Name: Anne Sai Venkata Naga Saketh
USC Email: annes@usc.edu
USC ID: 3725520208

Assignment-3 Homework Report:

Questions in Task 1:

What is the selected threshold for unknown words replacement?

Ans: 3

What is the total size of your vocabulary?

Ans: 16290

What are the total occurrences of the special token '<unk>' after replacement?

Ans: 32537

Explanation of the code:

I am using the training data to build a vocabulary dictionary and exclude terms that don't occur frequently enough by using a preset frequency threshold of 3. I am iterating over each line of the input files and parsing words, tags, and their frequencies. I am keeping track of the frequency of each term and tag. I am using <unk> token instead of a word when its appearance falls below the preset threshold. Later, I wrote the vocabulary dictionary containing word frequencies and indices in the text file.

Questions in Task 2:

How many transition and emission parameters in your HMM?

Ans: Number of transition parameters: 23373
Number of emission parameters: 1392

Explanation of the code:

Initially, I am setting up dictionaries to hold the transition and emission parameters. Then, I interpret the words, tags, and their frequencies as I go through each line of the training data iteratively. I am updating the transition probabilities between tags based on prior and current tags and incrementing the emission probability for each tag-word pair. If there is no prior tag, I am adding a new tag called 'start' with the initial probabilities. Then, I am normalizing these chances. I'm printing the number of emission and transition parameters right now in the console. Lastly, I'm storing the HMM model in a JSON file called 'hmm.json', which will store the model parameters for later usage.

Questions in Task 3:

What is the accuracy of the greedy decoding algorithm on the dev data?

Ans: Total: 131768, correct: 122390, accuracy: 92.88%.

Explanation of the code:

I have configured the output file name to "greedy_dev.out" for the predictions generated on the development data and “greedy.out” for the predictions generated on the test data. I'm initializing a list to hold data that will be written later. The variable {prev_tag} is initialized using the start token we defined in the model creation step. Then, I scan over each line of the development data, looking for a single element at the start of a new sentence. In this scenario, I reset the previous tag to the start token and append the line to the data that must be published. I get the current word and index if not. If the word is not in the vocabulary we generated in the first step, "" is used in its place. After initializing the probability and temporary tag variables, I go through each tag in the tag frequency dictionary. For every currently used tag, I check the emission and transition probabilities for the word and tag combination. I adjust the temporary tag if the possibility is now higher. After changing the previous tag with the temporary tag, I write the required line and add it to the data. In the end, I write all the data to the respective output file based on the input data.

Questions in Task 4:

What is the accuracy of the Viterbi decoding algorithm on the dev data?

Ans: Total: 131751, correct: 124384, accuracy: 94.41%.

Explanation of code:

I am implementing the Viterbi algorithm method for the prediction of the parts of speech. I am initializing the back pointer and Viterbi score matrices initially to empty. Then, given observed words, I compute the most likely sequence of tags by completing the initialization, recursion, and termination phases. Then, I update the Viterbi scores based on transition and emission probabilities by iteratively going through each word in the input data, either the development or the test data. Then, I translate state indices to state labels after fully decoding the language and returning the best route. During this procedure, I am also implementing the sentence boundaries and words that are not in the dictionary. Finally, I am writing the output of the predictions to the "viterbi_dev.out" for predictions on the development data and the output to the “viterbi.out” for predictions on the test dataset.