CSC384H1F

Assignment 4: HMM Part-of-Speech Tagging

Junrui Chen - chenj591

Due: December 1, 2022

Heuristic

1. The following code shows how the lines in training files are separated into sentences.

```
56  def start(file_lst: List[str]) -> None:
57      for file in file_lst:
58          f = open(file, "r", encoding='utf-8')
59          temp = []
60          for line in f:
61              n = line.index(" ")
62              word = line[:n]
63              tag = line[n+3:-1]
64              if tag not in init_prob and len(tag) == 7:
65                  tag = tag[4:] + "-" + tag[:3]
66              if tag not in init_prob:
67                  if word in emission_prob and len(emission_prob[word]) != 0:
68                      tag = max(emission_prob[word])
69                  else:
70                      tag = "ZZ0"
```

Notice that some words in training files don't have valid tags, that is, their tags are not in the list of all 76 the part-of-speech tags. These invalid tags may belong to the list of ambiguity tags consisting of two tags separated by a dash, but with a reversed order for the two tags. To fix this problem, we swap the position of the tags before and after the dash.
If this doesn't work, or the actual problem does not match this situation, we can check if this word has appeared. If so, its will be tagged with what it most likely is based on its emission probability. Else, we just tag it with a tag which appears most frequently. After counting we know that this tag is "ZZ0".

2. The following code is part of my Viterbi algorithm.

```
166     # final = max(prob[len(E)-1])
167     final = "PUN"
168     j = len(E) - 1
169     while j >= 0:
170         path.append(final)
171         final = prev[j][final]
172         j -= 1
173     return path[::-1]
```

Notice that each entry of $prev$, $prev[i][j]$, represents what the most likely previous state is when the word at index i of the sentence has tag j. In Viterbi algorithm, based on the data stored in $prev$, we can backtrack the path for the most likely sequence of hidden states.
When we start from the end of the path, instead of find the tag which has the highest probability for the word at the last index of the sentence, we can simply start from "PUN". Since "PUN" is general separating mark, including period("."), exclamation mark("!") and question mark("?"), which we used to split lines into sentences. In other words, almost all sentences end with a word/symbol whose tag is actually "PUN".