

Short Answer (10 Points)

1. Give two reasons why might we choose the ReLU activation function over a sigmoid activation function in a deep neural network? (5 points)

- Easy derivatives (0 or 1)
- Removes negative values
- Doesn't contribute to vanishing gradient

2. Explain the intuition behind Adaboost. Why do we use it? What does it give us?

Adaboost intends to create one strong learner by linearly combining weak learners.

We use it to take advantage of the strengths of many learners.

The idea is that many learners won't make the same mistakes, so we'll get better results when we create an ensemble of these methods.

Short Answer (10 Points)

3. List and define 5 hyperparameters for deep learning. (5 points)

learning rate - how much you update each weight
step size - how often you lower the LR.
momentum - amount of old update to use
layers - layers in the DNN.
non-linearity - what nonlinear function to apply?
where.

4. True/False: PCA always produces orthogonal eigenvectors. Briefly explain. (5 points)

PCA produces a new set of axis that describe the data's variance from largest to smallest.

The Covariance matrix is positive semidefinite so its eigenvectors are always \perp .

Adaboost (20 Points)

5. Assume you have four data points and you have four classifiers. The table below details which samples each classifier incorrectly classifies. Run two full iterations of adaboost, choosing the classifier with the lowest error at each iteration. Break ties by choosing h_i over h_k if $i < k$. In the table provided, indicate which classifier you chose, its error rate and the weights for each sample.

Classifier	Incorrect
h_1	s_1
h_2	s_1, s_2
h_3	s_2, s_3
h_4	s_4

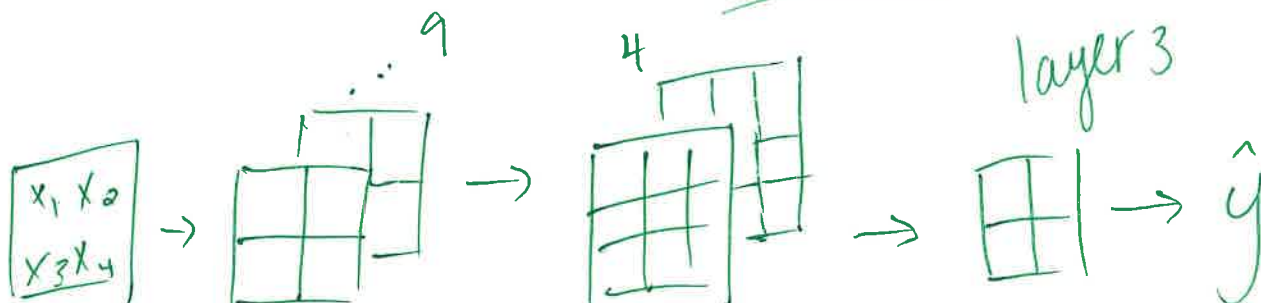
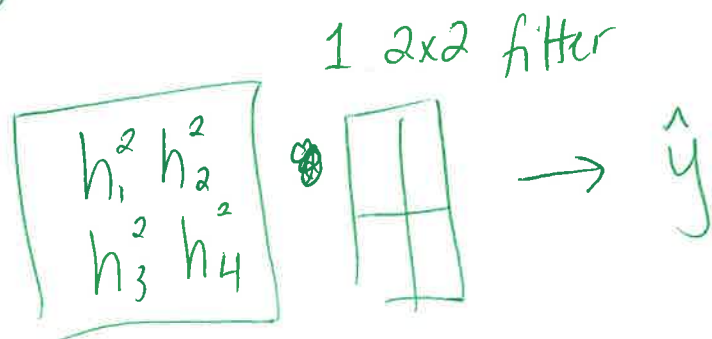
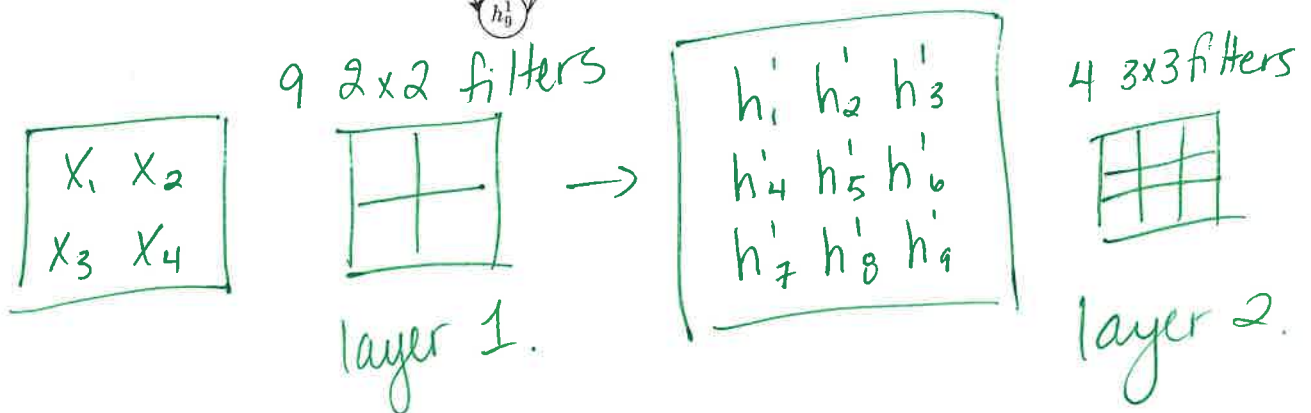
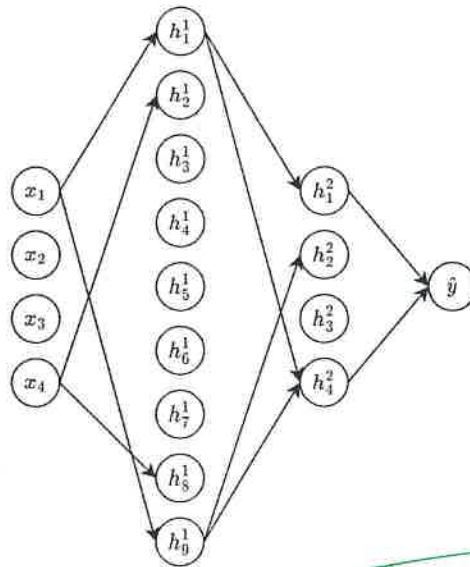
Sample	Iter 0	Iter 1	Iter 2
s_1	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{3}{10}$
s_2	$\frac{1}{4}$	$\frac{1}{6}$	$\frac{1}{10}$
s_3	$\frac{1}{4}$	$\frac{1}{6}$	$\frac{1}{10}$
s_4	$\frac{1}{4}$	$\frac{1}{6}$	$\frac{1}{2}$
h	h_1	h_4	h_3
Error	$\frac{1}{4}$	$\frac{1}{6}$	$\frac{2}{10}$

$$\frac{1}{2} + \frac{1}{6} + \frac{1}{6} = \frac{5}{6} \quad X = \frac{1}{2} \quad X = \frac{3}{5}$$

$$\frac{1}{2} \cdot \frac{3}{5} = \frac{3}{10} \quad \frac{1}{6} \cdot \frac{3}{5} = \frac{1}{10}$$

Deep Learning (10 Points)

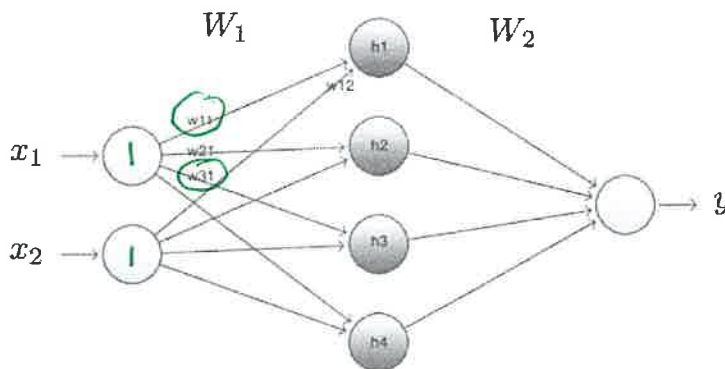
6. Convert the following fully connected network to an equivalent convolutional neural network. That is, the networks should be equivalent in the number and application of weights to inputs, and hidden nodes. Assume the network is fully connected. I just didn't include all the connections because it takes too long to draw. Explain your choices.



Neural Networks (20 Points)

only.

7. Given the following neural network, perform back propagation to update the following weights: w_{11} and w_{31} . Assume we are using a ReLU nonlinearity at the hidden nodes, a squared loss function $L(y, \hat{y}) = (y - \hat{y})^2$. Use the weights in the following table to start. Assume we have $x = (1, 1)$ as our input and $y = 1$ as our label. Also assume our learning rate is $\eta = 0.1$.



w_{11}	w_{21}	w_{31}	w_{41}	w_{12}	w_{22}	w_{32}	w_{42}	w_1^2	w_2^2	w_3^2	w_4^2
1	-1	2	-1	-2	1	1	2	-1	-2	1	0

$$h_1 = \text{relu}(1 \cdot 1 + -2 \cdot 1) = 0$$

$$h_2 = \text{relu}(-1 \cdot 1 + 1 \cdot 1) = 0$$

$$h_3 = \text{relu}(2 \cdot 1 + 1 \cdot 1) = 3$$

$$h_4 = \text{relu}(-1 \cdot 1 + 2 \cdot 1) = 1$$

$$\hat{y} = -1 \cdot 0 - 2 \cdot 0 + 1 \cdot 3 + 0 \cdot 1 = 3$$

$$\frac{\partial L}{\partial \hat{y}} = -2(y - \hat{y}) = -2(1 - 3) = 4$$

$$\frac{\partial \hat{y}}{\partial h_1} = w_{11}^2 = -1 \quad \partial \text{relu} = 0$$

$$\frac{\partial \hat{y}}{\partial h_3} = w_{31}^2 = 1$$

$$\frac{\partial h_1}{\partial w_{11}} = x_1 \cdot \partial \text{relu} = 1 \cdot 0$$

$$\frac{\partial h_3}{\partial w_{31}} = x_1 \cdot \partial \text{relu} = 1 \cdot 1$$

$$w_{11} = w_{11} - \eta \cdot 0 = w_{11} = 1$$

$$w_{31} = w_{31} - \eta (4 \cdot 1 \cdot 1) = 2 - 0.4 = 1.6$$

CNNs (10 Points)

8. Assume you have a CNN. The input is of size 10×10 and you can apply filters each of size 3×3 . How many layers would you need to get an output of size 2×2 ? How many parameters would this network have? Briefly explain.

$$10 \times 10 \xrightarrow{1} 8 \times 8 \xrightarrow{2} 6 \times 6 \xrightarrow{3} 4 \times 4 \xrightarrow{4} 2 \times 2$$

4 layers.

$$3 \times 3 \times 4 = 36 \text{ parameters}$$

(assuming no bias)

PCA (20 Points)

9. Use PCA to find the principal components of the following data. No need to center or standardize the data. Project the following point onto the first principal component: (3,3). Project that same point onto both of the principal components.

Sample	x_1	x_2	y
s_1	0	1	1
s_2	1	0	1
s_3	1	1	1
s_4	2	2	-1
s_5	2	1	-1
s_6	1	2	-1

$$\begin{bmatrix} 0 & 1 & 1 & 2 & 2 & 1 \\ 1 & 0 & 1 & 2 & 1 & 2 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 1 \\ 2 & 2 \\ 2 & 1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 11 & 9 \\ 9 & 11 \end{bmatrix}$$

$$(11-\lambda)(11-\lambda) - 81 = 0$$

$$121 - 22\lambda + \lambda^2 - 81 = 0$$

$$\lambda^2 - 22\lambda + 40 = 0$$

$$(\lambda - 2)(\lambda - 20) = 0$$

$$\lambda = 20, 2$$

$$\begin{bmatrix} 11 & 9 \\ 9 & 11 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{matrix} 20 \\ \text{or} \\ 2 \end{matrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

$$11v_1 + 9v_2 = 20v_1 \Rightarrow 9v_2 = 9v_1$$

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \lambda = 20$$

$$11v_1 + 9v_2 = 2v_1 \Rightarrow 9v_2 = -9v_1$$

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix} \lambda = 2$$

normalize $\begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \quad \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix}$



Equations

Adaboost

Algorithm 32 $\text{AdaBoost}(\mathcal{W}, \mathcal{D}, K)$

```
1:  $d^{(0)} \leftarrow \langle \frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N} \rangle$   
2: for  $k = 1 \dots K$  do  
3:    $f^{(k)} \leftarrow \mathcal{W}(\mathcal{D}, d^{(k-1)})$   
4:    $\hat{y}_n \leftarrow f^{(k)}(x_n), \forall n$   
5:    $\hat{\epsilon}^{(k)} \leftarrow \sum_n d_n^{(k-1)} [y_n \neq \hat{y}_n]$   
6:    $\alpha^{(k)} \leftarrow \frac{1}{2} \log \left( \frac{1 - \hat{\epsilon}^{(k)}}{\hat{\epsilon}^{(k)}} \right)$   
7:    $d_n^{(k)} \leftarrow \frac{1}{2} d_n^{(k-1)} \exp[-\alpha^{(k)} y_n \hat{y}_n], \forall n$   
8: end for  
9: return  $f(\hat{x}) = \text{sgn} [\sum_k \alpha^{(k)} f^{(k)}(\hat{x})]$ 
```

PCA

$$\text{Cov} = (X - \bar{X})(X - \bar{X})^T$$

$$|A - \lambda I| = 0$$

$$X^* = UX$$

$$[3, 3] \cdot \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = \boxed{\frac{6}{\sqrt{2}}}$$

$$[3, 3] \cdot \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} = 0$$

$$\begin{bmatrix} \frac{6}{\sqrt{2}} \\ 0 \end{bmatrix}$$

Short Answer (10 Points)

1. Give two reasons why we might choose the ReLU activation function over a sigmoid activation function in a deep neural network? (5 points)

Same as 422 #1

2. In theory, we only need 2 layers for a neural network to approximate any function. Why does this not work that well in practice, and what can we do to improve performance? (5 points)

The model becomes arbitrarily large, with too many weights. It also doesn't allow for learning hierarchical representations.

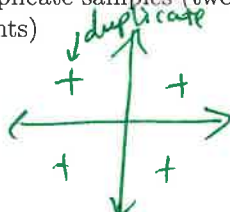
Increasing depth (# of layers) allows us to approximate complex functions with fewer parameters.

Short Answer (10 Points)

3. If I give you one principal component in 2D can you recover the other one with no additional information? If yes, how? If no, why not? (5 points)

yes. Find a vector \perp to the PC.
This will be our second P.C.
Normalize, of course.

4. Do duplicate samples (two samples with the same features) affect the output of PCA? Briefly explain. (5 points)



$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \\ -1 & 1 \\ -1 & -1 \end{bmatrix} \rightarrow \text{covariance matrix}$$

$$\begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}$$

eigenvectors $\rightarrow \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \\ -1 & 1 \\ -1 & -1 \end{bmatrix} \rightarrow \text{covariance matrix}$$

$$\begin{bmatrix} 5 & -1 \\ -1 & 5 \end{bmatrix}$$

$$(5-\lambda)(5-\lambda)+1=0$$

$$\lambda^2 - 10\lambda + 25 + 1 = 0$$

$$\lambda^2 - 10\lambda + 26 = 0$$

yes. This isn't perfect square.

Adaboost (20 Points)

5. Assume you have four data points and you have four classifiers. The table below details which samples each classifier incorrectly classifies. Run two full iterations of adaboost, choosing the classifier with the lowest error at each iteration. Break ties by choosing h_i over h_k if $i < k$. In the table provided, indicate which classifier you chose, its error rate and the weights for each sample.

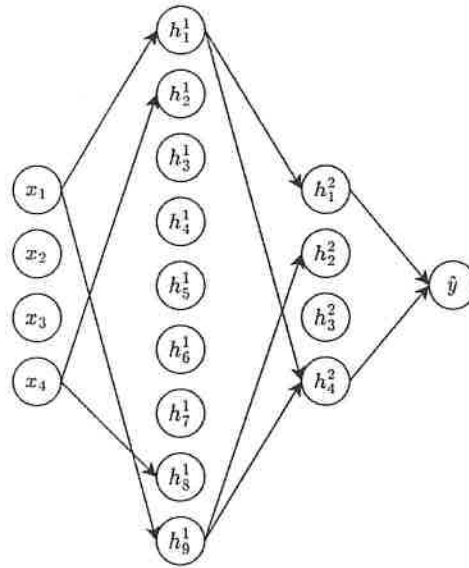
Classifier	Incorrect
h_1	s_1
h_2	s_1, s_2
h_3	s_2, s_3
h_4	s_4

Sample	Iter 0	Iter 1	Iter 2
s_1			
s_2			
s_3			
s_4			
h			
Error			

Same as 422 #5.

Deep Learning (10 Points)

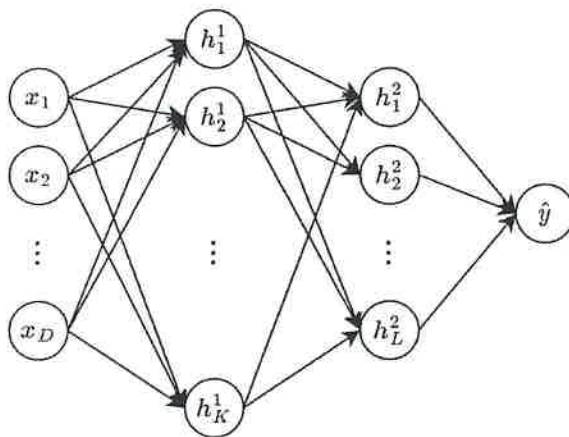
6. Convert the following fully connected network to an equivalent convolutional neural network. That is, the networks should be equivalent in the number and application of weights to inputs, and hidden nodes. Assume the network is fully connected. I just didn't include all the connections because it takes too long to draw. Explain your choices.



Same as 422 #6.

Neural Networks (20 Points)

7. Given the following neural network, give the general formula for updating a weight in the first layer (w_{ji}^1 , the weight going from input i to hidden node h_j^1). Weights in the second layer are w_{kj}^2 (the weight going from hidden node h_j^1 to h_k^2) and weights in the third layer are w_k^3 (the weight going from h_k^2 to \hat{y}). Assume your non-linearity is $z = (h)^2$ (you can think of z as being the output of each hidden node, where you just square the h value) and your loss is $L = (y - \hat{y})^2$. For the formula, do not simply write $\frac{\partial L}{\partial \hat{y}} \times \dots$, plug in the expressions for these. Please keep everything clean and clear for partial credit!



$$\frac{\partial L}{\partial w_{ji}^1} = \frac{\partial L}{\partial \hat{y}} \left[\sum_{k=1}^L \frac{\partial \hat{y}}{\partial z_k^2} \frac{\partial z_k^2}{\partial h_k^2} \frac{\partial h_k^2}{\partial z_j^1} \frac{\partial z_j^1}{\partial h_j^1} \right] \frac{\partial h_j^1}{\partial w_{ji}^1}$$

$$\frac{\partial L}{\partial \hat{y}} = -2(y - \hat{y}) \quad \frac{\partial \hat{y}}{\partial z_k^2} = w_k^3 \quad \frac{\partial z_k^2}{\partial h_k^2} = 2h_k^2$$

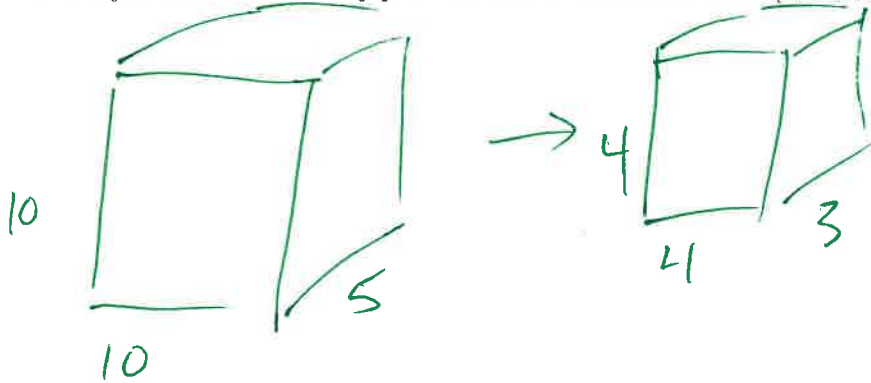
$$\frac{\partial h_k^2}{\partial z_j^1} = w_{kj}^2 \quad \frac{\partial z_j^1}{\partial h_j^1} = 2h_j^1$$

$$\frac{\partial h_j^1}{\partial w_{ji}^1} = x_i \quad \frac{\partial L}{\partial w_{ji}^1} = -2(y - \hat{y}) \left[\sum_{k=1}^L w_k^3 2h_k^2 w_{kj}^2 2h_j^1 \right] \cdot x_i$$

CNNs (10 Points)

8. Assume you have a CNN with one layer (input to output). The input is of size $10 \times 10 \times 5$. Assume your output is of size $4 \times 4 \times 3$. Assume you have 1 pixel of zero-padding on each side of the image. What size is your filter? How many parameters does this CNN have? (No need to include bias terms).

Stride
= 1.



$$10 \times 10 + 1 \text{ pixel padding} = 12 \times 12$$

$$12 - k + 1 = 4$$

$$13 - k = 4$$

$$9 = k.$$

$9 \times 9 \times 5$ and there
are 3 of them

$9 \times 9 \times 5 \times 3$ parameters

PCA (20 Points)

9. Use PCA to find the principal components of the following data. No need to center or standardize the data. Project the following point onto the first principal component: $(3,3)$. Project that same point onto both of the principal components.

Sample	x_1	x_2	y
s_1	0	1	1
s_2	1	0	1
s_3	1	1	1
s_4	2	2	-1
s_5	2	1	-1
s_6	1	2	-1

Same as 422 #9.

Equations

Adaboost

Algorithm 32 ADABOOST($\mathcal{W}, \mathcal{D}, K$)

```
1:  $\vec{d}^{(0)} \leftarrow \langle \frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N} \rangle$   
2: for  $k = 1 \dots K$  do  
3:    $f^{(k)} \leftarrow \mathcal{W}(\mathcal{D}, \vec{d}^{(k-1)})$   
4:    $\hat{y}_n \leftarrow f^{(k)}(x_n), \forall n$   
5:    $\hat{e}^{(k)} \leftarrow \sum_n d_n^{(k-1)} [y_n \neq \hat{y}_n]$   
6:    $\alpha^{(k)} \leftarrow \frac{1}{2} \log \left( \frac{1 - \hat{e}^{(k)}}{\hat{e}^{(k)}} \right)$   
7:    $d_n^{(k)} \leftarrow \frac{1}{Z} d_n^{(k-1)} \exp[-\alpha^{(k)} y_n \hat{y}_n], \forall n$   
8: end for  
9: return  $f(\hat{x}) = \text{sgn} [\sum_k \alpha^{(k)} f^{(k)}(\hat{x})]$ 
```

PCA

$$Cov = (X - \bar{X})(X - \bar{X})^T$$

$$|A - \lambda I| = 0$$

$$X^* = UX$$