Aravind Krishnan

Task 1 - Parsed the train dataset using csv.reader and used defaultdicts to keep count and remove words occurring less than a threshold value number of times. Selected the threshold to be 3. Created a vocab.txt by sorting word_count defaultdict.

1. Threshold = 3
2. Total size of vocabulary - 16920
3. Total occurrences of <unk> - 32537

Task 2 - Calculated transmission and emission probabilities by running through reader output and using the formula. Transmission counts are counted from s to s' and then divided by total counts of s to get transmission probability(s'|s). Emission counts are counted from number of counts of state giving a word and then divided by total counts of state to get emission probability(x|s). Since we are using a defaultdict, we did not fill in zero-values as they will be taken care of during runtime. Number of tags = 46 and number of words in vocabulary = 16920.

1. Transmission parameters - 1375
2. Emission parameters - 23373

Task 3 - Performed greedy decoding on the dev data by selecting the most likely state for the first word, and then used that to decode maximize P(s2|s1) to continue to choose local optimal value (local maxima).

1. Achieved an accuracy of 91.85% on the dev data.

Task 4 - Performed Viterbi decoding on the dev data using the Viterbi matrix and backpointer matrix. Once the Viterbi matrix is calculated using the formula, the best path is constructed backwards by selecting the maximum probability value in each column of the Viterbi matrix.

1. Achieved an accuracy of 93.23% on the dev data.