Artificial Neural Networks and Deep Architectures, DD2437

Short report on lab assignment 4
RBM and DBN

Jordan Wu, Daniel Janischowsky

Oktober 10, 2019

1 Main objectives and scope of the assignment

- Implement RBM and investigate the effects of hidden units
- Implement DBN by stacking and greedy training of RBMs. Investigate recognition and generation of images using DBN.
- Implement fine tuning of DBN using wake-sleep algorithm. Investigate its effects on performance.

2 Methods

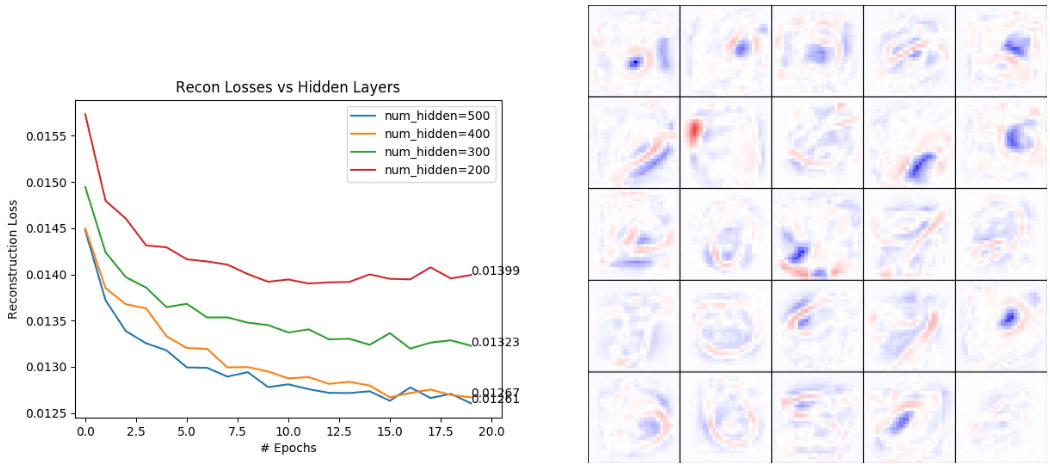
The entire lab was done in Python using the provided code.

3 Tasks

3.1 RBM for Reconstructing Images

One way to monitor convergence or stability is to follow the same approach as early stopping in other neural net training methods. We stop when the training losses (MSE between the original and reconstructed image) don't improve by a significant amount. Another way to measure convergence is to see if $\Delta W \to 0$.

We noticed that reconstruction loss increased for 300 and 200 hidden nodes. With 200 nodes, training losses plateaued out at about 10 epochs.



Learning Curves Receptive Fields

Figure 1: VIS-HID RBM

In the receptive fields representation above, we can see some semblance of various numbers jumbled together.

3.2 Greedy Layerwise Pre-training

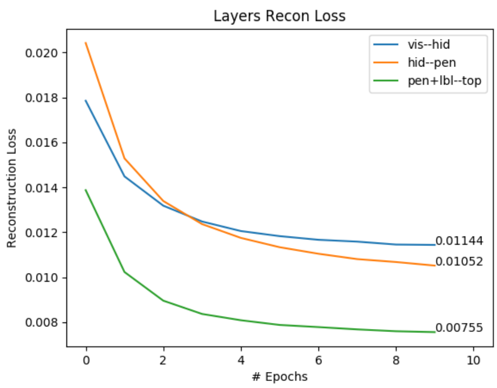


Figure 2: DBN Layers' Recon Loss

Dataset	Accuracy Mean	Accuracy Std
Train	90.15517	0.08177
Test	90.16383	0.05914

Figure 3: Recognition Accuracy (10 Runs)

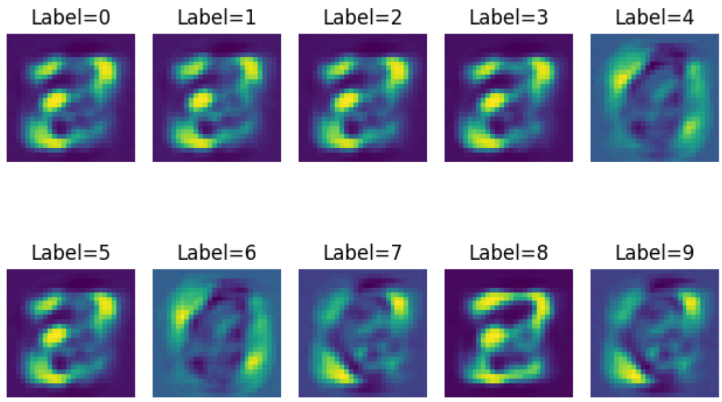


Figure 4: Generated Images for each Label

We can see differences within the generated samples. Although we can not match them with the ten numbers as we expected it to do. There seems to be two "sets" of similar images. Another thing we noticed was that $P(h|v)_{top}$ after Gibbs sampling was similar for the different labels, which explains the similar images.

3.3 Fine Tuning

3.3.1 Fine Tuning of Original Model

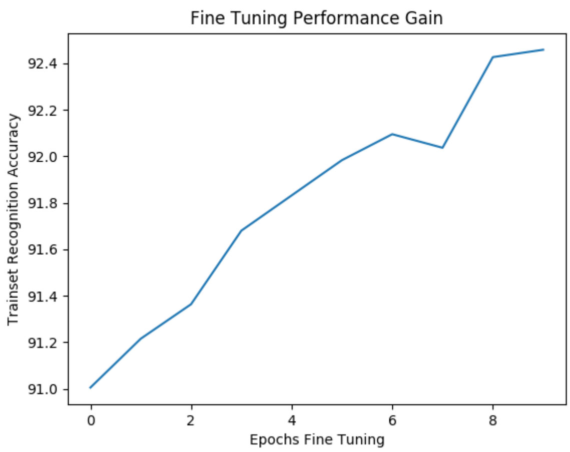


Figure 5: Recognition Accuracy Improvement

Dataset	Accuracy Mean	Accuracy Std
Train	92.42800	0.06725
Test	92.42617	0.04097

Figure 6: Fine Tuned Recognition Accuracy (10 Runs)

We saw about a 2% recognition performance increase after fine tuning, raising the recognition from 90% to 92.4%.

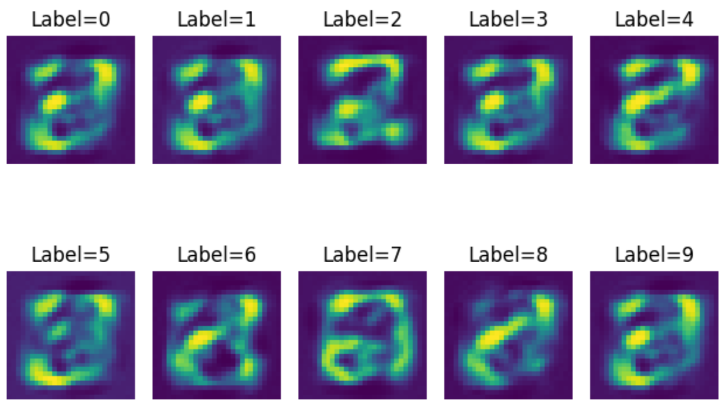


Figure 7: Generated Images After Fine Tuning

After fine tuning, all of the images now look similar to each other. With careful examination of each image, we can see that although the general shapes are similar, the brightest spots are different.

3.3.2 Simple DBN

Below are the results for a $784 \to (500 + 10) \to 2000$ DBN

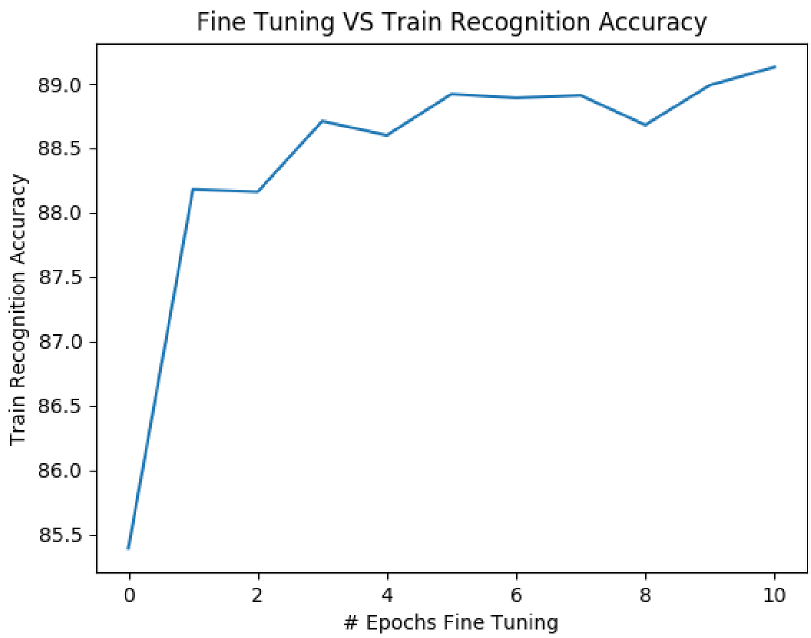


Figure 8: Recognition Accuracy Improvement

Dataset	Training	Accuracy Mean	Accuracy Std
Test	Greedy (10 Epochs)	85.628	0.22569
Test	Fine Tuned (10 Epochs)	89.24199	0.15432

Figure 9: Simple DBN Recognition Accuracy (5 Runs)

We found it quite interesting that this DBN achieved much higher performance gain after fine tuning compared to the original model. This was contrary to our initial intuitive assumption that fine tuning would provide more benefit more complex models. We reckon that with more epochs of fine tuning, this simple model can get close to our original DBN's performance.