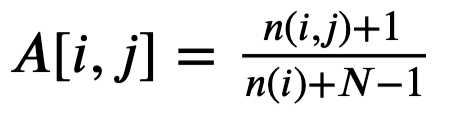
COMP9318 Project Report

First, read the three files State\_File, Symbol\_File and Query\_File and process them.

State\_File : parse\_state\_file\_to\_trans\_pro(State\_File)

The first line of this file is the number of the entire state(Num\_state), and the next Num\_state lines is the name of each state (index number from 1 to Num\_state +1). Save these names in a list in order, with an index for each name. The last two lines of these lines are the begin state and the end state. From Num\_state +1 to the end of the entire file, is the frequency of transitions between states. Each row has 3 values, each corresponding to state i, state j and number. My method is to create a new matrix and then fill in the probability of each state transition into the corresponding matrix. Use the smooth method to process before adding.



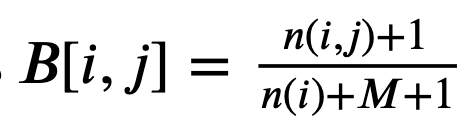
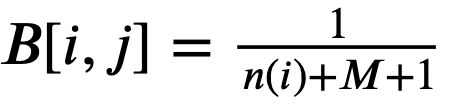
When there is no value from state i to j, 1/(n(i)+N -1) is the probability of A[i, j].

Since the begin state cannot be transferred, the end state cannot be rolled out, so their probability is 0. Finally, a matrix of num\*num is obtained, and the corresponding A[i, j] is the probability that state i goes to state j.

Mark this matrix as trans\_matrix.

Symbol\_File : parse\_symbol\_file\_to\_Emission\_pro(Symbol\_File, states)

Same as the State\_File, the first line of this file is the number of the entire symbol(Num\_symbol), and the next Num\_symbol lines is the name of each symbol (index number from 1 to Num\_symbol +1). Save these names in a list in order, with an index for each name. From Num\_symbol +1 to the end of the entire file, each row has 3 values, and each corresponding value is state, symbol, and the number of symbol. My method is to create a new matrix Num\_state\* (Num\_symbol +1). The last column corresponds to UNK.

By extracting the number of symbols in the corresponding position in the file, use the method of smooth, add 1 when the value is 0, and finally fill the probability into the matrix.

Mark this matrix as emit\_matrix.

Query\_File : parse\_query\_file\_to\_index(Query\_File, emit\_names)

Each line of this file is a query statement, firstly, split them and take a word from the statement. Every word and symbol must be retained. If the extracted word or symbol does not match a known symbol, it is classified as UNK. Output the index corresponding to each word or symbol.

Q1 : result\_viterbi(trans\_matrix, emit\_matrix, pi, obs)

For each query, mark its length as len\_of\_obs, and mark the number of states as LEN. First create a matrix of all 0s with length and width are len\_of\_obs and LEN, respectively. Each row corresponds to the next symbol sequence number.

First, for the first symbol, the flag state is A[i,i], and the symbol index is B[i,j], and the values of the corresponding positions trans\_matrix, emit\_matrix are calculated, and the probability of the position is their product.

Secondly, for each subsequent row, calculate the probability of all states from the previous symbol, multiply the current corresponding position of A[i,i] ,B[i,j] and the further probability, and take the maximum value among all the values. Enter it to the current matrix. Fill the dictionary with the state corresponding to the maximum value and save it as a path.

Finally, after traversing the complete matrix, the maximum value of the last row and the corresponding path are taken out, and the final maximum probability is log(Pro \* trans\_matrix[state][LEN - 1])

Q2 : top\_k\_viterbi(State\_File, Symbol\_File, Query\_File, k)

Similar to Q1, first I created two 3D matrices to store probability values ​​and paths, respectively. For a TOP k query, my new matrix specification is, the number of row is #state, the number of columns is #query\_symbol, and each empty grid contains k empty lists.

First, for the symbol of the first column, the flag state is A[i,i], and the symbol index is B[i,j], and the values ​​of the respective positions trans\_matrix, emit\_matrix are calculated, and the probability of the Position is their product. Save this probability value to the first of the k lists in this grid. Save the current status serial number to the path list of the corresponding location.

Secondly, for each subsequent column, calculate the current position and the current symbol A[i,i], B[i,j] and the further probability. Extract Top k values ​​among all the obtained probabilities, respectively stored them in the k empty list of the current grid, and in the corresponding path matrix, the path of the value and the index of the current state are stored, which is the path of the current probability.

Finally, after traversing the entire matrix, a total of (Num\_state \* k) values ​​and paths are obtained in the last column in probability matrix and path matrix. Top k values ​​and their corresponding paths are taken among all the probabilities, output the value and path.

Q3 :

First, we find the smooth value is not well for the data training, and then we change the smooth value to reduce the wrong label. Second, the model have a problem in training the 0th label and 9th label, so we adjust the weight value of them.