CS 615 Deep Learning
Summer 2021
Exam

Rules and Regulations

1. This exam must be taken over a **contiguous** span of 2hrs (120 minutes). Meaning once you start it, you have 2hrs to finish it. This reflects 1hr, 50 minutes for working on the exam and additional 10 minutes for preparing your submission.
2. Your solutions are to be **hand-written (**or written via stylus on a touch surface) photographed (or scanned), combined into a **single PDF** and uploaded to BBlearn. Any deviation from this will result in a deduction on the exam.
3. Any late submissions will be penalized by 5% **PER MINUTE.**
4. Although it is open book (in our case, notes, resources referenced in notes, listed textbooks, lectures, etc..), you **MAY NOT** post or research solutions in forums.
5. You also **may not** work with anyone else or discuss the exam with anyone else. You **MUST** either sign proper place on the cover sheet attesting to this, or, if you don't print the exam, write the pledge and sign it. Again, not doing this will result in a deduction on your exam and potentially a **zero** on it.
6. You **MAY** use a calculator and/or Matlab/Python to do computations for you.

I attest that I have worked on this exam alone, without the assistance of anyone else

Willie R Hood III

Name ***Signature***

Part	**Potential**	**Obtained**
Part I: Basic Theory	43	
Part II: Basic Computation	21	
Part III: Forward-Backwards Propagation	29	
Total	**93**	

Good Luck!

Part I: Basic Theory

For each of the following answer with a number, equation, word, or sentence. Keep things short (the more concise, the better)

1. For each objective function, what is the optimal value (2pts each)?
   a. Cross entropy

      A distribution

   b. Log Loss

      any number > -∞

   c. Squared error

      1 or 0

2. What is the range of values output by the following activation functions (2pts each)?
   a. Linear

      -∞ to ∞

   b. Logistic

      0 to 1

   c. Softmax

      0 to 1

   d. Hyperbolic Tangent

      -1 to 1

3. For each of the following objective functions, what is the logical activation function to have at the end of your architecture in order to produce $\hat{y}$ and **why**? (2pts each).
   a. Log Loss

      ReLu since it avoids negative numbers. Any activation function that does this will do great.

   b. Cross Entropy

      Soft max because cross entropy does great with distributions softmax is also very slow so ReLu would do fine as well

   c. Squared Error

      Linear Activation, this function is more for continuous targets As such linear will do fine.

4. Are we **more** or **less** likely to overfit as we **increase** the number of nodes in a hidden layer (2pts)?

more likely

5. Why do we often zscore our data when doing gradient-based learning (3pts)?

It is to standardize the data and reduce the number of outliers.

6. For each scenario state if a Log-Loss, Cross-Entropy, or Squared Error objective function is most appropriate, and **why** (2pts each).
   a. Building a system to determine if an audio clip is saying "alexa" or not.

   Logloss will work here. We are basically setting up a binary system which Logloss is best at.

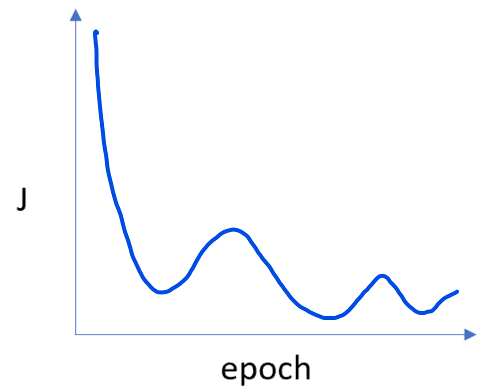
   b. Building a system to estimate the age of someone.

   This depends on how we set up this system. We could say for example if someone is 18 yes or no (binary) which Logloss works with. We could also use squared error since we are estimating.

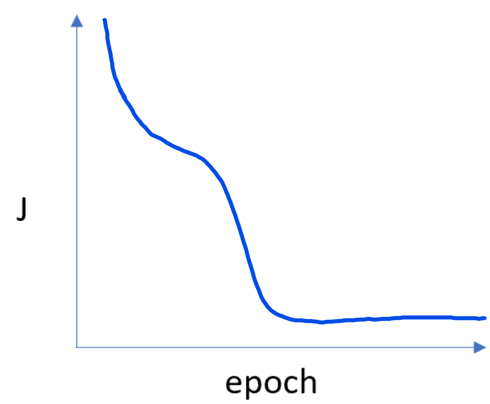
   c. Building a system that is attempting to determine which, out of five people, and image is of.

   Cross-Entropy works for this since it is the best for distributions (multi-classification).

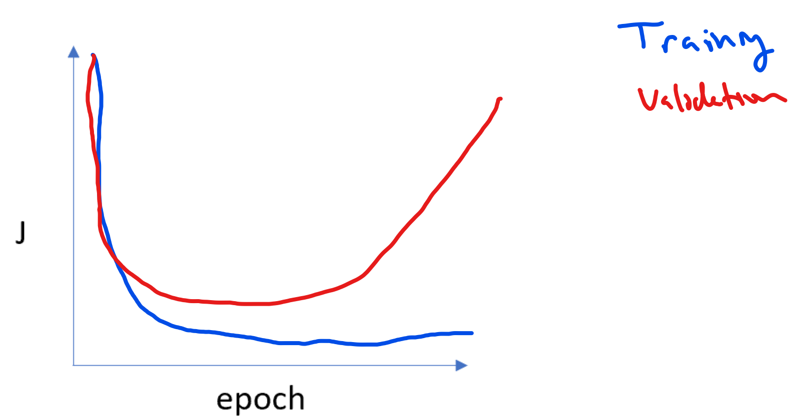
7. For each scenario draw $epoch$ vs. J, where J is the evaluation of an objective function that we are attempting to minimize. The scale of J is irrelevant. (2pts each)
   a. The gradient updates cause us to jump about a minima.



   b. The gradient updates cause our gradient update rules to "explode".



8. Draw a scenario when the training results in poor generalization by plotting $epoch$ vs J for both training and validation datasets (3pts).



9. Another common objective function is the *log likelihood*, defined as

$$J = yln(\hat{y}) + (1 - y)\ln(1 - \hat{y})$$

This is identical to the log-loss objective function, but without the negative sign. Describe Therefore this is something we want to **maximize.** Describe the changes you'd need to make to the learning process to accommodate this objective function (5pts).

We are now looking for a maximum. We would remove the negative sign from logloss with this convention our values would still be between 0 and 1 but the higher the output is the better.

Part II: Basic Computation

10. Given the following set of class labels, what is the one-hot encoding (3pts)?

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

11. Given the following target and computed values:

$$Y = \begin{bmatrix} 1 \\ 2 \\ 1 \\ -4 \\ 3 \end{bmatrix}, \hat{Y} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 5 \\ 2 \end{bmatrix}$$

RMSE $\sqrt{\left([1\ 1\ 1\ {-9}\ 1] \begin{bmatrix} 1 \\ 1 \\ 1 \\ -9 \\ 1 \end{bmatrix} \right) / 5}$ $= \sqrt{85/5}$ $= \sqrt{17}$

a. What is the root mean squared error (keep your answer in the form of a square root)? Show your work. (2pts)

MAPE
$$\begin{bmatrix} 1 \\ 2 \\ 1 \\ -4 \\ 3 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 0 \\ 5 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ -9 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ -9 \\ 1 \end{bmatrix} \Big/ \begin{bmatrix} 1 \\ 2 \\ 1 \\ -4 \\ 3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.5 \\ 1 \\ 2.25 \\ 0.33 \end{bmatrix} \xrightarrow{\text{sum}} 5.08$$

b. What is the mean absolute percent error (keep your answer in the form of fractions)? Show your work. (2pts) MAPE = 5.08 / 5

12. Given the following multi-class targets, and the output $\hat{Y}$ from a soft-max layer, what is the accuracy of the system (4pts)?

$$Y = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 2 \\ 2 \\ 0 \end{bmatrix}, \quad \hat{Y} = \begin{bmatrix} 0 & 0 & 1 \\ 0.2 & 0.5 & 0.3 \\ 0.5 & 0.2 & 0.3 \\ 0.9 & 0.0 & 0.1 \\ 0.3 & 0.3 & 0.4 \\ 0.5 & 0.1 & 0.4 \\ 0.8 & 0.1 & 0.1 \end{bmatrix}$$

1 0 0 – no
1 0 0 – no
0 1 0 – no
1 0 0 ✓ yes
0 0 1 – no
0 0 1 – no
1 0 0 ✓ yes

2/7 = 0.29%

13. Given the objective function $J = 2\underbrace{xw}_{\begin{bmatrix} -1 & -2 \\ 2 & 4 \end{bmatrix}} + 3$, where $x = [1, 2]$ (1x2) and w is initialized as $w = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$: 2x1

a. What is J for this observation x, before training (given this initialization) (2pts)

$$J = \begin{bmatrix} -2 & -4 \\ 2 & 8 \end{bmatrix} + 3 = \begin{bmatrix} 1 & -1 \\ 8 & 11 \end{bmatrix}$$

b. What is $\frac{\partial J}{\partial w}$ (as an equation, without plugging in numbers) (2pts)?

$$\frac{\partial J}{\partial w} = 2x$$

c. What will w become after one training iteration if our learning rate is $\eta = 1$? You may assume that this objective function is something that we want to maximize (3pts).

$$w = w + \eta\left(-\frac{\partial J}{\partial w}\right) = \begin{bmatrix} -1 \\ 2 \end{bmatrix} + \begin{bmatrix} -2 \\ -4 \end{bmatrix} = \begin{bmatrix} -3 \\ -2 \end{bmatrix}$$

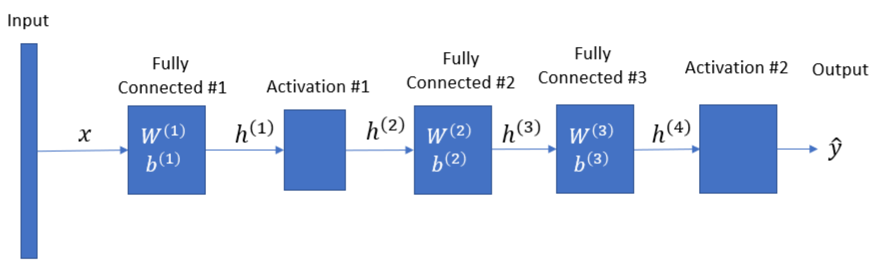
d. What is an intrinsic issue/problem with this objective function (3pts)?

The learning rate is 1 meaning we are forced to full changes due to our gradient.

Part III: Forwards-Backwards Propagation

For this section you'll use the architecture below.

Below is a deep-learning architecture:



Where:

- x has 3 features.
- Fully Connected Layer #1 has 3 outputs and is initialized as
$$W^{(1)} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ -2 & 2 & 1 \end{bmatrix}, b^{(1)} = [1 \quad -2 \quad 1]$$
- Activation Layer #1 is a function $f(x) = x^2$
- Fully Connected Layer #2 has 2 outputs and is initialized as:
$$W^{(2)} = \begin{bmatrix} -1 & -1 \\ 0 & 1 \\ 1 & 2 \end{bmatrix}, b^{(2)} = [1 \quad 0]$$
- Fully Connected Layer #3 has 1 output and is initialized as:
$$W^{(3)} = \begin{bmatrix} 2 \\ -1 \end{bmatrix}, b^{(3)} = [5]$$
- Activation Layer #2 is a **ReLu** activation function

Running out of time

14. What is the gradient of Activation Function #1? That is, what is $\frac{\partial h^{(3)}}{\partial h^{(2)}}$ (2pts)?

This is relu gradient times fully connected 3 gradient (W^T_3) times fully connected 2 gradient which is W^T_2

15. Given an objective function of $J = (y - \hat{y})^2 + 3y\hat{y}$ (all operations are element-wise), what is $\frac{\partial J}{\partial \hat{y}}$ (2pts)?

$$\frac{\partial J}{\partial y} = 2(y - \hat{y})(-1) + 3y$$

16. Given the observation below, X, perform *forward-propagation*, showing the computations of $h^{(1)}$ through $\hat{y}$ as you do so. All the math/numbers should be easy enough to compute that you don't need a calculator. For simplicity, ***do not*** zscore the input data (5pts).

$$X = \begin{bmatrix} 0 & -1 & 2 \\ 2 & 1 & 0 \end{bmatrix}$$

17. Given target values of $Y = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, if we are looking to minimize the objective function $J = (y - \hat{y})^2 + 3y\hat{y}$, using the computations you did in forward-propagation, *back-propagate* to update the weights in the fully connected layers. Again, all the math/numbers should be easy enough to compute that you don't need a calculator. **Make sure that you show the value of the gradient as you propagate backwards as well as the updated weights.** You may assume a learning rate of $\alpha = 1.0$ (10pts)