Theory of Neural Networks MP2

Michael Person

April 16, 2017

1 Task 1

1.1 Part A

Here I will argue that if we allow our weight initializations to be too large than learning will never occur because the gradient will always be zero and therefore our weights will never be updated.

$$\mathbf{a} = \widetilde{\mathbf{W}}\tilde{\mathbf{x}} \tag{1}$$

$$\boldsymbol{\delta} = \frac{\partial E}{\partial \mathbf{a}} \tag{2}$$

$$\frac{dE}{d\widetilde{\mathbf{W}}} = \boldsymbol{\delta}\mathbf{y}^T \tag{3}$$

If our weights, $\widetilde{\mathbf{W}}$, are too large than our activation, $\mathbf{a}$ will be very large. If our activation becomes too large than our sensitivities, $\boldsymbol{\delta}$, will become small. And if our sensitivities are small than our gradients will be small therefore reducing if not eliminating the learning that is possible by the network.

1.2 Part B

In order to allow our model to train, we need to be smart about how we initialize the weights. Here we prove how to initialize the weights so that our model will in fact learn. We will start will making the argument that if we use a tanh or a sigmoid activation function that if our activate signal becomes either too positive or too negative, the absolute value becomes too large, that the answer will be a flat 0 or 1. Therefore we set some number α_0 that the absolute value of our activation signal must be lower than.

$$|\alpha| \leq \alpha_0 \tag{4}$$

$$\alpha = \mathbf{w}^T\mathbf{y} + b \tag{5}$$

$$|\alpha| \leq |\mathbf{w}^T\mathbf{y} + b| \leq |b| + |\mathbf{w}^T\mathbf{y}| \tag{6}$$

$$|\mathbf{w}^T\mathbf{y}| \leq ||\mathbf{w}||_1 \underbrace{||\mathbf{y}||_\infty}_{=1} \tag{7}$$

$$|\alpha| \leq |b| + ||\mathbf{w}||_1 = ||\widetilde{\mathbf{W}}||_1 \leq (N+1)||\widetilde{\mathbf{W}}||_\infty \tag{8}$$

This last line is because the summation of a vector will always be less than or equal to the dimensionality times the maximum element that occurs in it.

$$|b| + ||\mathbf{w}||_1 \leq (N+1)||\widetilde{\mathbf{W}}||_\infty \tag{9}$$

$$|\alpha| \leq (N+1)||\widetilde{\mathbf{W}}||_\infty \leq \alpha_0 \tag{10}$$

$$||\widetilde{\mathbf{W}}||_\infty \leq \frac{\alpha_0}{N+1} \tag{11}$$

2 Task 2

2.1 Part A

For this part we were supposed to compare the approximate and instantaneous gradients. Unfortunately my approximation was very poor and was never even the same order of magnitude as

the instantaneous gradient. I am not sure why this was the case. It seemed as if my approximate gradient was smushed and would never get much larger than 1 therefore never being a good approximation of the instantaneous gradient which could take on seemingly any numerical value up to a few hundred.

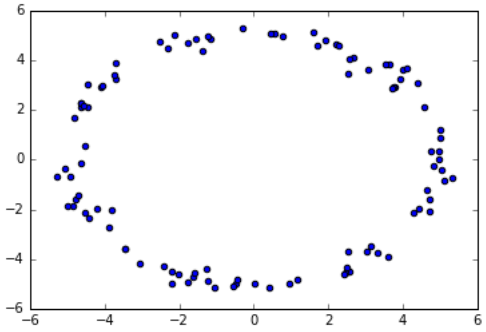

Figure 1: Noisy Circle

2.2 Part B

For this part I used tied weights with no bias node in order to maintain dimensionality. The error never seemed to converge regularly and would often times lead to infinite error therefore five weight initializations were not included since the infinite error would distort the rest of the decent errors.

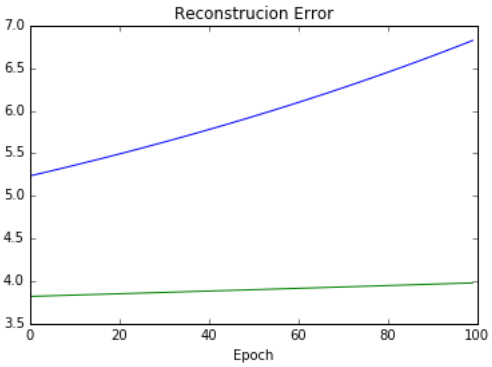

Figure 2: Error vs Epochs

2.3 Part C

For this part we were supposed to varying the output of the hidden node that varies between 0 and 1. This would correspond nicely to the 0 to 2π however since we only have 1 hidden node the most complex function we could approximate was a line. Altering the hidden nodes output did indeed trace out a line though.

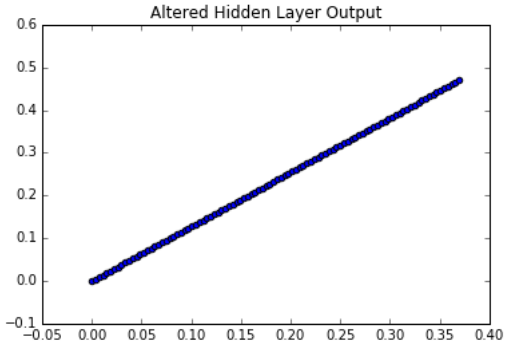

Figure 3: Attempted Circle Recreation

3 Task 3

3.1 Part A

Here we are supposed to create a plot of the data using an XOR classification based upon the sign of the product of the points coordinates. If the product is a negative then we assign it a label of 1 and if positive, it receives a label of 0. The labels of 0 are graphed in red and the labels of 1 are graphed in blue.

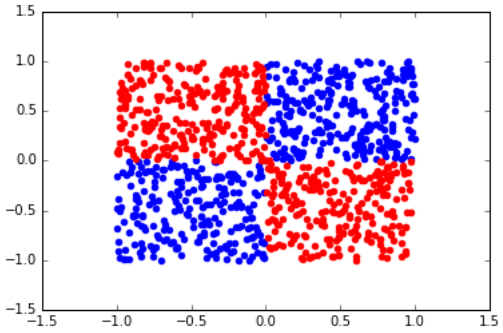

Figure 4: XOR Data

3.2 Part B

To train the network, I used 10 different weight initializations and trained them to see what error they got. I only updated the CE error at the end of every epoch. As can be seen from the graph, for the most part, not much learning occurs after about 800 epochs. Also it is interesting to note that the error stays relatively large during the training meaning that the predictions using two hidden nodes will most likely be poor.

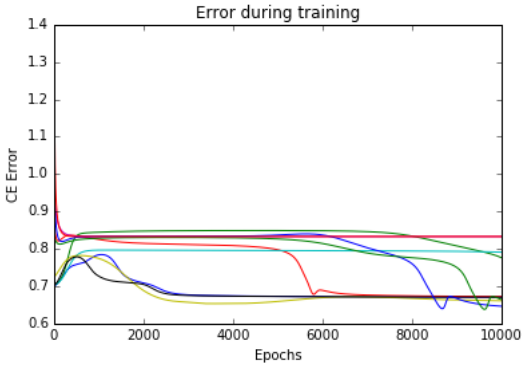

Figure 5: Error of 10 weight initializations

3.3 Part C

Next plots of the predictions on the testing set were made from the optimal weights obtained from 3.2. As can be seen in the Figure 3.3 the network does learn a slope for its classifications but it seems to be slightly more than a linear decision boundary. Since XOR requires 2 decision boundaries, it makes sense that this network will not be able to do a good job of estimating it because it only has 2 hidden nodes. The misclassification rate for this network was 29%.

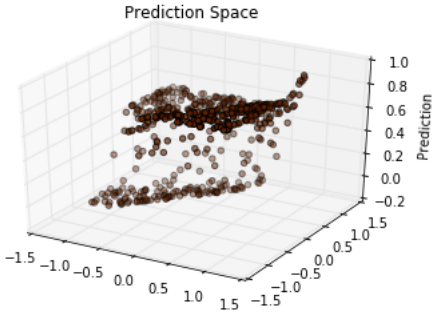

Figure 6: Prediction Space

In order to have a better visualization of the data, I made a 2D plot in Figure 3.3. It is very obviously only able to learn 1 of the 2 decision boundaries and what most likely happened it that it would learn 1 of the boundaries for the different weight initializations and this one happened to most accurately split the data.

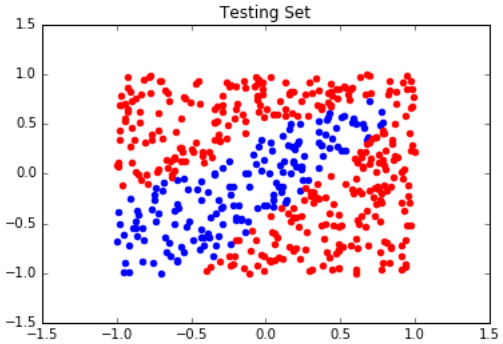

Figure 7: Testing Set XOR Prediction

3.4 Part D

For this part I used networks with hidden layers of varying dimensionality. I trained networks with 2, 3, 4, and 5 hidden nodes each with 10 different weight initializations. Each network's best model was saved and then each was tested against the validation set in order to find a champion model between all hidden nodes sizes. The champion model was then used for prediction on the unseen testing dataset. The prediction space can be seen in Figure 3.4. It clearly captures the classification very well. The champion model was indeed the network with 5 hidden nodes meaning that increasing the amount of nodes does increase the complexity that the decision boundary can take on. This added complexity was able to correctly reproduce the XOR gate. The misclassification rate for the champion model was 6.6% which was a marked improvement over the model with only 2 hidden nodes.

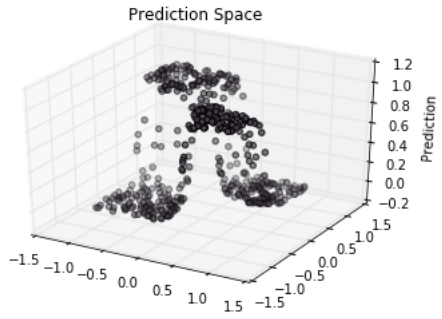

Figure 8: Prediction Space

In order to better visualize the trained models predictions, the original 2d plot was recreated and it is obvious that the model does almost a perfect job of classifying the testing set.

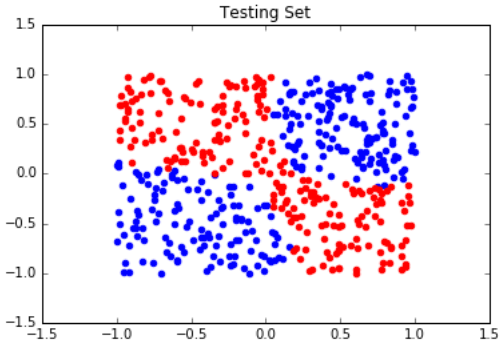

Figure 9: Testing Set XOR Prediction

WORK ORIGINATION CERTIFICATION

By submitting this document, I, Michael Person, the author of this deliverable, certify that

1. I have reviewed and understood the Academic Honesty section of the current version of FITs Student Handbook available at http://www.fit.edu/studenthandbook/, which discusses academic dishonesty (plagiarism, cheating, miscellaneous misconduct, etc.)
2. The content of this Mini-Project report reflects my personal work and, in cases it is not, the source(s) of the relevant material has/have been appropriately acknowledged after it has been first approved by the courses instructor.
3. In preparing and compiling all this report material, I have not collaborated with anyone and I have not received any type of help from anyone but from the courses instructor.

Signature Michael Person Date 4/16/17

6