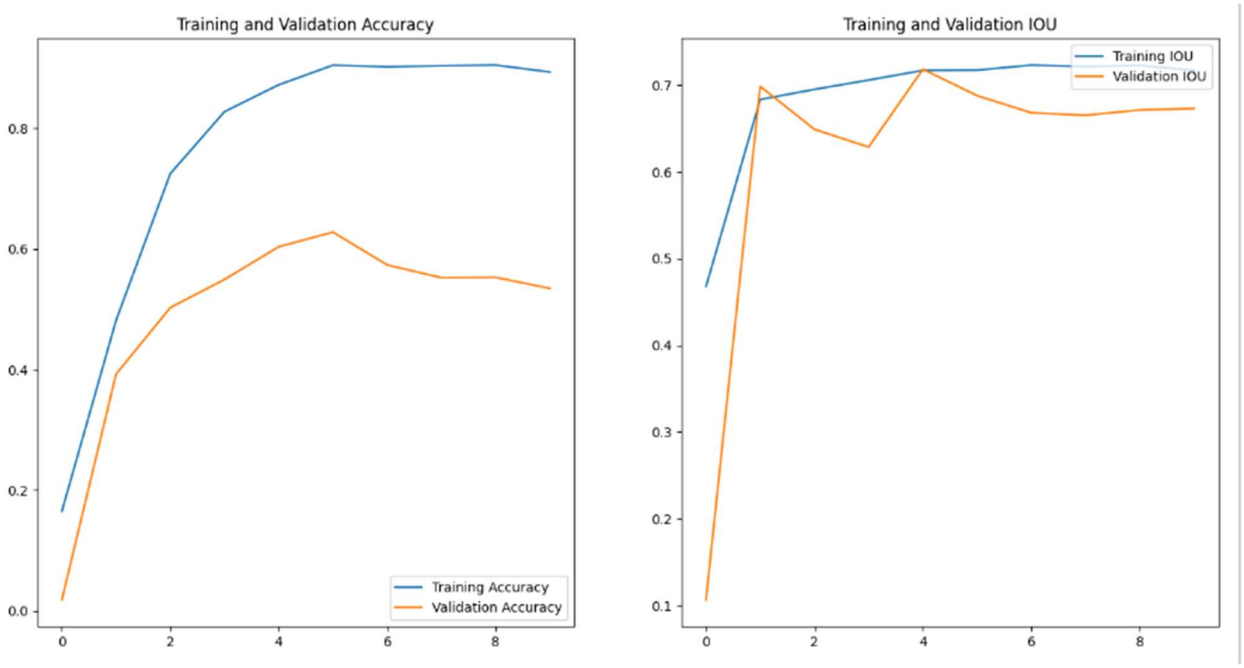


Fig18. First Phase results

Above figure shows the training vs validation accuracy and Training vs validation IOU in the first phase.

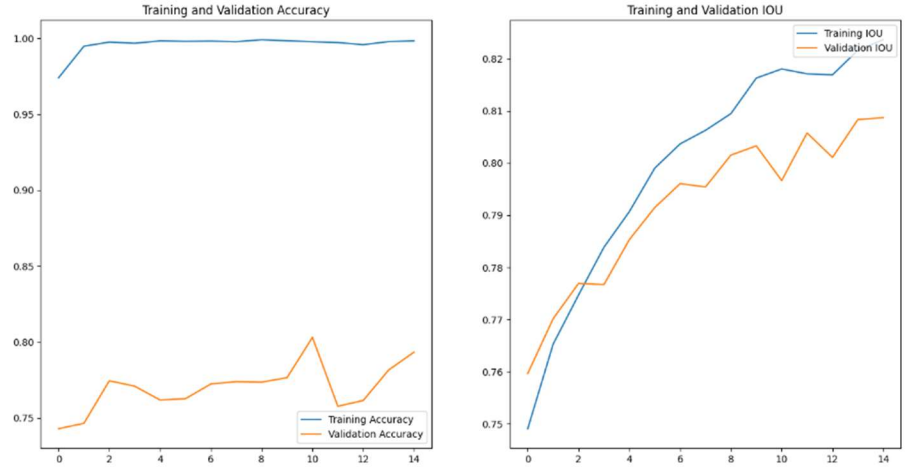


Fig19. Second Phase results

Above figure shows the training vs validation accuracy and Training vs validation IOU in the second phase.

The training vs validation accuracy in the first phase is having a gap between them but with the second phase the validation accuracy has improved. Although the observed gap is more, for a check on a few thousand sample data, the classification seems to be acceptable. The IOU of the model has performed pretty well and compared to resnet50 and MobileNet, the

bounding boxes on the sample data has shown improvement. Below mentioned are a few samples of the dataset.

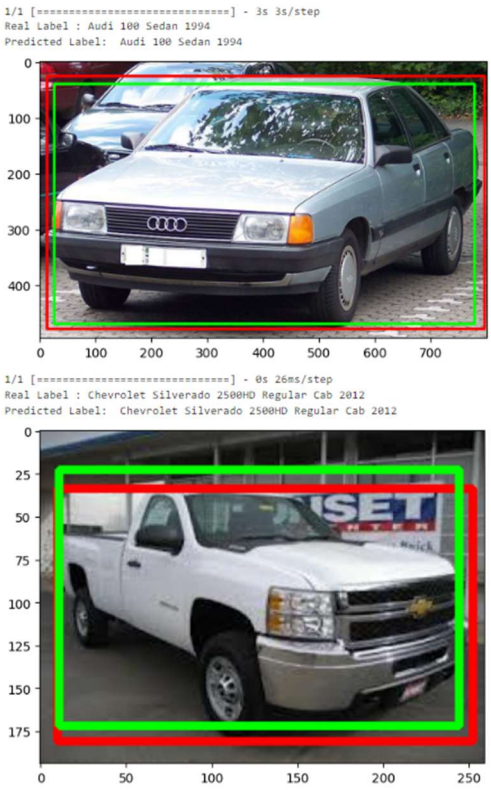


Fig20. Predicted image results

Based on the performance, the EfficientNet-B7 model is finally chosen, which is having the best performance among all the ones we have seen so far.

Classification report

Below mentioned table classification report for EfficientNet-B7 model. Overall Accuracy is 79%, Precision is 78%, recall is 78% and f1-Score is 78%. As observed in the classification report, for a few classes the number is low and the same is observed in test samples where the model is showing misclassification. The number of samples used for producing reports is 8041. In EfficientNet, the authorspropose a new Scaling method called Compound Scaling.

Classification report for EfficientNet-B7

	precision	recall	f1-score	support
Acura Integra Type R 2001	0.85	0.89	0.87	44
Acura RL Sedan 2012	0.90	0.84	0.87	44
Acura TL Sedan 2012	0.49	0.66	0.56	32
Acura TL Type-S 2008	0.73	0.81	0.77	43
Acura TSX Sedan 2012	0.79	0.81	0.80	42
Acura ZDX Hatchback 2012	0.77	0.60	0.68	40
AM General Hummer SUV 2000	0.60	0.77	0.67	39
Aston Martin V8 Vantage Convertible 2012	0.57	0.58	0.57	45
Aston Martin V8 Vantage Coupe 2012	0.69	0.66	0.68	41
Aston Martin Virage Convertible 2012	0.86	0.55	0.67	33
Aston Martin Virage Coupe 2012	0.70	0.84	0.76	38
Audi 100 Sedan 1994	0.59	0.65	0.62	40
Audi 100 Wagon 1994	0.73	0.76	0.74	42
Audi A5 Coupe 2012	0.53	0.78	0.63	41
Audi R8 Coupe 2012	0.85	0.79	0.82	43
Audi RS 4 Convertible 2008	0.86	0.69	0.77	36
Audi S4 Sedan 2007	0.78	0.71	0.74	45
Audi S4 Sedan 2012	0.58	0.38	0.46	39
Audi S5 Convertible 2012	0.67	0.81	0.73	42
Audi S5 Coupe 2012	0.45	0.36	0.40	42
Audi S6 Sedan 2011	0.78	0.78	0.78	46
Audi TT Hatchback 2011	0.41	0.40	0.41	40
Audi TT RS Coupe 2012	0.61	0.59	0.60	39
Audi TTS Coupe 2012	0.46	0.45	0.46	42
Audi V8 Sedan 1994	0.69	0.56	0.62	43
Bentley Arnage Sedan 2009	0.51	0.57	0.54	35
Bentley Continental Flying Spur Sedan 2007	0.69	0.71	0.70	41
Bentley Continental GT Coupe 2007	0.67	0.67	0.67	42
Bentley Continental GT Coupe 2012	0.73	0.80	0.77	41
Bentley Continental Supersports Conv. Convertible 2012	0.74	0.59	0.66	44
Bentley Mulsanne Sedan 2011	0.74	0.68	0.71	34
BMW 1 Series Convertible 2012	0.65	0.82	0.73	44
BMW 1 Series Coupe 2012	0.65	0.63	0.64	41
BMW 3 Series Sedan 2012	0.59	0.63	0.61	41
BMW 3 Series Wagon 2012	0.91	0.79	0.85	38
BMW 6 Series Convertible 2007	0.74	0.83	0.78	41
BMW ActiveHybrid 5 Sedan 2012	0.91	0.74	0.82	42
BMW M3 Coupe 2012	0.88	0.72	0.79	40
BMW M5 Sedan 2010	0.76	0.95	0.84	39
BMW M6 Convertible 2010	0.74	0.73	0.74	44
BMW X3 SUV 2012	0.62	0.72	0.67	46

BMW X5 SUV 2007	0.74	0.59	0.66	34
BMW X6 SUV 2012	0.77	0.67	0.72	36
BMW Z4 Convertible 2012	0.85	0.80	0.82	35
Bugatti Veyron 16.4 Convertible 2009	0.77	0.53	0.63	32
Bugatti Veyron 16.4 Coupe 2009	0.57	0.86	0.69	43
Buick Enclave SUV 2012	0.74	0.83	0.79	42
Buick Rainier SUV 2007	0.88	0.83	0.85	42
Buick Regal GS 2012	0.82	0.94	0.88	35
Buick Verano Sedan 2012	0.86	0.86	0.86	37
Cadillac CTS-V Sedan 2012	0.98	0.98	0.98	43
Cadillac Escalade EXT Crew Cab 2007	0.85	0.89	0.87	44
Cadillac SRX SUV 2012	0.90	0.88	0.89	41
Chevrolet Avalanche Crew Cab 2012	0.70	0.87	0.77	45
Chevrolet Camaro Convertible 2012	0.77	0.75	0.76	44
Chevrolet Cobalt SS 2010	0.67	0.78	0.72	41
Chevrolet Corvette Convertible 2012	0.92	0.56	0.70	39
Chevrolet Corvette Ron Fellows Edition Z06 2007	0.93	0.68	0.78	37
Chevrolet Corvette ZR1 2012	0.70	0.72	0.71	46
Chevrolet Express Cargo Van 2007	0.33	0.14	0.20	29
Chevrolet Express Van 2007	0.36	0.74	0.49	35
Chevrolet HHR SS 2010	0.88	0.97	0.92	36
Chevrolet Impala Sedan 2007	0.76	0.79	0.77	43
Chevrolet Malibu Hybrid Sedan 2010	0.70	0.74	0.72	38
Chevrolet Malibu Sedan 2007	0.78	0.70	0.74	44
Chevrolet Monte Carlo Coupe 2007	0.86	0.69	0.77	45
Chevrolet Silverado 1500 Classic Extended Cab 2007	0.63	0.88	0.73	42
Chevrolet Silverado 1500 Extended Cab 2012	0.57	0.67	0.62	43
Chevrolet Silverado 1500 Hybrid Crew Cab 2012	0.59	0.55	0.57	40
Chevrolet Silverado 1500 Regular Cab 2012	0.58	0.64	0.61	44
Chevrolet Silverado 2500HD Regular Cab 2012	0.79	0.61	0.69	38
Chevrolet Sonic Sedan 2012	0.89	0.77	0.83	44
Chevrolet Tahoe Hybrid SUV 2012	0.71	0.73	0.72	37
Chevrolet TrailBlazer SS 2009	0.84	0.90	0.87	40
Chevrolet Traverse SUV 2012	0.86	0.68	0.76	44
Chrysler 300 SRT-8 2010	0.82	0.58	0.68	48
Chrysler Aspen SUV 2009	0.87	0.91	0.89	43
Chrysler Crossfire Convertible 2008	0.95	0.91	0.93	43
Chrysler PT Cruiser Convertible 2008	0.95	0.89	0.92	45
Chrysler Sebring Convertible 2010	0.76	0.88	0.81	40
Chrysler Town and Country Minivan 2012	1.00	0.86	0.93	37
Daewoo Nubira Wagon 2002	0.88	0.84	0.86	45
Dodge Caliber Wagon 2007	0.68	0.81	0.74	42
Dodge Caliber Wagon 2012	0.68	0.68	0.68	40
Dodge Caravan Minivan 1997	0.89	0.93	0.91	43

Dodge Charger Sedan 2012	0.76	0.76	0.76	42
Dodge Charger SRT-8 2009	0.81	0.63	0.71	41
Dodge Dakota Club Cab 2007	0.89	0.84	0.86	38
Dodge Dakota Crew Cab 2010	0.87	0.80	0.84	41
Dodge Durango SUV 2007	0.82	0.73	0.78	45
Dodge Durango SUV 2012	0.95	0.81	0.88	43
Dodge Journey SUV 2012	0.90	0.84	0.87	44
Dodge Magnum Wagon 2008	0.93	0.93	0.93	40
Dodge Ram Pickup 3500 Crew Cab 2010	0.88	0.83	0.85	42
Dodge Ram Pickup 3500 Quad Cab 2009	0.73	0.68	0.71	44
Dodge Sprinter Cargo Van 2009	0.82	0.69	0.75	39
Eagle Talon Hatchback 1998	0.84	0.70	0.76	46
Ferrari 458 Italia Convertible 2012	0.86	0.93	0.89	27
Ferrari 458 Italia Coupe 2012	0.75	0.91	0.82	33
Ferrari California Convertible 2012	0.63	0.85	0.73	39
Ferrari FF Coupe 2012	0.65	0.71	0.68	42
FIAT 500 Abarth 2012	0.77	0.85	0.80	39
FIAT 500 Convertible 2012	0.94	0.79	0.86	42
Fisker Karma Sedan 2012	0.97	0.74	0.84	43
Ford E-Series Wagon Van 2012	0.92	0.97	0.95	37
Ford Edge SUV 2012	0.88	0.88	0.88	43
Ford Expedition EL SUV 2009	0.93	0.91	0.92	44
Ford F-150 Regular Cab 2007	0.89	0.87	0.88	45
Ford F-150 Regular Cab 2012	0.97	0.88	0.93	42
Ford F-450 Super Duty Crew Cab 2012	0.88	0.93	0.90	41
Ford Fiesta Sedan 2012	1.00	0.74	0.85	42
Ford Focus Sedan 2007	0.78	0.80	0.79	45
Ford Freestar Minivan 2007	0.88	0.95	0.91	44
Ford GT Coupe 2006	0.83	0.76	0.79	45
Ford Mustang Convertible 2007	0.79	0.75	0.77	44
Ford Ranger SuperCab 2011	0.85	0.79	0.81	42
Geo Metro Convertible 1993	0.84	0.86	0.85	44
GMC Acadia SUV 2012	0.74	0.70	0.72	40
GMC Canyon Extended Cab 2012	0.85	0.68	0.75	68
GMC Savana Van 2012	0.86	0.88	0.87	41
GMC Terrain SUV 2012	0.74	0.60	0.66	42
GMC Yukon Hybrid SUV 2012	0.93	0.86	0.89	44
Honda Accord Coupe 2012	0.76	0.74	0.75	43
Honda Accord Sedan 2012	0.76	0.82	0.79	39
Honda Odyssey Minivan 2007	0.72	0.79	0.76	39
Honda Odyssey Minivan 2012	0.60	0.76	0.67	38
HUMMER H2 SUT Crew Cab 2009	0.84	0.88	0.86	41
HUMMER H3T Crew Cab 2010	0.84	0.74	0.78	42
Hyundai Accent Sedan 2012	0.78	0.75	0.77	24
Hyundai Azera Sedan 2012	0.83	0.71	0.77	42

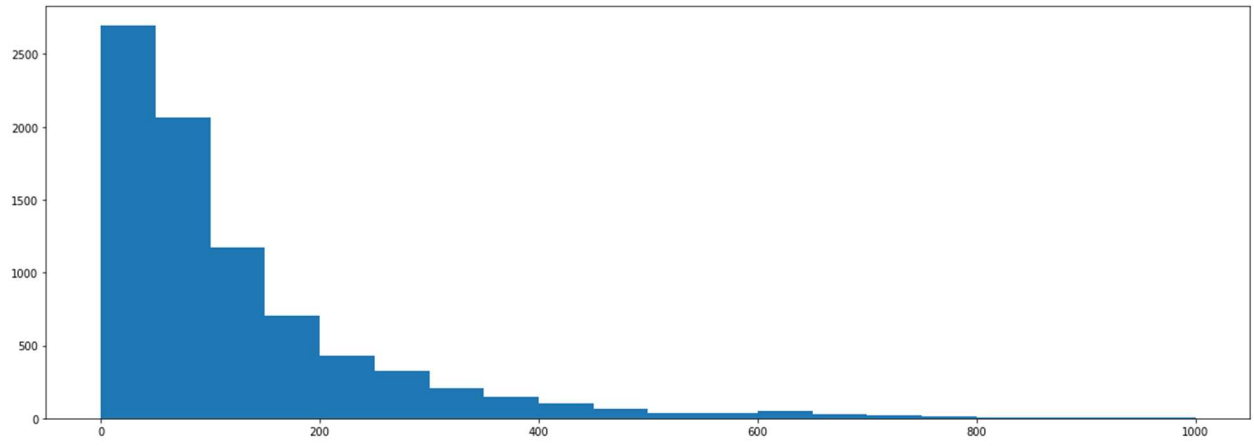
Hyundai Elantra Sedan 2007	0.82	0.79	0.80	42
Hyundai Elantra Touring Hatchback 2012	0.86	0.88	0.87	42
Hyundai Genesis Sedan 2012	0.88	0.88	0.88	43
Hyundai Santa Fe SUV 2012	0.92	0.81	0.86	42
Hyundai Sonata Hybrid Sedan 2012	0.83	0.88	0.85	33
Hyundai Sonata Sedan 2012	0.61	0.90	0.73	39
Hyundai Tucson SUV 2012	0.82	0.84	0.83	43
Hyundai Veloster Hatchback 2012	0.72	0.88	0.79	41
Hyundai Veracruz SUV 2012	0.84	0.62	0.71	42
Infiniti G Coupe IPL 2012	0.74	0.76	0.75	34
Infiniti QX56 SUV 2011	0.83	0.91	0.87	32
Isuzu Ascender SUV 2008	1.00	0.95	0.97	40
Jaguar XK XKR 2012	0.71	0.74	0.72	46
Jeep Compass SUV 2012	0.90	0.86	0.88	42
Jeep Grand Cherokee SUV 2012	0.85	0.76	0.80	45
Jeep Liberty SUV 2012	0.87	0.77	0.82	44
Jeep Patriot SUV 2012	0.86	0.86	0.86	44
Jeep Wrangler SUV 2012	0.83	0.93	0.88	43
Lamborghini Aventador Coupe 2012	0.85	0.79	0.82	43
Lamborghini Diablo Coupe 2001	0.89	0.89	0.89	44
Lamborghini Gallardo LP 570-4 Superleggera 2012	0.80	0.80	0.80	35
Lamborghini Reventon Coupe 2008	0.74	0.97	0.84	36
Land Rover LR2 SUV 2012	0.87	0.79	0.82	42
Land Rover Range Rover SUV 2012	0.88	0.86	0.87	42
Lincoln Town Car Sedan 2011	0.92	0.85	0.88	39
Maybach Landaulet Convertible 2012	0.88	0.97	0.92	36
Mazda Tribute SUV 2011	0.67	0.90	0.76	29
McLaren MP4-12C Coupe 2012	0.86	0.89	0.88	36
Mercedes-Benz 300-Class Convertible 1993	0.83	0.91	0.87	44
Mercedes-Benz C-Class Sedan 2012	0.88	0.94	0.91	48
Mercedes-Benz E-Class Sedan 2012	0.84	0.84	0.84	45
Mercedes-Benz S-Class Sedan 2012	0.68	0.44	0.54	43
Mercedes-Benz SL-Class Coupe 2009	0.66	0.93	0.77	44
Mercedes-Benz Sprinter Van 2012	0.84	0.89	0.86	36
MINI Cooper Roadster Convertible 2012	0.72	0.83	0.77	41
Mitsubishi Lancer Sedan 2012	0.81	0.74	0.78	47
Nissan 240SX Coupe 1998	0.75	0.87	0.81	46
Nissan Juke Hatchback 2012	0.91	0.70	0.79	44
Nissan Leaf Hatchback 2012	0.87	0.98	0.92	42
Nissan NV Passenger Van 2012	0.83	0.92	0.88	38
Plymouth Neon Coupe 1999	0.83	0.91	0.87	44
Porsche Panamera Sedan 2012	0.71	0.51	0.59	43
Ram C-V Cargo Van Minivan 2012	0.85	0.68	0.76	41
Rolls-Royce Ghost Sedan 2012	0.72	0.76	0.74	38

Rolls-Royce Phantom Drophead Coupe Convertible 2012	0.76	0.73	0.75	30
Rolls-Royce Phantom Sedan 2012	0.75	0.61	0.68	44
Scion xD Hatchback 2012	0.90	0.88	0.89	41
smart fortwo Convertible 2012	0.72	0.84	0.78	45
Spyker C8 Convertible 2009	0.73	0.52	0.61	42
Spyker C8 Coupe 2009	0.73	0.71	0.72	38
Suzuki Aerio Sedan 2007	0.80	0.80	0.80	46
Suzuki Kizashi Sedan 2012	0.82	0.74	0.78	42
Suzuki SX4 Hatchback 2012	0.65	0.60	0.62	40
Suzuki SX4 Sedan 2012	0.72	0.89	0.80	38
Tesla Model S Sedan 2012	0.97	0.88	0.92	40
Toyota 4Runner SUV 2012	0.92	0.77	0.84	43
Toyota Camry Sedan 2012	0.79	0.77	0.78	43
Toyota Corolla Sedan 2012	0.86	0.95	0.90	38
Toyota Sequoia SUV 2012	0.79	1.00	0.88	42
Volkswagen Beetle Hatchback 2012	0.78	0.83	0.80	46
Volkswagen Golf Hatchback 1991	0.95	0.88	0.92	43
Volkswagen Golf Hatchback 2012	0.93	0.93	0.93	45
Volvo 240 Sedan 1993	0.78	0.88	0.83	41
Volvo C30 Hatchback 2012	0.65	0.93	0.76	43
Volvo XC90 SUV 2007	0.97	0.93	0.95	40
accuracy			0.78	8041
macro avg	0.78	0.78	0.78	8041
weighted avg	0.79	0.78	0.78	8041

Fig26. Classification Report

SECTION 6: VISUALISATIONS

During exploratory data analysis several insights were made on the data provided. One of the useful insights was to check if all instances of images share the same location and orientation within the dataset. As it is observed from the graph below, the location of the car within the image varies and there are images where the appearance of the car is too small compared to the image itself.

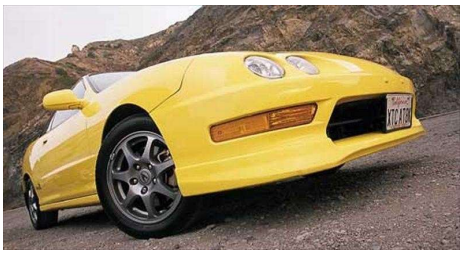


In the above graph, the y-axis shows the number of images (count) and the x-axis shows the orientation of the car in the image.

The Model performance is good if the provided image is having the whole car in the image, some

examples are as mentioned below. But if the image is having only the back side of the car or if the image is not having proper brightness or if the image is blurred then it is observed that the performance of the model is poor.
For below mentioned sample images the model performance is good.


For below mentioned sample images the model performance is poor.

SECTION 7: IMPLICATIONS

The 2018 Used Car Market Report & Outlook published by Cox Automotive reveals that 40 million used vehicles were sold in the US last year, constituting approximately 70% of total vehicle sales. A significant portion of these transactions already leverage online resources across various stages of the purchasing journey, including searching, pre-qualifying, applying, and ultimately buying. Noteworthy websites frequented by car buyers include AutoTrader.com, Kelly Blue Book, Cars.com, and Carvana.com.

While the Cox Automotive report suggests that many market leaders and Silicon Valley startups speculate about a complete shift of car sales to online retailing, such a scenario may be viewed as extreme. Nevertheless, these industry players are keen on enhancing the user experience of online car purchases and improving recommender systems for car searches. Similarly, peer-to-peer sales platforms such as Craigslist, Shift, and eBay Motors express interest in enhancing fraud detection and monitoring user postings.

The implementation of a car image classification system can address several key business cases:

1. **Ground-truthing of posted used car images on peer-to-peer sales platforms:** This involves verifying whether the images posted correspond accurately to the specified cars and determining if multiple exterior images represent the same vehicle.

2. **Organizing web-page displays based on user-uploaded images:** The system can facilitate the arrangement of web-page content based on images uploaded by users during their search for cars.

3. **Recommending alternative cars available in the inventory with similar looks and price:** Utilizing image classification, the system can recommend alternative vehicles from the inventory that share similar visual attributes and price points.
4. **Contributing to fine-grained feature identification for 3D object detection in self-driving cars:** The classification system can aid in identifying and understanding the nuanced features essential for 3D object detection, thereby contributing to advancements in self-driving car technology.

SECTION 8: LIMITATIONS

Limitations

Convolutional neural networks (CNNs) and recurrent neural networks (RNNs) operate by learning from training data through backpropagation algorithms. A critical aspect of their effectiveness lies in adapting the AI approach to the specific problem at hand and the availability of data. Given that these systems are "trained" rather than explicitly programmed, they often require substantial amounts of labeled data to perform complex tasks accurately.

However, obtaining large datasets can pose significant challenges. In certain domains, such datasets may simply be unavailable. Even when available, the process of labeling the data can be resource-intensive, demanding significant human effort.

Moreover, understanding how a mathematical model trained by deep learning arrives at a particular prediction, recommendation, or decision can be challenging. The inherent opacity of these models, often referred to as "black boxes," can limit their utility, especially in contexts where the outcomes have significant societal implications and individual well-being is at stake.

In such cases, users may require insights into the reasoning behind the model's operations. They may need to understand why an algorithm made specific recommendations, especially in scenarios with legal or regulatory repercussions, such as making factual findings or determining lending decisions. Understanding why certain factors were deemed critical while others were not can be essential for establishing trust and ensuring accountability in AI systems.

Scope for Enhancement

1. **Adjustment of Train-Test Split**: The original dataset's 50-50% train-test split can be revised to a more common 80-20% split for further analysis. This adjustment allows for a larger portion of data to be utilized for training while still retaining a sufficient test set for evaluation.
2. **Addressing Class Imbalance**: The dataset exhibits class imbalance, with certain classes being

underrepresented compared to others. Techniques such as random oversampling and Synthetic Minority Oversampling TEchnique (SMOTE) can be employed to balance class distribution. By generating synthetic samples for minority classes, these techniques can improve the model's ability to learn from underrepresented classes.

3. **Considerations for Image Segmentation Metrics**:
    - **Awareness of Mathematical Properties**: Understanding the fundamental mathematical properties of segmentation metrics like Dice Similarity Coefficient (DSC), Hausdorff Distance (HD), and Intersection over Union (IoU) is crucial for determining their applicability in specific scenarios. Addressing issues such as segmentation of small structures, annotation errors, shape awareness, and tendencies towards oversegmentation or undersegmentation is essential.
    - **Suitability for Task**: Ensuring that the chosen segmentation metrics align with the underlying image processing task is important. Metrics should be selected based on their suitability for tasks such as detection, localization, or segmentation.
    - **Metric Aggregation**: Aggregating metric values from individual images into an accumulated score requires careful consideration, especially when dealing with missing values or metrics with fixed boundaries. Strategies for handling missing values and aggregating metrics should be chosen judiciously to avoid bias and ensure consistency.
    - **Combining Metrics**: Using multiple metrics with different properties can provide a more comprehensive evaluation of algorithm performance. However, the selection of metrics should be deliberate, considering their mathematical relationships and ability to reflect various aspects of algorithm validation. Combining metrics that measure different properties can offer insights that a single metric may not capture adequately.

By addressing these aspects, the enhancement of the image classification and segmentation system can lead to improved performance, robustness, and applicability across diverse scenarios.

SECTION 9: REFLECTIONS

Improving the accuracy and F1 score of the final model required exploring various avenues for feature extraction and enhancing the overall pipeline. Here are the reflections on the techniques employed and potential areas for further experimentation:

1. **Histogram of Oriented Gradients (HOG):**
    - Utilizing HOG feature descriptors provides resilience to variations in perspective and shapes of objects like cars.
    - Experimenting with the Scikit-image Python library for calculating HOG features can enhance feature robustness.
    - Implementing sliding windows technique on sub-regions of images divided into grids allows for thorough model predictions, especially considering the varied sizes of cars in different parts of the image.
2. **Eliminating False Positives:**
    - Incorporating a redundancy approach akin to creating a heat map helps improve the accuracy by finding multiple hits for the object of interest in similar areas.
    - Leveraging multi-size sliding windows and labeling objects with overlapping windows using scipy.ndimage.measurements' label function aids in refining the detection process.
3. **Color Spaces Exploration:**
    - Exploring various color spaces such as HUV, HLS, YUV, YCrCb, and LAB can provide insights

into selecting the most suitable color space for the configuration.
    - This exploration is crucial as HOG features across RGB channels may not generate features with enough variations.
4. **Frame Aggregation:**
    - Experimenting with smoothening detected windows across multiple frames strengthens the pipeline.
    - Accumulating detected windows between frames and retaining objects with a certain count across frames helps in double filtering and improving detection accuracy.
5. **Image Augmentation:**
    - Implementing image augmentation techniques can augment the training dataset artificially, thereby enhancing the model's capacity to generalize.
    - Utilizing tools like the ImageDataGenerator class in the Keras deep learning neural network toolkit facilitates the augmentation process and enriches the training dataset.

Overall, these reflections highlight the importance of continuous experimentation and refinement in feature extraction techniques, false positive elimination strategies, color space exploration, frame aggregation methods, and image augmentation approaches to enhance the accuracy and robustness of the car image classification system. Continued exploration and integration of these techniques can lead to further improvements in model performance and generalization capabilities.

THANK YOU NOTE

A big thank you to the Great Learning team for their support throughout our AIML learning journey. Special thanks to our Mentor, Mr. Amit Kumar, for his invaluable guidance and insights. We're grateful for the opportunity and look forward to applying our newfound knowledge.

REFERENCES

https://keras.io/api/applications/

https://arxiv.org/abs/1704.04861

https://arxiv.org/abs/1801.04381

https://arxiv.org/abs/1512.03385

https://keras.io/api/applications/efficientnet/#efficientnetb7-function