Object Recognition and computer vision
Assignment 3: Image classification

Dadoun Hind
hind.dadoun@ens-paris-saclay

Abstract

The problem we have at hand is to train a model to classify 20 species of birds. We have about 55 training images for each bird. There are usually less than 8 validation images for each class and finally 517 test images. This is a reasonably small dataset and trying to train it from scratch using a hand made convolutional neural network performs very poorly.

1. A first glance at the data

The majority of the pictures in the training data set are very clear in the sense that we can see the bird entirely. On the contrary, in the test dataset, we have pictures of birds flying, or at the top of a tree where even a human eye couldn't distinguish them. On top of that some of them are very noisy. I decided to start by applying data augmentation methods on the training data set to obtain different orientation, location, scale, brightness etc using : Horizontal Flip, Random Crop and Rotation. This also allows me to enlarge my data set and get better performance.

2. Choosing a torch vision model

One of the most common datasets that are available for image classification is the Imagenet dataset. Which is why I decided to use the torchvision models, all of which have been trained on this 1000-class dataset. Based on an article (Krassimir Valev *et al.*) discussing the evaluation of recent deep learning architectures, I tried 3 models discussed in the paper: VGG Networks, Dense Networks and Residual Networks and finally decided to take Residual Networks which seemed to perform better on the task in hand. For all models I had to reshape the network because the final layer of a CNN model has the same number of nodes as the number of output classes in the dataset (1000) while our dataset only had 20 classes.

3. Tuning parameters

The authors of the article also said that the very deep ResNet-152 has so many parameters that with small amount of training data, if we start with a random initialization of the weights we may not arrive at a meaningful point in the objective function. They proposed to use fine-tuning (Initialization with ImageNet weights and training at a lower learning rate) which allowed me to further improvement in accuracy. I also tried feature-extraction which only updates the final layer weights from which we derive predictions and I found out I had better results using only fine-tuning. Finally, I tried different optimizers (SGD,Adam,ASGD..) and even though most literature stated Adam performed better, I had better results using SGD and changing the learning rate and momentum. Momentum is a method that helps accelerate SGD in the relevant direction and dampens oscillations ,the momentum term is usually set to 0.9. As for the learning rate I used Learning rate schedules which tries to adjust the learning rate during training byreducing the learning rate according to a pre-defined schedule or when the change in objective between epochs falls below a threshold.

4. Results

I achieved a test accuracy of 0.20% with the script given by the instructors,I managed to increase it by data augmentation. Then when I tried pre-trained models I realized a 0.57% score using VGG-16, 0.66% using Densenet-121 and 0.70% using Resnet-152. Finally tuning parameters allowed me to get 0.76% accuracy.

5. References

Krassimir Valev *et al.*: "A Systematic Evaluation of Recent Deep Learning Architectures for Fine-Grained Vehicle Classification".
Nathan Inkawhich : "Finetuning torch vision models tutorial", from which I based the majority of my script.