數位語音處理概論作業三 資工碩二 曾品翔 R08922126

開發環境:
Intel Xeon E5-2620 v4
Ubuntu 20.04.1 LTS x86_64
在作業提供的 docker (ntudsp2020autumn/srilm)下開發

程式使用說明:
$ make

$ make map
or
$ python mapping.py

$ ./mydisambig <segemented file> <ZhuYin-Big5 mapping> <language model> <output file>

segemented file 和 language model 皆由投影片上說明產生之。
ZhuYin-Big5 mapping 由 python mapping.py 產生
Iconv -f big5 可以看 output 結果

what you have done:
基本上我只有寫 mapping.py 和 mydisambig.cpp 兩個檔案，我只有寫 bigram 的部分，並沒有實作 trigram。
mapping.py 基本上使用 python 很容易就做好了，使用一個 dict，並且是用 word:List[word]，最後輸出成 ZhuYin-Big5 mapping 就好，並不困難。
mydisambig.cpp，先把 segemented file 和 ZhuYin-Big5 mapping 和 language model 讀入，之後 segemented file 一行一行去執行 viterbi，然後逐行輸出到 output file，並沒有太困難。
比較麻煩的事情是因為讀入的格式是 big5，在 ubuntu 上面會覺得很麻煩，因為一開始看都是亂碼，然後用程式讀其實很方便，因為一個 big5 字元就是兩個 char 的大小，所以滿方便的，只要去掉空白就可以順利讀完。

what you observed (e.g., disambig vs. MyDisambig):
自己寫的 viterbi 跟 srilm 寫的 disambig 其實翻譯出來的東西還滿像的，我想應該是因為用同一份 language model 跟 map 所以出來的結果差不多，當然還是會有不一樣的地方。不過我其實比較不出來誰翻得比較好，因為不一樣的地方其實用人眼看都是錯的。那麼兩個方法翻出來一樣也不代表翻對，也有可能是一

起翻錯成同一個字。兩個方法的比較我沒有觀察到太多的心得。
另外就是有觀察到執行的速度，可能就是我用了太多的 C++ vector push_back，導致程式速度沒有 srilm 那麼快，所以我的程式都要等個一下下才會好。

自己在寫 mydisambig 的時候，其實有想到一件事情，就是最後一個字和</s>再去比機率，也就是最後一個字出現在最後的機率，原本以為這樣會比較正確，後來發現用這個方法之後會錯得更嚴重。