Homework 5

Neural Network

By: Saurabh Hinduja

1)

Result of Auto Encoder:

```
Starting Epoch  0
Loss: 2.31
--------------------------
Starting Epoch  1
Loss: 2.30
--------------------------
Starting Epoch  2
Loss: 2.30
--------------------------
Starting Epoch  3
Loss: 2.30
--------------------------
Starting Epoch  4
Loss: 2.29
--------------------------
Starting Epoch  5
Loss: 1.85
--------------------------
Starting Epoch  6
Loss: 1.10
--------------------------
Starting Epoch  7
Loss: 0.66
--------------------------
Starting Epoch  8
Loss: 0.66
--------------------------
Starting Epoch  9
Loss: 0.38
--------------------------
Starting Epoch  10
Loss: 0.29
--------------------------
Starting Epoch  11
Loss: 0.30
--------------------------
---------------------
Testing on Test data
---------------------
Loss 0.15
Accuracy  95.52 %
```

2)

Result for RBM:

```
[BernoulliRBM] Iteration 1, pseudo-likelihood = -88.71, time = 25.87s
[BernoulliRBM] Iteration 2, pseudo-likelihood = -83.74, time = 29.37s
[BernoulliRBM] Iteration 3, pseudo-likelihood = -85.02, time = 29.88s
[BernoulliRBM] Iteration 4, pseudo-likelihood = -82.24, time = 31.78s
[BernoulliRBM] Iteration 5, pseudo-likelihood = -82.96, time = 29.41s
[BernoulliRBM] Iteration 6, pseudo-likelihood = -81.42, time = 29.29s
[BernoulliRBM] Iteration 7, pseudo-likelihood = -80.45, time = 29.08s
[BernoulliRBM] Iteration 8, pseudo-likelihood = -81.01, time = 29.29s
[BernoulliRBM] Iteration 9, pseudo-likelihood = -78.93, time = 29.49s
[BernoulliRBM] Iteration 10, pseudo-likelihood = -78.04, time = 29.19s
[BernoulliRBM] Iteration 11, pseudo-likelihood = -78.74, time = 28.57s
[BernoulliRBM] Iteration 12, pseudo-likelihood = -79.05, time = 29.09s
[BernoulliRBM] Iteration 13, pseudo-likelihood = -78.31, time = 28.93s
[BernoulliRBM] Iteration 14, pseudo-likelihood = -78.46, time = 29.20s
[BernoulliRBM] Iteration 15, pseudo-likelihood = -83.55, time = 28.89s
Initializing neural network with 3 layers, 400 inputs and 10 outputs.
  - Convl: Tanh       Output: (23, 23)   Channels: 5
  - Dense: Tanh       Units:  100
  - Dense: Softmax    Units:  10

Training on dataset of 52,500 samples with 21,525,000 total size.
  - Reshaping input array from (52500, 400) to (44625, 1, 20, 20).
  - Train: 44,625     Valid: 7,875
  - Early termination after 10 stable iterations.

Epoch       Training Error       Validation Error       Time
-------------------------------------------------------------
    1          1.912e+00             2.215e+00         25.8s
    2          1.815e+00             1.852e+00         25.8s
    3          1.744e+00             1.173e+00         33.9s
    4          1.659e+00             1.375e+00         30.4s
    5          1.530e+00             1.539e+00         26.1s
    6          1.409e+00             1.311e+00         27.8s
    7          1.239e+00             1.235e+00         26.1s
    8          1.217e+00             9.145e-01         25.6s
    9          1.239e+00             2.003e+00         25.6s
   10          1.224e+00             1.052e+00         27.8s
   11          1.226e+00             9.894e-01         27.0s
   12          1.196e+00             1.171e+00         29.2s
   13          1.060e+00             7.987e-01         27.1s
   14          1.031e+00             1.175e+00         25.3s
   15          9.447e-01             9.491e-01         25.3s
   16          9.654e-01             1.128e+00         25.2s
```

17	1.064e+00	1.268e+00	25.3s
18	9.327e-01	7.510e-01	27.4s
19	9.230e-01	9.508e-01	26.8s
20	9.352e-01	8.432e-01	26.2s
21	9.063e-01	8.921e-01	25.6s
22	9.498e-01	9.681e-01	27.7s
23	9.985e-01	7.117e-01	27.8s
24	8.851e-01	6.031e-01	27.8s
25	8.065e-01	8.783e-01	27.9s
26	8.681e-01	7.047e-01	27.6s
27	8.227e-01	6.475e-01	26.4s
28	8.878e-01	7.492e-01	28.3s
29	9.261e-01	1.164e+00	25.7s
30	8.065e-01	6.717e-01	25.3s
31	7.900e-01	7.143e-01	25.2s
32	7.804e-01	6.038e-01	25.1s
33	8.476e-01	7.348e-01	25.1s
34	7.304e-01	6.206e-01	25.3s

Early termination condition fired at 34 iterations.

Accuracy in training 0.861542857143
Accuracy in testing 0.855085714286

3)

The final project Proposal:
To determine the thumb placement of a robotic arm on objects.
The problem we want to solve and why:
Nowadays, everything is getting automated. Robots are being used everywhere from space mission, industrial assembly lines and very soon to help us perform daily tasks, such as cleaning the house, etc. For majority of these tasks the robots have to hold, lift, or grasps objects, example, a screwdriver in space, or a bringing you a hot cup of coffee. For all these our simple proposal is to find the thumb placement for the robotic arm. Once the robot knows where to place its thumb, then the remaining fingers can easily help in grasping.

What you need deep learning to solve it:
We want deep learning to help us finding the thumb placement on objects.

What kind of deep structure you plan to construct:
At this stage we are not sure which kind of deep learning structure will work well with our dataset, but we plan on using convolutional neural network as it is best for processing images.

How to train it:
We will be using images of objects from various angles without the thumb placement as the input to the neural network and the mask of the thumb placement as the output.

How to obtain training data:
Dr. Sun has provided us with 33 3D models of everyday objects with the thumb placement. We will use these 3D models remove the thumb placements and take pictures of the models from all 360 degrees, with increment of 1 degree in the horizontal and vertical planes. The total number of images for each object is 720. Then we take just the thumb placement of the object and again take 720 images. Therefore, for each objects we will have 1440 images making the total training dataset of 47520 images.
For getting the images we are using Unity 3D.

How to validate your trained deep network:
As we have only 33 objects, we want to train the network on 30 objects and validate 3 object.