Solutions for CSC321 Midterms

Notes about grading

We made some adjustments to grades on Markus after grading on paper. The annotations on Markus supersedes the hand-written marks.

Midterm A. Question 1

1. (A) Neural networks are not easily interpretable.
2. (E) The optimal choice of k depends on the problem; there are no trainable parameters in a k-NN model; the training accuracy for a 1-NN model is 100%; and the decision boundary is nonlinear. The only correct choice is the last one.
3. (C) The training cost is always positive because y_k is always between 0 and 1, so the two components of the cross-entropy loss $-\log y_k$ and $-\log(1-y_k)$ are both always positive.
4. (E) Gradient descent updates the parameters only. This is really a question regarding terminology: "layer" is not the correct terminology to use. "Activations" are the values of the hidden units and depend also on the input data, and are not updated during gradient descent.
5. (A) The learning rate was decreased, so we can get into a region of the cost function that we keep "stepping over"
6. (E) Decreasing the number of hidden units reduces the variance of the model, and helps prevent overfitting. Using a larger training set also helps prevent overfitting. Using a smaller batch size does not affect the fit of a model. The choice of optimizer also does not affect overfitting. **Edit: Half point for choices (C) or (D)**
7. (B) This is the tanh activation function, which you can identify both by its shape and its range (-1 to 1).

Midterm B. Question 1

1. (D) The first can be solved by computing whether an individual pixel is green, then aggregate the result; sorting integers can be done using techniques you learn in CSC148; checking spelling can be done by looking up words in a dictionary, again usign techniques you learn in CSC148. Determining whether a sentence is about Queen Elizabeth is difficult, since sentences about her might contain a wide range of words like "queen", "monarch", etc, and some sentences with those words might not be about Queen Elizabeth. It is unclear how we would write a program to solve this task. The best choice is (D).
2. (B) The point "p" is closest (Euclidean distance) to "O", and has 0 angle with the point "X"
3. (C) Using a linear activations means that the entire model is linear. We've seen in lecture that a neural network with at least 2 layers, and with either sigmoid/ReLU activations are universal approximators.
4. (A) If the batch size is the same as the training set, we are using full batch SGD, and the cost will always decrease. Training cost can decrease while the training accuracy stays the same. If the batch size is 1, then the training cost will generally be noisy, and might increase from time to time.
5. (D) Weight decay increases the training cost because there is an extra (positive) term in the training cost. Weight decay penalizes large **weights** and not activations. The choice of adding weight decay is independent of the choice of the optimizer. In lecture we discussed how weight decay can revive a "dead" neuron, but it does not help us get out of saddle points (e.g. the origin in the parameter space, as in Q5). **Additional note:** Dead/saturated neurons have activations that are always in the plateau area of the activation function (e.g. ReLU or sigmoid or tanh), caused by large (positive or negative) weights/biases. Weight decay reduces those parameters, so that the activations will not be consistently large.
6. (E) A high variance model is more prone to **over**fitting, not underfitting. It requires more training data to train. It is true that a high variance model will have a higher training accuracy (but potentially lower

validation and test accuracy). The batch size is an independent parameter, though you might be able to argue that a larger batch size is required to account for the variance. **Edit: Half point for choices (B) or (C)**

7. (E) The values should sum of to 1 (rules out choices A-B). The values are not linear transformations of the logits (rules out C). Also, e^x cannot be zero (rules out D).

Midterm A. Question 2

Part (a): One point for $\overline{y}$. One point for both of $\overline{w_1}$ and $\overline{w_2}$. One point for $\overline{m}$, with part marks if the sign is flipped. It's okay not to have $\overline{L}$ in the expression for $\overline{y}$.

$$\begin{aligned}\overline{\mathcal{L}} &= 1\\ \overline{y} &= \overline{L}(y-t)\\ \overline{w_1} &= \overline{y}x_1\\ \overline{w_2} &= \overline{y}x_2\\ \overline{m} &= \overline{w_1} - \overline{w_2}\end{aligned}$$

Part (b): The only thing that needs to change is

$$\overline{m} = \overline{w_1} - \overline{w_2} - 2\lambda(\frac{1}{2} - m)$$

Part (c): Initialize $\mathbf{p}$, the learning rate α, and the dampening parameter μ. Then, in each iteration, compute $\frac{\partial \mathcal{E}}{\partial m}$ and update $\mathbf{p}$ and m like this:

$$\begin{aligned}\mathbf{p} &\leftarrow \mu\mathbf{p} - \alpha\frac{\partial \mathcal{E}}{\partial m}\\ m &\leftarrow m + \mathbf{p}\end{aligned}$$

- 1 point for having an expression that mentions the learning rate α and that updates m in a reasonable way (e.g. gradient descent without momentum). A student can use another symbol other tha α
- 1 point for having an expression that uses the paramter μ in reasonable some way that looks like a momentum update. A student can use another symbol other than μ
- 2 points for the correct mathematical equations. (Subtract half mark for each minor mistake, like flipping signs etc)

Note that we **do not** need to update w_1 and w_2 explicitly, because these are computed based on the value of m. Answers that tried to update w_1 and w_2 using gradient descent are incorrect.

Midterm B. Question 2

Part (a): One point for $\overline{y}$. One point for both of $\overline{w_1}$ and $\overline{w_2}$. One point for $\overline{m}$, with part marks if the sign is flipped. (Be careful when grading because there are other ways to express some of the terms.)

$$
\begin{aligned}
\overline{\mathcal{L}} &= 1 \\
\overline{y} &= \overline{\mathcal{L}} \frac{y - t}{y(1-y)} \\
\overline{z} &= \overline{y} y (1 - y) \\
\overline{w_1} &= \overline{z} x_1 \\
\overline{w_2} &= \overline{z} x_2 \\
\overline{m} &= \overline{w_1} w_1 - \overline{w_2} w_2
\end{aligned}
$$

Part (b): There was a typo in this question, so the grading was more lax. The loss function changed, so pointing out that $\overline{y}$ needs to change is worth half a point.

The other thing that needs to change is:

$$\overline{m} = \overline{w_1} w_1 - \overline{w_2} w_2 + 2\lambda m$$

Pointing this out earns the other half point. Since the wording of the question suggests that the loss in unchanged, answers that points out only the change in $\overline{m}$ is worth the full 1 point.

Part (c): In each iteration, update:

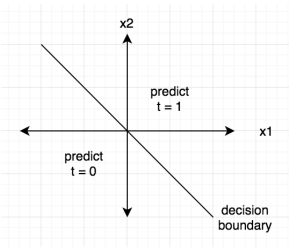
$$m \leftarrow m - \alpha \frac{\partial \mathcal{E}}{\partial m}$$

- 1 point for having an expression that mentions the learning rate α and that updates m in a reasonable way. A student can use another symbol other tha α
- 1 points for the correct mathematical equations. Subtract half mark for each minor mistake, but deduct a full mark for a flipped sign

Note that we **do not** need to update w_1 and w_2 explicitly, because these are computed based on the value of m. Answers that tried to update w_1 and w_2 using gradient descent are incorrect.

Part (d): If $m = 0$ then $w_1 = w_2 = 1$, and the decision boundary is $z = 0 = x_1 + x_2$. So if $x1 + x_2 >= 0$ we would predict $t = 1$, and if $x_1 + x_2 < 0$ we would predict $t = 0$.

Your plot of the input space should have the axes x_1 and x_2. (If you labeled your plot with axes y and x, that is incorrect. We don't even have a term x!)



Question 3

Part (a): One point for showing understanding of what the "stride" parameter does. One point for showing understanding of what the "padding" parameter does. One point for the computation of the convolution. (In the computation, forgive the first computation mistake. Additional mistakes warrants a half point deduction).

0.5	-0.5	0
0	1.0	0
0	1.5	-0.5

Part (b): $5 \times 10 \times 3 \times 3 + 10$. Half point each for the input channel, output channel, kernel squared, and bias.

Part (c): 0. Max pooling does no require any trainable parameters.

Part (d): $3 \times 7 + 7$. Half point for weights, half point for biaa

Midterm A. Question 4

Part (a): The derivative of this function is 0 everywhere, which means that if we use this activation function, we won't be able to use gradient descent to train our network.

One common misconception is regarding where activation functions are applied. Activation functions are applied at **every layer** of the neural network, not just the final layer!

Likewise, the fact that the derivative of the sign function does not exist at sign(0) is also not an issue: the derivative of sign(0) is close to 0 on either side.

The sign function is a nonlinear function, so the argument that linear functions should not be used as activation functions in a MLP does not apply.

The fact that this function is not defined at zero does not pose as a challenge: we can just define what sign(0) means. (We talked about this issue when using a similar function for logistic regression.)

Part (b): "Checkpointing" means to store our trained neural network weights every few iterations/epochs. Checkpointing is how we can implement early stopping, where we take our neural network weight at an earlier iteration, before our network has significant overfitting.

This is a question directly from project 2, where the checkpointing code was provided to you. One point for definition of checkpointing, and one point for its relationship to early stopping.

Midterm B. Question 4

Part (a): Because the computation for model training and computation for model evaluation differs, so we need to annotate whether we want to use training-time forward/backward computation, or evaluation-time forward computation.

(You are not required to write this but) during training, batch normalization normalizes the activations by computing the mean and standard deviation using the data from the current minibatch. During training, the batch normalization layer also keeps track of the mean and standard deviation of the activations. This is so that during testing, we can used the saved mean and standard deviation to perform the normalization.

Part (b): "Gradient Checking" means testing to make sure that the gradient computations are correct using finite differen approximations. That is $\frac{\partial \mathcal{E}}{\partial w}$ should be approximately equal to $\frac{\mathcal{E}(w+\epsilon,\cdots)-\mathcal{E}(w,\cdots)}{\epsilon}$ from the definition of the derivative. (Reasoning about $\mathcal{L}$ or f rather than $\mathcal{E}$ is okay.)

One point for a general answer. One point for the finite difference formula. This question is straight from project 1.

Question 5

Part (a): A saddle point $\mathbf{s}$ of a function $f(\mathbf{x})$ has $\frac{\partial f}{\partial \mathbf{s}} = \mathbf{0}$, but the point $\mathbf{s}$ is neither a maxima nor a minima.

Half point for pointing out that the derivative at a saddle point is zero, and half point for pointing out that the saddle point is neither a max or min.

Part (b): If we have $\mathbf{W}^{(k)} = 0$ and $\mathbf{b}^{(k)} = 0$, then $\mathbf{z}^{(k)} = \mathbf{W}^{(k)}\mathbf{h}^{(k-1)} + \mathbf{b}^{(k)} = 0$, and also $\mathbf{h}^{(k)} = 0$.

This also means that during the backward pass, we will have $\overline{W^{(k)}} = \overline{z^{(k)}}(h^{(k-1)})^T = \mathbf{0}$ and $\overline{b^{(k)}} = \overline{z^{(k)}} = \mathbf{0}$ and $\overline{\mathbf{z}^{(k)}} = \overline{\mathbf{h}} \circ I(\mathbf{z} > 0)$

- One point for understand the need to show that $\overline{\mathbf{W}^{(k)}} = 0$ and $\overline{\mathbf{b}^{(k)}} = 0$
- One point for working with the forward computation, for example to show that $h^{(k)} = 0$

- Half point for showing $\overline{W^{(k)}} = \mathbf{0}$
- Half point for showing $\overline{b^{(k)}} = \mathbf{0}$