Capstone 2 Consolidated Report

Problem Statement

This capstone project aims to build a machine-learning tool capable of precisely and accurately classifying cars by their make, model, body style and year from images acquired from the Stanford car dataset using deep learning.

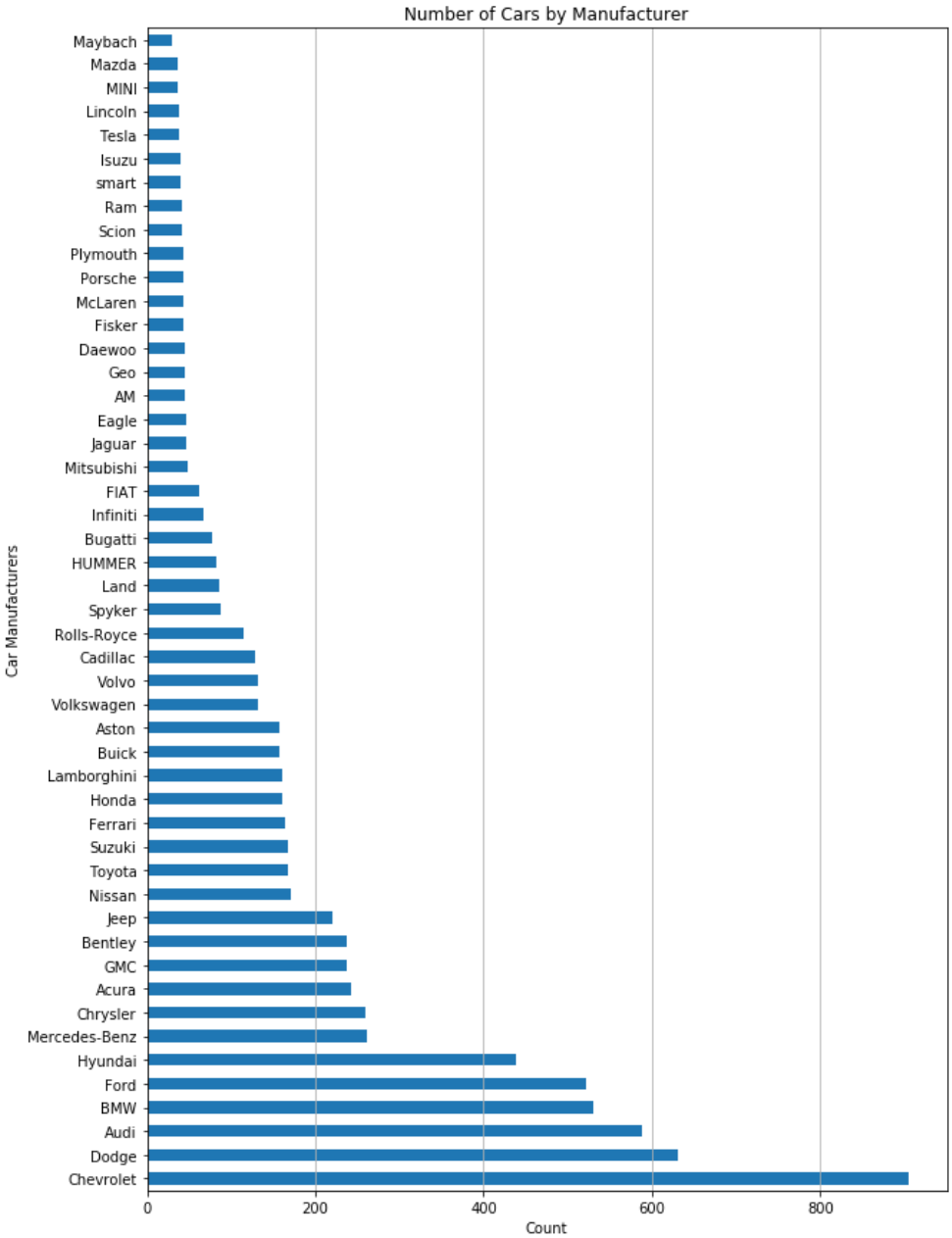
Data Wrangling

The data for the project was collected from a Kaggle dataset based on the Stanford car dataset, which included 196 classes for over 16,000 car images. Each class included the car manufacturer, the model, the body style, and the year the car was released. The data was roughly split 50-50 into training and test folders. All the images were stored in the JPG format within folders providing the class. Alongside the images, three CSV files were included. The first file called names contained the labels for each of the 196 classes within the dataset. The other two files provided insightful information about each image in the training and test data.

I loaded the train CSV file into a pandas dataframe examining the contents of the files given. The dataframe contained filenames of all the images within the train folder, the class number, which corresponded with the labels in the names CSV file mentioned above, and the positions of the boundary box for the vehicle in each image. While the column labels for the data files were listed on Kaggle the files did not have column headers. Once I updated the column names within the dataframe, I used the labels from the names file to add the make, model, and year for each row of the dataset. Along with that, I added two new columns taking the coordinates of the boundary box to give the height and width of the boundary boxes of each image. Specifically, to aid in my exploratory data analysis, in which I wanted to compare the cars within the dataset, I added columns for make, model, body style and year. The information for all these new columns was already established within the labels for each class, I used the builtin str method in pandas to split and isolate each piece of the label for their respective new columns. Once all the steps required to clean the data and make it serviceable for future action was revealed, I created a new python file called clean_data_files.py which would perform the necessary steps on the given CSV files and create a new clean CSV file. Once this script was run for both the train and test data files, I was ready to perform my exploratory data analysis.

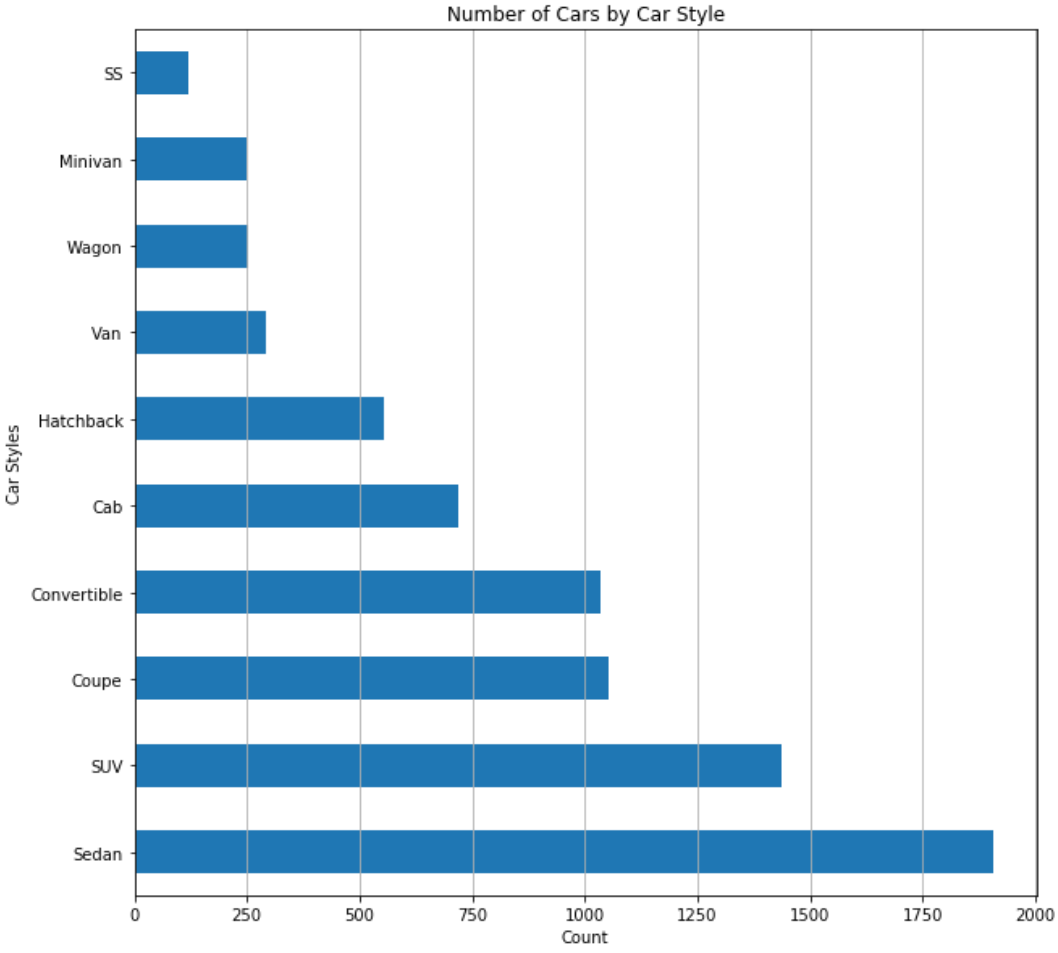
Exploratory Data Analysis

Having created new clean CSV files for both the training and test data, I loaded the clean train datafile into a pandas dataframe aiming to gain further insight into the data through visualization. Since I had isolated the make, model, body style, and release year of each vehicle in the dataset I wanted to aggregate the values to understand how diverse the data was. The graph below illustrates the different car manufacturers present in the dataset alongside the number of cars for each make featured in the training folder.



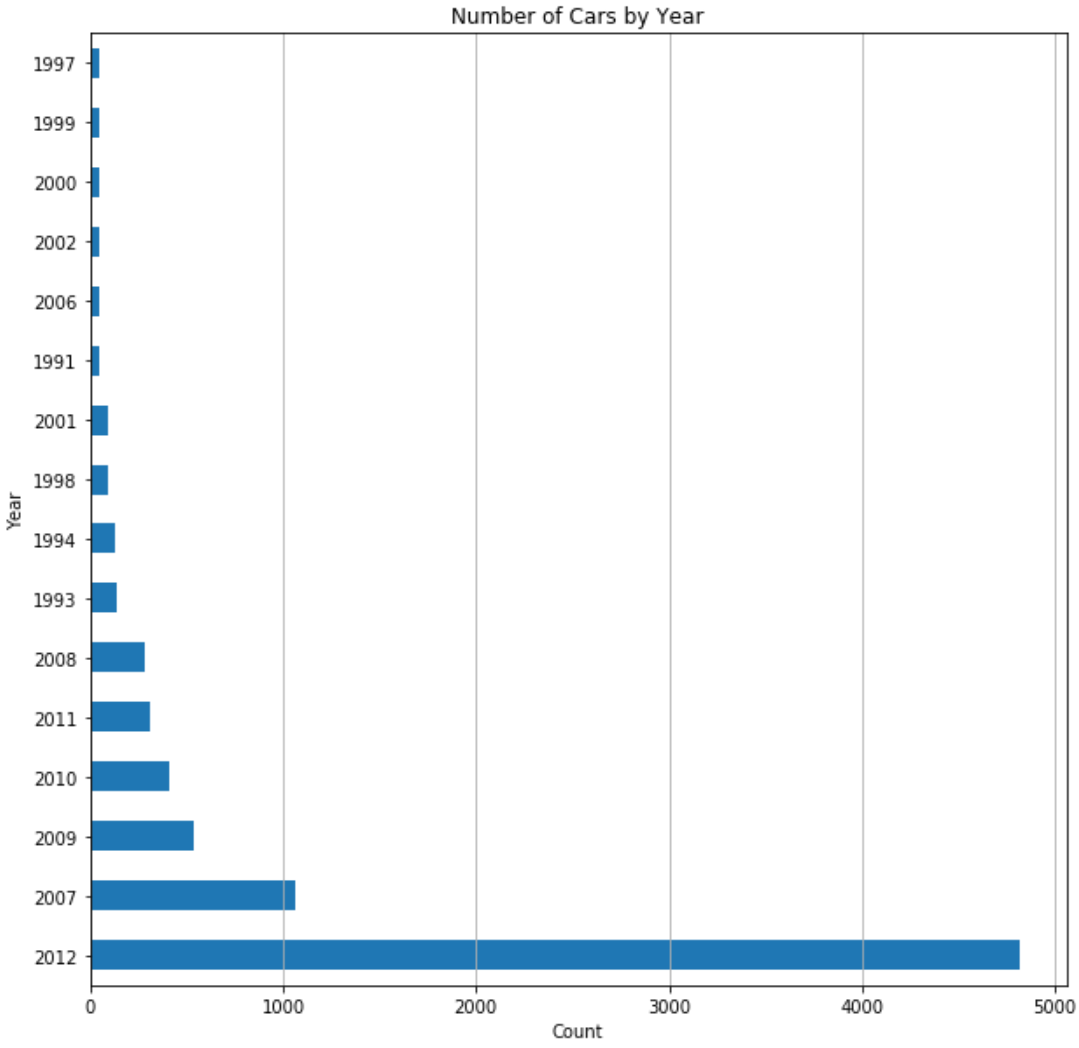
As can be observed from the graph above the dataset is diverse and all major car manufacturers are included. Chevrolet has by far the most vehicles in the dataset with nearly 300 more images than second-placed Dodge, with over 900 images roughly every ninth image in the dataset is a car made by Chevrolet. Twelve car manufacturers have over 200 images validating that a majority of vehicles that can be found on the streets and parking lots are available in the training dataset for our classification model. Many of the manufacturers that have limited photos make far fewer models and body styles than that of the top car manufacturers and so while the images may be fewer there is sufficient data available to classify such vehicles. For example, Tesla in 2012 which is the latest year of vehicles this dataset included made only one car the Model S available as a sedan, whereas Chevrolet manufactured 14 different car models alone not including the different body styles available for each model. The one car manufacturer that makes a variety of models but is largely underrepresented in this dataset is Mazda who has the second-fewest images in the Stanford dataset.

Let us now examine the different body styles of vehicles present in the dataset, due to many styles being unique to certain vehicles and having very few images only the ten most popular body styles were included in the graph below.



The graph above displays the body styles of the cars present in the Stanford car dataset. Mirroring the popularity of vehicles on the road, sedans and SUVs are the body classes that appear the most in the images provided. Sedans and SUVs account for over 3300 of the 8000 images present in the Stanford car dataset. Unlike the earlier graph, all the major and popular body styles included have a large number of images whereas earlier a leading car manufacturer such as Mazda has substantially fewer images compared to the competing automotive manufacturers of similar size.

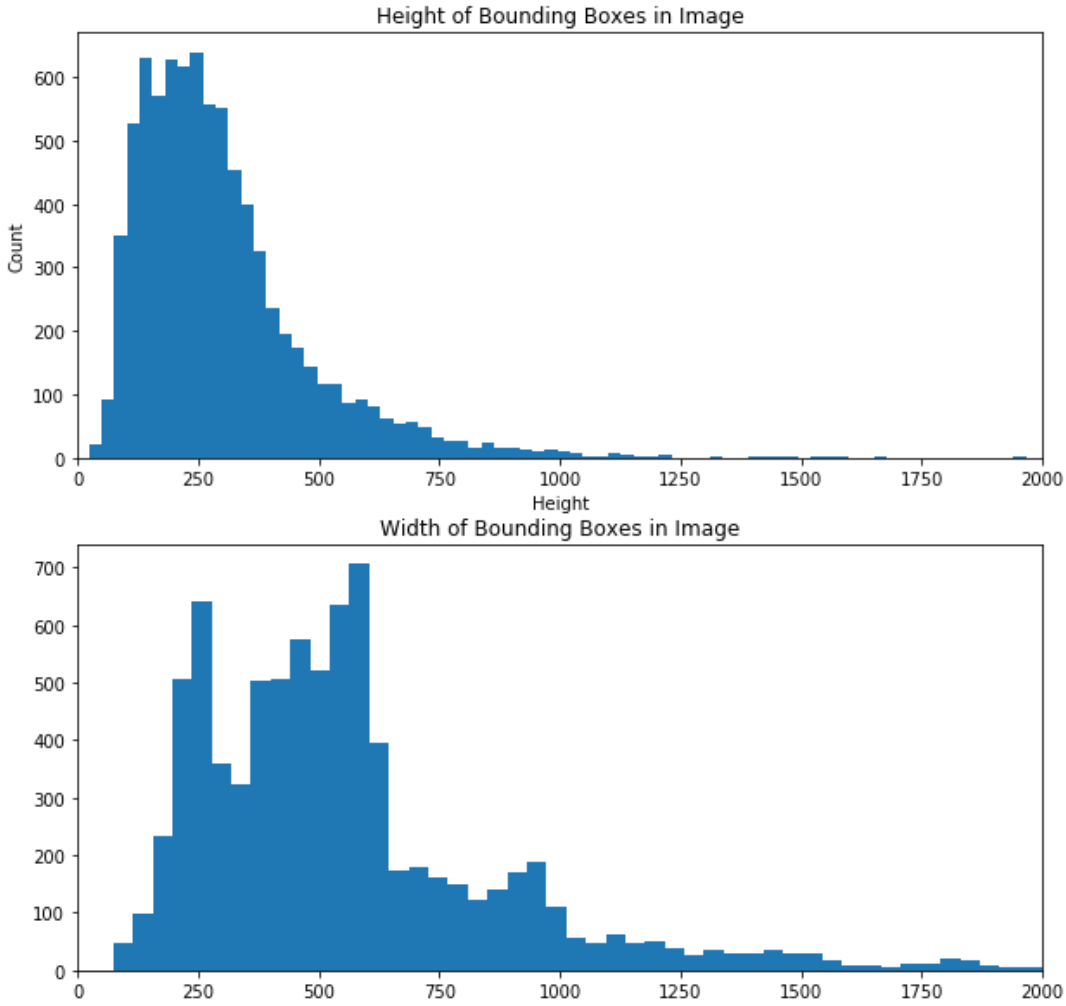
The below graph explores the number of images of the vehicles by their respective manufacturing year.



The recency bias on display in the dataset is acceptable for our model, primarily due to older vehicles being far less likely to be present on the road than newer models. 2012 remains the most popular and most recent manufacturing year for the dataset with nearly 5,000 of the 8,000 images being from cars released in 2012. In contrast, 2007 is the only other year in which cars

manufactured have over 1000 images present in the dataset. That is also in line with automotive manufacturing trends as a new model for a car manufacturer arrives every four to five years while vehicles get small incremental updates annually in the years in between. The one shorting coming of the model is that it contains no images for vehicles prior to 1991, while vehicles made earlier can rarely be seen on the road often in shows and parking lots owners have more vintage vehicles that gain popularity after being decades old. The inclusion of images of vehicles decades old today could have proven valuable for predicting vehicles that most would not be able to easily identify themselves, an ideal use case for our model.

Finally, the last graph below illustrates the deviation in the boundary boxes for each vehicle. The two histograms contrast the height and width of the boundary boxes for cars inside the images.



As observed above the height boundary compared to the width is more centralized in size. The width is more varied allowing for the boundary boxes to be wider and more diverse. This can be explained by the dimensions of a vehicle which are almost always wider than are tall. The

images plotted in the jupyter notebook displayed similar characteristics with the images being almost twice as wide compared to the height.

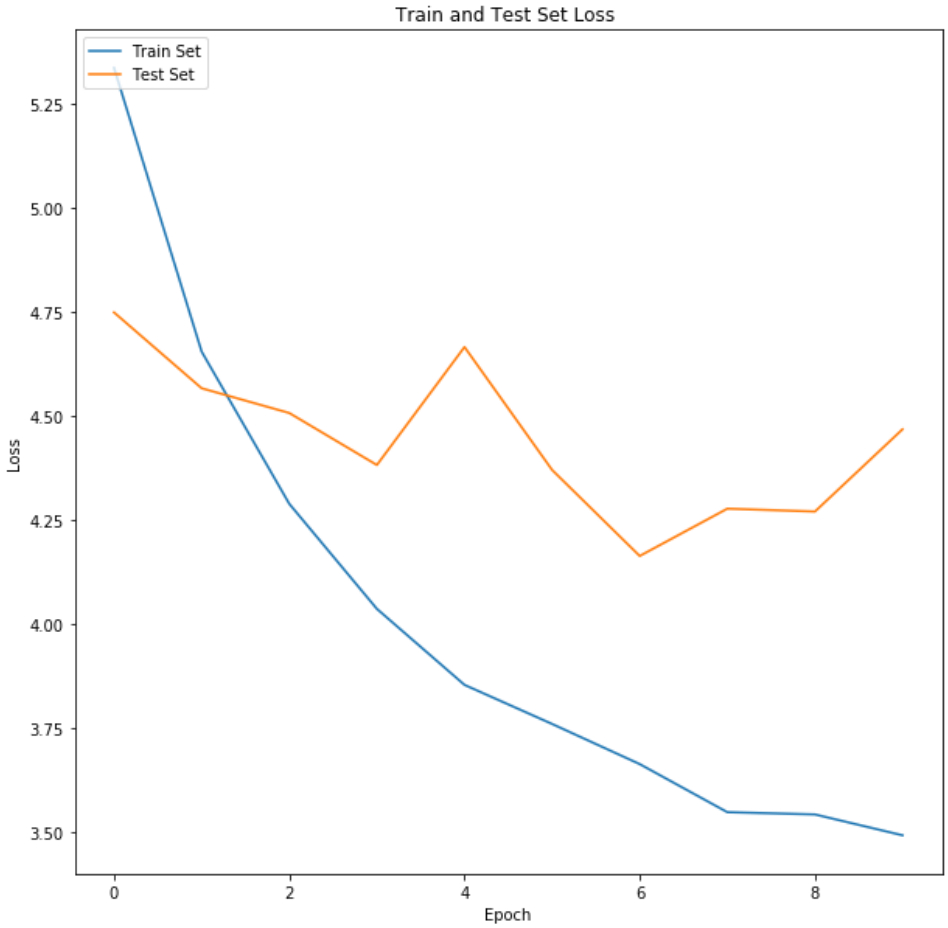
The visualizations yielded from the exploratory data analysis have garnered critical insights into our data. We know that the data is diverse amongst most leading automotive manufacturers, Mazda being the one exception. The most popular body styles for vehicles are captured in larger numbers in the dataset, and while the recency bias in manufacturing year is not in complete detriment to us, considering the use cases of our predictive model, the performance on classifying older vehicles through images will be something to pay attention to when modelling. The differences in the boundary boxes dimensions validate, images processed will, almost always, contain a larger number of pixels in an image's width compared to its height.

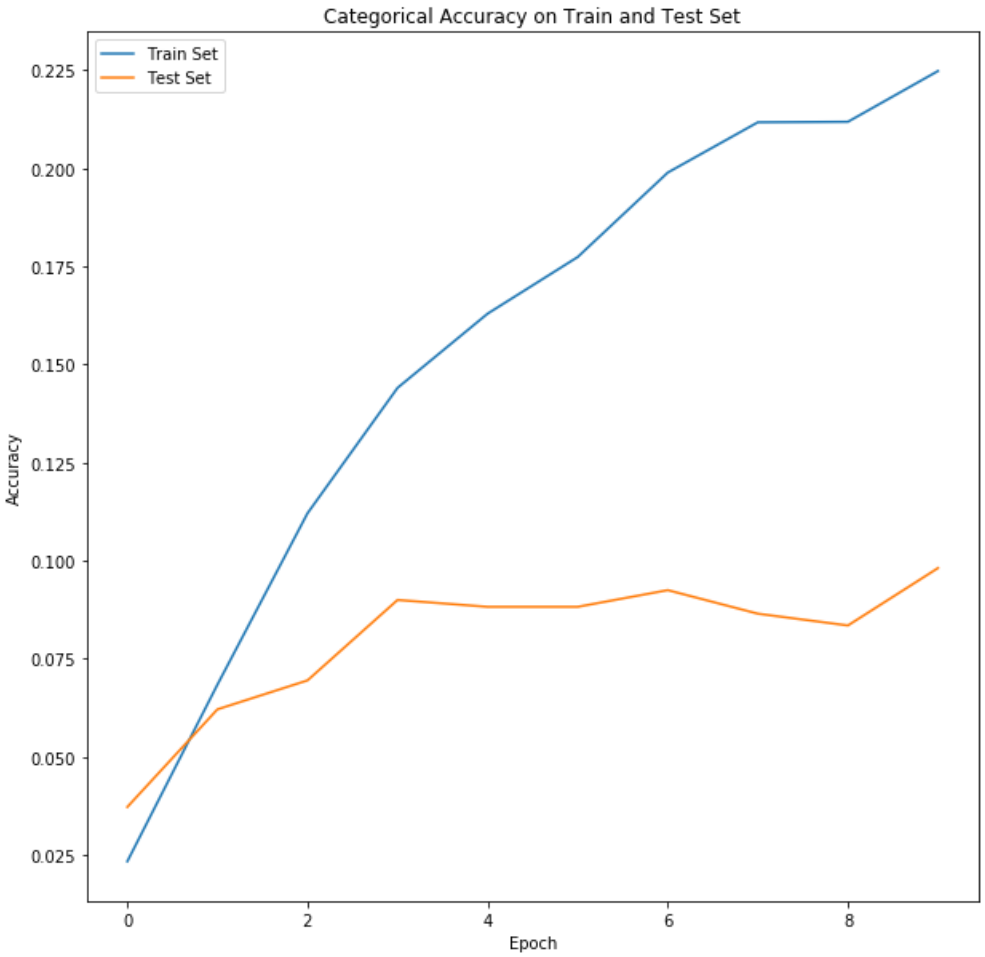
Modelling and Results

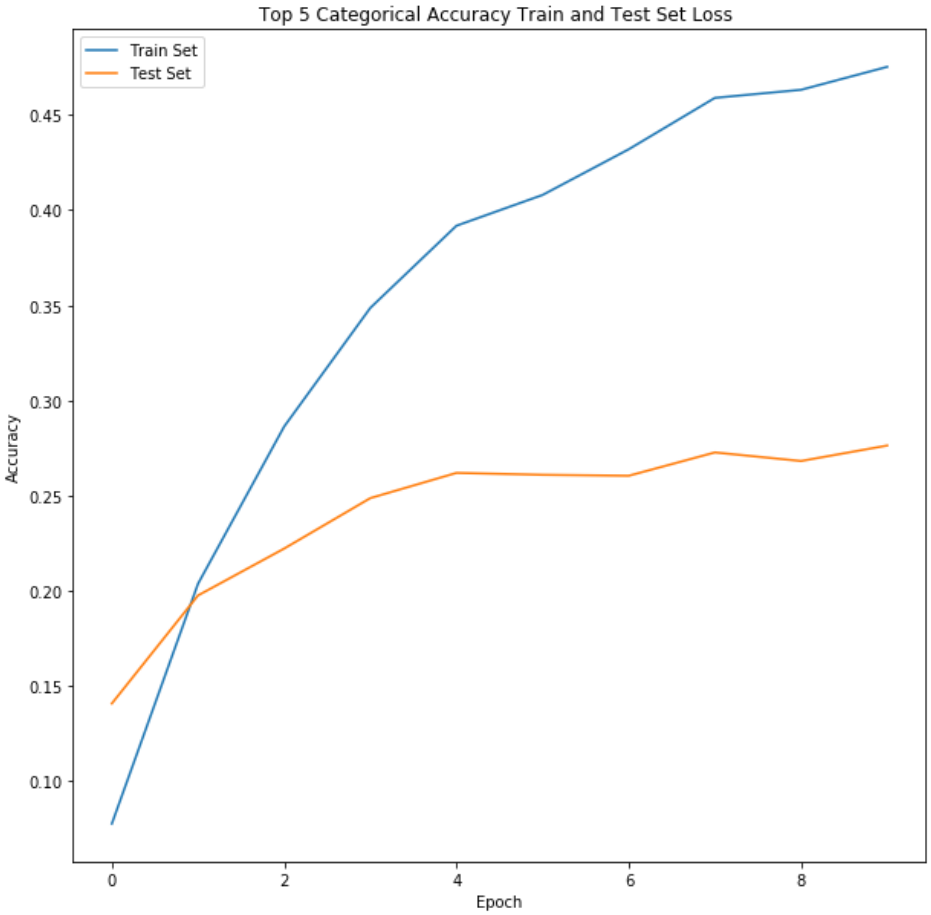
After gaining further insight into the Stanford car dataset through the exploratory data analysis, I was ready to begin modelling. To create a deep learning model capable of accurately predicting the make, model, body style, and year of given car images I would need to use Keras and TensorFlow. Instead of creating a neural network from scratch using the several pre-trained models available for image processing would allow me to save time and resources. All the photos for both the training and test sets were already split up into folders with their class names so I did not have split the data, having it split in roughly 50-50 was both convenient for setting up, but I was also aware having such a large test set comparable to the training set meant the model would take much longer to validate and the sheer size of unseen data would take longer to compute and access. The other benefit to such a large test set would be that performance on unseen data could be very indicative of how the model would perform on new data since the test set is roughly equal to the training set and overfitting will be readily apparent.

The two pre-trained models I used were InceptionV3 and MobileNetV2. To prepare the images to be used ImageDataGenerator was imported from the Keras preprocessing and it was applied to all images in both the training and test sets. Simple data augmentation plays a key role in preprocessing images and so images were stretched both horizontally and vertically to allow for the model to be able to function even when fed faulty images. Once the preprocessing was done the InceptionV3 model was the first pre-trained model and was the first pre-trained model used. The imagenet weights were loaded as the pre-trained weights. Before unfreezing several of the final layers in the model I ran the model as is first to assess performance and validate that the model showed significant improvement over the training cycle. Without unfreezing any layers the pre-trained InceptionV3 had 22 million parameters of which slightly more than 400 thousand were trainable as yielded from the summary method of our model. Alongside loss and accuracies, the other custom metric I introduced was the top-five categorical accuracy, to

highlight the performance on the best five classes using the top_k_categorical_accuracy function from the metrics library imported earlier. We established earlier, major car manufacturers and models released in more recent years were more prevalent earlier in the exploratory analysis and so this metric would give a better understanding of the performance at the high end. With the model compiled and the optimizer and loss functions defined using NAdam and categorical cross-entropy, I trained the model on 10 epochs with a batch size of 64 the graphs below display the improvements yielded throughout the training process.



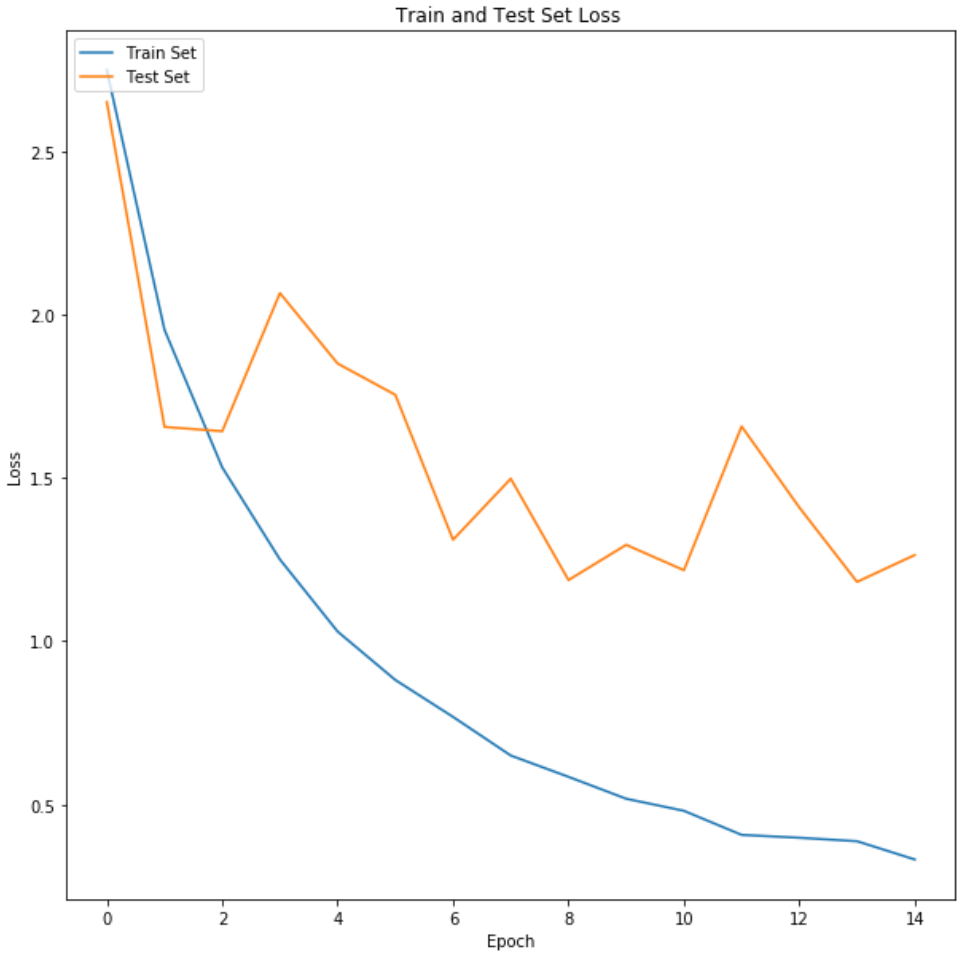

Categorical Accuracy on Train and Test Set
Train Set
Test Set
Accuracy
Epoch
0.225
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0
2
4
6
8

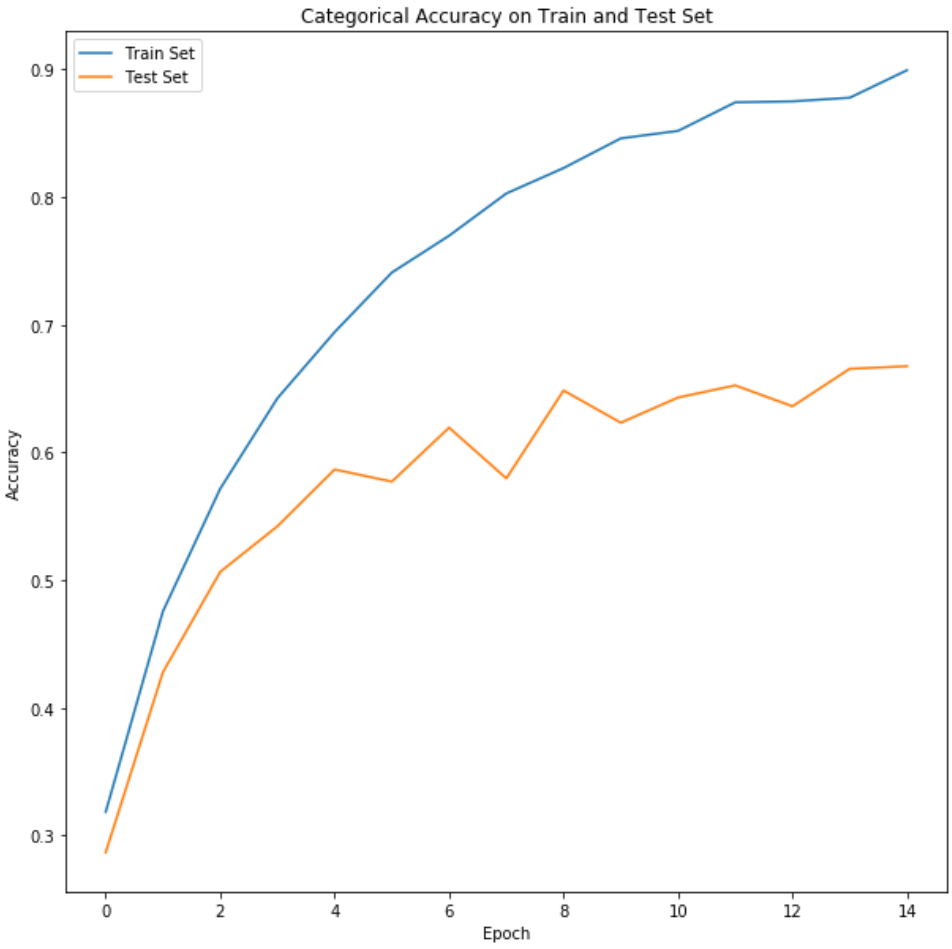


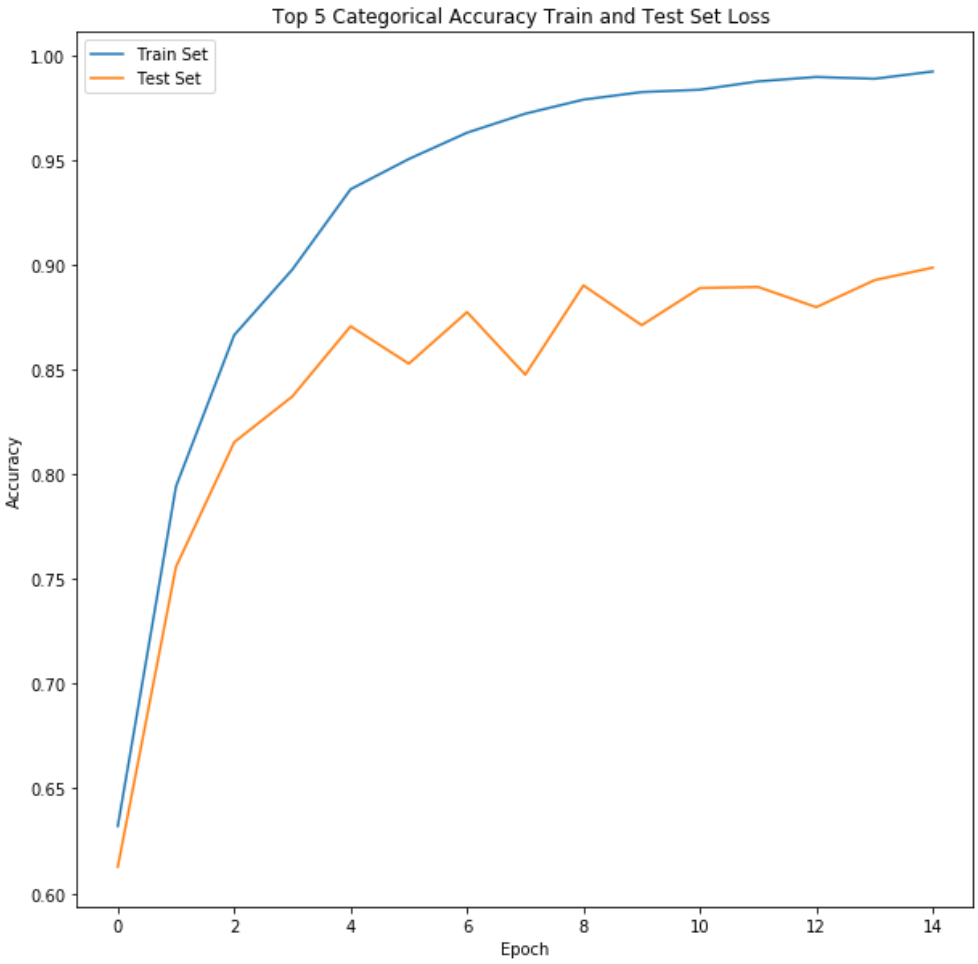
As can be observed by the three graphs above the performance of the predictive model improved significantly throughout the training process. The loss of the model decreased while the categorical accuracy and top-five accuracy increased for both the training and test set. The loss reduced from 5.34 to 3.49 for the training set and from 4.75 to 4.16 and then increased again finishing at 4.47. The loss for the validation data did not dramatically improve like with the training data and this was to be expected and can be seen in the accuracy metrics as well. The categorical accuracy grew from 2.3 percent to 22.5 percent for the training set and from 3.7 to 9.8 percent for the test data. Again we do not see the same rate of growth with the test set as we experienced with the training set. Finally, the top-five categorical accuracy conforms to similar disparity growing for 7.8 percent to 47.5 percent with the training data and from 14.1 percent to 27.6 percent in validation data. It is clear that using a pre-trained model with such few trainable parameters will not give us the performance our predictive model requires and that due to the size of the validation data more epochs will be required as the performance is still volatile with the loss increasing in the last two epochs instead of decreasing.

The InceptionV3 model's conv2d layers were slowing unfrozen to allow for the model to train more of its parameters and thus optimize weights for our needs rather than be the weights for a

different problem as they were before. The final model used unfroze three of the last conv2d layers to increase trainable parameters in the model to 11.5 million from the earlier 400 thousand. Keeping the loss function, optimizer, and batch size the number of epochs was increased to 15 and the improvement of this model's performance can be seen in the graphs below.



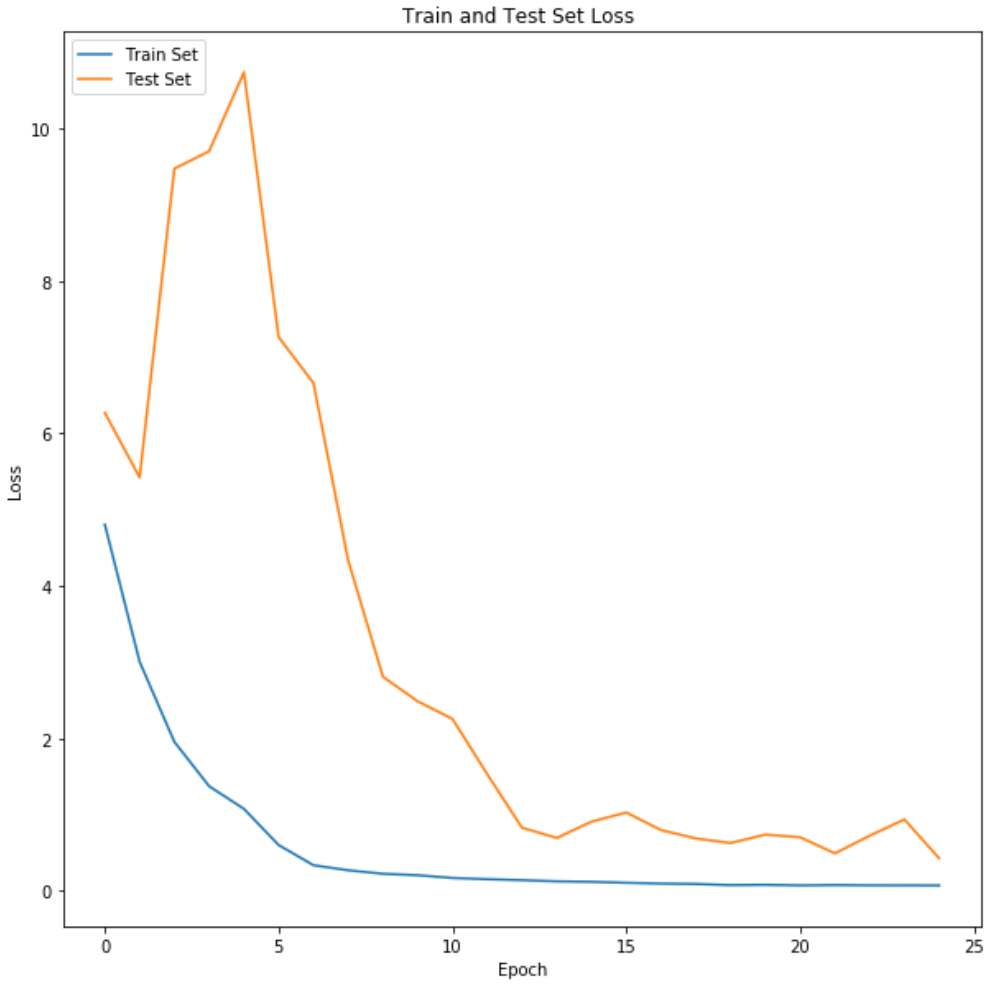

Categorical Accuracy on Train and Test Set
Train Set
Test Set
Accuracy
Epoch
0.3
0.4
0.5
0.6
0.7
0.8
0.9
0
2
4
6
8
10
12
14

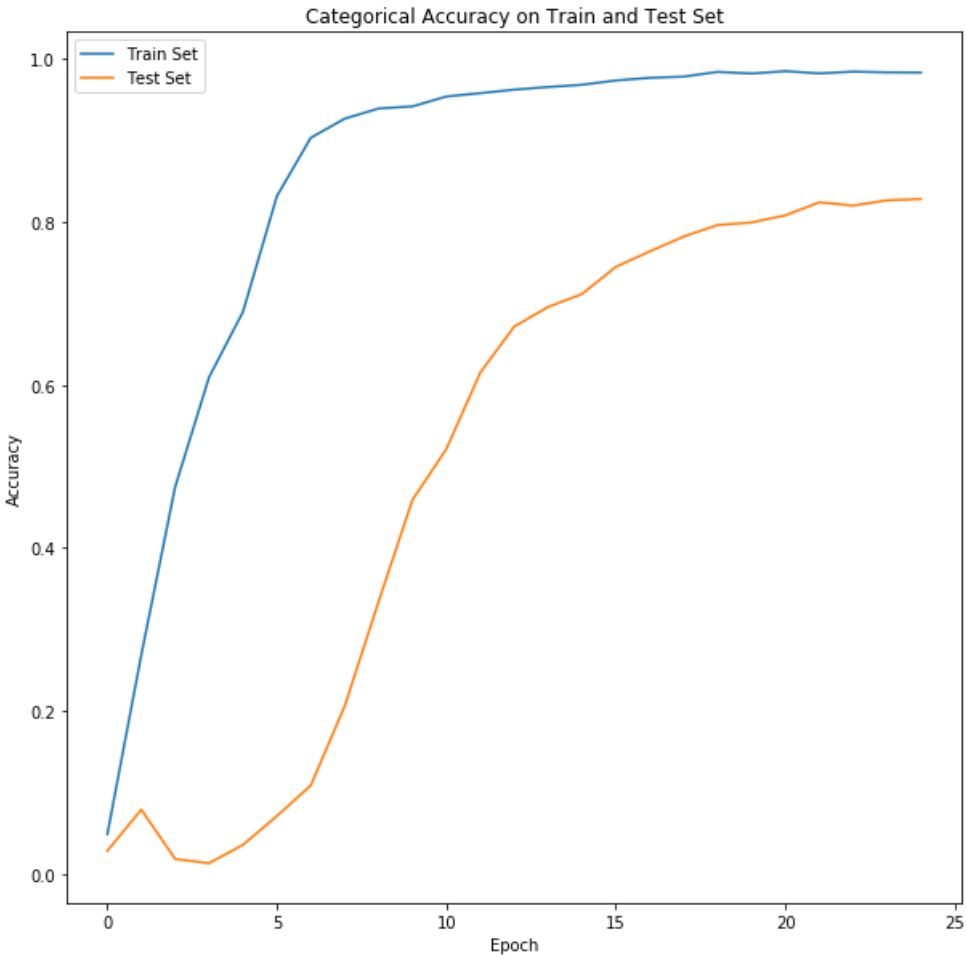


The improvement in the performance of this model with more trainable parameters is substantial compared to the base model earlier. All three key metrics are better in this model at the first epoch then they were at the end of the tenth in the original model. The loss starts at 2.75 and is 0.33 by the 15th epoch for the training set and improves from 2.65 to 1.26. The categorical accuracy on the training set improves boosts from 31.8 percent to 89.9 percent and from 28.7 to 66.8 percent. The top-five categorical accuracy also shows dramatic improvements from 63.2 to 99.24 percent for the training data and 61.3 to 89.9 percent in the validation data. While the improvements compared to the last model are substantial an overall categorical accuracy of 66.8 percent on the test set is still slightly under what we expect. We can validate from these results that unfreezing layers and having more trainable parameters and more epochs to train all contributed to the betterment of the performance of our predictive model and we will use what we learned when creating and compiling the next model.

The model will be based on the pre-trained MobileNetV2 imported from Keras with the same weights as earlier loaded imagenet. I did not bother training the base model this time and set trainable to true all the layers that could be trained. The optimizer for this model was changed to

SGD and had the learning rate and patience reducible when reaching plateaus in performance. Out of a total of 6.4 million parameters only about 48 thousand were untrainable. This allowed me to train almost all the parameters allowing me to optimize the weights of the model for my specific use case. I increased the number of epochs to 25 to allow the model more time to find the best model and be adaptable in changing the learning rate and patience when performance was plateauing. The loss function, metrics, and batch size were carried over from the last model. The graphs below display the training of the model and its performance throughout the 25 epochs.





While the performance metrics start slow for this model compared to the last, taking about five epochs before the learning rate is reduced and the performance starts to show improvement over the epochs. The loss in particular for the test set continually increases between the second and fifth epoch until the learning rate is reduced. Following that the learning rate is once again reduced in the seventeenth epoch for the final time. Despite a poor start with a loss of 4.80 the loss finishes after 25 epochs at 0.08 for the training set and improves from 6.27 to 0.49 for the test set. The categorical accuracy and top-five accuracy are bettered from 4.9 and 15.3 percent to 98.3 and 99.9 percent for the training set. The test data also displays the best performance yet rising from a categorical and top-five accuracy of 2.8 and 10 percent to 82.8 and 96.4 percent respectively. These huge strides of improvement throughout showcase that this model is not only exceptionally strong with classifying images it is trained on but predicting the classes of cars with unseen data as the strong metrics with the validation data suggest.

Conclusion

Using a pre-trained MobileNetV2 neural network and optimizing its parameters for the Stanford car dataset yielding accurate results in classifying car images by their make, model, body style and year. With an accuracy of 82.8 percent on the test images, roughly the same number as the training images used to train the model we have successfully built a predictive model capable of classifying cars by their images as we originally set out to do. Through our exploratory data analysis earlier we had identified key strengths and weaknesses within the Stanford car dataset, such as the lack of images for a major car manufacturer such as Mazda, and a skewed portion of images for more recent car models while ignoring any vehicles made before 1991 entirely. The faults in the dataset are inherent to the model's shortcomings on unseen data that is not available in the original dataset. As mentioned earlier many of these issues in the dataset can be ignored due to the occurrence of a vehicle made prior to 1991 on the road today in 2020 being rare.
To further enhance the performance of the model, the dataset first must be updated to include more images of marginalized car manufacturers and car models both older than 1991 and newer than 2012, the year the dataset was published. These changes to rectify the underrepresented vehicles in the original dataset can allow for better performance for the model amongst all vehicles that can be found on the road and photographed.