Project Report

Team Name: Trigger
Team Members: z5147976 Tingfeng Lin
z5125251 Yi Xiao

● Implementation details of Q1

Question 1 is just an application of HMM, Viterbi algorithm and Smoothing, and it does not require us to create new things to finish this question except sorting 'state path'. According to those data read from State file and Symbol file, we can generate two matrixes for transmission rate and emission rate. We use the Viterbi algorithm to calculate the probability of each symbol by these two matrixes.

If there is an Unknown symbol in the query, we should use the Laplacian Smoothing to generate its probability. Smoothing method is explained clearly in project specification and we do not want to meaninglessly repeat those words here.

Note that there may be several 'state paths' all having the largest probability to emission a symbol query, so we have to sort those 'paths' and select the best one following requirements. Q2 has a quite similar part although it requires the best K.

● Details on how to extended Q1 to Q2

For Q2, we need to slightly change original Viterbi algorithm in Q. Since Q2 require outputting top K probabilities and their paths, we can store at most top K probabilities and paths for each state in the iteration of each symbol query. If there are some paths having same probability 'compete' Kth position, we also need to execute sort function mentioned above and keep only K paths, because those paths having same probability outside K are in the same state which means they have same transmission rate and emission rate in later iteration of symbol query and their priority will not change.

Therefore, for each symbol query, we just need to keep the top K paths for each state.

At the end of the query, we sort N*K paths (N is the number of states) and top K paths of them are our result.

- **The approach of advanced decoding**

By observing the data of the symbol matrix, a lot of unknown can be found in this matrix, which means there are several noises. It might affect the accuracy of the result. In question 3, we use the Absolute Discounting Smoothing method instead of Laplacian Smoothing method to initialize the symbol matrix. The aim for the method is that from all non-zero probability symbol at every state, we subtract a small amount of probability, then this probability p is assigned to those symbols which the probabilities are zero by the using maximum likelihood estimate. After processing this symbol matrix, we still keep using the Viterbi algorithm to calculate the highest probability of the path.

- **The major differences between Q1 and Q3**

In Q1, we are using add-1 smoothing method to deal with zero probability symbol. But in Q3, we are using Absolute Discounting Smoothing method to deal with zero probability symbol. The formula of Absolute Discounting Smoothing is shown below.

$$P(V|S) = \begin{cases} P(V|S) - p & if\ P(V|S) > 0 \\ \dfrac{vp}{N - v} & otherwise \end{cases}$$

where

$$p = \frac{1}{T - v}$$

v is the number of symbols assigned a non-zero probability at a state S and N is the total number of symbols. T is the total number of symbol emitted at state S.

Using this method, it can make the probability of the unknown symbol more precise according to a different state.

- **How to execute the code**

In the submission file, there has a python file, which is called submission.py. In Q3, we directly reduced the Total Number of Incorrect Labels on the given dev set by a margin > 17. Actually, we reduce the Incorrect Labels from 134 (in Q1) to 114(in Q3). The improvement is 20 labels. If you are testing our code, just invoke advanced decoding method, that is our bouns part.