### CS 5984: Deep Learning

### Homework 2

### Due Date: October 26th, 2017 (4:00PM) Total: 100 Points

**Note: All implementations are required to be accomplished and submitted using the *Jupiter notebook*. Show your results and necessary comments in the notebook. *No handwritten homework is accepted.***

**Activation Functions and Dropout (30 points)**

1. In homework 1, we have built a deep feedforward neural network with two hidden layers and implemented the stochastic gradient decent (SGD) algorithm to train the network. Based on your codes, replace the ‘sigmoid’ activation functions with ‘relu’ and retrain the model. Compare the results with those in homework 1 (Question 2).
2. Suppose the dropout rate of neurons for layer 2 is 1/3, which means only two out of three neurons will be kept during the training for each batch, where the batch size is 1. Implement the backpropagation and train the network. Show the performance on the Test Data.

**Convolutional Neural Networks (35 points)**

1. Implement the following CNN using the tensorflow:
2. Input: Images
3. Convolutional Layer 1: filter (), stride (1, 1), activation (‘relu’), padding (None).
4. Max-pooling: filter/kernal (), stride (2, 2).
5. Convolutional Layer 2: Filter (), stride (1, 1), activation (‘relu’), padding (None).
6. Max-pooling: filter/kernal (), stride (2, 2).
7. Flatten.
8. Fully connected layer: number of neurons (1024), activation (‘relu’).
9. Output layer: number of neurons = number of classes, activation (‘softmax’).

Note: You are free to try out different parameters.

1. Train the above CNN on MNIST and Cifar-100 training datasets. Show the intermediate output, such as loss and accuracy. You can find the datasets from the following links:

<http://yann.lecun.com/exdb/mnist/>

<https://www.cs.toronto.edu/~kriz/cifar.html>.

1. Evaluation the performance of the CNN model using the testing datasets. You can try different evaluation metrics, like accuracy. Pick 9 examples from MNIST and Cifar-100 datasets, respectively, plot the images and show true labels and predicted labels.
2. Select one of the images from cifar-100 and visualize the feature maps of the first and second convolutional layers.
3. Plot the filters of the first convolutional layers.
4. Repeat 2-4 by adding additional convolutional layers, remove the max-pooling, and changing the number and the size of filters. (Not Required)

**Recurrent Neural Networks (35 points)**

1. Implement the following RNN using **tensorflow**



* 1. The input of the RNN is a sentence. Each word in the sentence is represented by a vector . The dimension of the word vectors is 50.
  2. consists of neurons. The activation function is tanh.
  3. The activation function for the output layer is a softmax function.
  4. The length of the sequence is 100, i.e., .

1. Apply your model to the attached sentiment analysis data. There are three files in this data:
   1. ‘train.csv’ contains 20,000 sentences. 10,000 of them are labeled as positive and the rest are labeled as negative. The labels of the sentences are in the first column of the file. Each sentence has at most 100 words. If the number of words in a document is less than 100, you should extend it to 100 by appending 0 (zero vectors).
   2. ‘test.csv’ has 5,000-labeled sentences for evaluating the performance of the models.
   3. In ‘word-vectors.txt’, each word is represented by a 50-dimensional vector.
2. Train the vanilla RNN model on the training data. Evaluate the performance of the model on the testing data. Show your results.
3. Implement the LSTM (64 cells) and GRU (64 cells) as well. Apply them to the sentiment analysis data.
4. Compare the performance of vanilla RNN, LSTM and GRU.