# Object Recognition and Computer Vision 2019/2020

Aicha BOUJANDAR
ENS Paris-Saclay & ENSTA Paris
aicha.boujandar@ensta-paris.fr

## Abstract

*The Kaggle competition "Object recognition and computer vision 2019/2020" consists on producing the model that gives the best accuracy on a test dataset containing the same categories as a subset of Caltech-UCSD Birds-200-2011 dataset. Since the images contained in this dataset overlap with the ImageNet dataset, I have used the transfer learning approach with the pretrained models available in Pytorch to produce the classification model.*

## 1. Transfer learning

The transfer learning approach consists on using models which were trained on datasets like ImageNet, Coco,.. in classification problems within other datasets instead of building models from scratch. As set in [1], this approach's main points are : Data augmentation and progressively unfreezing layers in a pretrained model.

## 2. Data augmentation

The training data contains 1087 images, while the validation data contains 103 images, which represents a small dataset. Thus, and in order to avoid overfitting problems, I have added different transformations to the training and the validations images using torchvision.transforms. For the training data : RandomHorizontalFlip, ColorJitter, Resize(224) and for the validation data Resize(224), CenterCrop(224).

## 3. Features extraction from pretrained models

There exists a large panel of pretrained models on the ImageNet dataset : [VGG, ResNet, DenseNet, ..]. Each of this models has its own variants. The pretrained models can be used in two main ways : using only the model's architecture and learn all its parameters, or keeping the first layers' parameters and learning only the last layers' parameters, and progressively unfreeze some layers' parameters and learn them in order to better fit the data. The second approach is justified by the fact that the first layers extract general features of the images (contours, edges,..) while the last layers extract specific features of the classes.
The ResNet50 model, which I have used in my model, contains 10 main layers : conv1, bn1, relu, maxpool, layer1, layer2, layer3, layer4, avgpool and fc. The last layer (fc) is a linear layer whose output's dimension is 1000 since the model was trained on ImageNet dataset. Thus, I have started by changing the output's dimension of this layer to the number of classes of birds dataset (20 classes), then I have started by only learning its parameters and keeping the others layers parameters unchanged with a learning rate = 0.01, momentum = 0.9 and a learning rate scheduler of parameters : gamma=0.1 and step 8. In order to improve the model's performance, I unfroze the layers 4 and 3 as well, with respective learning rates 0.01 and 0.001 and with the same scheduler as fc.
Using smaller learning rates as I unfreeze more layers is justified by the fact that as I mount towards the first layers, I learn generic features, so the parameters should not be drastically modified.

## 4. Results

By using ResNet50, and unfreezing layers fc and layer4 with lr=0.01 and layer3 with lr=0.001 both with the scheduler that I mentioned before, I have obtained 99% accuracy for the training data, 86% accuracy in the validation data, and 77.419% accuracy in the test data. And by using ResNet101 with the same parameters and an additional training data transformation (RandomResizedCrop(224)), I have obtained 93% accuracy on the training data, 91% accuracy on the validation data, and 76.774% accuracy on the test data.
The main problem that I have encountered are overfitting problems since my models are not able to generalize. Thus, I have tried to add other transformations in order to make them learn on a varied images, but the models either fail to fit the training and validation data, or they can not generalize for the test data.

## References

[1] Will Koehrsen. transfer learning with convolutional neural networks in pytorch, November 26th, 2018.