Automatic Automobile Taxonomy: Car Identification using a CNN Architecture

Samuel Frishman, Wilson Ruotolo
{samuel9, wruotolo} @ stanford.edu

CS230 Autumn 2018

Predicting Cars

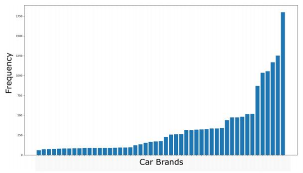
Car identification is critical in a diversity of applications ranging from vehicle appraisal to autonomous driving [1, 2]. We use a convolutional neural network (CNN) to identify a car's make (brand). The network is trained on the Stanford CARS dataset which includes over 16,000 images [3]. We ran experiments to test a range of hyperparameters and our final results indicate a ~88% test accuracy on the 49 class identification problem.

Data

Cars Dataset: 16,185 color images of cars photographed in a variety of everyday settings. We augmented data by flipping images for a total of 32,370 images. The following breakdown was used:

- Train: 25,248
- Dev: 6,313
- Test: 809 (randomly selected from images before augmentation)

The data contained 49 different car makers. The histogram shows the frequency breakdown for each brand.



Images/Features

The inputs for this task are either 64x64 or 128x128 pixel, three color channel images. An end to end learning approach was used in lieu of automatic feature detection algorithms.

Models and Results

We tested a variety of CNN models to understand the impact of different hyperparameters. The most notable experiments are shown below. For all tests we used softmax cross entropy loss and the following structure with a stride of 2. The number of filters increased with depth by a factor of 2 starting at 16. We used mini batch sizes of 32 with batch normalization.

[CONV2D ⟶ RELU ⟶ MAXPOOL] x4 ⟶ FC x2

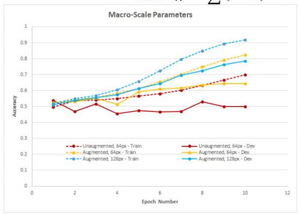
$$\sigma(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

$$L_{cross\ entropy} = -\sum y_j * \log \hat{y}_j$$

Parameters:

- Learning rate: 0.001
- Kernel Size: 3

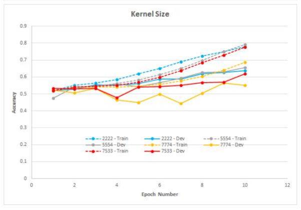
Takeaways: Data augmentation and increased image size are very helpful. The latter is computationally expensive.



Parameters:

- Learning rate: 0.001
- Kernel Size: Varied

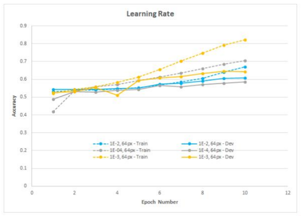
Takeaways: 5x5 filters were slightly more effective, but smaller filters still work as well to within 2% accuracy.



Parameters:

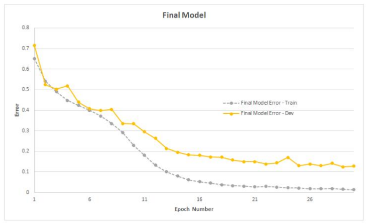
- Learning rate: Varied
- Kernel Size: 3

Takeaways: 0.001 is a reasonably optimal learning rate for this application.



Discussion

Using the takeaways from our preliminary results, we opted to add another CONV2D layer with 512 filters, and reverted back to the highly robust kernel size of 3. This expanded net on 128x128 pixel images reached a peak accuracy of ~99% and ~88% on the train and dev sets respectively.



Previous work on the same data set [5] achieved best results of ~80% accuracy using GoogleLeNet on a similar 196 class problem that further breaks down vehicles by model. Though our 49 class challenge is less complex, this indicates, our 88% accuracy is quite good given the resources available.

Future Work

- Data Augmentation was very effective → More data from scraping the internet will be a good next step
- Limited computational power → Try larger neural nets on larger image sizes
- Slight overfitting problems emerged → Increase regularization

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. 1¸seri and B. Karlık, "An artificial neural networks approach on automobile pricing,"Exper tSystems with Applications, vol. 36, no. 2, pp. 2155–2160, 2009.
[3] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," 2013.
[4] R. F. Rachmadi and I. Purnama, "Vehicle color recognition using convolutional neural network,"arXiv preprint arXiv:1510.07391, 2015.
[5] D. Liu and Y. Wang, "Monza: image classification of vehicle make and model using convolutionalneural networks and transfer learning," 2017.

We would like to thank the CS230 course staff.
Code Repository: https://github.com/sam-frishman/cs230_project_Frishman_Ruotolo (code adapted from cs230-code-examples)