Problem Set 2

Due: Tuesday, November 8, 3:30pm.

1. **Toy CNN:** In this problem you will build your own, small CNN and apply it to some problems with synthetic data. The goal here is to get some experience with CNNs and gain some intuitions. In all of these problems, you will have to make a number of choices about the CNN and the data. For example, in problem 1 you will have to decide how big an image to use, how big a filter to use in your CNN, and how big the circles and squares are. Make reasonable choices and explain what you have done.

   (a) **Shallow network:** We provide you with a program to generate synthetic images. Each image will contain either a circle or a square. The circle or square will be at a random location in the image, but it will always have the same size. Train a CNN to distinguish between images with a circle or a square. Your CNN should have just a single hidden layer. How many filters do you need to use to get good performance? Create a visualization of the filters that you learn, focusing on any that are particularly interesting. Can you say anything about why they look the way they do?

   (b) Try varying the size of your filter. How do the results vary? How big a filter do you need to distinguish these classes?

   (c) Now add some distractors. These could look like a half circle, or a rectangle or a pac-man. Each image now contains one circle or one square in a random location, and a bunch of distractors in random locations. How (if at all) do the results change as you add distractors? See if you can add distractors that will make the result more interesting.

   (d) **Optional problem for extra credit.** Now create random images with objects and distractors like the ones shown in Figure 1. Can you train the shallow network to tell the difference between these objects? How does the size of the filter you need differ from the previous problems?

   (e) **Optional problem for extra credit.** Try training a deeper network with more filters. Can you get good results with a deep network using fewer overall parameters than you would need to get good results in a shallow network?

2. **Real CNN:** In this problem we will look at a large-scale CNN that has been trained on real data, ImageNet. Unfortunately, training a network on ImageNet takes a lot of time, so it's hard to run experiments in which we explore changes in the hyperparameters of the system. So instead, we'll look at a network that has already been trained, and try to understand it.

   In this problem we will look at how active a neuron is when a particular image (or set of images) is fed into the network. One way to measure this level of activity is to look at the magnitude of the neuron's output. Feel free to propose other measures of activation. Keep in mind that the problem, as stated, might not have a very satisfying solution. Maybe the most active neuron isn't doing

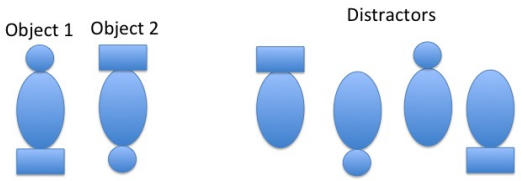

Figure 1:

anything very interesting. If this seems to be the case, try to find other ways to explore the questions raised by these problems, and report on what you find.

(a) Download a pretrained version of a network for ImageNet. If you want to use AlexNet, then for Matconvnet, you can find this at: http://www.vlfeat.org/matconvnet/pretrained/. AlexNet is not fully supported by LightNet, so if you want to use LightNet you can download VGG from the same location. Figure out how to run this on real images. Using a couple of images of your own, test AlexNet. Show the images you used and the classes that the network came up with.

(b) Pick an image that your network seems to classify correctly. Find the neuron that is most active in each layer of the network when that image is shown. Do these neruons seem to be spatially localized? For example, if you block out parts of the image, covering them with a black rectangle, how does this affect how active the neurons are? Does this seem to depend on which parts of the image you block out? Do your results vary depending on what layer of the network you look at?

(c) Now take two sets of images from two similar classes (for example, from two breeds of dogs). Find the neurons in each layer that have the greatest difference in their response to these two classes. Try to characterize what these neurons might be doing, for example, by seeing how their response changes when you black out parts of the images.

(d) Take two different images from the same class. Compare the pattern of activation of neurons created by these two images. For each layer of the network, how similar is the pattern of activation? Can you compare this to the similarity in the pattern of activation when you use images from different classes? For example, do two beagles produce more similar activations

than do a beagle and a refrigerator? Can we see that the similarity in activation becomes greater as we move higher in the network? Can you quantify this?

One way to measure the similarity in activation is to compute the magnitude of the output for each neuron. For each layer of the network, this would give you a vector of magnitudes for each image. Then you could compute the mean squared error between the two, as a measure of similarity. But maybe this isn't the best way to do this. Feel free to propose other methods.