Instructions: This assignment contains 8 pages (including this cover page) and 4 questions. The maximum number of points you can earn on the assignment is 10. Assignment (hard copy) due before class on Nov 28.

Grade Table

<table>
<thead>
<tr>
<th>Question</th>
<th>Points</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Total:</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

1. (3 points) **True or False**

(a) (0.25 points) The XOR operator can be modeled using a neural network with a single hidden layer (i.e. 3-layer network).

A. True  B. False

(b) (0.25 points) Backpropagation is motivated by utilizing Chain Rule and Dynamic Programming to conserve mathematical calculations.

A. True  B. False

(c) (0.25 points) All neural networks compute non-convex functions of their parameters.

A. True  B. False

(d) (0.25 points) A multi-layer neural network model trained using stochastic gradient descent on the same dataset with different initializations for its parameters is guaranteed to learn the same parameters.
(e) (0.25 points) Stochastic gradient descent results in a smoother convergence plot (loss vs epochs) as compared to batch gradient descent.

A. True  
B. False

(f) (0.25 points) Convolutional neural networks generally have fewer free parameters as compared to fully connected neural networks.

A. True  
B. False

(g) (0.25 points) The loss function for LeNet5 (the convolutional neural network by LeCun et al.) is convex.

A. True  
B. False

(h) (0.25 points) Convolution is a linear operation i.e. $(\alpha f_1 + \beta f_2) \ast g = \alpha f_1 \ast g + \beta f_2 \ast g$.

A. True  
B. False

(i) (0.25 points) The input image has been converted into a matrix of size 28 X 28 and a kernel/filter of size 7 X 7 with a stride of 1. The size of the convoluted matrix is 21 X 21.

A. True  
B. False

(j) (0.25 points) In the neural network, every parameter can have their different learning rate.

A. True  
B. False

(k) (0.25 points) Dropout can be applied at visible layer of Neural Network model?

A. True  
B. False

(l) (0.25 points) Increasing the number of hidden nodes in a multilayer perceptron improves generalization.

A. True  
B. False
2. (2 points) **Multiple Choice**

(a) (0.25 points) Given function \( f(x) = |x^2 + 3| = 1 \) defined on \( \mathbb{R} \):

A. Newton’s Method on minimizing gradients will always converge to the global optimum in one iteration from any starting location
B. The problem is nonconvex, so it is not feasible to find a solution
C. Stochastic Gradient Descent will always converge to the global optimum in one iteration
D. All of the above
E. None of the above

(b) (0.25 points) Neural networks:

A. optimize a convex cost function
B. can be used for regression as well as classification
C. always output values between 0 and 1
D. can be used in an ensemble

(c) (0.25 points) Which of the following are true of CNNs for image analysis?:

A. Filters in earlier layers tend to include edge detectors
B. Pooling layers reduce the spatial resolution of the image
C. They have more parameters than fully-connected networks with the same number of layers and the same numbers of neurons in each layer
D. A CNN can be trained for unsupervised learning tasks, whereas an ordinary neural net cannot

(d) (0.25 points) In neural networks, nonlinear activation functions such as sigmoid, tanh, and ReLU

A. speed up the gradient calculation in backpropagation, as compared to linear units
B. help to learn nonlinear decision boundaries
C. are applied only to the output units
D. always output values between 0 and 1

(e) (0.25 points) Which of the following functions can be used as an activation function in the output layer if we wish to predict the probabilities of \( n \) classes \( (p_1, p_2, \ldots, p_k) \) such that sum of \( p \) over all \( n \) equals to 1?

A. Softmax
B. ReLU
C. Sigmoid
D. Tanh
(f) (0.25 points) What are some practical problems with the sigmoidal activation function in neural nets?
   A. It is convex, and convex functions cannot solve nonconvex problems
   B. It does not work well with the entropy loss function
   C. It can have negative values
   D. Gradients are small for values away from 0, leading to the "Vanishing Gradient" problem for large or recurrent neural nets

(g) (0.25 points) Which of the following neural network training challenge can be solved using batch normalization?
   A. Overfitting
   B. Restrict activations to become too high or low
   C. Training is too slow
   D. None of the above

(h) (0.25 points) Which of the following statement is true regarding dropout?
   A. Dropout gives a way to approximate by combining many different architectures
   B. Dropout demands high learning rates
   C. Dropout can help preventing overfitting
   D. None of the above

3. (3 points) **Neural Networks and BackPropagation**

   (a) (1 point) Draw a neural network that represents the function \( f(x, y) \) defined below:
   Note that to get full credit, you have to write down the precise numeric weights (e.g., 1, 0.5, +1, etc.) as well as the precise units that you will use at each node (e.g., sigmoid, linear, simple threshold, tanh etc.).

<table>
<thead>
<tr>
<th>( x )</th>
<th>( y )</th>
<th>( f(x, y) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>-5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>-5</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
</tbody>
</table>
Solution:

Nodes labeled by 1 and 2 are simple threshold units while the node labeled by 3 is a linear unit.

A possible setting of the weights is given below. Recall that the simple threshold unit is given by:

\[
out = \begin{cases} 
  +1 & \text{if } \sum_i w_i x_i > 0 \\
  -1 & \text{otherwise}
\end{cases}
\]

\(o_1\), which is the output of node labeled by 1 implements the following function

\[
o_1 = \begin{cases} 
  +1 & \text{if } \neg x \land \neg y \text{ is true} \\
  -1 & \text{otherwise}
\end{cases}
\]

To achieve this, we can use \(w_0 = 1\) and \(w_1 = w_2 = -2\).

\(o_2\), which is the output of node labeled by 2 implements the following function

\[
o_2 = \begin{cases} 
  +1 & \text{if } x \land y \text{ is true} \\
  -1 & \text{otherwise}
\end{cases}
\]

To achieve this, we can use \(v_0 = -2\) and \(v_1 = v_2 = 1.5\).

\(o_3\) implements the following function

\[
o_3 = \begin{cases} 
  +10 & \text{if } o_1 = +1 \text{ or } o_2 = +1 \\
  -5 & \text{otherwise}
\end{cases}
\]

Note that since 3 is a linear unit, we need it to obey the following constraints:

\[
s_0 + s_1 - s_2 = 10 \text{ (if } o_1 = +1 \text{ and } o_2 = -1) \\
s_0 - s_1 + s_2 = 10 \text{ (if } o_1 = -1 \text{ and } o_2 = +1) \\
s_0 - s_1 - s_2 = -5 \text{ (if } o_1 = -1 \text{ and } o_2 = -1)
\]

Notice that the case \(o_1 = +1 \text{ and } o_2 = +1\) can never happen.

A solution to the three equations is \(s_0 = 10\) and \(s_1 = s_2 = 7.5\).
Consider the neural network given below: Assume that all internal nodes compute the sigmoid function. Write an explicit expression that shows how back propagation (applied to minimize the least squares error function) changes the values of \( w_1, w_2, w_3, w_4 \) and \( w_5 \) when the algorithm is given the example \( x_1 = 0, x_2 = 1 \), with the desired response \( y_1 = 0 \) and \( y_2 = 1 \) (\( x_0 = 1 \) is the bias term). Assume that the learning rate is \( \alpha \) and that the current values of the weights are: \( w_1 = 3, w_2 = 2, w_3 = 2, w_4 = 3 \) and \( w_5 = 2 \). Let \( O_1 \) and \( O_2 \) be the output of the output units 1 (which models \( y_1 \)) and 2 (which models \( y_2 \)) respectively. Let \( O_3 \) be the output of the hidden unit 3.

(b) (0.5 points) Forward propagation. Write equations for \( O_1, O_2 \) and \( O_3 \) in terms of the given weights and example.

\[
\text{Solution: } O_3 = \sigma(w_1 x_0 + w_2 x_1 + w_3 x_2) = \sigma(3 \times 1 + 2 \times 0 + 2 \times 1) = \sigma(5) \\
O_2 = \sigma(w_5 o_3) = \sigma(2\sigma(5)) \\
O_1 = \sigma(w_4 o_3) = \sigma(3\sigma(5))
\]

(c) (0.5 points) Backward propagation. Write equations for \( \delta_1, \delta_2 \) and \( \delta_3 \) in terms of the given weights and example where \( \delta_1, \delta_2 \) and \( \delta_3 \) are the values propagated backwards by the units denoted by 1 and 2 and 3 respectively in the neural network.

\[
\text{Solution: } \delta_1 = (y_1 - o_1) o_1 (1 - o_1) = o_1^2 - o_1^2 \\
\delta_2 = (y_2 - o_2) o_2 (1 - o_2) = o_2 (1 - o_2)^2 \\
\delta_3 = o_3 (1 - o_3) (w_4 \delta_1 + w_5 \delta_2) = o_3 (1 - o_3) (3\delta_1 + 2\delta_2)
\]
(d) (1 point) Give an explicit expression for the new (updated) weights $w_1, w_2, w_3, w_4$ and $w_5$ after backward propagation.

Solution: Let $\eta$ denote the learning rate

\[
\begin{align*}
    w_1 &= w_1 + \eta \delta_3 x_0 = 3 + \eta \delta_3 \times 1 = 3 + \eta \delta_3 \\
    w_2 &= w_2 + \eta \delta_3 x_1 = 2 + \eta \delta_3 \times 0 = 2 \\
    w_3 &= w_3 + \eta \delta_3 x_2 = 2 + \eta \delta_3 \times 1 = 2 + \eta \delta_3 \\
    w_4 &= w_4 + \eta \delta_1 o_3 = 3 + \eta \delta_1 o_3 \\
    w_5 &= w_5 + \eta \delta_2 o_3 = 2 + \eta \delta_2 o_3
\end{align*}
\]

4. (2 points) **Convolutional Neural Networks**

Below is a diagram of a small convolutional neural network that converts a 13x13 image into 4 output values. The network has the following layers/operations from input to output: convolution with 3 filters, max pooling, ReLu, and finally a fully-connected layer. For this network we will not be using any bias/offset parameters. Please answer the following questions about this network.

(a) (0.5 points) How many weights in the convolutional layer do we need to learn?

Solution: 48 weights. Three filters with 4x4=16 weights each.
(b) (0.5 points) How many ReLu operations are performed on the forward pass?

**Solution:** 75 ReLu operations. ReLu is performed after the pooling step. ReLu is performed on each pixel of the three 5x5 feature images.

(c) (0.5 points) How many weights do we need to learn for the entire network?

**Solution:** 348 weights. 48 for the convolutional layer. Fully-connected has 3x5x5=75 pixels each connected to four outputs, which is 300 weights. Pooling layer does not have any weights.

(d) (0.5 points) True or false: A fully-connected neural network with the same size layers as the above network (13 × 13 → 3 × 10 × 10 → 3 × 5 × 5 → 4 × 1) can represent any classifier that the above convolutional network can represent.

A. True  
B. False