黃郁庭 資工所碩二 R05922038

1. Environment

   Operating system: Ubuntu 14.04 (x86_64 GNU/Linux)

   Compiler: gcc version 4.8.4 (Ubuntu 4.8.4-2ubuntu1~14.04.3)

2. How to execute

   a. 編譯 train、test：

   輸入指令"make all" 編譯 train.cpp 與 tes.cpp(使用壓縮檔裡的 Makefile)，或輸入"g++ -O3 –lm train.cpp –o train" 編譯 train.cpp 產生 train、"g++ -O3 –lm test.cpp –o test" 編譯 test.cpp 產生 test。編譯時，hmm.h 需與 Makefile、train.cpp 與 tes.cpp 放在同一個(層)資料夾裡。

   b. 執行 train 產生 model_01~05.tx：

   以產生 model_01.txt 為例，輸入" ./train #iteration model_init.txt seq_model_01.txt model_01.txt"執行程式，其中#iteration 為 iteration 數目。執行檔 train 需與 model_init.txt、seq_model_01~5.txt 放在同一個(層)資料夾裡。

   c. 執行 test 產生 result1~2.txt：

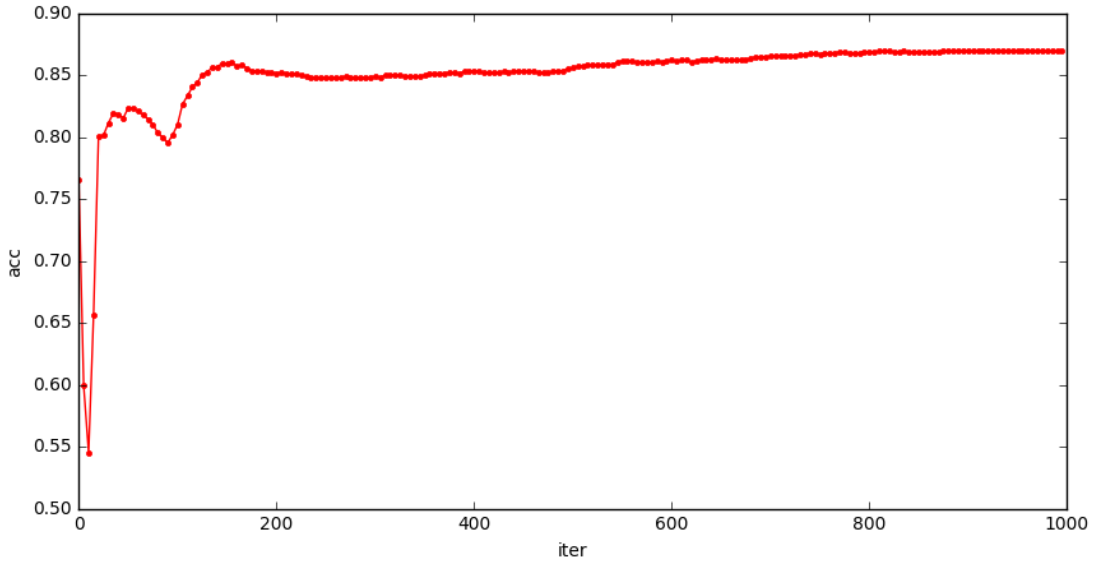
   以產生 result1.txt 為例，輸入" ./test modellist.txt testing_data1.txt result1.txt"執行程式。執行檔 test 需與 modellist.txt、testing_data1~2.txt 放在同一個(層)資料夾裡。

3. Results and analysis

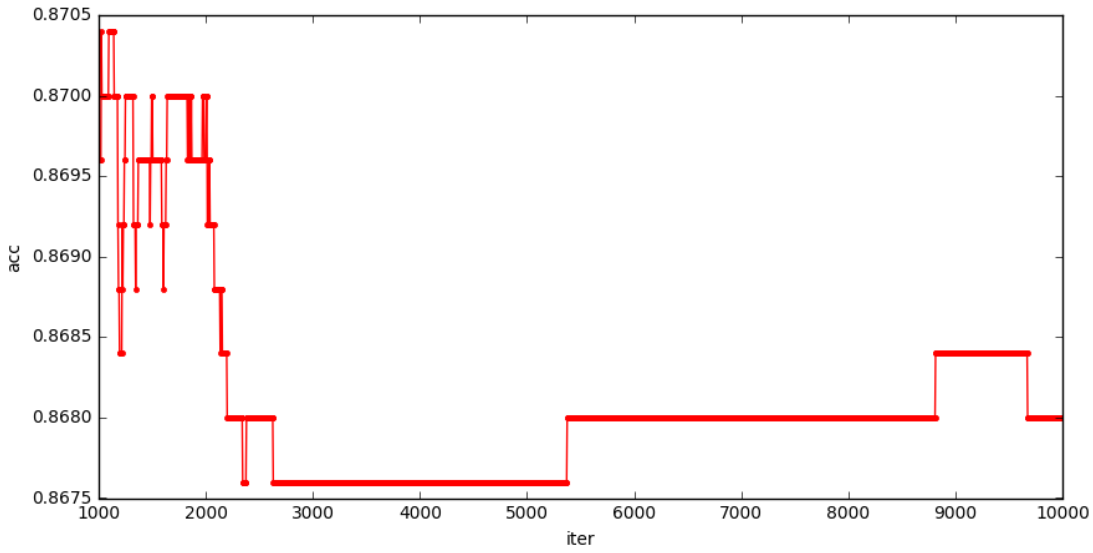
   我從第一個 iteration 開始，每隔五個 iteration 計算一次 accuracy，一直做到兩萬個 iteration，其中最好的 accuracy 為 0.8704(此時的 iteration 次數為 930)，如下方表格所示。

iter	1	11	21	101	201	501	930
Acc.(%)	76.6	54.52	80.08	81.04	85.2	85.64	87.04

在觀察 accuracy 的變化時，如下方圖表所示(橫軸是 iteration，縱軸是 accuracy)，我發現在第一個 iteration 做完後的 model 就可以有 0.766 的 accuracy，這說明了 model 的 initial 做得足夠好；但是之後的 iteration 的 accuracy 卻逐漸往下掉(我猜測可能是參數調到了某種地方會讓 model 需要先離開目前的地方才能往更好的地方走，例如在第 21 個 iteration 時可以達到 0.8008 比一開始的 0.766 還好，而離開當前 accuracy 較好的地方就很有可能讓 accuracy 暫時往下掉，或者是 overfitting 造成的)，在第十一個 iteration 做完的 model 的 accuracy 掉到 0.5452 後，accuracy 才開始往上升，然後大約在八、九百個 iteration 之後 accuracy 就沒有明顯的上升了，此時 model 應逐漸達到收斂狀態。



在第 1000 個 iteration 之後到第 10000 的 accuracy 如下圖所示，這段區間的 accuracy 都介於 0.8675 到 0.8705 之間，雖然 accuracy 整體趨勢略為下降，但是應該只是因為 iteration 次數做的不夠多所以 model 參數仍會有些微調整使 accuracy 略有變化。



在第 930 個 iteration 時 accuracy 可以達到 0.8704，在 2500 筆 test data 中有 324 筆 data 分類錯誤。而這 324 筆 data 中，有 142 個原屬於 model_04 的 data 被分錯，其中有 98 個被分到 model_05；另有 177 個原屬於 model_05 的 data 被分錯，其中有 142 個被分到 model_04。我發現 seq_model_04.txt 與 seq_model_05.txt 有 1115 個 observation 一模一樣，因此很有可能是這些 noise data 造成 model_04 與 model_05 不易區分這些分類錯誤的 data 使 accuracy 無法再上升。