Lecture 17.2 Exercises

2.1 Comparing network architectures

In this problem, you will compare the performance of three neural networks:

1. The two layer convolutional neural network we built in the tutorial above.
2. A one layer convolutional neural network obtained from the first network by removing the second convolutional layer and skipping straight to classifier layers.
3. A simple linear classifier: just the classifier part of the previous networks, without any convolutions.

(Note that to make the second network work, you will have to adjust the sizes of the reshaping and linear layers.)

2.2 Comparing performance and learning dynamics

For each of these networks do the following:

1. Train the network for 1,000 steps using batches of size 100. Record the test accuracy every 50 steps during training.
2. Plot the test accuracy against training iteration for each of the three networks on the same set of axes. Plotting in Torch is fairly straightforward; you can read the document here: https://github.com/torch/gnuplot.
   The curves you will see are called the model's learning curves. Do you think the model's learning dynamics are a good match for human learning?
3. Comment on the plot above. Why does it look how it does? Think about both the final accuracy the model achieves and the dynamics of how it gets there.

A note about optimization: a side effect of the fact that we've chosen our optimization algorithm for its simplicity and transparency rather than performance is that it's difficult to add regularization to the parameters of our model. Regularization prevents parameters from blowing up. As a result, you may see situations during training in which your network's weights become too large for the computer to handle (this will look like a flat learning curve stuck at chance accuracy). If this happens, you can try running the model again, or experiment with different learning rates and batch sizes.

2.3 Comparing invariance

As discussed in the chapter, an important feature of convolutional networks deals with translation invariance: if the model's input is shifted by a few pixels, then its predictions should not change much. Here, we'll study how the representations discovered by our three networks change as we shift their inputs.

The data files:

./data/translations/leftShifts.t7
./data/translations/rightShifts.t7
./data/translations/center.t7

contain shifted versions of the first 100 images from the mnist test. center.t7 contains the one 100 test images in their original centered position. leftShifts.t7 is a Lua table with five cells, the *i*th of which contains the 100 images shifted *i* cells to the left. Likewise, rightShifts contains five right shifts of the 100 test images.

You will feed these shifted images through the **trained** networks from part 1. For each network, you will:

1. Feed the center images through the network, obtaining a score vector. This will give you, for image *I*, a score vector $V^I_{\text{center}.}$
2. Find for each shifted image its score vector, and compute the distance between this vector and $V^I_{\text{center}.}$ for each shift distance *i* = 1, …, 5
3. Feed the batch of images leftShifts[i] through the network. For each image *I*, this will give you a score vector $V^I_{\text{left}}[i]$ from which you can compute the distance to the center vector, $\left|\left|V^I_{\text{center}} - V^I_{\text{left}}[i]\right|\right|$.

   To get a general picture of the effect a leftward shift of I pixels tends to have, average you're the distance above over all 100 images *I* to obtain:

$$\text{dist}_{\text{left}}[i] = \frac{1}{100}\sum_{k=1}^{100}\left|\left|V^{I_k}_{\text{center}} - V^{I_k}_{\text{left}}[i]\right|\right|,$$

   for *i* = 1, …, 5. The function avgDistance in util.lua can help compute the average over a batch of score vectors.

   Important: since we are interested in comparing the invariance properties of different networks, we have to account for the fact that some networks might generally tend to produce larger score vectors than others, meaning the distance between the shifted and center vectors for these networks could be large without reflecting a meaningful representational difference. To account for this possible distortion, force each of your network-output score vectors to have norm one, projecting it onto the unit ball without changing the relative sizes of it entries. The function normalize() in util.lua can do this for you.
4. Repeat step 2 for the right shifted images, getting a mean-distance vector $\text{dist}_{\text{right}}$.
5. Plot your distance vectors. There will be 11 tics on the x-axis of your plot, ranging from -5 to 5. The first five of these, -5, …, 1, will be $\text{dist}_{\text{left}}$, the last five will be $\text{dist}_{\text{right}}$, and the middle should be zero, corresponding to the distance between the centered image score vector and itself. Put the plots from all three networks on the same axes.

When you compare the invariance data for the three networks, what do you observe? Is it what you would expect from the networks' structure? How would you go about building a network with more invariance?

2.4 Comparing to neural data

As discussed in the chapter, CNNs are promising models of ventral stream visual processing. We reviewed some studies comparing neural networks to electrophysiological data. For many tasks, though, human fMRI data is much faster and easier to collect. How would you go about assessing the neural fidelity of a CNN model using fMRI data? Here are some papers you may find useful:

- Khaligh-Razavi S-M, Kriegeskorte N (2014) Deep Supervised, but Not Unsupervised, Models May Explain IT Cortical Representation. PLoS Comput Biol 10(11): e1003915. doi:10.1371/journal.pcbi.1003915
- Güçlü, Umut, and Marcel AJ van Gerven. "Deep neural networks reveal a gradient in the complexity of neural representations across the ventral stream." The Journal of Neuroscience 35.27 (2015): 10005-10014.
- Agrawal, P., Stansbury, D., Malik, J. & Gallant, J. L. Pixels to Voxels: Modeling Visual Representation in the Human Brain. ArXiv14075104 Cs Q-Bio (2014).
- Cichy, Radoslaw Martin, et al. "Comparison of deep neural networks to spatio-temporal cortical dynamics of human visual object recognition reveals hierarchical correspondence." Scientific reports 6 (2016).
- Cichy, Radoslaw M., et al. "Deep neural networks predict hierarchical spatio-temporal cortical dynamics of human visual object recognition." arXiv preprint arXiv:1601.02970 (2016).