RUTGERS ELECTRICAL AND COMPUTER ENGINEERING DEPARTMENT

Fine-tune Pre-trained Deep Neuron Network

Hua Liu
April 21, 2017

1 INTRODUCTION

OBJECTIVE In this project, we fine-tuned the Alexnet and VGG-16 pre-trained networks on the Labeled Faces in the Wild (LFW) dataset by constructing a siamese rectification network and training it with image pairs. In the training phase, The siamese network uses a contrastive loss function to learn a model in Caffe.

We compared the performance of Alexnet and VGG networks with and without fine-tuning. Specifically, to measure the performance of the networks, we extracted the fc8 layer output as transformed image features and plotted the ROC curves.

2 FINE-TUNING IMPLEMENTATION

TEST PRETRAINED DNN

- Download the .prototxt and the pretrained weights .caffemodel file
- Download the LFW data
- In pycaffe interface, load the pre-trained net and feed the input images(.jpg)
- Extract fc8 features, and plot the ROC curve.

CONSTRUCT THE SIAMESE NETWORK FOR FINE-TUNING

- Modify the training prototxt file
  - Replicate the original net work to create two identical branches.

  - Change the input layer to accept paired data. And followed by a slicing layer to extract input data for the two branches separately.
  - Remove the last layer (i.e. softmax layer), and use Contrastive Loss layer instead to take fc8 features and labels from two branches as input and produce loss.

- Generate the initial weights (i.e. .caffemodel file) for the siamese network by replicating the original pretrained weights.

FINE-TUNING

- Generate the LMDB file from the LFW image data
- Set the generate LMDB file path in training prototxt
- Decrease the base learning rate in solver file. Because the fine-tunning process needs a way smaller learning rate than the original one.
- Train the siamese network using caffe command line interface (i.e. CAFFE-HOME/build/tools/caffe). Use -solver to specify solver and -weight to specify initialization wight which is generated in the previous step.

TEST THE TUNED NETWORK

- Extract one branch from the siamese network and generate the trained .prototxt and .caffemodel file.
- Use the same way as testing the pretrained DNN to test the fine-tuned DNN.

3 EXPERIMENT AND RESULT

EXPERIMENTS DESIGN We compared the performance of Alexnet and VGG networks with and without fine-tuning. Specifically, to measure the performance of the networks, we extracted the fc8 layer output as transformed image features and plotted the ROC curves.

RESULTS OF FINE-TUNING are shown in Figure 3.1. The left figure is for the Alex net and the right one is for the VGG net. The orange lines are the ROC curve of pre-trained models and the black lines represent the ROC curve of the fine-tuned models. Findings:

- From Figure 3.1, it is shown that, for both Alex and VGG net, the the fine-tuned versions (i.e. the black line) perform slightly better than the original ones.
- The Alex net performs slightly better than the VGG net on our chosen data.
- Learning rate is critical to the performance. While a too small base learning rate could result in slow converging, a base learning rate which is too large could result in loss explosion. This is because the optimization process is a non-convex optimization

problem. And there even be noncontinuous point around the initial weights. **So in the fine-tunning optimization, the learning rate should be way much smaller than the original learning rate in the pre-trained model. And choosing the proper learning rate is more of an art than science.**

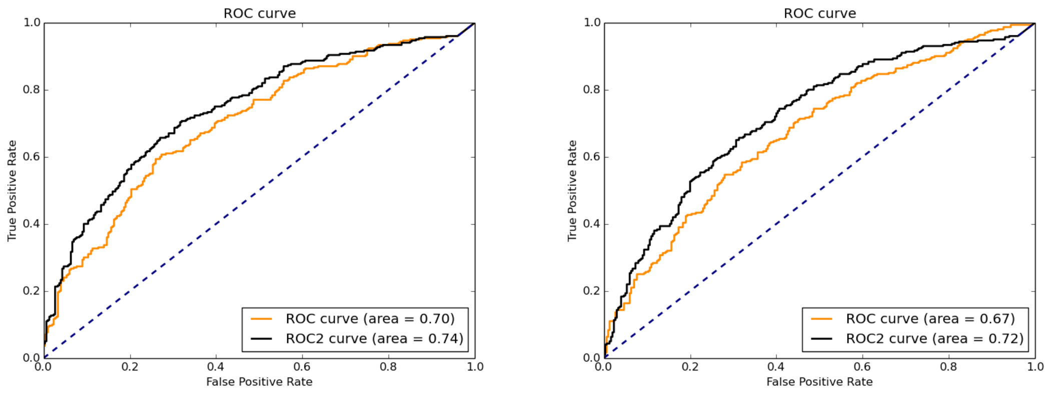

Figure 3.1: Eigenfaces algorithm accuracy measurement. Left figure: accuracy against training size p. Right figure: accuracy against total classes