RNN (Deep Leraning Assignment - 4)

In this assignment, you will learn to develop a recurrent neural network (RNN) from scratch. You need to train an RNN for language modelling and use the trained model for the following tasks;

1. Predicting the last word of a sentence.
2. Generating a sequence of words.

Dataset: Please find the train and test files.

The steps for preparing data the word-based language model are as follows;

1. Read data from the file.
2. For the purpose of preprocessing you can refer (https://medium.com/ml2vec/training-and-tuning-a-lstm-to-classify-yelp-reviews-4d37b8aa2e91). You can do further preprocessing of your choice.
3. Form a dictionary for unique words.
4. Form an embedding matrix using pre-trained embeddings.

The initial steps are as follows;

1. Define an embeddings matrix for the words.[**a. One-hot encodings b. Pre-trained embeddings**]
2. Break the entire training data into batches.
3. Allocate parameters: Define the parameters and hyper-parameters (weight matrices etc.)
4. Attach the gradients to the relevant parameters
5. *Define a function for clipping the gradients. Use value in the range [-10, 10].*
6. Define a function for softmax activation.
7. Define the RNN model using the parameters defined in Step 3.
8. Define a function for cross-entropy loss, calculating average loss over sequence and SGD optimizer.

Training phase:

```
For epoch in MAX_EPOCH:
    For batch in (training batches):
        1. Forward pass through the network
        2. Find the loss value
        3. Get the gradient
        4. Apply clipping
        5. Update weights
```

Print the average loss value for each epoch and plot the training error vs epochs. Save the parameters to a file after training.

Evaluation phase:
For each sentence in the test data;

1. **Evaluation function 1:** If the length of the sentence is ***n***, then encode the sentence upto ***n-1*** words and predict the ***n***th word.
2. **Evaluation function 2:** In this function, given the first half of a sentence, you have to generate the second half of a sentence. e.g., In the sentence, “I went to the market to buy some fruits”, you have to generate the words “to buy some fruits”. Given a sentence of length n, encode the sentence upto $floor[n/2]$ words and generate the rest of the words using the following steps;
    a. Run your model on the first half of the sequence to obtain the intermediate hidden state
    b. Use the obtained hidden state as the initial state of your RNN for predicting the second half.
    c. Pass the network the last word of the first half sequence.
    d. Run one step of forward propagation to predict the next word.
    e. For len(sequence to be generated):
    i) Use the predicted word in the previous step as input to the RNN to predict the next word.

3. Report the %age accuracy for the task of predicting the last word.
4. Report the accuracy for the task of sentence completion (Compute accuracy only for the part of the sentence generated).
5. **Repeat your experiments using a. One hot encoding b. Pretrained embeddings [You may use the fasttext embeddings]**

<u>**Deliverables:**</u>

1. <u>**Code in jupyter notebook [Training and evaluation]**</u>
2. <u>**Save the weights**</u>
3. <u>**Your results for the experiments in a report (pdf)**</u>
4. <u>**Zip your folder containing the code, report and parameter and name the folder as following**</u>
   <RollNumber_4.zip>