Assignment 2: Recognizing Object Categories

Abhishek Sinha
as76588

Goal

The goal is to implement an image recognition system for the given 25 object categories.

Approach

I used the approach of finetuning CNN descriptors. To be more precise, the bvlc_reference_caffenet model was finetuned which is a slight variation of the original Alex Net Model.

Implementation Details

The following steps were used to implement the approach

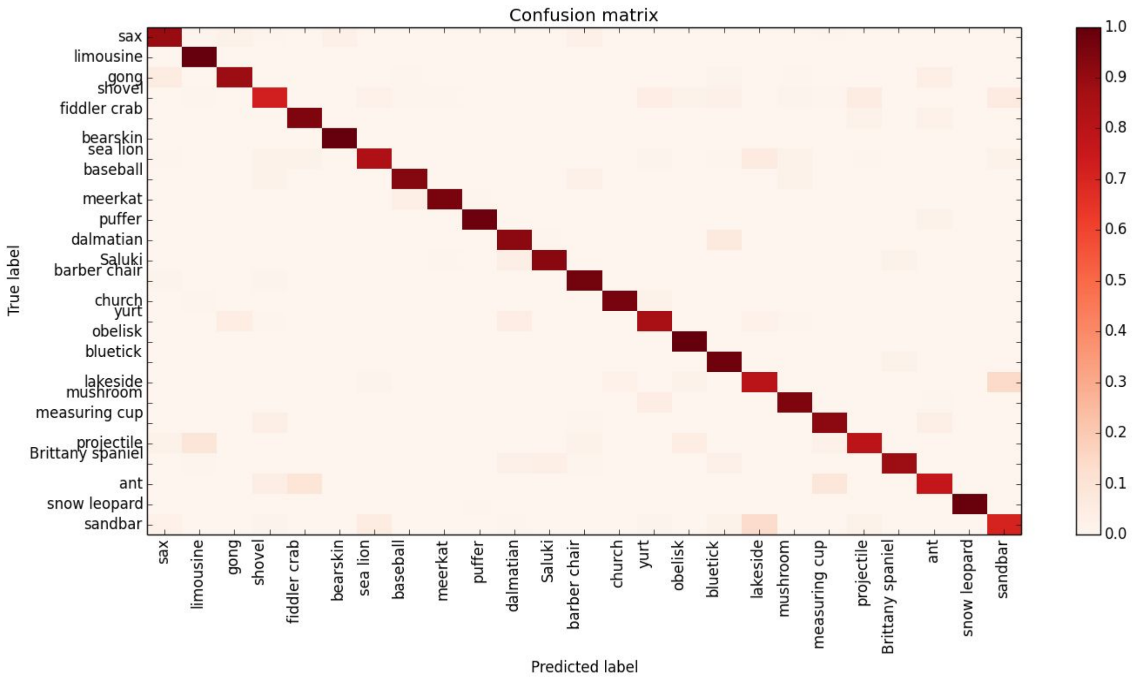
1. First an LMDB file was created for the training and test images
   i) For this, first a path label file was created using trainImNames and test1ImNames
   ii) After this an LMDB file was created using the create_imagenet.sh script given in the examples/imagenet folder in caffe.
2. The second step was to create the image mean file. For this the compute_image_mean executable was used which is present in the build/tools folder
3. The third step was to modify the prototxt files of bvlc_reference_caffenet. The following changes were made
   i) solver.prototxt:
      a) The test_iter parameter was set to 10. test_iter decides the number of batches to select each time we test on the testing data. The size of the batch was kept at 50. So this ensured that 500 test images were looked at each time during testing.
      b) The test_interval parameter was set to 100. This control the frequency at which we look at the test data. Since the training batch size is 256 (approximately 1/10th of 2500), this ensures that we look at the testing data once for every 10 times we look at the training data
   ii) train_val.prototxt
   a) The lmdb paths and mean file paths were modified.
   b) In the implementation with the best result, the learning rate multipliers for *all layers other than fc8 laye*r were set to *0*. The learning rate multiplier for the fc8 layers was retained at the original bvlc_reference_net values and the number of outputs was modified to 25.

Choices Made

The main choice to make was whether to tune only the fc8 layer or other layers too. Since the problem dataset is a subset of the entire Image Net dataset and the bvlc reference net is very well trained for that, it seemed intuitively that we should not make a lot of changes in the weights other than for the final 25 way final classifier.
We also tried the following 2 alternatives but they gave inferior results.
i) Finetuning both the fc7 and fc8 layers by keeping the original learning rate multipliers and setting all other multipliers to 0.

ii) Finetuning all the layers. Here we kept the original learning rate multipliers for the fc7 and the fc8 multipliers but took 1/10th of the original values for the other layers. The learning rate multiplier for the convolutional layers was kept lower since we did not want to loose the shared representations learnt in the original model.

Results (Only Finetuing fc8 layer)



Discussion of the Results

The overall test accuracy obtained was **93.8%**. The 4 best performing classes were:

i) obelisk

ii) bearskin

iii) limousine

iv) snow leopard

All of the above had 100% accuracy. These classes tend to perform well since some of them have canonical orientations (limousines) , some of them rich texture information (bearskin, snow leopard) and some have a combination of both.

The 4 worst performing classes were:

i) shovel - 74%

ii) sandbar - 78%

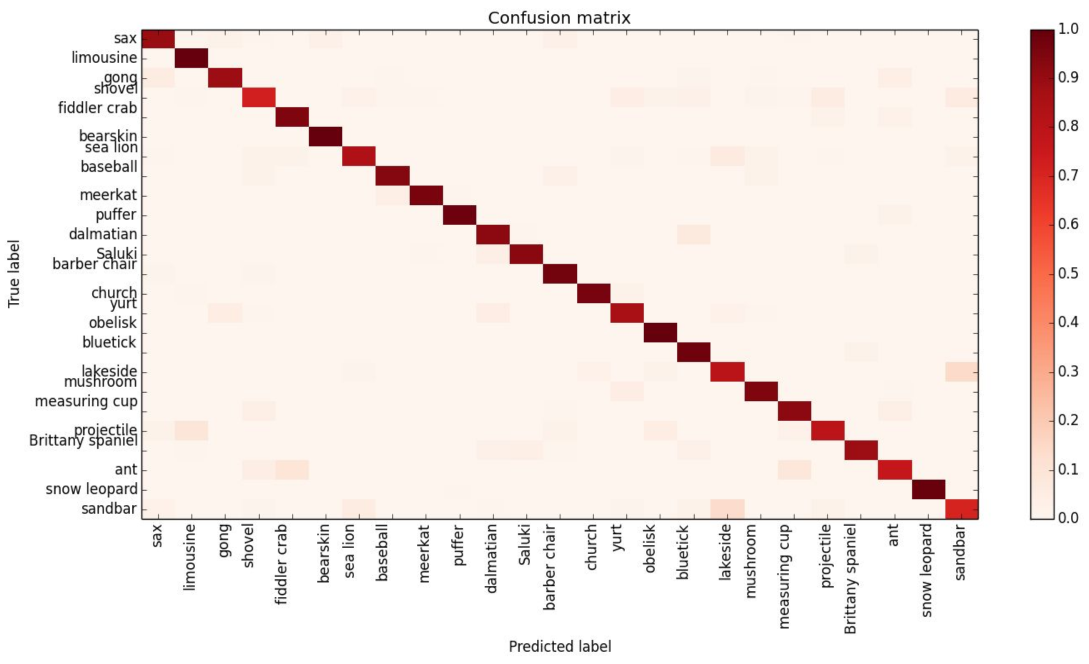
iii) ant - 79.5%

iv) projectile - 84%

The main reasons for the model performing weak on the above classes are:
1, None of the above 4 objects have very rich texture information.
2. The shovel and projectile classes tend to be found in a lot of different orientations .
3. For the ant, although there are some close-up images, in many of the images it occupies a very small portion of the entire image.

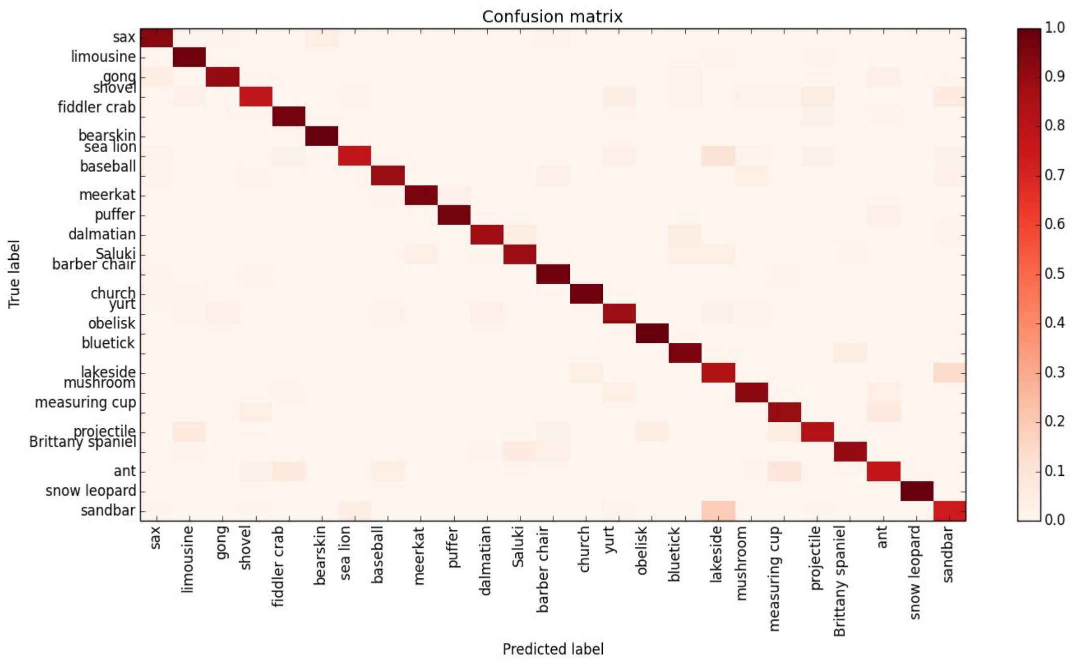
Some particular instances of mismatch are instructive to look at.
1. Ant is getting confused with fiddler crab. Both of these have similar structure in terms of the legs and general shape. But fiddler crab will generally occupy a much larger portion of the image and hence we will be able to learn a better representation for it and hence there will be more of a tendency for ant to be classified as fiddler crab than vice versa.
2. Another instructive case is sandbar and lakeside getting confused with each other. This also makes sense since both have some ‘water’ region and some ‘shore’ region.

Results on Variants
Finetuning fc7 and fc8 layers



In this case, the total accuracy dropped to **91.6%**

Finetuning all the layers
In this case, all the layers were finetuned . The learning rate multipliers for the last 2 layers was same as the original bvlc referrence caffenet model while those for the first six layers, it was 1/10th of the original multipliers.



In this case, the accuracy further dropped to **90.8%**. But the result wasn't very bad since we kept the learning rates for the convolutional layers to be small. Hence, we did not loose all the useful shared representations learnt by the original model.

References
Testing/convnet_test.py
This code applies the forward function and computes the confusion matrix. This code is entirely adapted from the following 2 sources with minor modifications.
i) https://gist.github.com/axel-angel/b2af7d980eb217a0af07#file-convnet_test-py
ii) https://github.com/sammy-su/CS381V-caffe-tutorial/
blob/master/CS381V_features.ipynb

Visualization/plot.py
Plotting the confusion matrix.

http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html

Other than this, the code for creating the lmdb files and image mean files were taken from the examples given in the caffe installation.