CS-564 ML: Assignment 3

Instructions:

1. Coding must be done using python and you are not supposed to use already available libraries, pytorch and tensorflow etc..
2. All the assignments should be completed and uploaded before the deadline.
3. Markings will be based on the correctness and soundness of the outputs. Marks will be deducted in case of plagiarism.
4. Proper indentation and appropriate comments are mandatory.
5. You should zip all the required files and name the zip file as roll_no.zip, eg. 1921cs28.zip. In case of groups, merge them as 1921cs28_1921cs29.zip
6. Upload your assignment (the zip file) in the following link:
7. Make necessary assumptions if required. For further clarification, you can write an email:
8. Reporting parts of the questions can be done as comments at the end of the file, or in a separate file.
9. You can make use of any existing implementation for Baum Welch algorithm

Dataset: Brown_dataset
https://www.dropbox.com/s/960u5jbd27epopl/Brown_train.txt?dl=0

ASSIGNMENT: You are to design and implement a HMM based model from scratch. The dataset has text stored in format ”token/PoS” where “token” is the word and “PoS” is the part of the speech tag of that token in the current context. Assume PoS as the hidden states and tokens as the visible outputs.

Steps to execute:

1. Format all sentences with “start of sentence” and” end of sentence” tokens.
2. Split the data in 80:20 for training and testing.
3. Create a vocabulary for all the tokens in the training text with their count.
4. Using the training dataset, create Initial probabilities (probability of the token given “start of sentence” token), State transition probabilities (probability of PoS given the previous PoS) and Emission probabilities (probability of a token given the current PoS). Use the Baum Welch Algorithm.
5. Remove the PoS information from the data.
6. Now code the solution to the following issues with HMM-

- **Evaluation problem**: Given the HMM and the observation sequence, calculate the probability that model M has generated sequence O. Do this for all the sentences in the test set. Report the probabilities for all utterances.

- **Decoding problem:** Given the HMM and the observation sequence, calculate the most likely sequence of hidden states that produced this observation sequence. Design the greedy/Viterbi solution here. Do this for all the sentences in the test set. Verify if they match the state sequence of the test set. Report the overall accuracy.

- For the decoding problem, now design and code the beam search variation. Do this for all the sentences in the test set. Verify if they match the state sequence of the test set. Report the overall accuracy. Compare with the Viterbi algorithm. Also verify that the beam search with beam_width=1 behaves exactly as the Viterbi algorithm(same sequences are generated for a given input text).

Documents to be submitted:

1. Steps to compute the Emission Probability and Transition Probability matrices
2. Overall accuracy of PoS Tagging
3. Consider a Trigram model of HMM and compare the accuracy with the Bigram model as stated above