HMM Training and Testing

Baseline

剛開始做這次作業時，遇到了一個困難：在 Ubuntu 64bit 下無法順利編譯 HTK。後來查了 Google 後發現要先修改 configure 檔案，刪除 -m32 參數，然後就能順利編譯。接下來連續執行助教提供的 scripts 就能得到 baseline 的結果了：

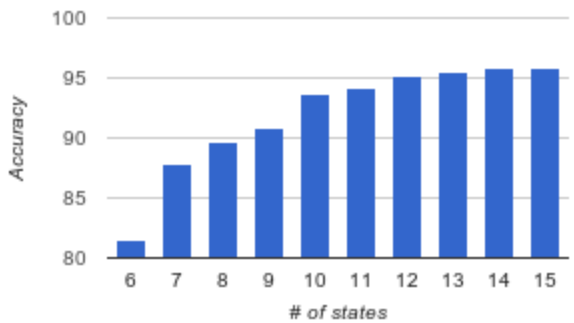
```
====================== HTK Results Analysis =======================
  Date: Fri Oct 25 16:59:25 2013
  Ref : labels/answer.mlf
  Rec : result/result.mlf
------------------------ Overall Results --------------------------
SENT: %Correct=38.54 [H=185, S=295, N=480]
WORD: %Corr=96.61, Acc=74.34 [H=1679, D=13, S=46, I=387, N=1738]
===================================================================
```

Increase the Number of States

接下來，我開始增加 state 數量，修改 lib/proto：

1. 將 <NumStates> 逐次加一。
2. 逐次增加一個新的 <State> 定義。增加的方法是直接複製前一個 <State> 的 <Mean> 和 <Variance> 並且將 state number 加一。
3. 同時修改 <TransP> 將數字加一，並將每一行的最後新增 0.0，並新增倒數第二行，新增的數字為全部 0.0，但最後兩個數字是 0.5。
   這麼做的效果是維持每個 2~n-1 的 state 都有一半機率停在原地或跳到下一個 state。

令人意外的發現是，accuracy 很快就超過了 95%，達到了作業的要求。但為了避免模型變得太長太奇怪，我決定在沒有明顯進步的 15 就停了下來。



增加 states 數後 accuracy 上升的趨勢

```
<NumStates> 15
<TransP> 15
0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.5 0.5
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
<EndHMM>
```

幾個 lib/proto 修改的部份

之所以可以提高準確度，應該是因為較多的 states 數可以較良好的 model 聲音的變化，也就是 states 數增加後可以增加 model 的能力。

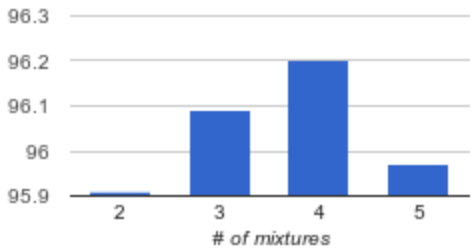
Increase the Number of Mixtures

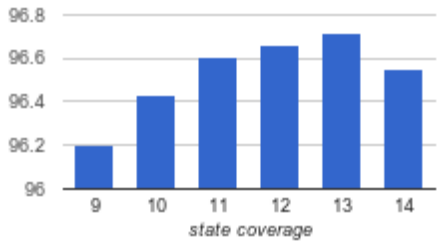
接下來，我調整了 lib/mix2_10.hed 中 sil 以外 mixtures 的數量，以及在數量為 4 情況下的涵蓋範圍。結果發現，剛開始增加 mixtures 數量時 accuracy 會上升，但後來就有下降情況。同樣的，增加涵蓋的範圍也會使 accuracy 上升，但最後也會有下降的情況，我最後選擇 MU 4 / [2-13].mix 作為最好結果。

之所以可以增加準確度，應該是因為多一點 mixture 可以 model 的 feature 分佈更多更精確。

```
MU 4 {liN.state[2-13].mix}
MU 4 {#i.state[2-13].mix}
MU 4 {#er.state[2-13].mix}
MU 4 {san.state[2-13].mix}
MU 4 {sy.state[2-13].mix}
MU 4 {#u.state[2-13].mix}
MU 4 {liou.state[2-13].mix}
MU 4 {qi.state[2-13].mix}
MU 4 {ba.state[2-13].mix}
MU 4 {jiou.state[2-13].mix}
MU 3 {sil.state[2-4].mix}
```

lib/mix2_10.hed 最後修改結果





Increase the Number of Iterations

接下來，我嘗試增加 iterations 的數量，但發現 accuracy 不進反退。這時想到作業 1 也遇到一開始下降的情形，假設行為相同的話，只要 iterations 數量夠高應該會穩定在可接受的高點才對，所以決定直接將 iterations 數字都一次增加到 200，看看 accuracy 會不會上升，結果果然上升了，成為 97.35。而之所以可以增加準確度，是因為 training 的演算法本來就是逐漸逼近逐漸優化，所以多跑幾次才能有好結果。

結論

```
====================== HTK Results Analysis =======================
  Date: Mon Oct 28 22:34:36 2013
  Ref : labels/answer.mlf
  Rec : result/result.mlf
------------------------ Overall Results --------------------------
SENT: %Correct=91.88 [H=441, S=39, N=480]
WORD: %Corr=97.47, Acc=97.35 [H=1694, D=33, S=11, I=2, N=1738]
===================================================================
```

由於這次的 training 過程較為複雜，不像之前變數較少，可以簡單的觀察不同變數的結果，所以其實無法完全確定要怎麼做才能達到最高的 accuracy。也因為這樣，過程中需要不少的嘗試。同時覺得光是增加 state 數量就能達到作業要求真是太好了，否則後來的調整其實也不是很確定能上升多少。

因為在考古題中出現了 HTK Result 的問題，所以我也做了一點調查。在結果中共有兩行，第一行 SENT 指的是有多少句子正確，第二行則是 word 的結果。而各英文字意義如下：

- Corr = Correct：正確辨認出的比率 = H/N
- Acc：準確度，除了正確辨認外，還要扣掉插入錯誤 = (H-I)/N
- H：正確辨認的數量，猜測應該是 Hit 的意思
- D：deletion error，刪除錯誤，即有一個樣本沒有被辨認出來，在結果中消失了
- S：substitution error，取代錯誤，即有一個樣本被辨認錯誤成別的結果
- I：insertion error，插入錯誤，即結果中出現不存在的樣本，也就是多了一個錯誤的物件
- N：總共的樣本數