Car Classifier

Raymond Adetola Ogunjimi, Cesar Nunez Rodriguez.

Department of Electrical and Computer Engineering, Villanova University

800 E Lancaster Avenue, Villanova, PA 19085, USA

rogunjim@villanova.edu, cnunezro@villanova.edu

Link to the code:
https://colab.research.google.com/drive/1x_UiB56QBGH9RhK6O99do2CUn51ZO_l5?usp=sharing

***Abstract*— This project aims to classify images of different makes and models of vehicles with high accuracy and efficiency. This image classification application is very popular in the field of machine learning whether it is binary or multi-class classification. This project aims to implement multi-class classification to some of the most popular vehicle designs on the market.**

***Keywords*— Artificial Intelligence, Machine Learning, Deep Learning, Convolutional Neural Networks, Image Classification**

I. Introduction

In the car industry a lot of companies take inspiration from their competitors in terms of design. Tesla is a newer company and its designs are subjective and polarizing, but also widely accepted as some of the most iconic and beautiful in the industry. Many car enthusiasts have been trying to trace Tesla's designs back to a root and an inspiration. Among the top contenders are Porsche, Lotus, and Aston Martin. Tesla is obviously tied to lotus as the first generation roadster was basically an electrified version of an existing Lotus vehicle. Porsche and Aston Martin simply show a very strong resemblance to a frog-like design. The intention of this project is simply to see if a machine to make a machine a car enthusiast. As we grow older we become more and more receptive to the design principles of different companies which in a way is very similar to the way machines learn as will be further discussed in this paper.

II. Machine learning Approach

The machine learning approach taken was the use of Google's open-source machine learning framework called TensorFlow in combination with the neural network framework named Keras. Although PyTorch and SciKit frameworks also have advantages and competitive products, TensorFlow was chosen due to its ease of access, connection to Google Colab, active community, and great support. Further specifications of our implementation are listed below.

A. Specifications

Each item below was chosen considering the specific application and constraints of this implementation and should be altered for any other application. For instance, for our experimental application, a sequential rather than a functional model was chosen as detailed below in further detail.

- Sequential 5-layer neural network
  i. 2D Convolution
  ii. Max Pooling
  iii. Fully Connected: Dense
  iv. Flatten
- ReLU activation Function
- Adam optimizer with learning_rate = 0.0001
  i. tf.keras.optimizers.Adam
- train/test split = 0.9/0.1

Some parameters are used to control the iterations through the neural network.

- epochs = 20
- Batch_size = 16

III. Data

The input data for the neural network was collected through a batch image downloader for publicly available images. The method for this was a publicly available web scraper chrome extension. However, this plug-in has no functionality to filter images. Images had to be filtered for the correct extensions and for corrupt files using utility scripts

and manual labor which was one of the most inefficient and intensive parts of the project. Ironically, due to Google's keyword search machine learning algorithms, some image data points were also simply incorrect and initially confused the model before they were discarded and the data was further cleaned and specified.

IV. Results

The predictions and results for the Neural Network Model are seen in Table 1.

Table 1: Prediction Results for the Neural Network Model

Models	**Number of Images**	**Training Time**	**Accuracy**
Porsche (911 and Macan) vs Tesla (Model Y and Model 3)	388 / 307	27s x 20 epochs (9 Minutes)	72.46%
Porsche Taycan vs Porsche Macan	267 / 193	17s x 20 epochs (6 minutes)	66.67%
Porsche Taycan vs Tesla Model 3	267 / 184	18s x 20 epochs (6 minutes)	68.89%
Porsche Taycan vs Toyota Tacoma	267 / 203	19s x 20 epochs (6 minutes)	78.26%
Tesla Model 3 vs Tesla Model Y	184 / 122	13s x 20 epochs (4 minutes)	56.67%
Tesla Model 3 vs Toyota Tacoma	184 / 203	16s x 20 epochs (5 minutes)	89.47%

The results align with the expected results. For example, it was difficult for the model to distinguish between models from the same car manufacturer. The model accuracy to classify between Porsche vehicles was relatively low at 66.67%. The worst accuracy results came from the Tesla Model 3 and Model Y comparison, at 56.67%. It is obvious that the design ideologies of these car manufacturers can make recognizing different car models very difficult. This can also be observed when comparing two models from different brands that have similar design ideologies, as the model also relatively struggled when it tried to differentiate between the Taycan and the Tesla Model 3.

In addition, the model was most accurate at classifying between the two most different vehicles, the Toyota Tacoma and the Tesla Model 3. The model also did well when comparing the Taycan and the Tacoma.

V. Conclusions

Overall the outlined project took a great deal of effort in terms of the acquisition of data, the training of the model, the tuning of the performance, and the validation using both new images and images split for testing. The accuracy of the model has been outlined in Section IV. Data is certainly better than a random guess, but there is more to the assessment of a machine learning implementation than just the data. For the experimental application that this model was intended for, it works very well, and fairly efficiently with a decent amount of space and time usage. For better performance, however, a better machine and a cleaner and larger dataset are necessary.

Acknowledgment

The group wishes to acknowledge Google TensorFlow and Keras for its important contribution to the open-source community and the field of machine learning as a whole. Through their support and documentation, it was possible to troubleshoot problems and tweak performance with great ease.

References

[1] "Basic Classification: Classify Images of Clothing : Tensorflow Core." *TensorFlow*, Google, https://www.tensorflow.org/tutorials/keras/classification.

[2] "Image Classification : Tensorflow Core." *TensorFlow*, Google, https://www.tensorflow.org/tutorials/images/classification.

[3] "Teachable Machine." *Google*, Google, https://teachablemachine.withgoogle.com/.