# Image Classification CNN for Car Make and Model

David Opie
University of Maryland Baltimore County
CMSC 691 Final Report
Spring 2022, Group #16

***Abstract-***
***I present image classification convolutional neural network for determining the make and model of a car when shown its image. I have created trained CNN models with high accuracy using a Stanford dataset with over 15000 images of cars (found here: [https://ai.stanford.edu/~jkrause/cars/car_dataset.html](https://ai.stanford.edu/~jkrause/cars/car_dataset.html)). Using those models, I attempt to accurately predict the car make, model, and year from an image.***

## 1. INTRODUCTION

Vehicle detection and identification is critical for enforcing traffic control, statistical analysis, and supporting law enforcement. When identifying a vehicle there are several different ways we can categorize, from the general like trucks and cars to the detailed like make and model. In this implementation of a vehicle detection and identification convolutional neural network we need to know the make, model, and year of manufactory, e.g. 2012 Tesla Model S or 2012 BMW M3 coupe. We use the Stanford dataset, a dataset that contains "16,185 images of 196 classes of cars. The data is split into 8,144 training images and 8,041 testing images."(Krause, 2013) The goal of this project is to create a CNN model that correctly identifies a vehicle with a relatively high accuracy. To create such a model, I started with a basic CNN that only trained on 5 epochs. To achieve a higher accuracy, I implemented a ResNet50 CNN framework and trained using more epochs.

## 2. RELATED WORK

The beginning of vehicle detection and classification seems to have started almost 10 years ago with 3D Object representations for fine-grained categorization (Krause et al. 2013). Over those 10 years the machine learning community and technology as a whole has made leaps and bounds in advancements. Even so the methodology behind their work still stands and you can see their dataset is one of many standards used for training today. Those achievements made in between then and now have given better access to stronger computations and stronger neural networks.

Liu et. al (2015) created a baseline for Image classification of vehicle make and model and are what I base my project on. However, their CNN models have become somewhat outdated and current models can achieve much better results.

Maity et. al (2021) builds the foundation for my future attempt with object detection using Faster R-CNN and YOLO based Vehicle detection.

Bautista et. al (2016) on Convolutional neural network for vehicle detection in low resolution traffic videos does not attempt to identify vehicles by make and model like my implementation will.

Potdar et. al (2018) on A Convolutional Neural Network based Live Object Recognition System as Blind Aid uses a different dataset for normal objects and not vehicles.

Benjdira et. al (2019) on Car Detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3 attempts to look at vehicles from a sky view unlike mine.

## 3. METHODOLOGY

My methodology is to create strong CNN models that can accurately predict cars from their images. Strong CNN models need a large dataset, strong architecture, and a long enough time to train. I began the project by looking for a dataset and settled on the Stanford dataset; a collection of 197 different classes of car separated by car make, model, and year using over 16000 images. The Stanford dataset contains "16,185 images of 197 classes of cars. The data is split into 8,144 training images and 8,041 testing images." (Krause et al. 2013) The large dataset contains images with varying size but CNNs require datasets to be uniform. Resizing the images into sizes of 224 by 224 distorts the data but creates a uniform model. In my initial assessment I used a basic CNN model shortly trained on only 5 epochs. Increasing the training to 50 epochs increased the accuracy but not to an acceptable level. The model summary can be seen below in Figure 1.

| Layer (type) | Output Shape | Param # |
|---|---|---|
| sequential (Sequential) | (None, 200, 200, 3) | 0 |
| conv2d (Conv2D) | (None, 200, 200, 17) | 476 |
| max_pooling2d (MaxPooling2D) | (None, 100, 100, 17) | 0 |
| dropout (Dropout) | (None, 100, 100, 17) | 0 |
| conv2d_1 (Conv2D) | (None, 100, 100, 39) | 6006 |
| max_pooling2d_1 (MaxPooling2D) | (None, 50, 50, 39) | 0 |
| conv2d_2 (Conv2D) | (None, 50, 50, 87) | 30624 |
| max_pooling2d_2 (MaxPooling2D) | (None, 25, 25, 87) | 0 |
| conv2d_3 (Conv2D) | (None, 25, 25, 87) | 68208 |
| max_pooling2d_3 (MaxPooling2D) | (None, 12, 12, 87) | 0 |
| conv2d_4 (Conv2D) | (None, 12, 12, 87) | 68208 |
| max_pooling2d_4 (MaxPooling2D) | (None, 6, 6, 87) | 0 |
| dropout_2 (Dropout) | (None, 6, 6, 87) | 0 |
| flatten (Flatten) | (None, 3132) | 0 |
| dense (Dense) | (None, 500) | 1566500 |
| dense_1 (Dense) | (None, 197) | 98697 |

Total params: 1,838,719
Trainable params: 1,838,719
Non-trainable params: 0

*Figure 1. Basic CNN model summary.*

The product and need for higher accuracy from the models can be seen in the demonstration. To demonstrate the models, I created a prediction program that would load in the models and use them to predict the label of an image. There we can see a clear success or failure of the model. To achieve success with my models I needed to move to a different framework.

ResNet50, a deep neural network framework is a popular and accurate framework that can accurately be used for car image detection (Tahir et. al, 2021). ResNet50 is a variant of the ResNet model with 48 convolutional layers that allows for ultra-deep neural networks. Creating a new model with a stronger architecture like ResNet50, with adequate time to train, produced a highly accurate model. I then used this model for my prediction model.

## 4. RESULTS

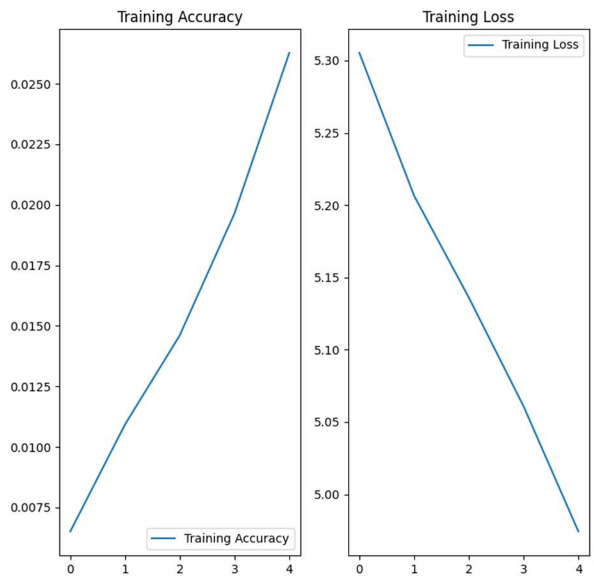



*Figure 2. Basic CNN (5 epochs) Training Accuracy and Loss.*

In my initial test the trendline seen in Figure 2 indicates a clear growth rate for the basic CNN model. The model is clearly training on the Stanford dataset, getting more accurate, and decreasing its loss. I then increase the epochs to 50, which is my standard for my other model. The basic CNN model trained on 50 epochs demonstrates a significant accuracy growth and decrease in loss. However, at only ~35% accuracy with 50 epochs It would take another 50 epochs to achieve ~70% accuracy, which is almost usable for the predicter if it follows the trendline represented below in Figure 3.

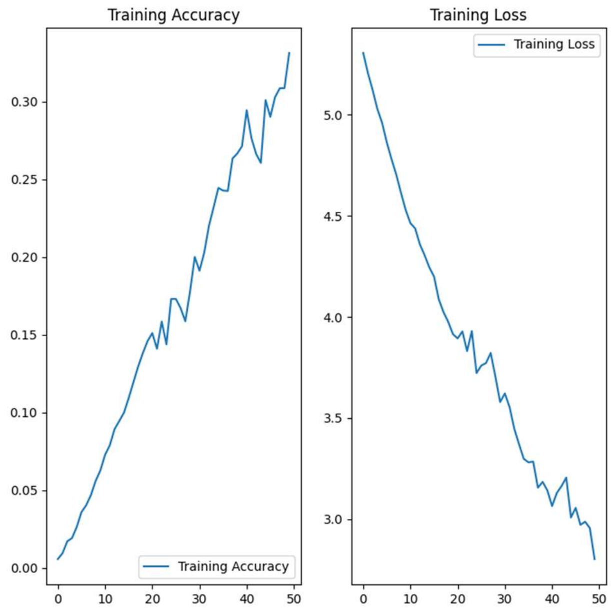
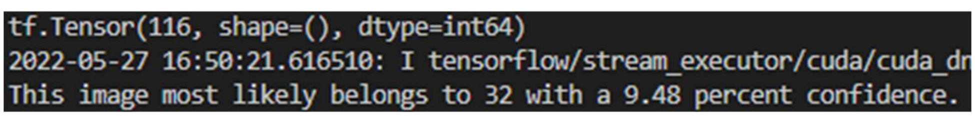



*Figure 3. Basic CNN (50 epochs) Training Accuracy and Loss.*

Using the prediction program, on the basic CNN model at 50 epochs, I demonstrate the failure of the model to accurately predict the correct vehicle, as seen in Figure 4. The prediction program uses the basic CNN model to predict the image belongs to class 32 with 9.48% confidence when the image belongs to class 116. This specific image can be seen in Figure 5.



```
tf.Tensor(116, shape=(), dtype=int64)
2022-05-27 16:50:21.616510: I tensorflow/stream_executor/cuda/cuda_dn
This image most likely belongs to 32 with a 9.48 percent confidence.
```

*Figure 4. Basic CNN (50 epochs) Prediction.*

*Figure 5. Prediction Program Result Image.*

A ~35% accuracy model consistently fails to produce accurate results unless we use a better CNN model and train with a larger epoch set. Moving to the ResNet50 model we find a highly accurate model (~90%) after only ~20 epochs. The superior framework quickly builds accuracy and decreases loss as seen in Figure 6 below.

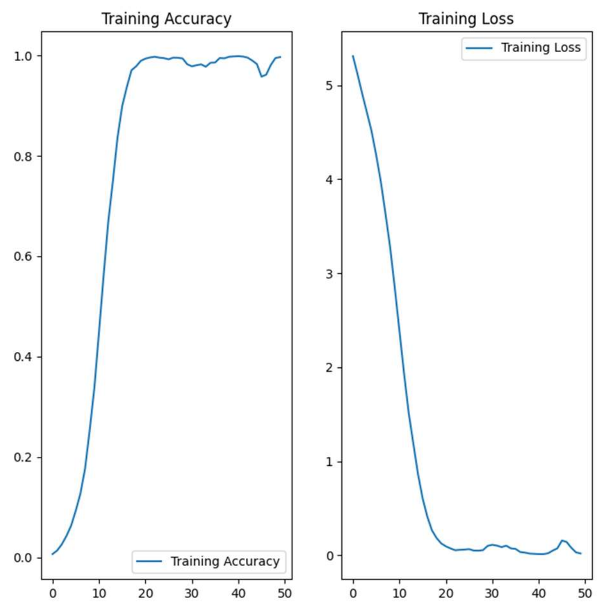



*Figure 6. ResNet50 CNN (50 epochs) Training Accuracy and Loss.*

However, using this model with my prediction program I receive an unexpected result as seen in Figure 7. The model produces an incorrect result with a lower percent confidence.

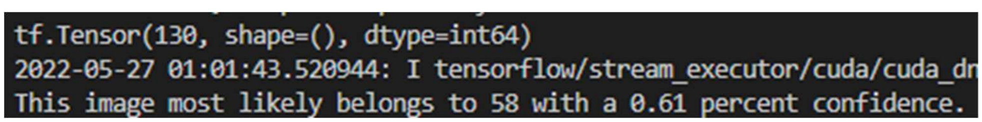



*Figure 7. Basic CNN (50 epochs) Prediction.*

## 5. CONCLUSION

I present an image classification convolution neural network and prediction program for determining the year, make, and model of a car using the Stanford dataset. I begin with simple CNN models like my Basic CNN model and move on to a more complex framework. Upgrading the framework of my CNN models I achieve better accuracy and lower loss. I attempt to use these models along with the prediction program to accurately classify images but fail even when using the ResNet50 framework. The ResNet50 model produces high accuracies but fails to classify images. The disassociation between my model accuracy and my model prediction is not yet understood. Moving forward I will attempt to solve this mystery and expand into object detection with YOLO or Faster-CNN.

REFERENCES

Krause, J., Stark, M., Deng, J., & Fei-Fei, L. (2013). 3D object representations for fine-grained categorization. *2013 IEEE International Conference on Computer Vision Workshops*. doi: https://doi.org/10.1109/iccvw.2013.77

Liu, D. & Wang, Y. (2015). Monza: Image Classification of Vehicle Make and Model Using Convolutional Neural Networks and Transfer Learning.

M. Maity, S. Banerjee and S. Sinha Chaudhuri, "Faster R-CNN and YOLO based Vehicle detection: A Survey," 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), 2021, pp. 1442-1447, doi: 10.1109/ICCMC51019.2021.9418274.

C. M. Bautista, C. A. Dy, M. I. Mañalac, R. A. Orbe and M. Cordel, "Convolutional neural network for vehicle detection in low resolution traffic videos," 2016 IEEE Region 10 Symposium (TENSYMP), 2016, pp. 277-281, doi: 10.1109/TENCONSpring.2016.7519418.

Potdar, K., Pai, C. D., & Akolkar, S. (2018). A Convolutional Neural Network based Live Object Recognition System as Blind Aid. doi: https://doi.org/10.48550/arXiv.1811.10399

B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar and K. Ouni. (2019). Car Detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3. *1st International Conference on Unmanned Vehicle Systems-Oman (UVS)*, 2019, pp. 1-6, doi: 10.1109/UVS.2019.8658300.

H. Tahir, M. Shahbaz Khan and M. Owais Tariq, "Performance Analysis and Comparison of Faster R-CNN, Mask R-CNN and ResNet50 for the Detection and Counting of Vehicles," 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), 2021, pp. 587-594, doi: 10.1109/ICCCIS51004.2021.9397079.