Classifying Dog Breeds: Performance Analysis of Frequently Used CNN Models

Mingming Lang, William Lee, Mac Stark, Han Wang

1. INTRODUCTION

Image recognition is a classic problem in computer vision in which the goal is to identify objects in images. This is a difficult task for computers as an object looks different based on lighting, point of view, and its surrounding environment. Among multitude of image recognition strategies, Convolutional Neural Network (CNN) is frequently used for image recognition. CNN is a class of deep neural network that consists of an input layer, an output layer, and hidden layers, where hidden layers are sequential combination of convolutional, activation, pooling, fully connected, and normalization layers.

Many CNN architectures have been developed using different structures of layers. This project implements four frequently used CNN models VGG 16, VGG 19, Inception-V3, and InceptionResnet-V2 to train a model that categorizes dogs into different breeds.

2. METHOD

We obtained training data from the Stanford Dogs Dataset [1], which includes 120 kinds of breeds, and 20,580 images of dogs with their breeds as labels. For training time feasibility, we reduced the sample to 10 breeds of dogs and 1,789 images. Then, we trained the data set on four CNN architectures described below using Keras with trained parameters on ImageNet [2]. Images were preprocessed prior to training to meet each input requirement. VGG 16, Inception-V3, and InceptionResnet-V2 were run on NVidia K80 GPUs on Kaggle cloud and VGG 19 was run on NVidia GTX 1080 on UNC Longleaf cluster.

2.1 VGG

VGG 16 and VGG 19, consisting of 16 and 19 weight layers respectively, placed first and second in the image classification task of Large Scale Visual Recognition Challenge (IISVRC) 2014 [3]. By using an architecture with small 3 x 3 convolutional filters, VGG pushed the depth to 16 - 19 layers [4]. This was considered very deep in contrast to prior CNN models as IISVRC 2012 winner (Alexnet) and IISVRC 2013 winner only had 8 layers [3]. VGG is characterized by architectural simplicity, but at the cost of high computation time [6].

2.2 Inception-V3

The inception network is a series of CNN classifiers which avoid repeatedly stacking convolution layers as previous classifiers did, and thus considerably improve the speed and accuracy of the inception classifiers. Inception-V3, along with Inception-V2, is considered as a big improvement of the previous version, Inception-V1. It factorizes the traditional 7 x 7 convolution into three 3 x 3 convolutions, and uses RMSPProp optimizer and batch-normalization to improve accuracy [6]. As suggested in [6], we expected that Inception-V3 would have higher accuracy than VGGNet.

2.3 InceptionResnet-V2

Inception ResNetV2 is a convolutional neural network which, like VGG16, VGG19, and ResNetV2, is trained on more than one million images from the ImageNet database. Unlike ResNetV2, Inception ResNetV2 uses Inception image pre-processing, and has a default image input size of (299, 299). With 781 layers and 153,700 trainable parameters, compared to VGG16, which has 4 layers and 5,130 trainable parameters, Inception ResNetV2 is an incredibly complex CNN that is capable of classifying images into up to 1,000 different categories.

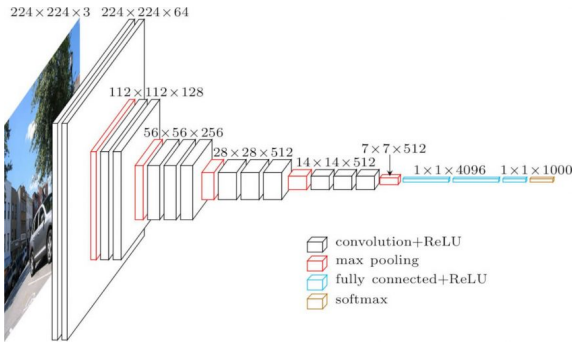

Figure 1: VGG architecture with 16 layers [5]

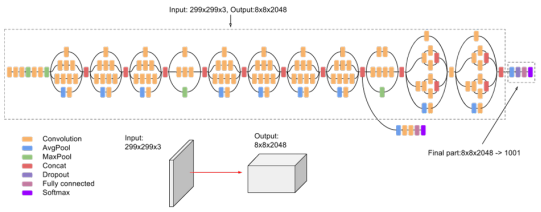

Figure 2: Inception-V3 Architecture [7]

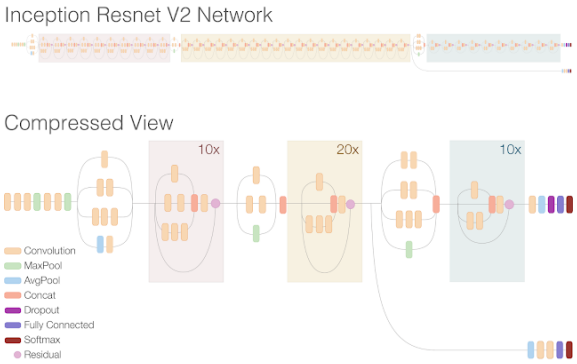

Figure 3: Inception Resnet - V2 Architecture [8]

3. RESULTS

3.1 Accuracy and Loss

3.1.1 VGG 16

We trained the model for 20 epochs, 1000 steps per epoch, and 1000 validation steps. Each epoch took approximately 73 seconds.

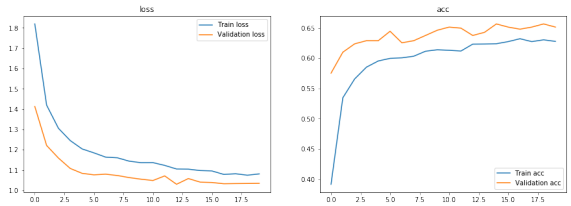

Figure 4: VGG 16 Accuracy and Loss

VGG 16 reached highest validation accuracy 65.12 percent at epoch 19. We observe that oscillations in the graphs were negligible with respect to overall growth (which is not true for the other architectures below), so we suspect that running more epochs would have increased the validation accuracy.

3.1.2 VGG 19

We used the same parameter as VGG 16, but trained for 10 epochs instead of 20. Each epoch took approximately 577 seconds (Recall that VGG 19 was tested on different architecture).

VGG 19 reached highest accuracy 77.64 percent. We observe that validation accuracy grew very rapidly in contrast to VGG 16. In particular, VGG 19 had 14 percent higher

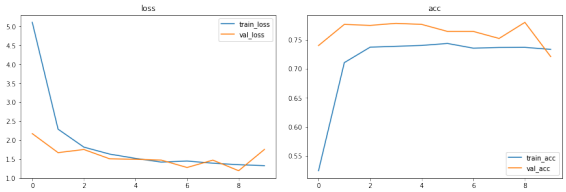

Figure 5: VGG 19 Accuracy and Loss

accuracy than VGG 16 at only 5 epochs. VGG 19 test accuracy was 77.64 percent at epoch 5, compared to 63.77 from VGG 16 at epoch 20.

3.1.3 Inception-V3

Applying this model to ten breeds, we acquired 20,490 trainable parameters. We trained the model for 30 epochs, with 1000 steps per epoch, and 1000 validation steps. Each epoch took approximately 75 seconds, and it achieved the highest accuracy 55.88 percent after the 29th epoch. Additionally, it reached 55.57 percent accuracy after the 10th epoch and has not improved significantly since then.

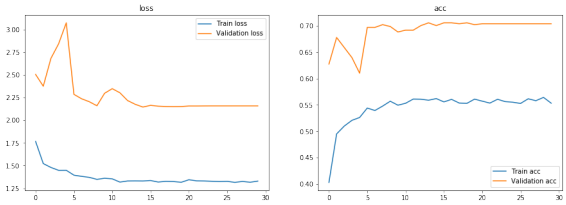

Figure 6: Inception-V3 Accuracy and Loss

3.1.4 InceptionResnet-V2

We chose Inception ResNetV2 as a model that we thought would have the highest correct classification rate, and anticipated it to easily outperform the other three CNN models that we trained. However, the incredibly computational complexity of training this model impeded our ability to fully train this model to its potential. The VGG-16 model was capable of training on an Intel Core i7 8th Generation processor in around 2 hours. Despite leveraging the computational power of several GPUs to train the Inception-Resnet V2 model, we were not able to train the Inception ResNetV2 model to meet the correct classification rates to any of the other aforementioned models. We made the decision to dedicate our computational resources and time to training, tweaking, and retraining the VGG16, VGG19, and ResNetV2 models. For this reason, we do not include our results for the Inception ResNetV2 model in the graphs and data, as we feel that with the proper hardware and time this model would have exhibited drastically higher classification accuracy than we were able to achieve.

3.2 Comparison

Table 1 summarizes accuracy and loss at the final epoch for each architecture.

Model	Epoch	Training		Validation	
		acc	loss	acc	loss
VGG 16	20	0.636	1.070	0.643	1.234
VGG 19	10	0.733	1.328	0.721	1.756
Inception-V3	30	0.549	1.342	0.704	2.158

Table 1: Training and Validation Data for Each Model

4. CONCLUSION

In this project, we implemented VGG 16, VGG 19, and Inception-V3 to design a dog breed classifier. We observe that VGG 19 was the most effective in classifying dog breeds. Not only does it have the highest training and validation accuracy, it also reached them in fewest epochs. However, we note here that runtime per each epoch was unusually high considering that it has analogous structure to VGG 16 but took approximately 8 times more time. We suspect that there was a misallocation of GPU resources in the UNC longleaf cluster.

Although we expected Inception-V3 would have higher accuracy than VGG, VGG19 turned out to be more effective. In particular, we see that the training accuracy was 0.549, which was lower than both VGG models. Validation accuracy, however, was comparable to VGG 19. For future investigation, we suggest that Inception-V4 and Inception-ResNetV1 may lead to better breed classification results, because both classifiers have more uniform modules and have reduced complicated modules to boost performance.

There are several possible improvements to our project. First, each breed only had 170 images in average. Acquiring more images for each breed would have increased model accuracy. Next, we can improve the sample breed size, perhaps up to all 120 provided breeds. This would in turn increase the computational time dramatically. Increasing sample size would also improve the quality of performance benchmarks.

5. REFERENCES

[1] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao and Li Fei-Fei. Novel dataset for Fine-Grained Image Categorization. First Workshop on Fine-Grained Visual Categorization (FGVC), IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011.

[2] Chollet, Francois. Keras. https://github.com/fchollet/keras, 2015.

[3] Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. IJCV, 2015.

[4] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015.

[5] Hassan, Muneeb ul. VGG16 - Convolutional Network for Classification and Detection. VGG16 - Convolutional Network for Classification and Detection, 21 Nov. 2018, neurohive.io/en/popular-networks/vgg16/.

[6] Szegedy, Christian et al. Rethinking the Inception Architecture for Computer Vision. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016): 2818-2826.

[7] Running Inception on Cloud TPU. Google Cloud. Google, cloud.google.com/tpu/docs/tutorials/inception.

[8] Improving Inception and Image Classification in TensorFlow. Google AI Blog, 31 Aug. 2016, ai.googleblog.com/2016/08/improving-inception-and-image.html.