APS360 Summer 2019 Midterm Solutions

Question 1

- Part (a)
  - Answer: C
  - For choice A, one would solve the problem by running the Python code For choice B, one could easily write a program to solve the problem
- Part (b)
  - Answer: A
  - The softmax output should be a probability distribution. Choices B, C, D do not sum up to 1 and so can be ruled out. Choice E can also be ruled out because the input to the softmax need to be four equal numbers for the output distribution to be unifrom.
- Part (c)
  - Answer: C
  - Each number in the one-hot encoding of a word is independently interpretable.
- Part (d)
  - Answer: A
  - Decreasing the learning rate will make the training curve less noisy The size of the training set is unlikely to affect the noisiness of the curve, as is the number of parameters in the network.
- Part (e):
  - Answer: C
  - Since the optimizer is minimizing the training loss across the training set, the training loss should decrease in each iteration (assuming a good learning rate)
  - If the batch size is 1, then the training loss will be noisy, and may not decrease in each iteration.
  - For larger inputs, a smaller batch size is often used so a batch can fit in memory.
  - The autoencoder architecture choice is independent of the batch size choice.
  - Training accuracy can stay the same, even when the training loss is decreasing: in that case, the network has the same number of correct predictions, but the probability estimate of the correct predictions are higher (e.g. 98% vs 55% predicted probability of the ground truth)
- Part (f):
  - Answer: D
  - Number of layers and more training epochs increase chances of overfitting.
  - Larger batch size shouldn't affect overfitting. (In lab 3, you may have notice that if you double your batch size but keep the same number of epochs, you actual *half* the number of weight updates that you perform.)
- Part (g):
  - Answer: C
  - GoogLeNet introduced the Inception submodule
- Part (h):
  - Answer: A
  - It doesn't make sense to dropout a neural network output, because no weights are actually trained using those values as input.
  - Applying dropout to an input layer is often done. Data augmentation can be applied to non-image inputs (e.g. audio, as in the weekly problems). Transfer learning is much more effective than weight decay at preventing overfitting.
- Part (i):
  - Answer: D
  - The encoder and decoder of an autoencoder do not have to be symmetrical (see the midterm preparation worksheet)
  - Denoising autoencoder has the same architecture as a normal autoencoder, just different inputs.
- Part (j):
  - Answer: E
  - Words with similar GloVe embeddings often has opposite meanings, but appear in similar contexts. We saw in class that word2vec/GloVe models learns the bias in the text that it is trained on. The word2vec architecture takes the one-hot encoding of a word as input, which is fixed in length.

Question 2

- Part (a):
  - relu(-10) = relu(0) = 0
  - relu(10) = 10
  - Need to be able to read off relu(10) to earn full points
- Part (b):
  - tanh(-10) = -1 (approximately)
  - tanh(10) = 1 (approximately)
  - tanh(0) = 0

Question 3

- Part (a):
  - Training set: Tune model parameters
  - Validation set: Tune hyperparameters
  - Test set: Evaluate model on unseen data (likely to appear in actual use)
- Part (b):
  - Using a standard test set means that models that different people build are comparable.
  - Otherwise, different test set choices can vary in difficulty
- Part (c):
  - No. Data augmentation on the test set doesn't help us prevent overfitting since it does not affect training. Data augmentation on test set also makes the test set different from the kind of data that the network would see in practice.

Question 4

- [10, 32]
- [20, 7, 12, 12]
- [20, 3, 8, 8]
- [20, 7, 8, 8]
- [20, 1, 31, 31]
- [20, 1, 36, 36]

Question 5

- Part (a):
  - Layer 1 input should have `32*32` features
  - Layer 2 input should be consistent with layer 1 output
  - Add activation function between layer 1 and layer 2
- Part (b):
  - Flattens an input `img` tensor of shape [-1, 32, 32] (or something else) to the shape [-1, 32*32]. This 2D shape is required by the fully connected layer input.
  - Restating the API documentation is not enough: we are looking for specific, unambiguous interpretations of what the code does in this specific case

Question 6

- Part (a): N, L, F, B, E, S
- Part (b): (g)
- Part (c):
  - Dropout layer is the issue; different neurons could be dropped out
- Part (d):

  - Need `model.eval()` to be called.
  - NOTE: this question was ambiguous. If you wrote any reasonable code that was missing in the `train` function from part (a), then you were awarded part/full marks. We did not accept answers like "reinitialize model weights" since the model is not defined inside the `train` function.

Question 7

- Part (a):
  - `1*16*3*3 + 16 + 16*32*3*3 + 32 + 32*64*7*7 + 64`
- Part (b):
  - Activation function will remove information, and the decoder would need to learn the inverse of the activation function.
  - We are not passing the output of the encoder to a loss function, so numerical stability is not a valid answer to this question.
  - NOTE: We tried our best to grade the best possible interpretation of your answer, and give part marks whenever possible. Unfortunately, there's not much we could do if we can't understand your answer, or if there was too much uncertainty about what you could mean.
- Part (c):
  - No, because there is no guarantee that the encoder had ever encoded any actual digits near `emb`. If no actual MNIST digit is encoded near `emb`, then the encoder would not learn to decode embeddings in that region to MNIST digits.
  - NOTE: Again, we tried our best with part marks + interpreting answers.

Question 8

- Part (a):
  - 10000 * 10 / 100 = 1000
- Part (b):
  - The network quickly learn to output "large digit" for every input to obtain a 70% accuracy. After some epochs, the network starts to learn to actually distinguish small vs large digits.
- Part (c):
  - Yes, but not severe overfitting because the validation accuracy is not decreasing.
  - That we see some evidence of overfitting in early epochs (~200) is not a cause of concern (we have a choice of how long to train)
  - The difference between validation and training accuracy is also not a good measure of severity of overfitting: the validation accuracy could still be increasing, even if the difference between training/validation accuracy is high.

Question 9

- Part (a):
  - The last layers will have the most high-level features (features that describes a larger portion of the image). Using a later layer activation means we need to train fewer layers ourselves.
- Part (b):
  - GloVe tends to embed words with similar contexts together (context = words appearing before/after a certain word). Male names tend to appear in similar contexts.
- Part (c):
  - The neural network is just as biased as the data that it is trained on.
  - NOTE: Neural networks are trained using examples. Other AI techniques can be rule-based, but the question talks about neural networks only.