王浚亦 r11922159

What I observation:

disambig 這個 function 跟我寫的 MyDisambig 比起來更有彈性，可以在輸入時就用參數決定要用 N 為多少的 Ngram 來進行預測，且執行速度比起我的程式來得更快，但除此之外我認為沒有其他不同之處，在我用 bigram 實作的 MyDisambig 中，預測出的結果跟其完全相同，僅有少數實作細節部分可能有差。

What I have done:

Mapping 的實作：

我是用 python 實作的，第一個遇到的問題是 encoding 的問題，因為注音的部分我是直接用打字建立一個 key map 的，所以整個 code 必須宣告成用 big5 解碼。關於注音的對應，我讀進 Big5-ZhuYin.map 後先將其用 split('/')分割各種破音字，然後只看第一個注音來建立 map。剩下的國字部分就直接建立一個一一對應的 map，然後將兩個 map 接起來之後直接寫入檔案。

MyDisambig 的實作：

大致上分為幾個步驟:

1. 讀入 LM 時，我會先將他分成一個 key 組成的 vector 跟其對應字所組成的 vector，例如 key[0]="ㄅ"，則 values[0]=[八, 匕, 卜,...]。
2. 接下來開始讀入要預測的檔案，一次處理一行，每次處理時我會先建立一個 vector 紀錄每個 input 字對應的所有可能字，例如 input 句子為"ㄈㄓ"，那我會建立一個 map_vector，其中 map_vector[0]=[凡, 丰, 分...], map_vector[1]=[丈, 中, 之,...]，最後我還會在句尾補上一個</s>。
3. 再來就是 Viterbi 的實作，我會動態建立兩個矩陣，一個用來記錄機率的 prob 跟一個用來記錄上一個字的 last_pos，實作時是按照 FAQ 中 2018 年一位同學整理的公式實作的，由於 LM 中給的是 log probability，所以機率是用加的而非乘的。
4. 做完 Viterbi 後，先從最後一排的對應字中找到機率最大的，然後用 last_pos 一個個 backtracking 回去到第一個字即可。做完之後就可以開始寫 result.txt。

(加分題) MyDisambig_trigram 的實作：

第一個遇到的問題是因為我在做 Viterbi 時需要用到三維矩陣的 prob 跟 last_pos，因此似乎有過大而造成 segmentation fault 的問題，後來我用動態宣告矩陣的方法解決。我在前兩個步驟處理資料的方法跟 MyDisambig 相同，因此在此不贅述，這邊從第三

點開始修改。

3. Viterbi 的實作，先定義下面三個矩陣:

map_vector	例如 input="ㄈ宅"，則 map_vector[0]=[凡, 丰, 分...], map_vector[1]=[宅]。
prob[k][I][j]	紀錄機率，其表示第 k 個字對應到 map_vector[k]中第 i 個字，第 k-1 個字對應到 map_vector[k-1]中第 j 個字。
last_pos[k][I][j]	紀錄最大機率上一個點，其表示在第 k-2 個字的所有可能對應字中，算出 prob[k][I][j] + 第 k-2 到第 k 個字的 trigram 機率，並紀錄最大的那個字。 例如現在第 k-2 到 k 個字是"ㄊ葫蘆"，假設算出來機率最大是"糖葫蘆"，則 last_pos 就會記錄"糖"在 map_vector[k-2]中的位置，並且將其機率記錄在 prob 中。

Viterbi 公式:

$$\delta_t(q_i, q_j) = \max_{qk} P(q_i|q_j, q_k)\delta_{t-1}(q_j, q_k)$$

這裡的初始值 prob[0][i][j] 我用的是 bigram，因為只有兩個字(包括<s>跟第一個字)，似乎是這個問題造成準確率有小誤差，但大概就錯 1~2 個字。

4. 接下來做 backtracking，首先選擇最後兩個字機率最大的 i, j，意即 (i*, j*) = argmax(prob[N-1][i][j])，這裡的 N 表示字串長度。加上倒數第三個字的位置，即 last_pos [N-1][i*][j*]，然後一個個往回推，最後將結果寫入 result.txt。

5. 如何執行我的 MyDisambig_trigram：

1) make
2) ./mydisambig_trigram $1 $2 $3 $4 $1 segemented file to be decoded $2 ZhuYin-Big5 mapping $3 language model $4 output file