Course: DD2427 - Exercise Set 4

Due date: 06/05/2016
Cut-off date: 07/06/2016

In this exercise you will investigate the effectiveness of a ConvNet image feature to represent an image. The exercise is very similar to Exercise 3 and you will re-use the code you wrote in that assignment to find the hyperplane, using SGD to optimize the SVM cost function, that discriminates between images of containing people those containing no people.

First download a subset of the *Inria Person dataset* contained in the `.mat` files `inria_train.mat` and `inria_test.mat` on the course website. Both these files contain images from the negative and positive classes. Each `mat` file contains a cell array containing images and a vector with the label for each image.

Exercise 1: *Downloading and applying a pre-trained ConvNet to an image*

Your first task is to download and intsall `version 1.0-beta11` of `MatConvNet` from the website MatConvNet: CNNs for MATLAB. You should follow the instructions at Installing and compiling the library to set-up the package and the environment. After you have succeeded here the next task is to download the parameters of a pre-trained ConvNet. If you visit the webpage Pretrained models and look at the code in the section **Using the pretrained models** you will see how to

- download a pre-trained ConvNet model
- load the model and then
- apply it to an image.

For this exercise you should download the `imagenet-vgg-f.mat` model.

Exercise 2: *Learn linear separating hyperplane for a ConvNet representation*

You are ready to write the code for this exercise. Your first task is to extract the responses from the 18th layer of the ConvNet `net` where

```
net = load('imagenet-vgg-f.mat');
```

when applied to an image. These responses will serve as the representation of your image. The code to extract these responses after you have applied the ConvNet to a properly normalized image `im_`, (see the code from the section **Using the pretrained models**) is

```
l = 18;
res = vl_simplenn(net, im_);
rep = squeeze(gather(res(l+1).x));
```

BTW you can get the information about the layer with the commands

```
net.layers(l);
disp(A{1}.type)
```

Write the code to extract this vector of responses from all the training images and store them in the array `trainX` (where each column of `trainX` is the representation of one image). Once you have this array and the ground truth label for each array then you can train a linear hyperplane in exactly the same way as in Homework Exercise 6. I used the same settings for this assignment as I use in that assignment except that I set the maximum number of epochs during training to 60. The classifier I learn has an accuracy of ∼`.99` on the training data.

Exercise 3: *Learn linear separating hyperplane for a ConvNet representation*

Next write code to compute the accuracy of the linear SVM you have learnt on the test images. I got an test accuracy of ∼`.96`.

Exercise 4: *Optional: Find the best layer to use for the representation*

It is not obvious what the best layer is to use for the representation. You can use cross-validation on the training set to empirically find the best layer for this dataset.

Upload to course web:

- *your code and*
- *in the comment section state the accuracy of your classifier on the test set.*