黃于真 R06725028 資管碩二

[DSP-hw1 Report]

• environment

下圖為我所執行的linux環境，gcc版本為5.4

```
lily@Weal222C:~$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/5/lto-wrapper
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 5.4.0-6ubuntu1~16.04.10' --with-bugurl=fil
e:///usr/share/doc/gcc-5/README.Bugs --enable-languages=c,ada,c++,java,go,d,fortran,objc,obj-c++ --prefix
=/usr --program-suffix=-5 --enable-shared --enable-linker-build-id --libexecdir=/usr/lib --without-includ
ed-gettext --enable-threads=posix --libdir=/usr/lib --enable-nls --with-sysroot=/ --enable-clocale=gnu --
enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-default-libstdcxx-abi=new --enable-gnu-unique-o
bject --disable-vtable-verify --enable-libmpx --enable-plugin --with-system-zlib --disable-browser-plugin
 --enable-java-awt=gtk --enable-gtk-cairo --with-java-home=/usr/lib/jvm/java-1.5.0-gcj-5-amd64/jre --enab
le-java-home --with-jvm-root-dir=/usr/lib/jvm/java-1.5.0-gcj-5-amd64 --with-jvm-jar-dir=/usr/lib/jvm-expo
rts/java-1.5.0-gcj-5-amd64 --with-arch-directory=amd64 --with-ecj-jar=/usr/share/java/eclipse-ecj.jar --e
nable-objc-gc --enable-multiarch --disable-werror --with-arch-32=i686 --with-abi=m64 --with-multilib-list
=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-checking=release --build=x86_64-linux-gnu --
host=x86_64-linux-gnu --target=x86_64-linux-gnu
Thread model: posix
gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.10)
```

• execute

下圖為makefile，使用make run即可跑training和testing

Makefile

```
.PHONY: all clean

CFLAGS+=
LDFLAGS+=-lm     # link to math library

#TARGET=test

#all: $(TARGET)
# type make/make all to compile test_hmm
all:
    gcc train.cpp -o train
    gcc test.cpp -o test

run:
    ./train 1000 model_init.txt seq_model_01.txt model_01.txt
    ./train 1000 model_init.txt seq_model_02.txt model_02.txt
    ./train 1000 model_init.txt seq_model_03.txt model_03.txt
    ./train 1000 model_init.txt seq_model_04.txt model_04.txt
    ./train 1000 model_init.txt seq_model_05.txt model_05.txt
    ./test modellist.txt testing_data1.txt result1.txt
    ./test modellist.txt testing_data2.txt result2.txt

clean:
    rm ./train
    rm ./test
    #$(RM) $(TARGET)   # type make clean to remove the compiled file

```

• results

[Training] 在決定iteration 數目上，嘗試了300~1500，每次增加100，在1000時有最好結果，也就是我最後所上傳的model檔，而訓練時間大約在三分鐘內，執行畫面如下左圖。

Iter	Acc
800	0.868800
1000	0.869600

黃于真 R06725028 資管碩二

[Testing]
測試時間約在三秒內可以跑完，執行畫面如下右圖。

```
load data ok
iter:986
load data ok
iter:987
load data ok
iter:988
load data ok
iter:989
load data ok
iter:990
load data ok
iter:991
load data ok
iter:992
load data ok
iter:993
load data ok
iter:994
load data ok
iter:995
load data ok
iter:996
load data ok
iter:997
load data ok
iter:998
load data ok
iter:999
load data ok
```

```
test : 2476 ...
test : 2477 ...
test : 2478 ...
test : 2479 ...
test : 2480 ...
test : 2481 ...
test : 2482 ...
test : 2483 ...
test : 2484 ...
test : 2485 ...
test : 2486 ...
test : 2487 ...
test : 2488 ...
test : 2489 ...
test : 2490 ...
test : 2491 ...
test : 2492 ...
test : 2493 ...
test : 2494 ...
test : 2495 ...
test : 2496 ...
test : 2497 ...
test : 2498 ...
test : 2499 ...
```

[learning]
這次的作業實作，讓我對於hmm模型是如何運作更加熟悉，也更實際去理解到每個公式推導背後代表的物理意義，不過由於需要計算的變數很多，哪些需要持續累積，哪些需要每次歸零，都要小心處理，對於每個指標代表什麼屬性也都很容易出錯，使得我花很多時間debug，但也藉此釐清許多觀念，此外也讓人見識到hmm模型架構的厲害，以及語音辨識的複雜，像是最後測試算出來的機率，其實小到10^-40次方，可見每個模型的差異即使很小，但還是能正確預測。而在iteration數目部分，即使只有十次時，準確率也可以很快地達到五六成，但接下來要再繼續進步就要增加iteration數目到一千，不過實際看模型內的機率，其實也只有少數機率差零點零零零幾，但還是可以讓準確率增加，這也對應到其實差10^-40次方就可以影響預測結果。