Homework 3

Guillaume Petit
g.petit98@gmail.com

November 26, 2019

Abstract

In this Kaggle project, we will try to perform a CNN to classify several bird species from the Caltech-UCSD Birds 2010-2011 database.

1 Introduction

Our Caltech database is here reduced to about twenty species, instead of the original 200. Our training base is composed of 1082 images and the validation base of 103. We will have to try several methods to fill the small size of our training base, which prevents our network from training well.

2 Preprocessing

After importing the data, we observe that the photos are of different sizes, the birds are in different positions, the background varies and sometimes elements partially hide the birds (and can therefore have an influence on the capacities of our network). We therefore decide to segment with YOLOv3 trained on COCO. Despite extensive documentation, this technique has never been effectively implemented on Pytorch. Another idea was to go through Tensorflow because they have an implemented function that allows to cut the photo at the segmentation level, however the tensor being built differently, this complicates the task to submit my network on kaggle. To try to avoid this complex code, we attempt to truncate the images. However, we notice that several images show a truncated bird. Therefore, we do not truncate. We therefore decided to increase our training database and enlarge the images with pytorch to improve the results of our CNN.

2.1 Home-made CNN

At first glance, I still decide to create my own neural network with the help of several resources so that it is developed enough. Nevertheless, the results are already disappointing on our validation data. We abandon the idea of creating our own network from scratch.

2.2 Transfer Learning

For this part, we decide to use networks renowned for their image recognition efficiency (googlenet, resnet152, vgg16), pre-trained on Imagenet which has no link with our database. We decide to freeze all the weights except the last layer and modify the number of ouputs to match our twenty classes. The following results obtained are:

Network	validation	Kaggle
vgg16	80%	66 %
resnet152	84 %	70 %
googlenet	75 %	60 %

We now consider only resnet152. We decide to go against the theory of transfer learning and unfreeze the weights. We think that these weights will be perfect for initialization and that the adjustments made will be specifics to our images. This is the result :

Network	validation	Kaggle
resnet152	95 %	78 %

Unfortunately, our failures at the preprocessing stage and time prevent us from getting any better.