Bird Recognition Challenge

Idriss MGHABBAR

1 Introduction

From the engineering point of view, the purpose of computer vision is to achieve similar tasks than the human visual system.

To this end, convolutional neural networks are often used as a deep learning architecture given their high performance.

In this project, the goal is to classify bird images. The models built will be fine-tuned based on their accuracy.

2 Dataset

We will work on a susbset of Caltech-UCSD Birds-200-2011 dataset.

Dataset	Size	Number of classes
Training set	1082	20
Validation set	103	20
Testing set	517	–

Actually, some images in this dataset overlap with images in ImageNet. We need then to be cautious when using pretrained neural networks on ImageNet. This will be dealt with in the preprocessing (data augmentation) and the learning steps (retraining of layers).

3 Pre-processing

In order to deal with the previous problem and to avoid overfitting models, we will perform data augmentation techniques on the training images.

- **Resize** We resize to 224*244 in order to use the pretrained neural networks on Pytorch.
- **Random Horizontal Flip** We perform a horizontal flip on the image with a probability of 0.75.
- **Random Vertical Flip** We perform a vertical flip on the image with a probability of 0.5.
- **Random Affine** We perform a random affine transformation using the following parameters: Degrees = [-30,30] - Translate = [0.15,0.15] - Scale = 0.5,2
- **Normalizing** We normalize the tensor of the image to zero mean and unit variance.

4 Model

The model that enables me to get the best score is VGG19 using convolution layers (filters 3x3), max-pooling layers (2x2). We added two dense layers (25088,2048) and (2048,64) and eventually a classifier layer.

In addition to performing data augmentation to avoid overfitting and cheating, we will freeze the first three blocks of layers and retrain the rest.

We've chosen to freeze the first four convolution layers for two reasons:

- Their purpose is to create simple features like edges. Hence, we can keep weights generated from ImageNet dataset.
- Computation time : We only need to backpropagate the gradient for the next layers.

5 Results

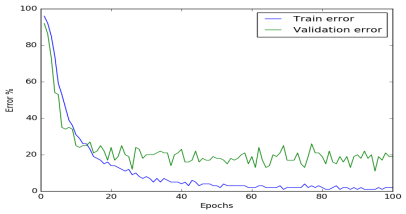

Figure 1: Train/Validation errors on 100 epochs

6 Conclusion

We will use the model built at the 64th epoch as it results in the first minimal of both the training error and the validation error.