Questions

"Training"

1. Describe the idea of a "distributed" encoding.
2. What is data augmentation?
3. Describe the following strategies for preventing overfitting: data augmentation, data normalization, model averaging, dropout, weight decay, and early stopping.
4. In PyTorch, why is it important to mark whether a model is currently used for *training* or for *testing*?
5. Describe, intuitively, why large weights are indicative of overfitting.
6. What is an autoencoder?
7. What is the loss function used when training an autoencoder?
8. What is an "embedding"?
9. How are word vectors (word2vec, GloVe) trained?

"Generalization"

1. Consider augmenting an image dataset by shifting the training images by a random (small) number of pixels. For which type of neural network would this type of data augmentation have a bigger impact: a convolutional network or a fully-connected network?
2. Suppose you have two networks: network A has 10 million hidden units (artificial neurons) and 1 million weights, network B has 10 million weights and 1 million hidden units. Which of the two networks has higher capacity? Which of the two networks is more likely to overfit?
3. In PyTorch, the implementation of weight decay in an optimizer (e.g. `optim.SGD(model.parameters(), weight_decay=0.0001)`) performs weight decay for **all** parameters, including biases. Why might we want to allow large biases, and only decay neural network *weights*?
4. We discussed data augmentation for images. Suppose that you are instead working with voice recordings. What are some ways to augment a dataset of voice recordings?
5. What do you think would happen if the encoder output of an autoencoder is **larger** than the image size? You may assume that both the encoder and decoder have high capacity.
6. What do you think would happen if the encoder output of an autoencoder is **too small**?
7. What does **overfitting** look like in the context of an autoencoder?
8. Why do we use the sigmoid activation in an autoencoder of images?
9. Suppose that an autoencoder used only convolutional layers, with no fully-connected layers, and also no global pooling layers. Can the autoencoder be used to generate images of different sizes? If so how?

10. How can you use autoencoders for data augmentations?