數位影音報告　　By　r01922164 資工所 李揚

Environment and how to execute it:

我有在工作站和 windows7 環境下跑過基本上都 OK

然後操作方法就跟助教規定的一模一樣:

./test modellist.txt testing_data.txt result.txt

./train iterations model_init.txt seq_model_01.txt model_01.txt

有一點值得特別注意的是，我的 test 有順便去算 testing_data1 的 accuracy，所以在這支程式裡會去跟 testing_answer.txt 去做比較然後 output 出 acc.txt。所以一定要放 testing_answer.txt。

我的 accuracy:

0.8696 in 2000 iterations' models

心得:

我是這一學期進入台大資工，所以這次我進台大的第一個程式作業，所以還算是蠻興奮的，也很高興最後能達到 accuracy 的要求。

我共花了十天做這個作業，前面三天架構，後面七天 debug。我覺得這次作業整體來說不會太難。

但有以下幾點需要注意:

1. Debug 時，如果是一些只有在 accuracy 看得出來的微小的錯誤，比較難以看出是錯在 test 還是 train，所以最好一開始寫好可以獨立測試這兩個部分的測資。
2. 這之作業裡有用到很多二維甚至三維的 index，要非常小心處理標號間的命名，以免發生錯亂。(我個人是 I for time, j for state, k other needs)
3. 測試時間會有點久，我寫了一支 autotest.c，可以在一支程式裡 train 五個 model，測驗時會比較方便。

程式架構:

我是用 c 語言去寫這支程式，考量蠻簡單的，因為 matlab 是付費軟體，而且用 C 可以藉此機會用一下系上的工作站。

程式大致上可分為三部分:1.讀檔.計算 3.寫檔

我的命名方式如下:

gamma: transition 分子　　sumga: sum of gamma(for all training sequence)

ep : transition 分母　　sumep: sum of ep(for all training sequence)

gammaOK: observation 分子　sumgaOK:sum of gammaOK(for all training sequence)

test 方面，則用 viterbi 陣列去計算。

註解部分是我的 debug 的測試

希望這部分方便助教讀我的 code。

我的一些發現:

1. forward, backward variable 的數值分析:
   我覺得這點會是我在實做之前很少考量到的一點，forward 和 backward

variable 在計算時會一列一列變小。
我覺得原因如下
Forward(t+1,j)
=sigma(all state i)(transition(I,j))*forward(t,i))observation(seq(t+1),j)
transition(i,j)，跟 observation(seq(t+1),j)都是對所有 state 累加會為一的值。
也就是說，但我們累加時卻只有累加 number of states 次，所以我每進入下一個時間的 forward variable，總值會變成(1/number of states)。
同理，backward variable 和 viterbi 都會有這樣的現象
這時候就要注意精準度問題了!不過這次作業只有六個 state 和 50 個 time frame，所以最後的值還在 double 可以計算的範圍之中。
(1/6)^50=1.2371931e-39
所以還 OK，但如果要做更大規模的實做時(state 和 time frame 增加)，數值上勢必要做一些特殊處理。

2. Some tests on the training set
   我把兩千次 train 出來的 model，餵進 seq_model_01, seq_model_02,…, seq_model_05 的 sequence。
   會得到以下的結果:
   model_01: 0.999300
   model_02: 0.999100
   model_03: 0.999600
   model_04: 0.673400
   model_05: 0.678600
   可以很明顯的發現，model_04、model_05 是這整套系統 accuarcy 的 bottle neck。
   而我去把 testing_data1 的答案一個一個 output 出來，發現大部分的錯誤都是發生在答案為 model_04.txt 和 model_05.txt。
   所以，我們如果要提高這個辨識系統的 accuracy 很明顯要從 model_04,model_05，這兩個 model 著手。然後我去看 model_04 和 model_05 的 learning curve(其實就是把各個 iteration train 出來的精準度拿出來看)，發現這兩個 model 也快要 converge 了。我可以判斷，model_04、和 model_05，有可能收斂到一個不好的 local maximum，所以如果要提高這個辨識系統的精準度，可以嘗試改變 model_04 和 model_05 的初始。
   不過我們隨便亂挑的一個 initialization 居然可以在五個 model 裡 fit 好三個 model 的 training set，表示這演算法其實有很高的機率收斂到 global maximum。