Digital Signal Processing Homework 3

Decode the ZhuYin-mixed sequence

b05901033 電機三 莊永松

Environment

```
OS: Ubuntu 16.04.5 LTS (GNU/Linux 4.15.0-36-generic x86_64)
Compiler: g++ (Ubuntu 5.4.0-6ubuntu1~16.04.10) 5.4.0 20160609
CPU: Intel(R) Xeon(R) CPU E5-1620 v3 @ 3.50GHz
RAM: 32G
---
Python 3.5.2 (default, Nov 12 2018, 13:43:14)
[GCC 5.4.0 20160609] on linux
```

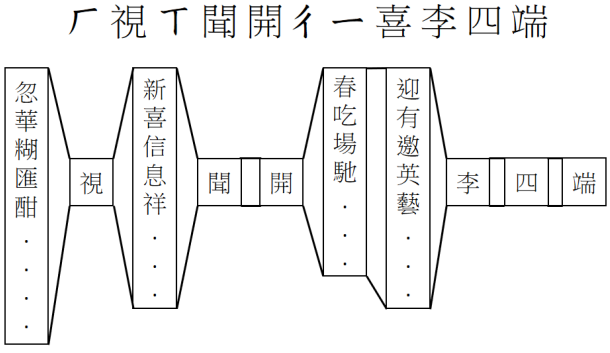
How to compile & execute?

- Use `make map` to generate map file `ZhuYin-Big5.map` before `make`.
- Use the command `make` to compile executable file `mydisambig`.
- Use the following command to execute the program on single file: `./mydisambig -text [Input file] -map [Map file] -lm [LM file] -order 2 > [Output file]`. (※Notice that my program does not support trigram LM. If you use `-order 3`, will make no difference. And the order of these arguments is not changeable.)
- Use `make run` to run over the testing file `1.txt` to `10.txt.`
- Use `make clean` to remove all compiled files.
- Specify the SRILM path and LM path if needed, as well as machine type: (eg. `make SRIPATH=/home/ta/srilm-1.5.10 LM=bigram.lm MACHINE_TYPE=i686-m64 all`)

What I have done?

Viterbi Algorithm

整個 Viterbi 的 search space 是由一句話中每個字的的可能選項所組成，如果該字已經是非注音的中文，那麼選項只有一個字；如果該字是注音符號，選項是他查詢 map 所對應到的所有可能字。



進行 Viterbi 時，先定義變數 $\delta_t(q_i) = \max_{W_{1:t-1}} P(W_1, \ldots, W_{t-1}, W_t = q_i)$，$q_i$指的是在$W_t$的位置上所有可能字的選項。

對第一個字的所有選項，我們計算$\delta_1(q_i) = Prob(W_1 = q_i)$，也就是直接從 SRILM 的 Model 中取得BigramProb(q_i, <s>)。

對於句子中間的字的所有選項，我們計算$\delta_t(q_i) = \max_{q_j} Prob(q_i|q_j)\delta_{t-1}(q_j)$，其中$Prob(q_i|q_j)$是由 SRILM 的 Model 中取得BigramProb(q_j, q_i)。

因為每個字擔任句尾最後一個字的機率不同，不能只看前面的字就算機率，也要考慮到他是句尾(也就是他的下一個字是</s>)，所以對於句尾的字的所有選項，我們計算
$\overline{W_T} = \underset{q_i}{argmax} P(W_1, \ldots, W_{T-1}, W_T = q_i, W_{T+1} = eos) = \underset{q_i}{argmax} \delta_T(q_i) Prob(eos|q_i)$，

其中$Prob(eos|q_i)$是由 SRILM 的 Model 中取得BigramProb(q_i, eos)。

據此，我們得到預測機率最高的句尾的字$\overline{W_T}$後，即可透過back-tracking，得到能走到$\overline{W_T}$機率最大的一條path。

效能比較

在同樣環境下，

使用 srilm 的 disambig 跑完10個測試檔的時間: 0m51.496s
使用自己寫的disambig 跑完10個測試檔的時間: 0m42.432s

自己寫的 code 效能上較 SRILM 更佳。