CS390-NIP - Fall 2020 Name: Michael Lee

Email: lee2965@purdue.edu Date: September 21, 2020

# Lab 1 Report

Basic Image Classification
Due Date: September 18, 2020

https://github.com/michaeldlee23/cs390-lab1

### Resources used

- Explanation of calculus https://www.youtube.com/watch?v=GlcnxUlrtek&t=390s
- Starting point for determining number of hidden neurons to use https://stats.stackexchange.com/a/136542
- Tensorflow and Keras model creation, training, predicting, optimizing https://www.tensorflow.org/api\_docs/python/tf/keras/

### Completed portions of lab

- 1. [30] Custom Neural Net
  - $\checkmark$ [5] Sigmoid and sigmoid derivative functions
  - $\checkmark$  [10] Train Function (using backpropagation properly)
  - √[15] Fully functioning 2-layer neural net
  - $\sqrt{[+3]}$  EC: implement an option to make the network 3 layer
  - $\checkmark$  [+5] EC: make the network N layer, where N is given in the constructor
  - $\checkmark$ ?[+1] EC: implement a second activation function (ReLU) that your network can use. The activation should be specified in the constructor
- 2. [30] TF Neural Net
  - √[30] Implement a function that builds a 2 layer neural net using either Tensorflow or Keras.
  - [+6] EC: Get accuracy to 99%.
- 3. [20] Pipeline & Misc.
  - $\bullet~ \checkmark [5]$  Preprocess data such that values are between 0.0 and 1.0
  - $\sqrt{[15]}$  Create an F1 score confusion matrix in the evaluation function and print it
  - $\checkmark$  [+5] EC: Look up and learn about the iris dataset. Download the dataset and write a program to classify it using your custom neural net.
- 4.  $\checkmark$ [20] Report

## **Summary**

• Custom Neural Network

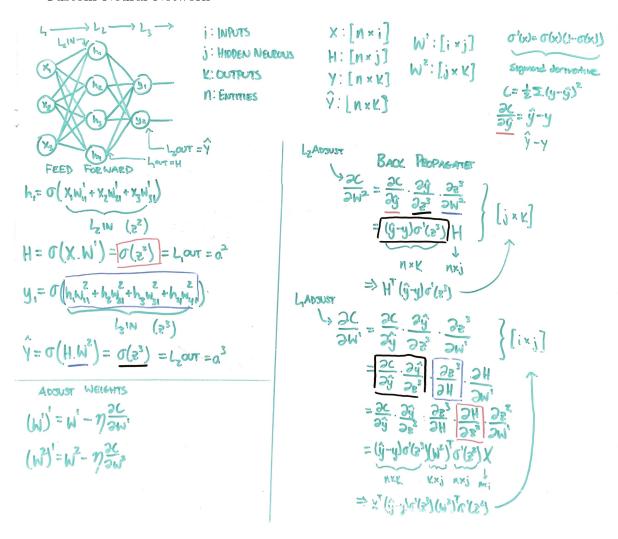


Figure 1: Work through backpropagation calculus

In order to understand how to implement the backpropagation algorithm, I needed to fully understand the math behind the process. I heavily relied on my whiteboarded work and constantly referenced it while I implemented the custom neural net, as it proved to be extremely helpful in keeping track of the dimensions of inputs. It also helped me keep track of variable names, as there were so many different values and matrices to keep track of.

In order to make testing easier, I implemented command line arguments to adjust hyperparameters:

### Usage:

- -a [algorithm | guesser, custom\_net, tf\_net]
- -e [number of epochs]
- -b [batch size]
- -n [number of hidden neurons]
- -l [initial learning rate]
- -f [activation function | sigmoid, relu]

### • Tensorflow Neural Network

The TensorFlow network was implemented in a similar manner that was shown on the lecture slides. Playing around with the hyperparamters (number of epochs and number of hidden neurons), we can see that the TensorFlow network performs significantly better than the custom net, as expected. In attempts to achieve 99% accuracy, I added layers, increased the number of hidden neurons, and tried different optimizers and batch sizes. Accuracy persistently hovered at around 98.2%.

# Relevant outputs

### 1. F1 Confusion Matrix

			rithm: d									
			racy: 57		<b>3</b> %							
			usion ma									
Ι.	931.	2.	73.	73.	7.	646.	12.	770.	736.	40.		
[	0.	1086.	22.	23.	3.	7.	1.	17.	12.	1.		
[	17.	20.	882.	23.	3.	22.	41.	30.	43.	2.		
[	2.	15.	7.	853.	4.	66.	0.	23.	37.	14.	1021.]	
[	7.	2.	11.	3.	892.	26.	30.	26.	21.	772.	1790.]	
[	2.	1.	2.	11.	0.	29.	0.	2.	5.	2.	54.]	
[	3.	6.	25.	0.	26.	6.	844.	2.	14.	5.		
[	9.	2.	0.	13.	11.	56.	17.	86.	60.	31.	285.]	
[	2.	1.	8.	8.	0.	10.	0.	14.	19.	0.	62.]	
[	7.	0.	2.	3.	36.	24.	13.	58.	27.	142.	312.]	
[	980.	1135.	1032.	1010.	982.	892.	958.	1028.	974.	1009.	10000.]]	
Cla	ssifie	r F1 s	core mai	rix:								
[0	.4361	nan	nan	nan	nan	nan	nan	nan	nan	nan	1	
1	nan	0.9415	nan	nan	nan	nan	nan	nan	nan	nan	]	
ī	nan	nan	0.834	nan	nan	nan	nan	nan	nan	nan	]	
ī	nan	nan	nan	0.84	nan	nan	nan	nan	nan	nan	]	
ī	nan	nan	nan	nan	0.6436	nan	nan	nan	nan	nan	]	
ī	nan	nan	nan	nan		0.0613	nan	nan				
ĩ	nan	nan	nan	nan			0.8936	nan				
ī	nan	nan	nan	nan	nan	nan		0.131	nan			
ī	nan	nan	nan	nan		nan	nan		0.0367	nan		
ř	nan	nan	nan	nan	nan	nan	nan	nan		0.215		

EDDADA	METERC			-							
			500	batci	nSize: :	100 ne	uronsPe	rLayer:	10	learningRate:	0.1
				0%							
					0.4		4	,	,	000 1	
700.	1155.	1032.	1010.	702.	072.	/50.	1020.	//	1007.	10000.11	
ssifie	r F1 s	rore mai	trive								
				nan	nan	nan	nan	nan	nan		
nan	nan	nan			nan	nan	nan	nan			
nan	nan	nan	nan		0.8314	nan	nan	nan	nan		
nan	nan	nan	nan	nan	nan	0.9108	nan	nan	nan		
nan	nan	nan	nan	nan	nan	nan	0.9141	nan	nan		
nan	nan	nan	nan	nan	nan	nan	nan	0.8577			
nan	nan	nan	nan	nan	nan	nan	nan	nan			
	: custifie ssifie ssifie ssifie ssifie ssifie 909. 0. 7. 11. 14. 5. 1. 1. 980. ssifie .9276 nan nan nan nan nan nan nan nan nan na	: custom_net ssifier algo ssifier accur ssifier confi 909. 0. 0. 1100. 7. 9. 11. 1. 31. 1. 14. 5. 5. 2. 1. 16. 1. 1. 980. 1135. ssifier F1 sc. 9276 nan	ssifier algorithm: cssifier accuracy: 96 ssifier confusion mm 909. 0. 8. 0. 1100. 2. 7. 9. 927. 11. 0. 27. 1. 1. 9. 31. 1. 1. 14. 5. 17. 5. 2. 15. 1. 16. 22. 1. 1. 4. 980. 1135. 1032. ssifier F1 score mannan nan nan nan nan nan nan nan nan	: custom_net epochs: 500 ssifier algorithm: custom_ssifier accuracy: 90.10000/ssifier confusion matrix: 909. 0. 8. 20. 0. 1100. 2. 0. 7. 9. 927. 21. 11. 0. 27. 871. 1. 1. 9. 3. 31. 1. 1. 26. 14. 5. 17. 7. 5. 2. 15. 13. 1. 16. 22. 43. 1. 1. 4. 6. 980. 1135. 1032. 1010. ssifier F1 score matrix: .9276 nan nan nan nan nan nan nan nan nan na	: custom_net epochs: 500 batcl ssifier algorithm: custom_net ssifier accuracy: 90.100000% ssifier confusion matrix: 909. 0. 8. 20. 0. 90. 1100. 2. 0. 0. 7. 9. 927. 21. 5. 11. 0. 27. 871. 0. 1. 1. 9. 3. 915. 31. 1. 1. 26. 4. 14. 5. 17. 7. 23. 5. 2. 15. 13. 1. 1. 16. 22. 43. 4. 1. 1. 4. 6. 30. 980. 1135. 1032. 1010. 982. ssifier F1 score matrix: .9276 nan nan nan nan nan nan 0.9722 nan nan nan nan nan 0.9724 nan nan nan nan nan 0.9845 nan nan nan nan 0.9645 nan nan nan nan nan 0.924 nan	: custom_net epochs: 500 batchSize: 3 ssifier algorithm: custom_net ssifier accuracy: 90.100000% ssifier confusion matrix: 909. 0. 8. 20. 0. 24. 0. 1100. 2. 0. 0. 4. 7. 9. 927. 21. 5. 8. 11. 0. 27. 871. 0. 38. 1. 1. 9. 3. 915. 5. 31. 1. 1. 26. 4. 730. 14. 5. 17. 7. 23. 29. 5. 2. 15. 13. 1. 5. 1. 16. 22. 43. 4. 37. 1. 1. 4. 6. 30. 12. 980. 1135. 1032. 1010. 982. 892. ssifier F1 score matrix: .9276 nan nan nan nan nan nan nan nan nan na	: custom_net epochs: 500 batchSize: 100 ner ssifier algorithm: custom_net ssifier accuracy: 90.100000% ssifier confusion matrix: 909. 0. 8. 20. 0. 24. 8. 0. 1100. 2. 0. 0. 4. 3. 7. 9. 927. 21. 5. 8. 10. 11. 0. 27. 871. 0. 38. 9. 11. 1. 9. 3. 915. 5. 9. 31. 1. 1. 26. 4. 730. 14. 14. 5. 17. 7. 23. 29. 898. 15. 2. 15. 13. 1. 5. 1. 1. 16. 22. 43. 4. 37. 6. 1. 1. 4. 6. 30. 12. 0. 980. 1135. 1032. 1010. 982. 892. 958. ssifier F1 score matrix: .9276 nan nan nan nan nan nan nan 0.9722 nan	: custom_net epochs: 500 batchSize: 100 neuronsPersifier algorithm: custom_net ssifier accuracy: 90.100000% ssifier confusion matrix: 909. 0. 8. 20. 0. 24. 8. 1. 0. 28. 11. 7. 9. 927. 21. 5. 8. 10. 28. 11. 0. 27. 871. 0. 38. 9. 7. 11. 1. 9. 3. 915. 5. 9. 17. 31. 1. 1. 26. 4. 730. 14. 0. 7. 11. 1. 26. 4. 730. 14. 0. 17. 7. 23. 29. 898. 3. 5. 2. 15. 13. 1. 5. 1. 920. 1. 16. 22. 43. 4. 37. 6. 4. 1. 1. 4. 6. 30. 12. 0. 37. 980. 1135. 1032. 1010. 982. 892. 958. 1028. ssifier F1 score matrix: 9276 nan nan nan nan nan nan nan nan nan na	: custom_net epochs: 500 batchSize: 100 neuronsPerLayer: ssifier algorithm: custom_net ssifier accuracy: 90.100000% ssifier confusion matrix: 909. 0. 8. 20. 0. 24. 8. 1. 4. 0. 1100. 2. 0. 0. 4. 3. 11. 4. 7. 9. 927. 21. 5. 8. 10. 28. 8. 11. 4. 11. 0. 27. 871. 0. 38. 9. 7. 21. 1. 1. 9. 3. 915. 5. 9. 17. 8. 31. 1. 1. 26. 4. 730. 14. 0. 50. 14. 0. 50. 14. 5. 17. 7. 23. 29. 898. 3. 15. 5. 2. 15. 13. 1. 5. 1. 920. 4. 1. 16. 22. 43. 4. 37. 6. 4. 847. 1. 1. 4. 6. 30. 12. 0. 37. 13. 980. 1135. 1032. 1010. 982. 892. 958. 1028. 974. ssifier F1 score matrix: .9276 nan nan nan nan nan nan nan nan nan na	: custom_net epochs: 500 batchSize: 100 neuronsPerLayer: 10  ssifier algorithm: custom_net ssifier accuracy: 90.100000% ssifier confusion matrix: 909. 0. 8. 20. 0. 24. 8. 1. 4. 6. 0. 1100. 2. 0. 0. 4. 3. 11. 4. 4. 7. 9. 927. 21. 5. 8. 10. 28. 8. 1. 11. 0. 27. 871. 0. 38. 9. 7. 21. 21. 1. 1. 9. 3. 915. 5. 9. 17. 8. 34. 31. 1. 1. 26. 4. 730. 14. 0. 50. 7. 14. 5. 17. 7. 23. 29. 898. 3. 15. 3. 5. 2. 15. 13. 1. 5. 1. 920. 4. 19. 1. 16. 22. 43. 4. 37. 6. 4. 847. 21. 1. 1. 4. 6. 30. 12. 0. 37. 13. 893. 980. 1135. 1032. 1010. 982. 892. 958. 1028. 974. 1009.  ssifier F1 score matrix: .9276 nan nan nan nan nan nan nan nan nan na	: custom_net epochs: 500 batchSize: 100 neuronsPerLayer: 10 learningRate: ssifier algorithm: custom_net ssifier accuracy: 90.100000% ssifier confusion matrix: 909. 0. 8. 20. 0. 24. 8. 1. 4. 6. 980.] 0. 1100. 2. 0. 0. 4. 3. 11. 4. 4. 1128.] 7. 9. 927. 21. 5. 8. 10. 28. 8. 1. 1024.] 11. 0. 27. 871. 0. 38. 9. 7. 21. 21. 1005.] 1. 1. 9. 3. 915. 5. 9. 17. 8. 34. 1002.] 31. 1. 1. 26. 4. 730. 14. 0. 50. 7. 864.] 14. 5. 17. 7. 23. 29. 898. 3. 15. 3. 1014.] 5. 2. 15. 13. 1. 5. 1. 920. 4. 19. 985.] 1. 16. 22. 43. 4. 37. 6. 4. 847. 21. 1001.] 1. 1. 4. 6. 30. 12. 0. 37. 13. 893. 997.] 980. 1135. 1032. 1010. 982. 892. 958. 1028. 974. 1009. 10000.]] ssifier F1 score matrix: .9276 nan nan nan nan nan nan nan nan nan na

Epochs: 500, Neurons/Layer: 5

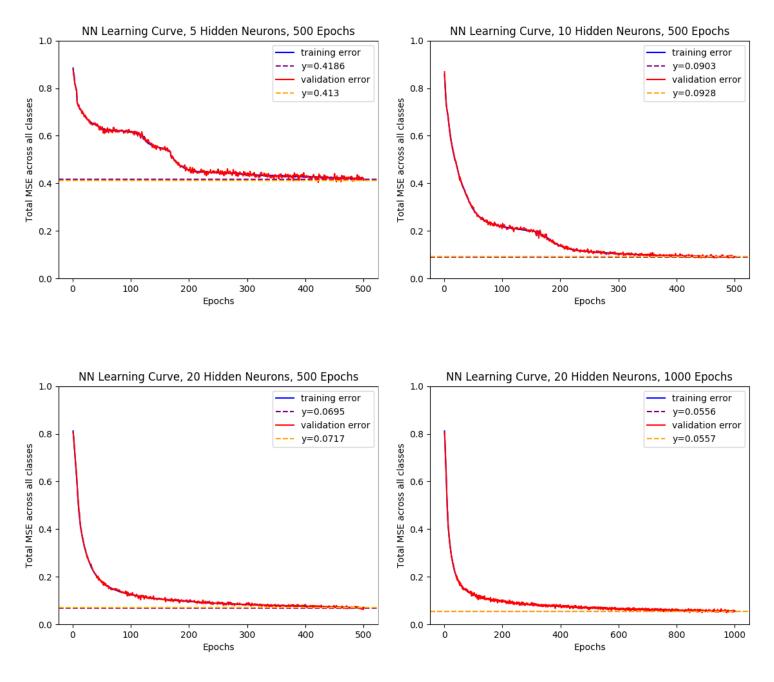
Epochs: 500, Neurons/Layer: 10

		AMETERS	: epochs:	. E00	botol	-6i-o. 1	100 001	ıranaDa:	rLaver:	20	learningR	loto: A 1
116	t. cus	com_nec	ероспъ	. 300	Date	13126.	roo net	ITOHSE.	LLayer.	20	Teatilitingh	ace. 0.1
C1	assifi	er algo:	rithm: d	custom_	net							
C1	assifi	er accu	racy: 92	2.12000	3%							
C1	assifi	er confi	usion ma	atrix:								
[[	952.	0.	6.	2.	2.	11.	13.	1.	3.	9.	999.]	
[	0.	1115.	1.	1.	2.	3.	3.	11.	5.	4.	1145.]	
[	3.	0.	923.	27.	7.	4.	9.	22.	8.	1.	1004.]	
[	0.	7.	14.	914.	0.	32.	1.	13.	31.	21.	1033.]	
[	1.	0.	12.	0.	922.	5.	9.	5.	12.	38.	1004.]	
[	4.	0.	8.	29.	3.	786.	18.	0.	21.	8.	877.]	
[	12.	3.	18.	5.	9.	17.	899.	3.	12.	1.	979.]	
[	3.	1.	13.	14.	1.	12.	1.	949.	10.	24.	1028.]	
[	5.	9.	31.	13.	4.	19.	5.	3.	861.	12.	962.]	
[	0.	0.	6.	5.	32.	3.	0.	21.	11.	891.	969.]	
[	980.	1135.	1032.	1010.	982.	892.	958.	1028.	974.	1009.	10000.]]	
			core mat									
ΙŪ	0.9621	nan	nan	nan	nan	nan	nan	nan	nan	nan]		
_ ī		0.9781	nan	nan	nan	nan	nan	nan	nan	nan]		
Ĺ	nan		0.9067	nan	nan	nan	nan	nan	nan	nan]		
Ĺ	nan	nan		0.8948	nan	nan	nan	nan	nan	nan]		
Ĺ	nan	nan	nan		0.9285	nan	nan	nan	nan	nan]		
اً ا	nan	nan	nan	nan		0.8886	nan	nan	nan	nan]		
Ī	nan	nan	nan	nan	nan		0.9282	nan	nan	nan]		
Ē		nan	nan	nan	nan	nan		0.9232	nan	nan]		
[		nan	nan	nan	nan	nan	nan		0.8895	nan]		
Γ.	nan	nan	nan	nan	nan	nan	nan	nan	nan	0.9009]		

Epochs: 500, Neurons/Layer: 20

Epochs: 1000, Neurons/Layer: 20

### 2. Error Plots



To evaluate my model's learning curve, I used cross validation with a 90-10 split. At the beginning of each epoch, I randomly shuffle the inputted data sets and partition 10% to be used for validation, while the rest is then used to train the model. At the end of the epoch, I run the model against the training and validation set and calculate the total mean square error across all output classes. As we can see from the outputted graphs, the model appears to not overfit at all, as the training and validation error over time remains practically identical.

To observe the effects of certain hyperparameters (namely epoch number and hidden

layer size), I tested the network under several different cases. We can see that increasing the hidden layer size drastically increases accuracy from 5 to 10, but its effects diminish as we go from 10 to 20.

### 3. TensorFlow Output

```
HYPERPARAMETERS:
                                                                                   learningRate: 0.1
                                     batchSize: 100 neuronsPerLayer: 512
net: tf_net
                  epochs: 50
Classifier algorithm: tf_net
Classifier accuracy: 98.210000%
Classifier confusion matrix:
                                                                       2.
                                                                                     984.]
    970.
               1.
                       2.
                               0.
                                               4.
                                                       3.
                                                               0.
                                                                                1.
                                       1.
       0.
           1126.
                       2.
                               0.
                                                                       0.
                                                                                3.
                                                                                    1137.]
                                               0.
                                                       2.
                                                               3.
                                               0.
                                                                                    1043.]
       3.
               1.
                   1016.
                               8.
                                       3.
                                                       1.
                                                               6.
                                                                       4.
                                                                                1.
                                                                                    1006.]
       0.
               1.
                       0.
                             987.
                                       1.
                                               8.
                                                                       2.
                                                                                5.
                                                       1.
                                                               1.
       0.
               0.
                               0.
                                     961.
                                               1.
                                                       1.
                                                               0.
                                                                                     971.]
                       1.
                                                                       1.
                                                                                6.
                                             870.
                                                                                     886.]
       0.
               1.
                       0.
                               5.
                                       0.
                                                       3.
                                                               1.
                                                                       1.
                                                                                5.
                                                                                     964.]
                                                     946.
                                                               0.
       4.
               2.
                       2.
                               0.
                                       4.
                                               4.
                                                                       1.
                                                                                1.
                                       2.
                                               0.
                                                                                    1029.]
                               2.
                                                       0.
                                                            1010.
                                                                       5.
       1.
               1.
                       4.
                                                                                4.
       2.
               2.
                       5.
                                       0.
                                               3.
                                                       1.
                                                               4.
                                                                     956.
                                                                               4.
                                                                                     981.]
                               4.
                                                                             979.
                                                       0.
       0.
               0.
                       0.
                               4.
                                       9.
                                               2.
                                                               3.
                                                                       2.
                                                                                     999.]
                                     982.
                                                            1028.
    980.
           1135.
                   1032.
                           1010.
                                             892.
                                                     958.
                                                                     974.
                                                                            1009. 10000.]]
Classifier F1 score matrix:
[[0.9878
             nan
                      nan
                              nan
                                      nan
                                              nan
                                                      nan
                                                              nan
                                                                      nan
                                                                              nan]
     nan 0.9912
                      nan
                              nan
                                                                      nan
                                                                              nan]
                                      nan
                                              nan
                                                      nan
                                                              nan
              nan 0.9793
                              nan
                                                                               nan]
     nan
                                      nan
                                              nan
                                                      nan
                                                              nan
                                                                      nan
                          0.9792
     nan
              nan
                      nan
                                      nan
                                              nan
                                                      nan
                                                              nan
                                                                      nan
                                                                               nan]
                                  0.9841
             nan
                                                      nan
                                                                               nan]
     nan
                      nan
                              nan
                                              nan
                                                              nan
                                                                      nan
                                      nan 0.9786
                              nan
                                                                      nan
                                                                               nan]
     nan
             nan
                      nan
                                                      nan
                                                              nan
                                              nan 0.9844
             nan
                              nan
                                                              nan
                                                                      nan
                                                                               nan]
     nan
                      nan
                                      nan
                                                      nan 0.982
     nan
             nan
                              nan
                                      nan
                                              nan
                                                                      nan
                                                                               nan]
                      nan
                                                              nan 0.978
                                                      nan
                                                                               nan]
     nan
             nan
                      nan
                              nan
                                      nan
                                              nan
             nan
                                              nan
                                                      nan
                                                              nan
                                                                      nan 0.9751]]
     nan
                      nan
                              nan
                                      nan
```

Epochs: 50, Neurons/Layer: 512

### 4. Extra Content

### • N-Layer Custom Net

I implemented an N-Layer custom net in extra.py. I also implemented command-line argument parsing to easily set hyperparameters; using the -1 flag will set the number of hidden layers. Below is the output for a network with 3 hidden layers, each layer having 10 neurons, trained over 500 epochs.

```
HYPERPARAMETERS:
                                     batchSize: 100 hiddenSize: 10 hiddenLayers: 3 activation: sigmoid
net: custom_net epochs: 500
Classifier algorithm: custom net
Classifier accuracy: 88.470000%
Classifier confusion matrix:
                                                                9.
                                                                                     1000.]
    922.
               0.
                      16.
                              18.
                                       0.
                                              11.
                                                       15.
                                                                        3.
                                                                                6.
       0.
           1095.
                      21.
                               5.
                                       2.
                                               2.
                                                       2.
                                                               16.
                                                                        9.
                                                                                1.
                                                                                     1153.]
                                                      14.
                                                               39.
                                                                                     1014.]
       9.
               9.
                     877.
                              29.
                                       2.
                                              11.
                                                                       20.
                                                                                4.
                             858.
                                              55.
       4.
               5.
                      21.
                                       0.
                                                       1.
                                                               11.
                                                                       12.
                                                                               12.
                                                                                      979.]
               0.
                       7.
                               1.
                                     881.
                                              11.
                                                       20.
                                                                9.
                                                                        7.
                                                                               57.
                                                                                     1000.]
               3.
                      10.
                              55.
                                       1.
                                             731.
                                                        7.
                                                                2.
                                                                       68.
                                                                               17.
                                                                                      901.]
                                      24.
                                                     891.
                                                                0.
     17.
               5.
                      23.
                               3.
                                              17.
                                                                       12.
                                                                                0.
                      10.
                               8.
                                       5.
                                              11.
                                                              898.
                                                                       12.
                                                                                      974.]
               3.
                                                        2.
                                                                               16.
       5.
              14.
                              24.
                                      15.
                                              37.
                                                              13.
                                                                      814.
                      41.
                                                        6.
                                                                               16.
                               9.
                                      52.
                                                               31.
                                                                       17.
                                                                              880.
       0.
               1.
                       6.
                                               6.
                                                        0.
                                                                                     1002.]
           1135.
                    1032.
                            1010.
                                     982.
                                             892.
                                                     958.
                                                             1028.
                                                                      974.
                                                                             1009.
                                                                                   10000.]]
Classifier F1 score matrix:
[[0.9313
                                                       nan
             nan
                      nan
                              nan
                                      nan
                                              nan
                                                              nan
                                                                       nan
                                                                               nan]
     nan
          0.9572
                      nan
                              nan
                                      nan
                                              nan
                                                       nan
                                                              nan
                                                                       nan
                                                                               nan]
     nan
             nan
                  0.8573
                              nan
                                      nan
                                              nan
                                                       nan
                                                              nan
                                                                       nan
                                                                               nan]
                          0.8627
     nan
             nan
                      nan
                                      nan
                                              nan
                                                       nan
                                                              nan
                                                                       nan
                                                                               nan]
     nan
             nan
                      nan
                              nan
                                  0.889
                                              nan
                                                       nan
                                                              nan
                                                                       nan
                                                                               nan]
     nan
             nan
                      nan
                              nan
                                      nan
                                             .8154
                                                      nan
                                                              nan
                                                                       nan
                                                                               nan]
                                                   0.9138
     nan
             nan
                      nan
                              nan
                                      nan
                                               nan
                                                              nan
                                                                       nan
                                                                               nan]
                                                           0.8971
                                                                               nan]
     nan
             nan
                      nan
                              nan
                                      nan
                                              nan
                                                       nan
                                                                       nan
                                                                               nan]
     nan
             nan
                      nan
                              nan
                                      nan
                                              nan
                                                       nan
                                                               nan
                                                                   0.831
                                      nan
                                              nan
                                                      nan
                                                              nan
                                                                       nan 0.8752]]
     nan
             nan
                      nan
```

Epochs: 500, Neurons/Layer: 10, Hidden Layers: 3

### • ReLU Activation

I attempted to implement the option to use a ReLU activation function instead of sigmoid, however there are some unexpected results. I have implemented the <code>\_relu()</code> and <code>\_reluDerivative()</code> functions that appear to behave as intended, but when actually running the model using these activation functions we get very poor accuracy — the model tends to only pick 0s and 1s. I tried using a smaller learning rate (0.001 instead of 0.1) which had minimal improvement. I also tried using ReLU on just the hidden layer, and kept the output layer as sigmoid, but that also had minimal improvement.

### • Iris Dataset

The option to run the neural net on the iris dataset was implemented in extra.py, which can be specified using the -d iris flag. Here are the results of running the

custom net on the iris dataset with 1 hidden layer, 20 hidden neurons, and 1000 epochs:

```
HYPERPARAMETERS:
net: custom_net epochs: 1000
                                batchSize: 100 hiddenSize: 20 hiddenLayers: 1
Classifier algorithm: custom_net
Classifier accuracy: 93.333333%
Classifier confusion matrix:
[[15. 0. 0. 15.]
 [ 0. 13. 2. 15.]
 [ 0. 1. 14. 15.]
[15. 14. 16. 45.]]
Classifier F1 score matrix:
            nan
                   nan]
    nan 0.8966
                   nan]
            nan 0.9032]]
    nan
```

Epochs: 1000, Neurons/Layer: 20, Hidden Layers: 1

The iris dataset was obtained through python's **sklearn.datasets** package, and contains 150 training samples. Since no test set is provided, I had to shuffle the given data and arbitrarily took 45 samples to set aside as the test set. The remaining 105 sample were used to train the model.