EECS 498.007 / 598.005
Deep Learning for Computer Vision
Fall 2019 Midterm Exam

October 21, 2019

Full Name: Mingyu Yang

UM Uniqname (Username): mingyuy

UM-ID (Number): 50978233

Question		Score
True/False	(16 points)	
Multiple Choice	(32 points)	
Short Answer	(52 points)	
Total	(100 points)	

- The exam is **1 hour 15 minutes**
- The exam should have **9 pages** – make sure you have them all
- The exam is closed-book, closed-internet
- You may use one page (max 8.5" × 11") of handwritten notes

I understand and agree to uphold the Honor Code during this exam.

Signature: Mingyu Yang Date: 10/21/2019

Good luck!

This page is left blank for scratch work only. DO NOT write your answers here.

1 True / False (16 points)

Fill in the circle next to True or False, or fill in neither. Fill it in completely like this: ●. No explanations are required.

Scoring: Correct answer is worth 2 points. To discourage guessing, incorrect answers are worth -1 points. Leaving a question blank will give 0 points.

1.1 When using a K-nearest neighbor classifier for image classification, increasing the value of K will also increase the accuracy on the test set.

● True ○ False

1.2 Suppose we have trained a softmax classifier (with weights W) that achieves 100% accuracy on our training dataset. If we change the weights to $2W$, the classifier will maintain the same accuracy while have a smaller total loss on the training dataset. (Assume cross-entropy loss without regularization)

○ True ● False

1.3 The loss function will always decrease after performing one iteration of full-batch gradient descent (no minibatches, compute loss and gradient on the full training dataset).

○ True ● False

1.4 The derivative of the loss with respect to some weight in your network is -3. Decreasing this weight (by a tiny amount) will cause the loss to decrease.

○ True ○ False

1.5 During backpropagation, as the gradient flows backward through a tanh non-linearity, it will always become smaller or equal in magnitude. (Recall that if $z = \tanh(x)$ then $z = (e^x - e^{-x})/(e^x + e^{-x})$ and $\frac{\partial z}{\partial x} = 1 - z^2$)

● True ○ False

1.6 Consider a nonlinear activation function $z = f(x)$ where f is any of sigmoid, tanh, or Leaky ReLU (with positive α). During backpropagation, as the gradient flows backward through f, each element of the downstream gradient $\frac{\partial L}{\partial x}$ will always have the same sign as the corresponding element of the upstream gradient $\frac{\partial L}{\partial z}$.

○ True ● False

1.7 You are training a CNN classifier to recognize handwritten digits 0 to 9. Your training dataset has the same number of samples for each of the ten digits. In order to increase the effective size of your training set, it is a good idea to use data augmentation which randomly applies horizontal and vertical flips to the images before feeding them to the CNN.

○ True ● False

1.8 One of the main benefits of dynamic computational graphs in a deep learning software framework is that they give the framework the ability to optimize the graph before it is run.

○ True ● False

2 Multiple Choice (32 points)

Fill in the circle(s) next to your answer(s). Fill them in completely, like this: ●.
No explanations are required. For each question, mark all answers that apply.

Scoring: Each question is worth 4 points. Each answer within each question is worth one point. Example: If the correct answers are A and B, and you choose A and C, then you receive 2/4 points for the question: one point for correctly choosing A, and one point for correctly not choosing D.

2.1 In the context of binary classification on images, a classifier is called *linear* if its decision boundary on the input space is a linear function: positive and negative examples are separated by a hyperplane. Which of the following is/are correct about various image classifiers?

- ○ A: KNN with $k = 1$ is a linear classifier.
- ○ B: KNN with $k = 5$ is a linear classifier.
- ● C: SVM [1] with hinge loss is a linear classifier.
- ○ D: CNN with a ReLU activation is a linear classifier.

2.2 Consider a two-layer-network $s(x) = W_2 f(W_1 x + b_1) + b_2$ with $W_1 \in \mathbb{R}^{H \times D}, W_2 \in \mathbb{R}^{C \times H}$. Which of the following statements about $f(x)$ are correct?

- ○ A: Choosing $f(x)$ to be an additional layer $f(x) = W_f x + b_f$, where $W_f \in \mathbb{R}^{H \times H}$, will allow the model to represent a wider class of functions compared to $f(x) = \text{ReLU}(x)$.
- ○ B: Choosing $f(x)$ to be an additional layer $f(x) = W_f x + b_f$, where $W_f \in \mathbb{R}^{H \times H}$, will likely increase the training accuracy compared to $f(x) = x$ (no activation).
- ● C: Generally, $f(x) = \text{ReLU}(x)$ usually works well in practice.
- ○ D: Choosing $f(x) = \text{LeakyReLU}(x)$ might introduce the vanishing gradient problem to our network.

2.3 Which of the following are true about optimization methods used to train neural networks?

- ○ A: A common strategy for training neural networks is to slowly increase the learning rate throughout the entire training process
- ● B: The bias correction factor in the Adam optimizer helps prevent very large updates in the first few iterations of training
- ○ C: Adding a momentum term to stochastic gradient descent makes it more likely to be caught in bad local minima.
- ○ D: Second-order methods such as Newton's method or L-BFGS are less expensive than first-order optimization algorithms, and they perform well in stochastic settings.

2.4 You train a Neural Network classifier and find that the accuracy on the training set is much higher than the accuracy on the validation set. Which of the following would you expect to decrease this gap?

- ● A: Train on a larger dataset
- ○ B: Add more hidden units
- ● C: Increase the strength of L2 regularization
- ○ D: Decrease the dropout probability

[1] SVM with a trivial (linear) kernel. Ignore this if you don't know what an SVM kernel means.

2.5 Which of the following convolution layers will have an output with the same spatial size as its input?

- ○ A: 3×3 convolution with stride 1 and pad 0
- ● B: 3×3 convolution with stride 1 and pad 1
- ○ C: 1×1 convolution with stride 1 and pad 1
- ○ D: 5×5 convolution with stride 2 and pad 2

$F = H \times W \times C^2 \times 9$

2.6 Consider a neural network layer that recieves an input with C channels, and spatial size $H \times W$; assume H, W, and C are divisible by 4. Let F be the number of floating-point operations required to compute the forward pass of a 3×3 convolution layer with C filters, with stride 1 and pad 1. For which of the following does computing the forward pass require more than F floating-point operations?

- ○ A: A 2×2 max-pooling layer with stride 2
- ○ B: A 5×5 convolution layer with $C/2$ filters, stride 1, and pad 2 $H \times W \times \frac{C^2}{4} \times 25$
- ○ C: A 4×4 convolution layer with $4C$ filters, stride 4, and pad 0 $\frac{W}{4} \cdot \frac{H}{4} \times 16C^2 \times 16$
- ● D: A 1×1 convolution layer with $5C$ filters, stride 1, and pad 0 followed by another 1×1 convolution layer with C filters, stride 1, and pad 0. $H \times W \times (5C)^2 +$

2.7 You train a deep convolutional network for image classification with ReLU nonlinearities and you find that no matter what learning rate you try, the training loss does not decrease much from its initial value during optimization. Which of the following changes might help the network train more effectively?

- ○ A: Replace all ReLU nonlinearities with tanh
- ○ B: Add more layers to the network
- ● C: Add batch normalization after each convolutional layer
- ● D: Use additive skip connections between every other convolutional layer

2.8 Which of the following sequences of layers have the same effective receptive field size in their input?

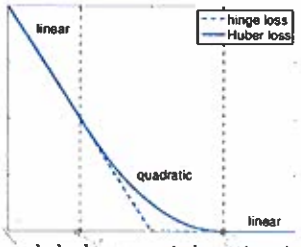
- ● A: 5×5 convolution (stride 1, pad 2) followed by 1×1 convolution (stride 1, pad 0)
- ● B: 3×3 convolution (stride 1, pad 1) followed by 3×3 convolution (stride 1, pad 1)
- ○ C: 2×2 max-pooling (stride 2, pad 0) followed by 3×3 convolution (stride 1, pad 1)
- ○ D: 3×3 convolution (stride 2, pad 1) followed by 2×2 max-pooling (stride 2, pad 0)

3 Short Answer (52 points)

3.1 Hinge Loss Variants (22 points)

In class, we learned that one characteristic of multi-class hinge loss was that it was not differentiable at the hinge point; in this question we consider two variations that address this problem.

Huber loss: One variant of the Hinge loss is the Huber classification loss; the following plot shows the comparison between the two.



Recall that given input vector x_i, true label y_i, weight matrix W, the class score is calculated as $s = x_i^T W = (s_1, s_2, \ldots s_C)$, and the hinge loss of class $j \neq y_i$ for some given margin $\Delta > 0$ is

$$L_j(x_i) = \max(s_j - s_{y_i} + \Delta, 0)$$

Now given parameters a, b, and $\Delta > 0$, the Huber classification loss is:

$$L_j(x_i) = \begin{cases} a \cdot \max(s_j - s_{y_i} + \Delta, 0)^2, & \text{if } s_j \leq s_{y_i} \\ b \cdot (s_j - s_{y_i} + 0.5\Delta), & \text{otherwise} \end{cases}$$

(a) **(5 points)** Same as the multiclass hinge loss, the Huber loss incurred by one point is the sum of the losses over each incorrect class, i.e. $L(x_i) = \sum_{j \neq y_i} L_j(x_i)$. Derive the partial derivative of the single-datapoint Huber loss $\frac{\partial L}{\partial s_j}$ for $j \neq y_i$ (No regularization term needed.)

(b) **(6 points)** For this loss function to be differentiable everywhere, the function value and the first order partial derivative have to be consistent at the hinges, i.e. the function values and gradients obtained by approaching from $s_j \leq s_{y_i}$ and from $s_j > s_{y_i}$ should be the same. Given a fixed Δ, what constraint(s) should the values of a and b satisfy?

Smooth Hinge Loss: Now let's consider the following form:

$$L = \max(\max_{j \neq y_i}\{s_j\} - s_{y_i}, 0)$$

This is slightly different than what we've seen in class, where instead of summing over all the non-ground-truth classes, we take the maximum of them.

(c) **(5 points)** Please rewrite the above expression into a smooth (i.e. differentiable) version of hinge loss using the following continuous relaxation:

$$\max(x, y) = \lim_{k \to \infty} \frac{1}{k} \log(e^{kx} + e^{ky}) \approx \log(e^x + e^y) \quad (1)$$

$$\max(x_1, \ldots, x_n) = \lim_{k \to \infty} \frac{1}{k} \log(\sum_{i=1}^{n} e^{kx_i}) \approx \log(\sum_{i=1}^{n} e^{x_i}) \quad (2)$$

Your answer should resemble a loss function that we've seen in class.

$\max_{j \neq y_i}\{s_j\} \approx \log\left(\sum_{j \neq y_i} e^{s_j}\right)$

$\Rightarrow L \approx \max\left(\log\left(\sum_{j \neq y_i} e^{s_j}\right) - s_{y_i}, 0\right)$ using (2)

$\approx \log\left(e^{\left(\log\left(\sum_{j \neq y_i} e^{s_j}\right) - s_{y_i}\right)} + e^0\right)$ using (1)

$= \log\left(\sum_{j \neq y_i} e^{s_j} / e^{s_{y_i}} + 1\right)$

$= \log\left(\sum_{j} e^{s_j} / e^{s_{y_i}}\right) = -\log\left(e^{s_{y_i}} / \sum_{j} e^{s_j}\right)^{-1}$ softmax loss

(d) **(6 points)** A margin Δ is usually introduced in hinge loss to encourage large intra-class distances. This takes the form

$$L = \max(\max_{j \neq y_i}\{s_j\} - s_{y_i} + \Delta, 0)$$

Using your answer above, please write down the smooth hinge loss when margin Δ is used.

$L \approx \max\left(\log\left(\sum_{j \neq y_i} e^{s_j}\right) - s_{y_i} + \Delta, 0\right)$

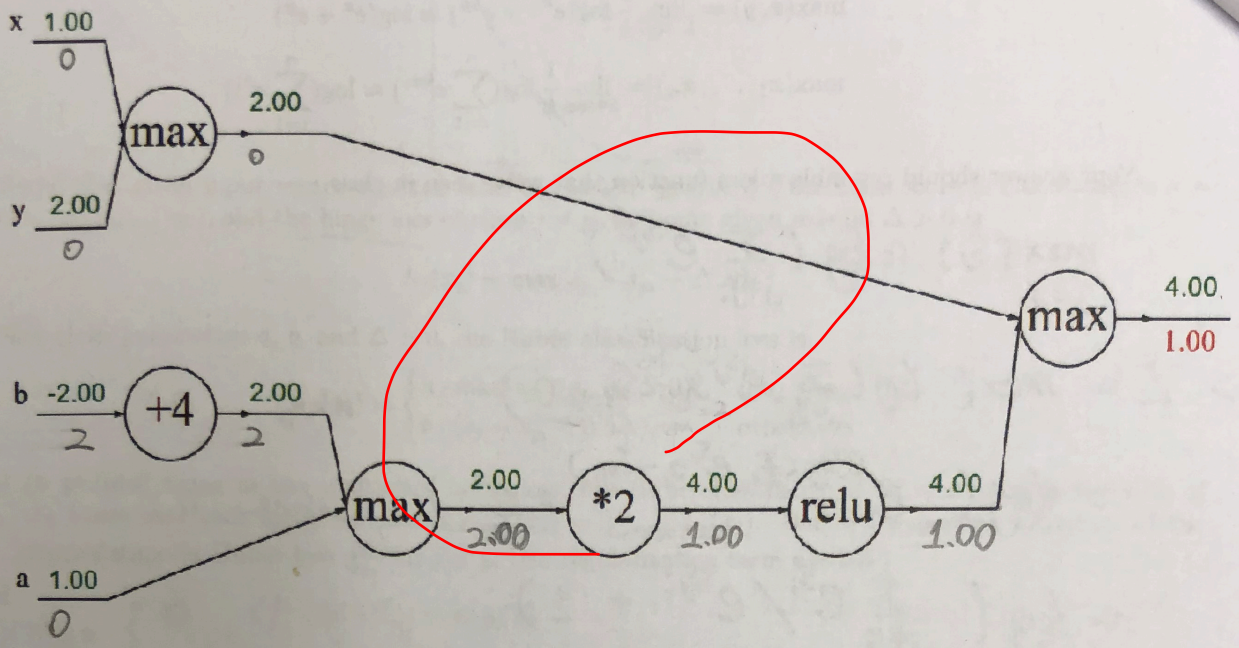
$= \max\left(\log\left(\sum_{j \neq y_i} e^{s_j}\right) - s_{y_i}, -\Delta\right) + \Delta$

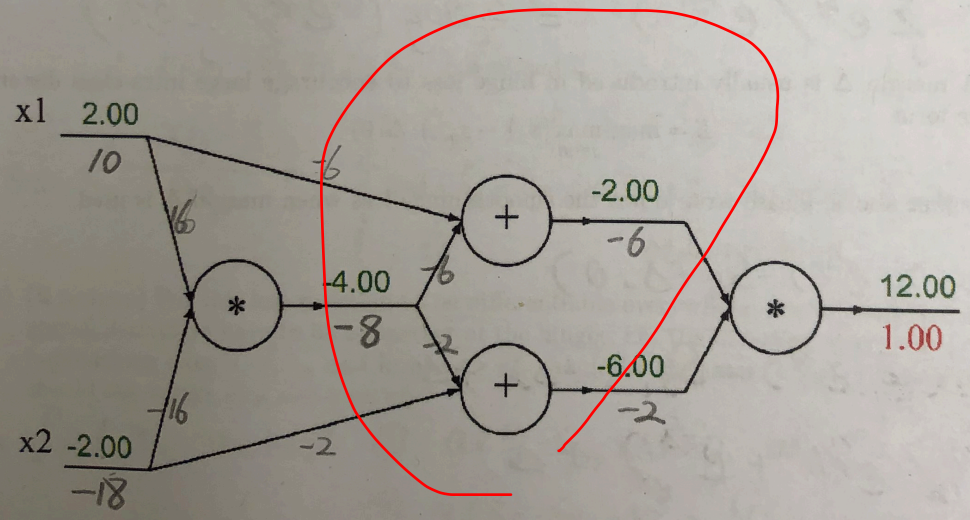
$\approx \log\left(\sum_{j \neq y_i} e^{s_j} / e^{s_{y_i}} + e^{-\Delta}\right) + \Delta$

$= \log\left(\sum_{j \neq y_i} e^{s_j} + e^{-\Delta} e^{s_{y_i}}\right) + \Delta$

$= -\log\left(e^{s_{y_i}} / \left(\sum_{j \neq y_i} e^{s_j} + e^{s_{y_i} - \Delta}\right)\right) + \Delta$

should fill in a gradient value below each activation value. (We have already end for you). (5 points each)





3.3 Convolutional Network Architectures (20 points)

(a) **(8 points)** Consider the convolutional network defined by the layers in the left column below. The input to the network is a single three-channel image (RGB) with spatial size 32 × 32. Fill in the size of the activation volumes (*channels* × *height* × *width*) output from each layer, and the number of parameters at each layer (weights and biases). You may write your answer as a multiplication and a sum (e.g. 3 × 128 × 128 + 3). You must fill out all cells of the table to receive full credit.

- CONV3-N is a convolutional layer (with bias) with N 3 × 3 filters and stride 1 pad 1.
- POOL2 is a 2 × 2 max-pooling layer with stride 2, pad 0.
- FC-N is a fully-connected layer (with bias) with N outputs. If necessary this layer flattens its input into a vector before computing its output.

Layer	Activation Volume Dimensions (memory)	Number of parameters
INPUT	3 × 32 × 32	0
CONV3-8	8 × 32 × 32	8×3×3×3 + 8
CONV3-16	16 × 32 × 32	16×3×3×8 + 16
POOL2	16 × 16 × 16	0
CONV3-32	32 × 16 × 16	32×3×3×16 + 32
CONV3-32	32 × 16 × 16	32×3×3×32 + 32
POOL2	32 × 8 × 8	0
FC-256	256	32×8×8×256 + 256
FC-10	10	256×10 + 10

(b) **(3 points)** Suppose that we wanted to process larger images of size 3 × 128 × 128 (i.e. four times as wide and high) with the same architecture. How many more parameters would this involve and where would they come from (if any)?

We will [illegible] 32×256×(32×32 − 8×8) more parameters and they all come from FC-256.

(c) **(3 points)** Suppose that we wanted to process images at the original size (32 × 32), but in grayscale (so the input image only has a single luminance channel). How does the total number of parameters in the ConvNet change?

The number of parameters in CONV3-8 will decrease to 8×3×3×1 + 8

So the total number of parameters will drop by 8×3×3×2 = 144

(d) **(3 points)** Your friend wants to use a ConvNet like the one above, but they want to have as few parameters as possible. What is a sensible change you would recommend to your friend for reducing the number of parameters, but still keeping reasonable performance? Briefly explain.

① using a global sum at the end instead of FC-256 and FC-10 since the main amount of parameters come from the two fully connected [illegible]

② replace 3×3 convolution with [illegible]. [illegible] CONV3-32 [illegible] 32×3×3 + 32×3[illegible]

(e) **(3 points)** Your other friend is mostly concerned about the memory taken up while computing the activations of the network. What changes to the above architecture would you recommend to your friend and why?

① use [illegible] downsampling at the very beginning to decrease [illegible] of images, thus the memory [illegible]

② [illegible] the kernel size and stride [illegible] layers ⇒ reduce feature size [illegible] stride [illegible] feature [illegible].