CSCI544: Homework Assignment No.3

TASK 1: VOCABULARY CREATION

What is the selected threshold for unknown words replacement?

Ans: 1

What is the total size of your vocabulary?

Ans: 23183

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

Ans: 20011

Firstly, extracted words and its pos tag from the training dataset file using pandas. Then, grouped the dataframe by their occurrences. Replaced the words which have occurrences less than equal to 1 with <unk> tag.

Created a new dataframe containing only distinct words with their occurrences. Find the index at which <unk> tag placed. Saved that row into a variable then dropped the index containing <unk>. Concatenated the vocab dataframe placing <unk> at top with its count and rest of the vocab in descending order by their occurrences. Added a id column for indexing the words and swapped it with occurrence column to be consistent with file format as stated in the task. Saved the vocab dataframe in a text file.

Extracted all the distinct pos tags from the

TASK 2: MODEL LEARNING

How many transition parameters in your HMM?

Ans: 2070 [(length of pos tags x length of pos tags) +initial probability (45)]

How many emission parameters in your HMM?

Ans: 1043235 [(length of pos tags x length of vocabulary)]

Here we are calculating the transition matrix, emission matrix and initial transition probability.

Given a 2D list of sentences where at each index we have one complete sentence and for each sentence we have tuples of word and its pos tag. Then we have list of distinct tags. Given the

formula given in the HW pdf we have calculated transition and emission matrix. After that we stored all the possible states into a dictionary and saved into a json file.

TASK 3: MODEL LEARNING

What is the accuracy on the dev data?

Ans: 0.94

Extract the sentences and their tags from the validation dataset in a similar way we did for training dataset.

Evaluate our trained HMM model using greedy decoding. In greedy decoding we first take maximum of the t(s1) * em(x|s) and store the tag as previous tag and use it for the next state till we exhaust whole sentence and save all the pos tags in a list. Now we have predicted for all the sentences we calculate the accuracy.

Extract the sentences for testing dataset and evaluate using greedy decoding and save the sequence of tags into a file ‘greedy.out’ in a similar format as mentioned in the HW pdf file.

TASK 4: VITERBI DECODING

What is the accuracy on the dev data?

Ans: 0.95

The sentences we extracted in the task 3 we will use it for Viterbi decoding. Viterbi decoding use all the tags and calculate the score, save it into a list and use it for the next state and calculate the score for all the tags again, till we get at the end of the sentence. Cache dictionary will store at what position of the sentence and pos tag we got the score.

Now we will back propagate through cache list taking only maximum probability at each position and building sequence for each position of the sentence.

Calculate the accuracy on our newly generated sequence with validation sentences.

For testing dataset, we save the best sequence in an output file as ‘viterbi.out’.