Programming Assignment 4

Background
Let's build a science fiction writer! How about Jules Verne reloaded? Haha
Anyway, here are going to design and implement two recurrent neural networks.

Dataset
Here is a list of Jules Verne's 10 masterpieces. Choose any 3 (or more) texts, and concatenate them which you would want your neural network to learn the structure of writing by this great sci-fi legend. RNNs have been trained on highly diverse texts (novels, song lyrics, etc.) with success. The links below are from Gutenberg Books which is a source of free (legal) books where you may download full novels in a .txt format (i.e., Plain Text UTF)

1. The Mysterious Island [http://www.gutenberg.org/ebooks/1268]
2. A Journey to the Center of the Earth [http://www.gutenberg.org/ebooks/18857]
3. Around the World in Eighty days [http://www.gutenberg.org/ebooks/103]
4. Five weeks in a Baloon [https://www.gutenberg.org/files/3526/3526-h/3526-h.htm]
5. Twenty thousand leagues under the sea [http://www.gutenberg.org/ebooks/2488]
6. A Journey to the Center of the Earth [http://www.gutenberg.org/ebooks/18857]
7. From the Earth to the Moon [http://www.gutenberg.org/ebooks/83]
8. Five weeks in a balloon [http://www.gutenberg.org/ebooks/3526]
9. Off on a comet [http://www.gutenberg.org/ebooks/1353]
10. The master of the world [http://www.gutenberg.org/ebooks/3809]

Character Level Recurrent Neural Network (RNN)

You are going to use a character-level representation for this model. To do this, you may use extended ASCII with 256 characters. As you read your chosen training set, you are going to read in the characters one at a time into a one-hot encoding, which I introduced in class, that is, each character will map to a vector of ones and zeros, where the one indicates which of the characters is present. Your RNN will read in these length-256 binary vectors as input.

Task 1: Vanilla RNN (character-level)

Implement Backpropagation through time: In a vanilla RNN with a single hidden layer you must have three sets of weights: U, V, W. And you use softmax units for the output layer and tanh units for the hidden layer.

- Now, train your recurrent neural network using the dataset you created above. You are free to choose learning parameters (sequence length, learning rate, etc.).
- Report the training loss vs epochs as a plot.
- During training, choose 5 breakpoints (e.g., you train the network for 100 epochs and you choose the end of epoch 20, 40, 60, 80, 100) and show how well your network learns through time. You can do it by feeding in the network a chunk of your training text and show what is the output of the network. Also, report about gradient check routine.
- You are going to explore how the network learns when we change the following parameters:

(a) Number of hidden units: Try doubling and halving your number of hidden units. And after training, plot the training loss vs the number of training epochs, and show the text sampling results. Discuss your findings.

(b) Sequence length: Try doubling and halving your length of sequence that feeds into the network. And after training, plot the training loss vs the number of training epochs, and show the text sampling results. Discuss your findings.

Task 2: LSTM (character-level)

Here, you will be generalizing your RNN as in Task 1 to utilize the LSTM units. Repeat doing the training and showing/reporting results in the format from your Task 1.

Word Level Recurrent Neural Network

You are going to use a word-level representation for this model. To do this, you may want to use a vocabulary size of 5000 most frequently used words by Jules Verne. As you read your chosen training set, you are going to read in the words one at a time into a one-hot encoding, which I introduced in class, that is, each character will map to a vector of ones and zeros, where the one indicates which of the words is present. Your RNN will read in these length-5000 binary vectors as input.

Task 3: Vanilla RNN (word-level)

Implement Backpropagation through time: In a vanilla RNN with a single hidden layer you must have three sets of weights: U, V, W. And, you use softmax units for the output layer and tanh units for the hidden layer.

- Now, train your recurrent neural network using the dataset you created above. You are free to choose learning parameters (sentence length, i.e., number of words in each, learning rate, etc.).
- Report the training loss vs epochs as a plot.
- During training, choose 5 breakpoints (e.g., you train the network for 100 epochs and you choose the end of epoch 20, 40, 60, 80, 100) and show how well your network learns through time. You can do it by feeding in the network a chunk of your training text and show what is the output of the network. Also, report about gradient check routine.
- You are going to explore how the network learns when we change the following parameters:
  (a) Number of hidden units: Try doubling and halving your number of hidden units. And after training, plot the training loss vs the number of training epochs, and show the text sampling results. Discuss your findings.
  (b) Sentence length: Try doubling and halving your length of sentence that feeds into the network. And after training, plot the training loss vs the number of training epochs, and show the text sampling results. Discuss your findings.

Task 4: LSTM (word-level)

Here, you will be generalizing your RNN to utilize LSTM units. Repeat doing the training and showing/reporting results in the format of Task 3.