HW 3

Cohen Ethan

Abstract

We perform birds image classification on a subset of Caltech-UCSD Birds-200-2011 dataset without any external annotation. We train a CNN from scratch to establish a baseline, and then substantially improve our results with transfer learning.

1. Introduction

The dataset we were supplied with is a subset of the original Caltech-UCSD Birds-200-2011 dataset, reduced to 20 classes, instead of the original 200 bird species. There are 1082 training images and 103 validation images.

2. Preprocessing

The pictures have different shapes and within a given species, birds exhibit varying sizes and poses. There are also significant changes in illumination and background. We have also observed occlusion by twigs or leaves. Given the size of the dataset, it was necessary to perform some data augmentation beforehand. We simply used the transforms `RandomRotation` and `RandomHorizontalFlip` available in Pytorch to make our networks more robust.

3. Baseline and Home-made CNN

To create a baseline, we began by training the model with the baseline given by the instructors. We had an accuracy of 14% with that baseline on kaggle so we decided to go through an other ”Home-made” CNN inspired by the tutorial on CNN by Medium . We trained a CNN by stacking 14 convolutional layers, each one followed by batchnormalization and ReLU. We also added some pooling layers, and the network ends on a linear layer. The CNN was trained using stochastic gradient descent and the number of epochs was chosen manually. There were clear generalization issues as the validation accuracy was significantly lower than the training one, so we added some dropout to mitigate overfitting.

Training	Test	Kaggle
61%	48%	34%

Table 1: Accuracy for our CNN

4. Tuning Parameters and Transfer learning

Since the dataset is quite small, training a network from scratch is not the best thing to do. Instead we imported pretrained model(Resnet-152) with elaborate architectures and modified their last linear layer to fit our needs. 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. We tried feature-extraction which only updates the final layer weights from which we derive predictions and we found out we had better results using only fine-tuning. Finally, we tried different optimizers (SGD,Adam,ASGD..) and even though most literature stated Adam performed better, we 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. We then try to use a model compose with three networks : resnet152, inception et vgg19. We trained it by using with AdaBound and SGD. By doing the same transformation than we did with resnet and using SGD optimisation, we managed to get 79% on Kaggle which is our best score so far.

	Training	Test	Kaggle
Resnet-152	92%	90%	76%
Multi model	89%	89%	79%

Table 2: Accuracy using transfer learning