Automatic Automobile Taxonomy: Car Identification using a CNN Architecture

Samuel Frishman
Department of Mechanical Engineering
Stanford University
samuel9@stanford.edu

Wilson Ruotolo
Department of Mechanical Engineering
Stanford University
wruotolo@stanford.edu

Abstract

Car identification is critical in a diversity of applications ranging from vehicle appraisal to autonomous driving. We present a deep learning approach to solve this task using a convolutional neural network (CNN) to identify a car's make (brand). The network is trained on the Stanford CARS dataset of over 16,000 images. We ran experiments to test a range of hyperparameters and our final results indicate 88% test accuracy on the 49 class identification problem. Results and causes of the remaining error are explored with an occlusion based feature activation map and class frequency analysis.

Code at: https://github.com/sam-frishman/cs230_project_Frishman_Ruotolo

1 Introduction

Recognizing vehicle make has relevance in several domains(1). For autonomous cars, identification is important in order to estimate surrounding vehicle capabilities and in car appraisal identification largely influences price while on-the-spot estimation is helpful in second hand purchases (2).

Further, brand classification is a critical first step for many other vehicle identification tasks. Assessing model, year, and other characteristics will often benefit from initially grouping by producer, indicating that a reliable automated solution for this step could have additional applications. Also, investigating what core features define a given brand's aesthetics could have interesting implications for automobile design if they can be properly analyzed and quantified.

We investigate this problem with deep learning using a CNN to predict car parameters. The input to our network is a color image of a car and the output is the make (brand) of car. The baseline network contains four conv-pool layers followed by two fully connected layers and a softmax output layer. The algorithm uses an Adam Optimizer with minibatches of size 32. The baseline model was expanded on and tuned to achieve a final accuracy of $\sim$88%.

2 Related Work

Car identification is an exciting area for deep learning with an abundance of relevant work. Liu et al. (3) explore the same Cars dataset with standard networks (e.g. GoogLeNet, CaffeNet, VGGNet) on a similar 196 class problem that further breaks down the vehicle models. The best performance reported was $\sim$80% accuracy using the GoogleLeNet architecture where they trained all the layers from scratch. Though our 49 class challenge is less complex, this gave us an idea of accuracies we should expect to achieve. The paper also provides the results on an SVM classifier which achieved only 3% accuracy.

Rachmadi et al. (4) explore the application of CNNs to identifying vehicle color. This can be a challenging task in poor lighting or partially shaded scenes. They experimented with different color spaces and achieved 95% accuracy. Though a different classification task, some core features carry over to general vehicle identification. With this in mind, we used their architecture as partial motivation for our initial design. Similarly, Simonyan et al. (5) gave us a starting point on filter size (3x3 convolutional filters).

3 Dataset

We use the Cars dataset originally described in (6) that includes 16,185 images of 49 brands of cars spanning from 1991 to 2012. The images are colored and taken in everyday settings. The data set can be found at `https://ai.stanford.edu/~jkrause/cars/car_dataset.html`. We agumented the data by flipping the images for a total of 32,370 images. We split the data into 25,248 train, 6,313 train, and 809 test. The test data was randomly selected from the original dataset before augmentation.

Figure 1 illustrates the breakdown of brands in the full dataset. The large categories are made up of common brands (e.g. Chevrolet, Toyota, Acura, etc.) while the smaller categories are composed of more niche brands (e.g. Hummers, Ferrari, etc.). We pre-processed the images and associated labeling files using a python script to include the brand label in the file name. We then resized the images to both 64x64 and 128x128 pixel resolution for experiments.

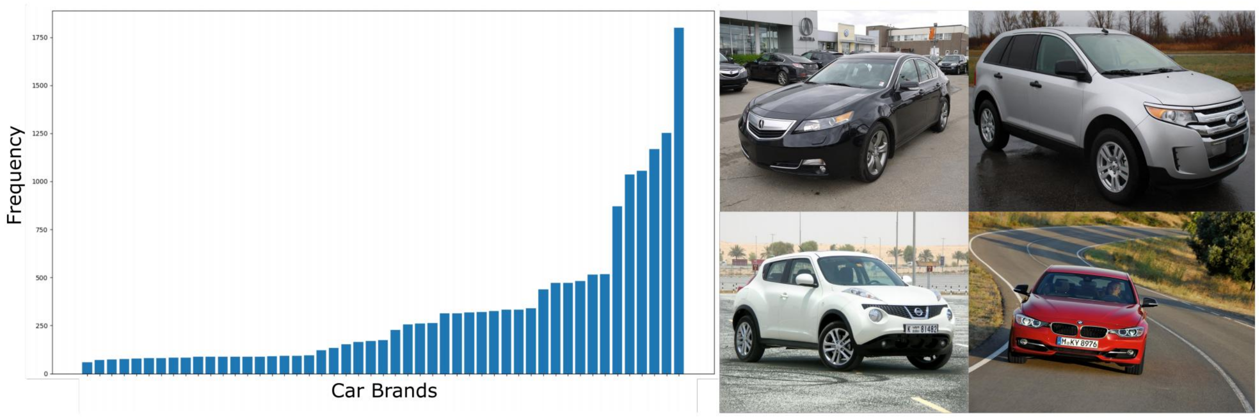

Figure 1: Left: Frequency plot of brands in the dataset. Right: Sample images from the set.

4 Methods

We conducted a variety of experiments focused on hyperparameter tuning. In addition to creating an accurate car identifier, a primary learning goal was to build intuition about what such a tuning process entails, leading us to stick with a relatively consistent architecture rather than testing with the different open source models. Resulting from these two design constraints, the project followed a "panda" like nurturing method of carefully tweaking a single model rather than face the exploding computational cost of running many different models in parallel.

Given the proven efficacy of convolutional neural nets for visual problems, a basic set of two dimensional filters was modified with progressive tuning. A six layer neural net model was based on examples provided for the class. This started with four layers of two dimensional convolutional filters of kernel size three, with a progressively doubling number of channels in each layer and "same" padding. These were followed by two fully connected layers and, finally, a softmax, multiple classification output. Further, each convolution operation was followed by a Relu activation function and then a max pooling layer with both size and stride of two. Batch normalization was used on all trials and learning was run with an Adam optimizer.

Attempting to orthogonalize our tuning process as much as possible, several comparisons were performed against the constant, base model. This structure was run on its own, and then again after data augmentation (horizontal flip) and higher resolution images were used. The accuracy of these models across a 10 epoch training cycles is plotted in the top left slot of Fig. 2. This demonstrates that both methods of increasing available data (data augmentation + increasing resolution) for the challenge did improve performance. Despite this, the larger, 128x128 pixel image training process also took four times as long to run than the 64x64 images, so it was deemed more efficient to tune hyperparameters on the faster, smaller images, and then seek to extrapolate later on in the process to their larger counterparts.

Next, we faced the challenge of optimizing the number of filters, depth of net, and kernel size. While all three of these parameters are important, the first two represent a strictly increasing amount of nodes that (typically) consistently improve performance at the cost of added computational cost. The latter parameter, however, was of more interest because, depending on both application and layer depth, both smaller and larger sizes of filters can be superior. As such, we focused on this parameter for the next set of trials, the performance of which is plotted in Fig. 2, middle. Lastly, larger and smaller learning rates were tested to converge to a sufficiently optimal value of 0.001.

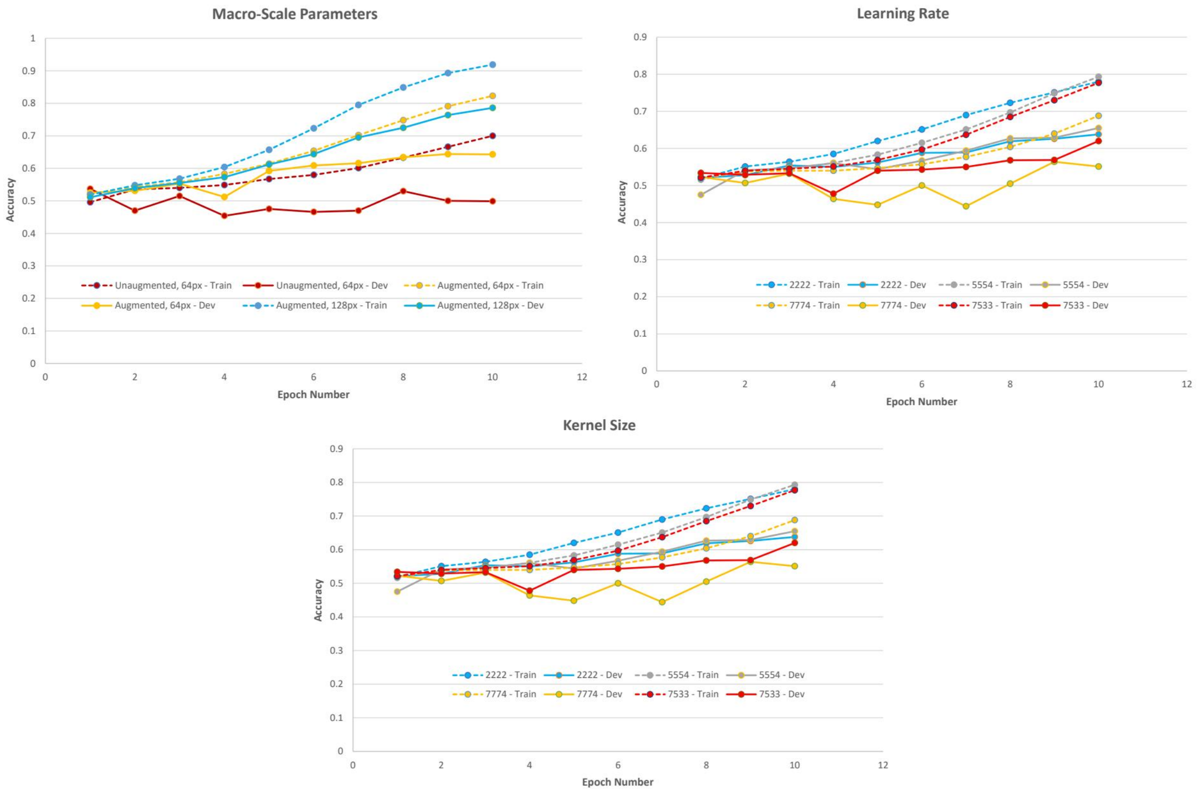

Figure 2: Top Left: Impact of less compressed image size and data augmentation on accuracy. Top Right: Accuracy from a three order of magnitude spread of learning rates tested on the same base model. Bottom Middle: Impact of different kernel sizes on model performance.

Comparing the results from the various tests described above led to a final model of six convolutional neural net layers with kernel size of 3 and stride of 1 in all. The structure is illustrated in Fig. 3. 5x5 filters were shown to be slightly superior in performance on 64x64 images, though the $\approx 1\%$ increase was not worth the significantly longer computation time. The output from these layers was then passed through three fully connected layers and to a softmax classifier. The number of filters in each layer equals $16 * 2^L$ where L represents the integer number of layer depth starting with zero for the very first convolution.

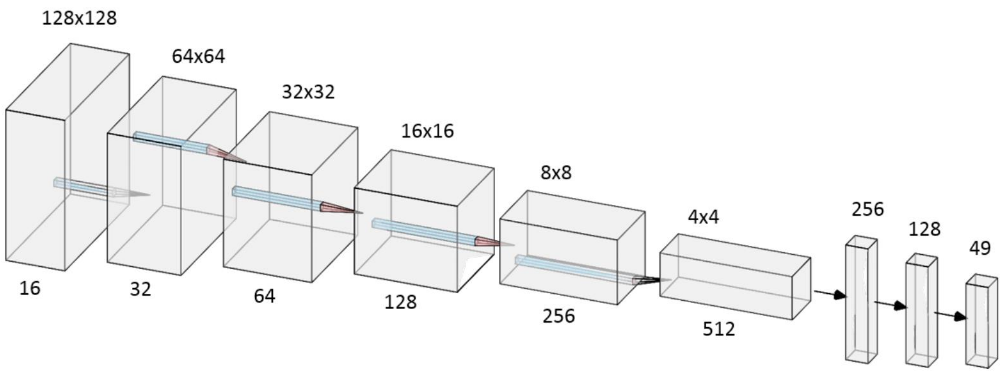

Figure 3: Final convolutional neural net structure shown in AlexNet style (7).

5 Results and Discussion

We ran the model described above for 30 epochs and the error is presented in Fig. 4. Error drops to 0.2 within the first 15 epochs, but the accuracy still continues to improve slowly after that for the remainder of the training, finishing at $\approx 88\%$. Training error, however, can be seen to drop to almost 1% by the end of epoch 30, indicating that overfitting remains a problem.

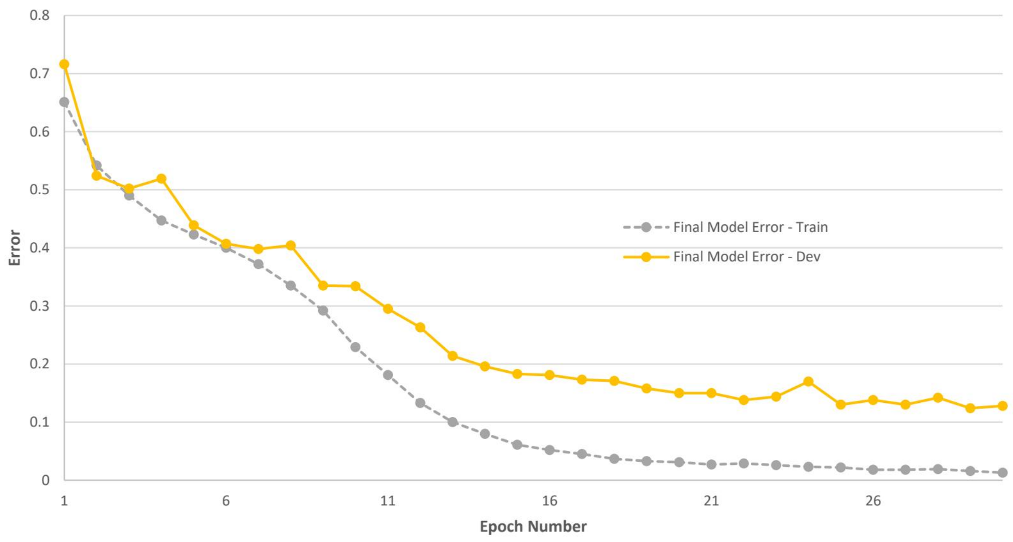

Figure 4: Final error results on the train and dev sets using the tuned model.

To understand the origins of this error further and identify which features the network is focusing on, we conducted a numerical error analysis by classification bin (Fig. 6) and created an activation heat map (Fig. 5) using an occlusion method. A new set of images generated by passing a pseudo-filter of black pixels across a correctly classified image and continually calculating the probability strength for a given image permutation is then plotted to generate a moving average of feature importance. This two dimensional numerical structure was converted to a heat map and is shown in the right image of Fig. 5. The original image is shown on the left, and the two versions overlaid are shown in the middle. From the center visualization, we see that the network has focused on several key components of the vehicle that a human would also use for identification, including the logo, headlight, and cab roof.

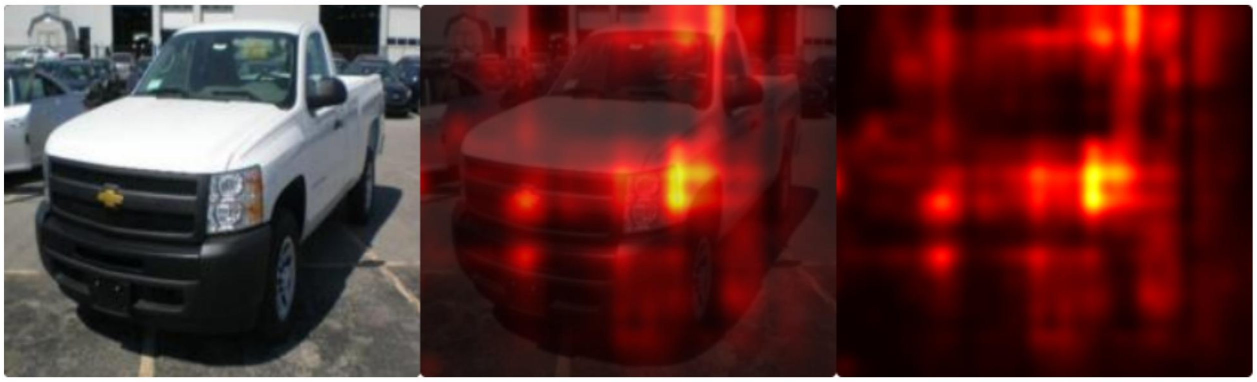

Figure 5: Occlusion heat map of feature importance. Left: Original image. Right: Raw heat map. Middle: Both images overlaid to clarify feature location.

We also sought to identify any large sources of error to help direct the tuning process. To this end, we randomly sampled a subset of the incorrectly classified results and plotted the relative frequency of each classification in these failed examples alongside a normalized version of the original frequency plot for comparison.

Based on this data, it appears that a disproportionate amount of error comes from the classes with fewer training examples. Particularly, the middle band of the frequency plot makes up a large part of the total error, for the very rare cars are so infrequently encountered that they do not contribute much error, and the more common cars have the extra data necessary for a more complete training process.

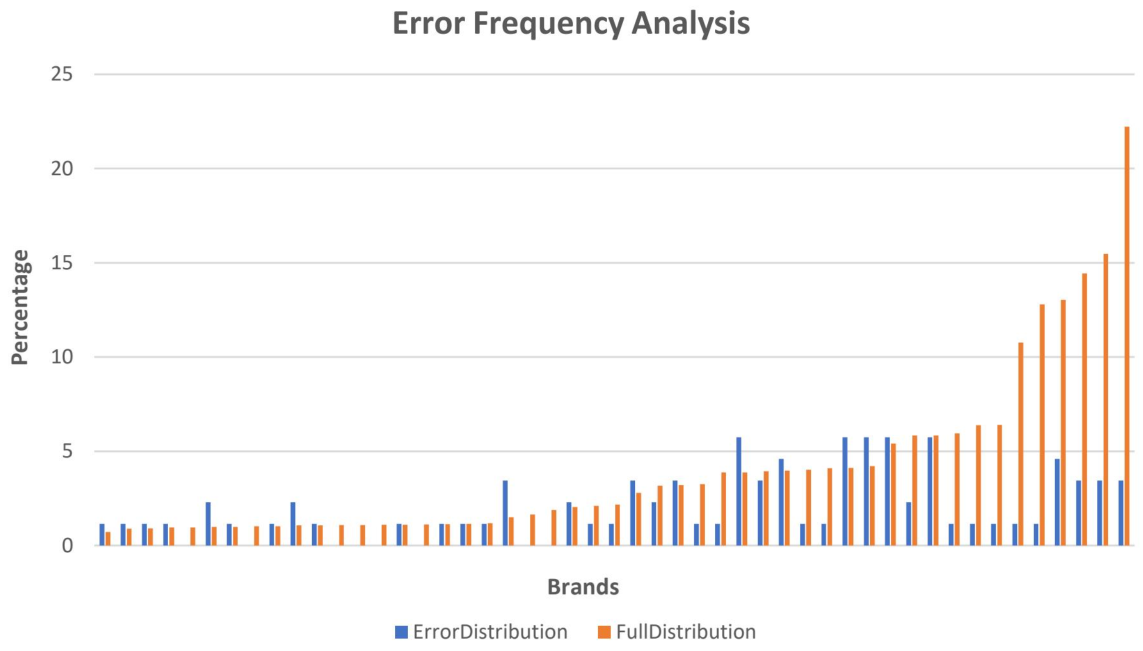

Figure 6: Comparative frequency analysis of full data set and error occurrence by make classification. Both results are normalized to a percentage of total occurrences.

6 Conclusion and Future Work

Given that the similar work presented in (3) achieved a maximal performance of $\approx 80\%$, these initial tuning efforts resulted in improved accuracy. However, (3) were also focused on the larger problem of identifying both make and model of vehicle, which entails a necessarily larger number of possible classes. To both improve our model's accuracy as well as its application range, it would be necessary to eventually expand to a finer grained classification as well. This could be accomplished with a similar end-to-end approach as used in this work, or the model and weights produced herein could serve as the initial components of a transfer learning method that eventually sub-classifies each brand into its constituent vehicle models.

From the heatmap analysis, we were excited to see that the algorithm was using features we would expect. We found it interesting that the brightest spots were on the headlights, indicating that there may be a lot of variance in car maker headlight design and that the algorithm could not rely on the car emblem as it is often not visible in the images. We realized that such heatmaps could be used to identify which features are unique to different car makers and what defines a particular brand. This information could further be used to inform design decisions and provide a quantifiable metric for defining a brand's style.

Improvements to performance could be achieved by adding more filters and depth to the current model and then running the result on less compressed images. Such changes were consistently shown to be very effective, and were only limited by computational time.

The error analysis (described above) showed that vehicles less represented in the dataset were resulting in a disproportionate part of the error, To deal with the middle sized categories (those large enough to contribute significant error to the total but too small to have been trained robustly) acquisition of more data seems the most viable approach since they are still numerous and such data is more easily available. To deal with the very rare vehicles such as Hummers and Ferraris, it makes more sense to ultimately lump all of these less likely classes into one macro group. Ironically, these more atypical vehicles are much easier for humans to identify because of their remarkable differences from the majority of the cars on the road, meaning they could be easily flagged and passed on for a human to identify manually.

Lastly, consultation with automobile experts could provide an improved knowledge of Bayes error for this classification challenge that can help guide improved regularization efforts as well as help set a minimum pixel number for input images.

Contributions

Sam Frishman was primarily responsible for data preprocessing and augmentation, as well as generating partially occluded images for our heat map. Wilson Ruotolo focused on hyperparameter tuning and error frequency analysis. Both team members contributed to all components of the work in some capacity, and worked together strongly to generate the core base model and associated code as well as idea generation, poster preparation, and paper writing.

References

[1] K. Noor and S. Jan, "Vehicle price prediction system using machine learning techniques," *International Journal of Computer Applications*, vol. 167, no. 9, 2017.

[2] A. İşeri and B. Karlık, "An artificial neural networks approach on automobile pricing," *Expert Systems with Applications*, vol. 36, no. 2, pp. 2155–2160, 2009.

[3] D. Liu and Y. Wang, "Monza: image classification of vehicle make and model using convolutional neural networks and transfer learning," 2017.

[4] R. F. Rachmadi and I. Purnama, "Vehicle color recognition using convolutional neural network," *arXiv preprint arXiv:1510.07391*, 2015.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[6] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," 2013.

[7] A. Romero, N. Ballas, S. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Imagenet classification with deep convolutional neural networks," in *International Conference on Learning Representations*, 2015.