HW 3

Gabriel ROMON

gabriel.romon@ensae.fr

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.

While birds are always in the focus of the picture, there may be larger objects also in the foreground, such as the trunk of a tree. This motivates the need for segmentation of each image. Even though bounding boxes are available in the original dataset, they were not provided for the competition. To achieve segmentation, we have experimented with edge detection techniques (canny detector followed by dilatation and erosion), as well as histogram-based (Otsu binarization) and color-based techniques (color spaces). While these approaches work well on individual images after some parameter tuning, they do not generalize well. We also contemplated using a pretrained Mask R-CNN model, but we did not have enough time to implement it.

We gave up on segmentation and decided to use deep learning methods. 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. Home-made CNN

To create a baseline 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	Validation	Kaggle
71%	50%	51%

Table 1: Accuracy for our CNN

4. Transfer learning

Since the dataset is quite small, training a network from scratch is not the best thing to do. Instead we imported pretrained models with elaborate architectures and modified their last linear layer to fit our needs. These models were originally trained on Imagenet, which has a nonempty intersection with Birds-200-2011, so results should be taken with a grain of salt.

	Training	Validation	Kaggle
Resnet-152	91%	90%	83%
Densenet-161	90%	89%	73%
PNASNet-5-Large	92%	93%	80%

Table 2: Accuracy using transfer learning

These models could be stacked to get a model with better accuracy and less variance, but we did not have time to implement this.