CSE 493 G1/ 599 G1
Deep Learning
Spring 2025 Practice Exam

SOLUTIONS

May 15, 2025

Full Name: _____

UW Net ID: _____

Question		Score
True/False	(20 pts)	
Multiple Choice	(40 pts)	
Backpropagation	(20 pts)	
Convolution & Pooling	(20 pts)	
Total	(100 pts)	

Welcome to the CSE 493 G1 Exam!

- The exam is 80 min and is **double-sided**.
- No electronic devices are allowed.

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

Signature: _____ Date: _____

Good luck!

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

1 True / False (20 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 Training on higher-level image features (e.g. color histograms or Histogram of Gradient features) instead of raw pixels improves image classification accuracy across all types of models.

◯ True

◯ False

SOLUTION:
False. While we saw better performance in A1 for an SVM and a two-layer neural network, with more expressive models like neural networks (and sufficient training data), it is often better to allow the model to learn its own features from raw pixels directly.

1.2 If your datapoints are linearly separable, then training a linear classifier with a Softmax loss and no regularization will achieve zero training loss at convergence.

◯ True

◯ False

SOLUTION:
False. The output of the softmax function only achieves 1 in the limit as values to go +infinity, and therefore in practice a loss of exactly 0 will not be observed.

1.3 During training, you examine the loss curve, and you notice that your training loss is plateauing out very early at a high value. One valid method to try is to increase the learning rate.

◯ True

◯ False

SOLUTION:
FALSE: The learning curve indicates that the learning rate is already too high, so increasing it would not help.

1.4 If the loss function is convex, Stochastic Gradient Descent always reaches the global minimum.

◯ True

◯ False

SOLUTION:
False, when step size is large, it is not guaranteed

1.5 Unlike the sigmoid activation function, tanh does not suffer from vanishing gradients.

◯ True

◯ False

SOLUTION:
False. The backpropagated gradient for tanh is in fact less than or equal to the upstream gradient so it can cause gradients to die in deeper networks.

1.6 A convolutional layer can express any function a fully connected layer can express.

◯ True

◯ False

SOLUTION:
False. It's the opposite: fully connected layers are actually strictly more expressive than convolutional layers, which are a particular instance of a weight matrix multiplication. For example, a fully connected layer could learn to process the top pixels of an image differently than the bottom pixels, while a convolutional layer applies the same kernel to all regions of the image. This is usually a good thing, since we want to detect an object the same way regardless of where it is in the image. It is this property of convolutions that makes them efficient and generalizable, but we are necessarily giving up some expressivity. Note that this question is asking about the general case, so you cannot assume for example that the kernel is the same size as the input.

1.7 If Model A has a lower test loss on a dataset than Model B, then Model A must have a higher accuracy on the test dataset than Model B.

◯ True

◯ False

SOLUTION:
False. Consider a dataset of 2 images, image 1 has ground truth value of Cat and and Image 2 has ground truth value of Dog. Let Model A and Model B have SVM loss with margin 1. Model A says that the first image score is [Dog: 1.1, Cat: 1.0] and the second image score is [Dog: 1.0, Cat: 1.1]. It therefore has accuracy of 0, but loss of 2.2. Model B says that the first image score is [Dog: 1.0, Cat: 1.1] and the second image score is [Dog: 1.0, Cat: 200.0]. This model has accuracy of 50%, but loss 200.9

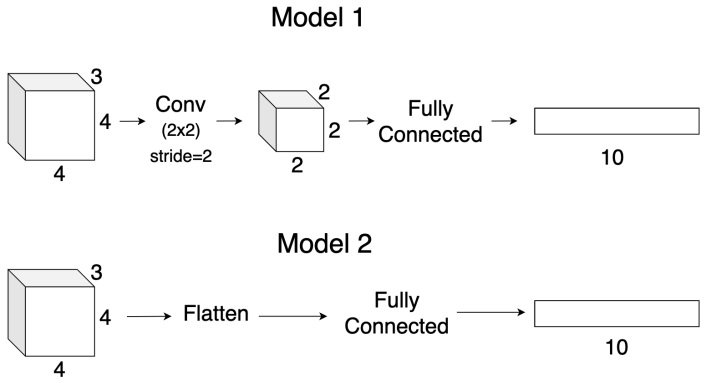
1.8 Consider a fully connected layer $\mathbf{W}$ just before a ReLU function in a network. If an element $w_{i,j}$ of the weight matrix $\mathbf{W}$ has a negative value then the gradient of the loss with respect to this weight is guaranteed to be zero.

◯ True

◯ False

SOLUTION:
False. ReLU acts on the activations, not the weights.

1.9 Consider 2 models which take as input an image of dimensions $4 \times 4 \times 3$ and output scores over 10 classes. Model 1 is made up of 1 convolution and 1 fully connected layer. Model 2 is made up of 1 fully connected layer. All input and output dimensions are noted in the figure above. Assert the following statement: Model 1 has more parameters than Model 2.



◯ True

◯ False

SOLUTION:
False. Model 1 has 2*2*3*2 + 2*2*2*10 = 104 params. Model 2 has 4*4*3*10 = 480 params.

1.10 You train one model with L2 regularization (model A) and one without (model B). The weights of model A will most likely be smaller in magnitude than those of model B.

◯ True

◯ False

SOLUTION:
True

2 Multiple Choices (40 points)

Fill in the circle next to the letter(s) of your choice (like this: ●). No explanations are required. **Choose ALL options that apply.**

Each question is worth 5 points and the answer may contain one or more options. Selecting all of the correct options and none of the incorrect options will get full credits. For questions with multiple correct options, each incorrect or missing selection gets a 2.5-point deduction (up to 5 points).

2.1 Which of the following statements are correct about kNN and Linear SVM?

- ◯ A: kNN has less runtime complexity than linear SVM during test-time.
- ◯ B: kNN often works better than linear SVM for highly nonlinear data.
- ◯ C: kNN does not require high memory for storing data, while linear SVM does.
- ◯ D: In kNN, a larger k results in a smoother classification boundary.
- ◯ E: None of the above.

SOLUTION:
BD. A) kNN has O(n) complexity, while SVM has O(1). C) kNN needs to store all of the training data.

2.2 Given the 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 increase the model's tendency to overfit on the training data 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: Using $f(x) = \text{ReLU}(-x)$ to train and test our network will lead to a much lower accuracy than $f(x) = \text{ReLU}(x)$.
- ◯ D: Choosing $f(x) = \text{LeakyReLU}(x)$ might introduce the vanishing gradient problem to our network.
- ◯ E: Choosing $f(x) = \text{ReLU}(x^3)$ helps the network train better than $f(x) = \text{ReLU}(x)$.
- ◯ F: None of the above.

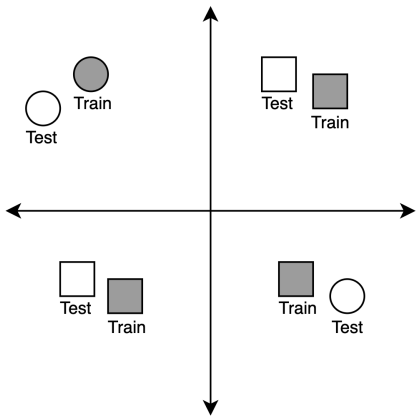
SOLUTION:
F. A) Another linear layer makes everything a linear classifier; B) same as A, it would be the same as no activation; C) It doesn't matter; D) A LeakyReLU doesn't have vanishing gradients; E) exploding gradients.

2.3 Assume an input feature map has a shape of (W_i, H_i, C_i) (where $W_i > 10, H_i > 10$), select all layers that are able to reduce the width and height dimension of the output feature map?

- ◯ A: 1×1 convolution layer with stride 1
- ◯ B: Average-pooling layer with stride 2
- ◯ C: Convolution layer with kernel size 3×3, padding 1, stride 1
- ◯ D: Batch normalization layer
- ◯ E: Convolution layer with kernel size 5×5, padding 2, stride 2
- ◯ F: None of the above

SOLUTION:
B,E

2.4 Consider the dataset pictured below. The features of each datapoint are given by its position. So the datapoint (0,1) appears at position (0,1). The ground truth label of the datapoint is given by its shape, either circle or square. You have a test set of datapoints, shown with no fill, and a train set of data, shown with a grey fill. Which of the following statements are true about classifying this data?



◯ A: It is possible for a linear SVM to have 100% train accuracy
◯ B: It is possible for a linear SVM to have 100% test accuracy
◯ C: KNN with K=1 has higher test accuracy than with K=4
◯ D: KNN with K=1 has higher train accuracy than with K=4
◯ E: None of the above

SOLUTION:
A, C, D

A is True because we are using a linear SVM and it is possible to linearly separate the train data.

B is False because it is not possible to linearly seperate the text data.

C is True because test accuracy is 75% when K=1, but 50% when K=4.

D is True because train accuracy is 100% when K=1, but 75% when K=4.

E is False

2.5 Let x be some image with ground truth label y for a classification problem between K classes. Let s_j denote a network's score for for class j. Consider the modified version of cross-entropy loss where we have variables a, b and c.

$$\mathcal{L}(x) = -\log\left(\frac{a \cdot e^{b \cdot s_y}}{\sum_{j=1}^{K} e^{s_j}}\right) + c$$

Now let $\frac{\partial \mathcal{L}}{\partial w}$ denote the gradient of the loss with respect to the final layer in the network. Which of the following is true about the relationship between $\frac{\partial \mathcal{L}}{\partial w}$, a, b, and c.

◯ A: Increasing a could increase $\frac{\partial \mathcal{L}}{\partial w}$.

◯ B: Decreasing a could increase $\frac{\partial \mathcal{L}}{\partial w}$.

◯ C: Increasing b could increase $\frac{\partial \mathcal{L}}{\partial w}$.

◯ D: Decreasing b could increase $\frac{\partial \mathcal{L}}{\partial w}$.

◯ E: Increasing c could increase $\frac{\partial \mathcal{L}}{\partial w}$.

◯ F: Decreasing c could increase $\frac{\partial \mathcal{L}}{\partial w}$.

SOLUTION:
C, D

Let us rewrite the expression as follows

$$\mathcal{L}(x) = -\log\left(\frac{a \cdot e^{b \cdot s_y}}{\sum_{j=1}^{K} e^{s_j}}\right) + c$$

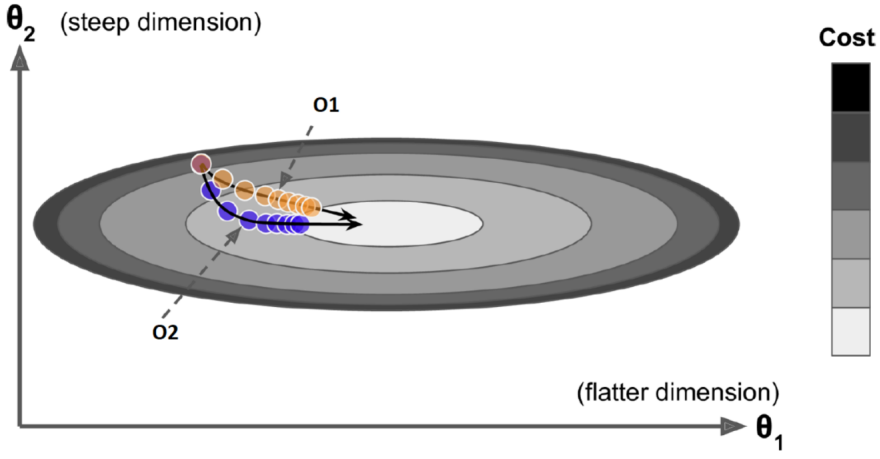
$$= -\log\left(a \cdot \frac{e^{b \cdot s_y}}{\sum_{j=1}^{K} e^{s_j}}\right) + c$$

$$= -\left(\log(a) + \log\left(\frac{e^{b \cdot s_y}}{\sum_{j=1}^{K} e^{s_j}}\right)\right) + c$$

$$= -\log(a) - \log\left(\frac{e^{b \cdot s_y}}{\sum_{j=1}^{K} e^{s_j}}\right) + c$$

We can now see that a and c cannot affect $\frac{\partial \mathcal{L}}{\partial w}$, so A, B, E and F are False. However, changing b will affect $\frac{\partial \mathcal{L}}{\partial w}$ differently depending on if the current score is positive or negative and so therefore increasing or decreasing b could increase $\frac{\partial \mathcal{L}}{\partial w}$, so C and D are True.

2.6 The figure below compares AdaGrad and Gradient Descent(without momentum) optimization. Which of the following is/are true?



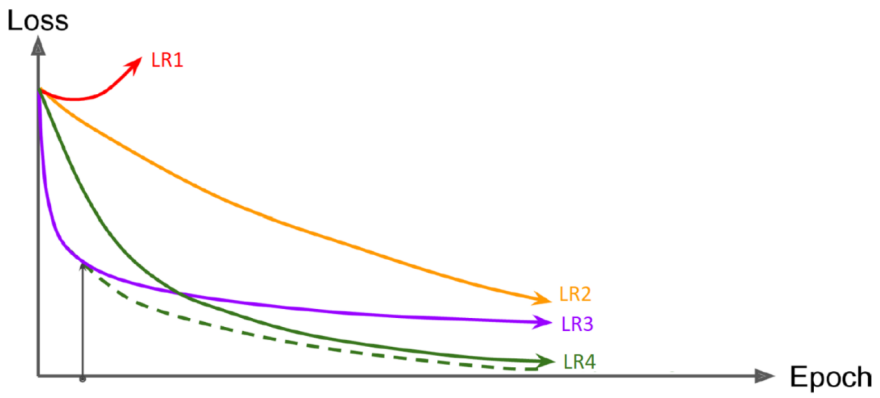
◯ A: O1 corresponds to AdaGrad while O2 corresponds to Gradient Descent

- ◯ B: O1 corresponds to Gradient Descent while O2 corresponds to AdaGrad
- ◯ C: O1 helps point the resulting updates more directly toward the global optimum compared to O2
- ◯ D: O2 helps point the resulting updates more directly toward the global optimum compared to O1

SOLUTION:
A, C

AdaGrad has a smoother trajectory than GD without momentum and generally leads to a more direct path towards the global optimum.

2.7 The below figure shows learning curves for various learning rates. What is the correct order of learning rates?



- ◯ A: $LR1 > LR2 > LR3 > LR4$
- ◯ B: $LR2 > LR3 > LR1 > LR4$
- ◯ C: $LR1 > LR3 > LR4 > LR2$
- ◯ D: $LR3 > LR1 > LR2 > LR4$

SOLUTION:
C

L1 is the largest learning rate as we see it diverges away from a low loss. Additionally, we know that LR3 > LR2 as LR3 drops quickly but then plateaus while LR2 continues to drop in loss, but very slowly. Therefore the correct answer is C.

2.8 Your notice your vanilla RNN has a vanishing gradient problem. Which one(s) of the following methods can help?

- ◯ A: Use gradient clipping
- ◯ B: Add more RNN layers.
- ◯ C: Add more training data.
- ◯ D: Replace vanilla RNN with LSTM or GRU.

SOLUTION:
D

A if False because gradient clipping can help exploding gradients, not vanishing gradients

B is False because adding more layers can only increase the vanishing gradient problem.

C is False because the amount of training data has not affect on vanishing gradients.

D is True because LSTMs and GRUs have a cell state that acts as a “gradient highway” to prevent vanishing gradients.

3 Backpropagation (20 points)

Please make sure to write your answer only in the provided space.

Consider the following function:

$$f(x_0, x_1, w_0, w_1, w_2, b_0, b_1) = max(w_2 \times (w_0 \times x_0 + w_1 \times x_1 + b_0) + b_1, 0)$$

1. (4 points) Having a computation graph for computing the function f would help you solve the rest of this question. We already give you all the input bubbles as follows. Finish the computation graph and label each edge with the values produced during the forward pass. The input values are:

$$w_0 = -1, w_1 = 3, w_2 = 4, x_0 = -3, x_1 = 2, b_0 = -8, b_1 = 1$$

.

w_0

x_0

w_1

x_1

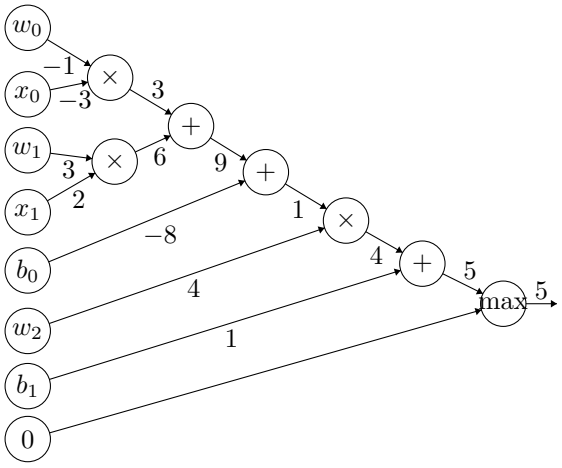
b_0

w_2

b_1

0

SOLUTION:



2. (12 points) Let function g represent the computation that is done at a node of the computation graph for f. At the computation node for g, we receive upstream gradient $\frac{\partial f}{\partial g}$. Show that the following patterns in gradient flow hold.

(a) Let $g(x, y) = x + y$. Show that g is a gradient "distributor" for $\frac{\partial f}{\partial g}$, i.e. show

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial g} \quad \text{and} \quad \frac{\partial f}{\partial y} = \frac{\partial f}{\partial g}.$$

SOLUTION:

$$\frac{\partial g}{\partial x} = 1, \quad \frac{\partial f}{\partial x} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial x} = \frac{\partial f}{\partial g} \cdot 1 = \frac{\partial f}{\partial g}$$
$$\frac{\partial g}{\partial y} = 1, \quad \frac{\partial f}{\partial y} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial y} = \frac{\partial f}{\partial g} \cdot 1 = \frac{\partial f}{\partial g}$$

(b) Let $g(x, y) = x \times y$. Show that g is a gradient "swap multiplier", i.e. show

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial g} \cdot y \quad \text{and} \quad \frac{\partial f}{\partial y} = \frac{\partial f}{\partial g} \cdot x.$$

SOLUTION:

$$\frac{\partial g}{\partial x} = y, \quad \frac{\partial f}{\partial x} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial x} = \frac{\partial f}{\partial g} \cdot y$$
$$\frac{\partial g}{\partial y} = x, \quad \frac{\partial f}{\partial y} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial y} = \frac{\partial f}{\partial g} \cdot x$$

(c) Let $g(x, y) = max(x, y)$. Show that g is a gradient "router", i.e. show

$$\frac{\partial f}{\partial x} = \begin{cases} \frac{\partial f}{\partial g} & \text{if } x \geq y \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \frac{\partial f}{\partial y} = \begin{cases} \frac{\partial f}{\partial g} & \text{if } y \geq x \\ 0 & \text{otherwise} \end{cases}.$$

SOLUTION:

$$\frac{\partial g}{\partial x} = \begin{cases} 1 & \text{if } x \geq y \\ 0 & \text{otherwise} \end{cases}, \quad \frac{\partial f}{\partial x} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial x} = \begin{cases} \frac{\partial f}{\partial g} & \text{if } x \geq y \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{\partial g}{\partial y} = \begin{cases} 1 & \text{if } y \geq x \\ 0 & \text{otherwise} \end{cases}, \quad \frac{\partial f}{\partial y} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial y} = \begin{cases} \frac{\partial f}{\partial g} & \text{if } y \geq x \\ 0 & \text{otherwise} \end{cases}$$

3. (4 points) Compute the following gradients corresponding to function f and values of the inputs given in part 1.

(a) $\frac{\partial f}{\partial w_0} =$

SOLUTION:
$\frac{\partial f}{\partial w_0} = -12$

(b) $\frac{\partial f}{\partial w_1} =$

SOLUTION:
$\frac{\partial f}{\partial w_1} = 8$

SOLUTION:
$\frac{\partial f}{\partial w_2} = 1$

(c) $\frac{\partial f}{\partial b_0} =$

SOLUTION:
$\frac{\partial f}{\partial b_0} = 4$

(d) $\dfrac{\partial f}{\partial b_1} =$

SOLUTION:
$\frac{\partial f}{\partial b_1} = 1$

4 Convolution & Pooling (20 points)

Please make sure to write your answer only in the provided space.

We have previously seen 2D convolution networks being successfully used on image data. Video data are essentially temporal series of images, where each video as an input has dimensions (channels C, time T, height H, width W).

In this problem, we look at 3D convolutions for video data, where the convolution happens not only spatially but also temporally. 3D convolution operates on 4 dimensional input tensors of size $C \times T \times H \times W$ with 3D kernels, in contrast to the 2D case where we have input size $C \times H \times W$ and 2D kernels.

1. (8 points) Consider a 3D convolutional network with its layers specified in table 1. For each layer, fill in the table with the size of the activation volumes, and the number of weight and bias parameters. The INPUT layer is provided; this network takes in 8 frames of 64-by-64 pixel RGB images.
Please write your answer as a multiplication (e.g. 128 × 128 × 128) or points may deducted.
The layer descriptions are as follows:

- CONV3-N-p1-s2 is a convolutional layer with N $3 \times 3 \times 3$ kernels, padding 1 and stride 2 (in height, width, and temporal dimension) with ReLU activation. Similarly, CONV5-N-p2-s1 is a convolutional layer with N $5 \times 5 \times 5$ kernels, padding 2 and stride 1.
- POOL2 denotes a $2 \times 2 \times 2$ max-pooling layer, with stride 2 and no padding in all dimensions.
- FC-N denotes a fully-connected layer with N neurons, with ReLU activation.

Layer	Activation Volume Dimensions	# Weights	# Biases
INPUT	$3 \times 8 \times 64 \times 64$	N/A	N/A
CONV5-16-p2-s1			
POOL2			
CONV3-16-p1-s1			
POOL2			
CONV3-32-p1-s2			
FC-256			
FC-10			

Table 1: Dimensions and parameters.

SOLUTION:

2. (4 points) In a 3D convolutional network we can define *spatiotemporal batch normalization* similarly to operate on a batch of 4D feature maps of shape $N \times C \times T \times H \times W$, normalizing over the N, T, H, and W dimensions. If we now add one spatiotemporal Batch Normalization (BN) after each convolution layer, how many additional parameters will be introduced? Please put your answers in Table 3.

SOLUTION:

Layer	Activation volume dimension	# Weights	# Biases
INPUT	$3 \times 8 \times 64 \times 64$	N/A	N/A
CONV5-16-p2-s1	$16 \times 8 \times 64 \times 64$	$5 \times 5 \times 5 \times 3 \times 16$	16
POOL2	$16 \times 4 \times 32 \times 32$	0	0
CONV3-16-p1-s1	$16 \times 4 \times 32 \times 32$	$3 \times 3 \times 3 \times 16 \times 16$	16
POOL2	$16 \times 2 \times 16 \times 16$	0	0
CONV3-32-p1-s2	$32 \times 1 \times 8 \times 8$	$3 \times 3 \times 3 \times 16 \times 32$	32
FC-256	256	$32 \times 1 \times 8 \times 8 \times 256$	256
FC-10	10	256×10	10

Table 2: Dimensions and parameters.

Layer	# BN parameters
CONV5-16-p2-s1	
CONV3-16-p1-s1	
CONV3-32-p1-s2	

Table 3: Number of BN parameters.

3. (4 points) 3D convolution network is computationally more expensive than its 2D counterpart. Assuming convolutions are implemented as matrix multiplications[1], let's analyze the computational complexity of convolutions in terms of the number of floating point multiplications they perform.

First calculate the complexity of 2D convolution, and then generalize to 3D convolution. Please write your answer as a multiplication (e.g. $128 \times 128 \times 128$) in Table 5.

SOLUTION:

4. (4 points) There are three ways to learn feature vectors for video inputs:

(a) 3D convolution as we introduced above.

(b) 2D convolution computes one feature vector for each frame independently, and stacks them together.

(c) 2D convolution computes one feature vector for each frame independently, and feeds the sequence of feature vectors to an LSTM.

If your input are long videos (e.g. with more than 100 frames), which of the 3 models above would you choose? Please rank them from the best temporal feature to the worst temporal feature, and explain why. (Hint: compare the temporal receptive field of the 3 models. You don't need to consider the computation efficiency here.)

SOLUTION:
(c) > (a) > (b)

2D-CNN + LSTM is preferred for modeling long sequences, since it can potentially have an infinite large temporal receptive field.

[1]Fun fact: there exist alternative ways of computing convolutions such as Fast Fourier Transform / Winograd convolutions, but don't consider them for this problem.

Layer	# BN parameters
CONV5-16-p2-s1	16×2
CONV3-16-p1-s1	16×2
CONV3-32-p1-s2	32×2

Table 4: Number of BN parameters.

Input tensor dimension	Convolutional layer type	# floating point multiplication
$3 \times 64 \times 64$	2D Conv, N=16, kernel=3×3, pad=1, stride=1	
$3 \times 64 \times 64$	2D Conv, N=32, kernel=5×5, pad=2, stride=1	
$3 \times 8 \times 64 \times 64$	3D Conv, N=32, kernel=$5 \times 5 \times 5$, pad=2, stride=1	

Table 5: Computation costs.

2D-CNN stacked doesn't aggregate any temporal information across the frames.

Input tensor dimensions	Convolutional Layer Type	# floating point multiplication
$3 \times 64 \times 64$	2D Conv, N=16, kernel=3×3, pad=1, stride=1	$3 \times 3 \times 64 \times 64 \times 3 \times 16$
$3 \times 64 \times 64$	2D Conv, N=32, kernel=5×5, pad=2, stride=1	$5 \times 5 \times 64 \times 64 \times 3 \times 32$
$3 \times 8 \times 64 \times 64$	3D Conv, N=32, kernel=$5 \times 5 \times 5$, pad=2, stride=1	$5 \times 5 \times 5 \times 8 \times 64 \times 64 \times 3 \times 32$

Table 6: Computation costs.