**CPEG 586 – DEEP LEARNING**

**HOMEWORK 7**

**Name:** Ivan Sangines Escrig

**ID#:** 968606

**Instructor:** Dr. Mahmood

**Date:** May 13, 2019

**TABLE OF CONTENT**

[**INTRODUCTION**](#_Toc446970371) 3

[**SCREENSHOTS:**](#_Toc446970373) 4

[**CONCLUSION:**](#_Toc446970375) 14

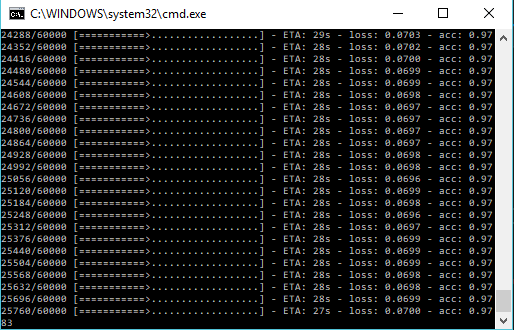
**INTRODUCTION:**

The purpose of this assignment is to start practicing and get better at Tensor Flow library. The idea of this assignment is to start with the really basics of tensor flow, in order to understand how to make a network work in tensor flow. The first exercises consist in creating simple operations such a simple sum or square roots to understand how the nodes are related to each other and how to activate them. The next exercises will get more complicated since I will be creating more complex architectures such as, neural network, Siamese network or triplets.

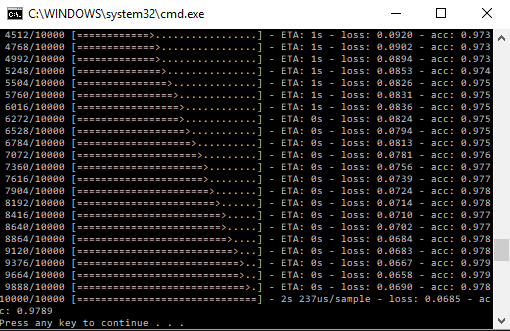
**SCREENSHOTS**

**Problem 1:**

The first problem of this assignment consisted in creating a small NN using Keras.

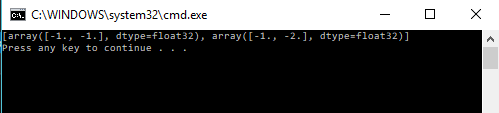


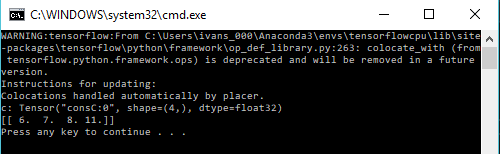
On the following screenshot, we can see the final results of our NN after using Keras. We obtained an accuracy of almost 98% with an error of 0.07.



Exercise 2/3:

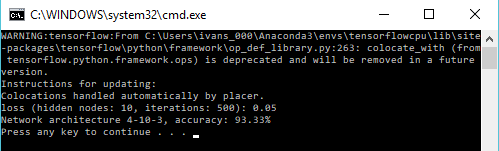
The following exercises of the assignment were related with Tensor Flow library. The idea is to start understanding how Tensor Flow work and learn basic concepts such as, the different between place holders and variables, how to activate an specific node so the whole network is executed. The following screenshots will show the results after creating simple operations such as square roots or a simple equation.



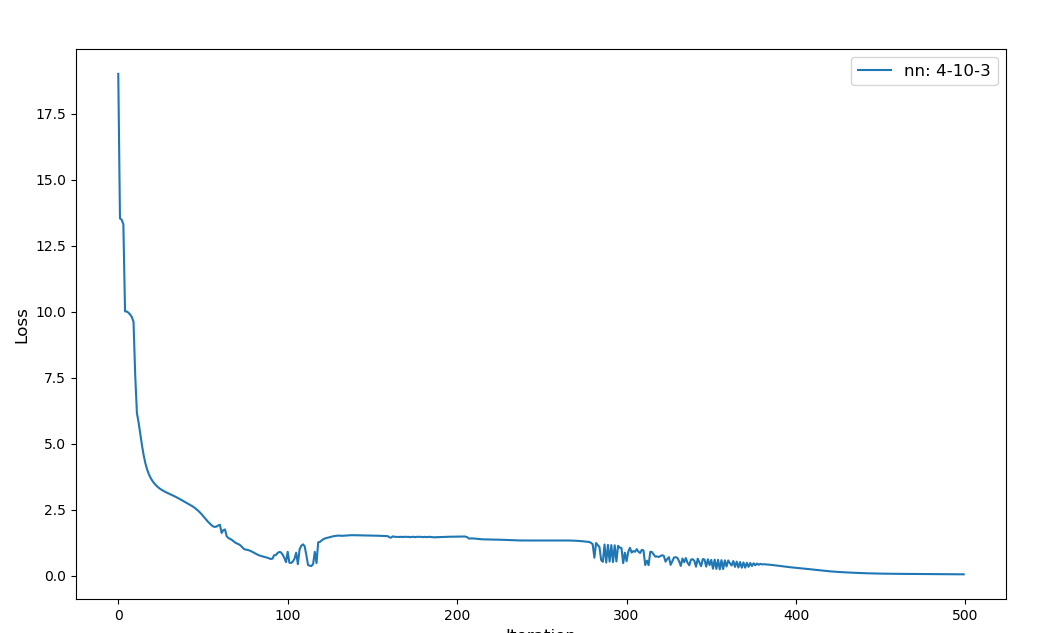


Exercise 4 (NN):

The following exercise will show the results after creating a regular NN using tensor flow. As we can see in the results, we obtained an accuracy of 94% with an error of 0.05

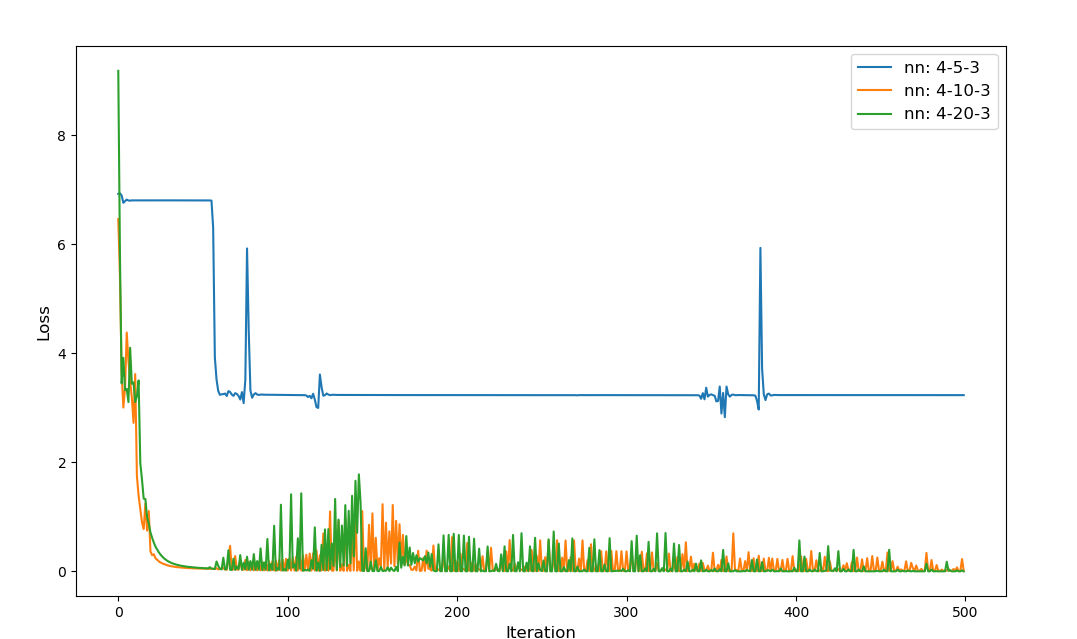


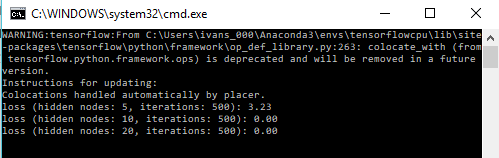
On the following screenshot, we can see the chart for the error minimization of our NN.



Exercise 5 (NN with Adam optimizer):

The objective of the following example was to compare which architecture does a better job for our data. We created 3 different NN, one with an architecture of 4-5-3 neurons in the different layers, second one with 4-10-3 and the last one with an architecture of 4-20-3. Also, I tried to enhance the previous exercise by using the Adam optimizer in the different architectures.

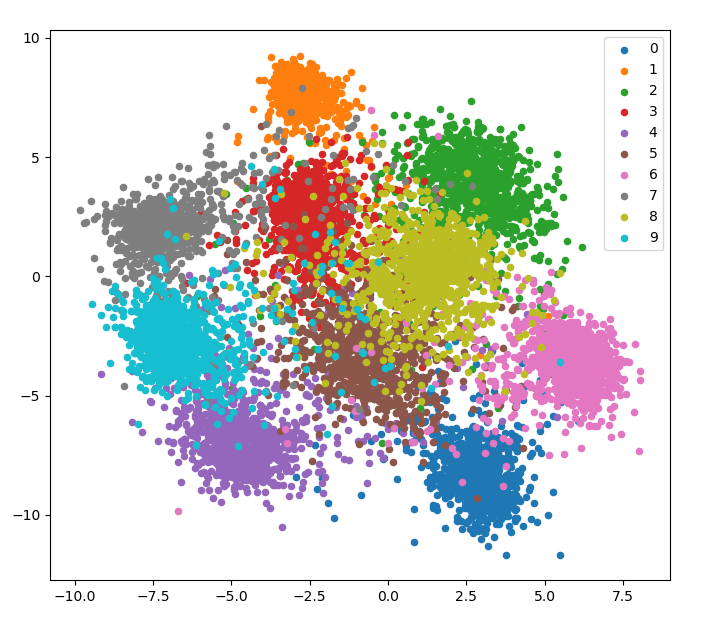


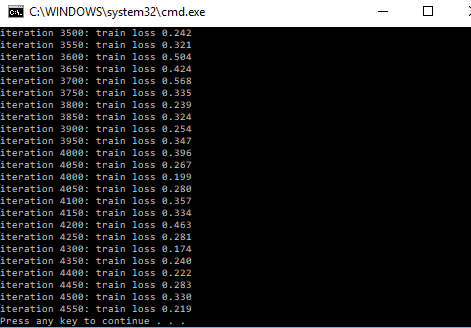
As we can see in the above chart or in the following results, the architectures that produces better results are the 4-10-3 and 4-20-3.

Exercise 6 (Siamese):

The objective of this assignment was to create the first Siamese network. I had different problems while trying to read the data from the website, since it was an https protocol and for some reason, my laptop cannot read that format. The solution I found was to read the data from a local folder and try to format and read it in a way that can simulate the Siamese architecture.

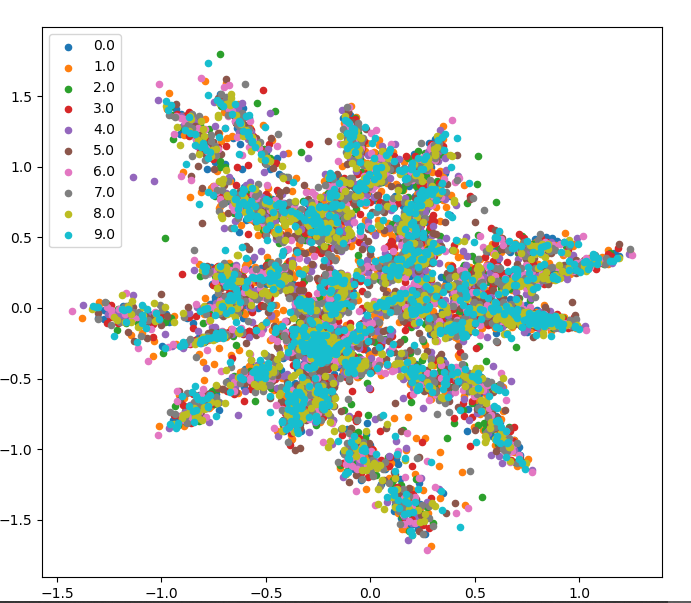
In this screenshot we can see the results of a Siamese Network with an embedding in two dimensions.

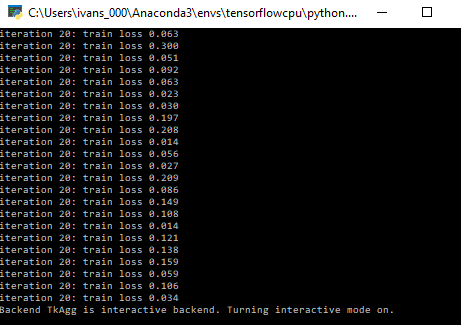


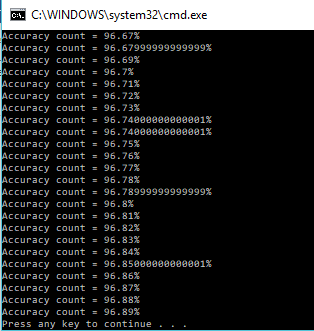


Exercise 7 (Siamese with an embedding of 100 dimensions)

The following example was to recreate the previous Siamese network but using a more “realistic” embedding. In this example, the embedding is on the 100 dimensions.



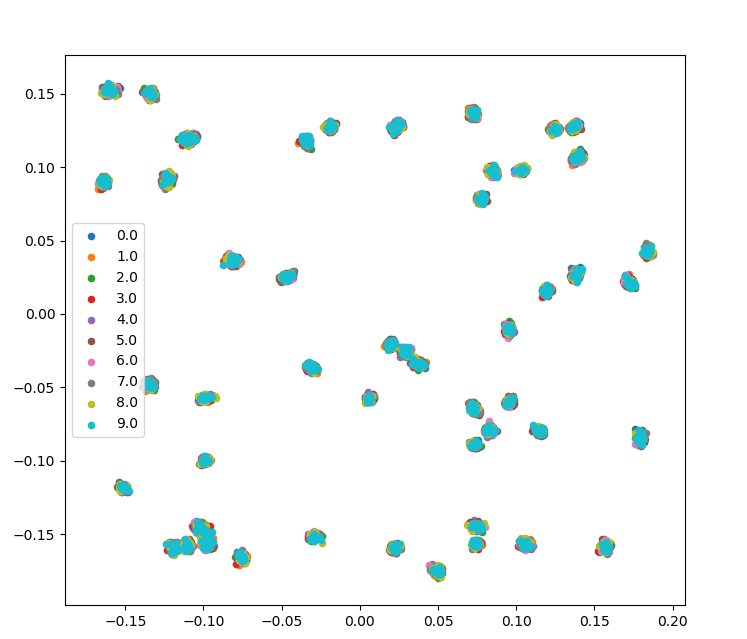


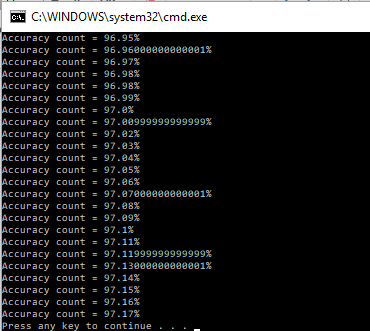


As we can appreciate in the above results, we can see a great improvement in the efficiency while using a Siamese network instead of a simple NN. The loss for the Siamese network is about 0.03, which is not much different from the one obtained in the NN. However, the big different comes when we compare the accuracy of the NN and Siamese network, while the NN did not give me any accuracy better than 93%, the Siamese network produces an accuracy of 96%.

**Problem 2 (Siamese using a new loss function)**

In the following exercise, I was trying to improve the above Siamese network by using the new loss function described in the handout. This new loss function is based on a combination of the both embedding obtained from the different inputs. As we can see in the results, the improvement is not really big (it went from an accuracy of 96.8 to 97.1), but one of the main reasons might be the type of data we have. The MNIST data does not have much information, meaning the data is too simple. If we were to use a different data, we might be able to see a better improvement.





**Problem 3:**

The goal of this assignment was to create a triplet network (using three different Siamese networks). The idea of the triplet is to create a stronger and better embedding by having three instead of just two networks in our architecture. The idea is that two of the networks in the triplet will share input (will always be a match or the same) and the third one will be the one that will have different input and embedding trying to “fool” the system. The idea is that now we will have two outputs or embedding, which will be based on the comparison of two single outputs or embedding making the new embedding more strong and unique.

The results obtained after creating the triplet were not too different from the ones obtained on the simple Siamese with new loss function. However, as mentioned before, the main reason for that to happen is that our data is too simple and does not contain much meaningful information.

**CONCLUSION:**

This assignment really helped me understand how Tensor Flow library work. I now can understand how are all the nodes and architectures created or related to each other and how to activate a node to make the whole architecture run. I have also learned how to create variables and place holders to further use them.

This assignment also helped me to learn new loss functions for a Siamese network based on the combinations of the two different outputs. I have also learned how a triplet architecture work and how the efficiency can be improved by creating a triplet architecture. The results might not be showing a huge improvement of the triplet while comparing with the regular Siamese architecture, one of the reasons can be the type of data we are using. The MNIST data does not have much information on it, however, if we were using another type of data with more relevant information in it, we will be seeing a good improvement from the triplet architecture.