**Report – ICP5**

1. **Question:** Firstly, change the dataset being used in the use case explain in the class.

**Answer:** I changed the dataset to a text which contains “Alice in wonderland story”

1. **Question:** Write down the code to save the graph then plot the graph in tensor board

**Answer:** I have defined a FileWriter and SummaryWriter in tensor board where the overall Summary is written to file path “./graphs/rnn” and SummaryWriters over loss and accuracy are merged and written to the above file path. This writer is attached to the graph after the session has been initialized and the parameters are running on the variables to give output.

**my\_writer = tf.summary.FileWriter("./graphs/rnn")**

**with tf.Session() as session:**

**session.run(init)**

**my\_writer.add\_graph(session.graph)**

**my\_writer.add\_summary(smm,step)**

and finally, summary is written at each “display\_step” for all the “training\_iterations”

A close up of a map

Description generated with very high confidence

1. **Question**: Write down the code to save accuracy and loss then plot accuracy and loss in tensor board

**Answer: loss\_summary = tf.summary.scalar("loss", cost)**

**acc\_summary = tf.summary.scalar("accuracy", accuracy)**

The code is explained above

A screenshot of a computer

Description generated with very high confidence

1. **Question:** Change the parameters like Learning rate change the n\_input and n\_hidden parameters and plot the result to see the change, also see the printed version changed.

**Answer:** I have changed the parameters of n\_hidden and n\_input which speak about the number of layers and the no. of words that it considers in each stride like 3 or 4 which are changed to 1024 and 4 from what is given in inClass.

1. Change the optimizer and plot to see the result

I have changed the optimizer to AdadeltaOptimizer is an algorithm for gradient-based optimization that does just this: It adapts the learning rate to the parameters, performing larger updates for infrequent and smaller updates for frequent parameters. For this reason, it is well-suited for dealing with sparse data.

**optimizer = tf.train.AdadeltaOptimizer(learning\_rate=learning\_rate).minimize(cost)**

1. Change the code from one stack LSTM to two-layer stack LSTM then see the result

**rnn\_cell = rnn.MultiRNNCell([rnn.BasicLSTMCell(n\_hidden),rnn.BasicLSTMCell(n\_hidden)])**

It increases the accuracy of the current Single -Layer stacked LSTM(Long term short memory)