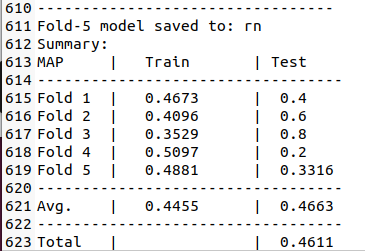
java -jar RankLib-2.18.jar -train 47\_SBFL/SBFL\_chart.txt -ranker 1 -kcv 5 -kcvmd 47\_SBFL\_result/chart/ -kcvmn rn -metric2t MAP -metric2T MAP -tvs 0.8 >> 47\_SBFL\_result/chart\_RN\_train.txt

這是已經有優化數據的資料用來訓練模型。

訓練結果如下，感覺跟沒有優化數據之前的差不多，不過目前還沒辦法確定，拿來應用在實際的資料上面看看。



java -jar RankLib-2.18.jar -load 47\_SBFL\_result/chart/f4.rn -rank normalize/SBFL\_closure.txt -indri 47\_SBFL\_result/chart\_f2.rn\_in\_closure\_result.txt

feature數量不同，所以先暫時擱置，把closure的資料也整理出來。

應用在closure上面。

java -jar RankLib-2.18.jar -load 47\_SBFL\_result/chart/f4.rn -rank 47\_SBFL/SBFL\_closure.txt -indri 47\_SBFL\_result/chart\_f2.rn\_in\_closure\_result.txt

無SBFL以及無patch的結果closure-1

大家的排名都一樣，符合結果。

而討論有在top找到，卻只有產生plausible的結果，closure-2

確實有比其他的還要來的高。

根據目前有的結果來看，加上了有在top-10找到，並且加上找到時使用的懷疑值，有能夠增加結果。

沒有top，而有正確patch的情況，closure-4。

因為僅有TBar生成正確patch，因此結果就是TBar的結果都是在前面的，其他都是沒有top也沒有patch，因此排名都次等且相同分數，合理。

僅有top，沒有產生任何的patch，closure-7。

因為主要是討論有沒有生成patch為主，因此沒有任何的patch生成，分數都一樣的話，那就沒什麼差別了。

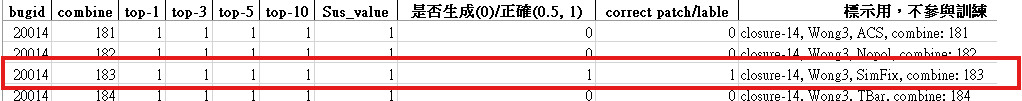
Closure-9也怪怪的。

Closure-14也跟原本得到一樣的結果。

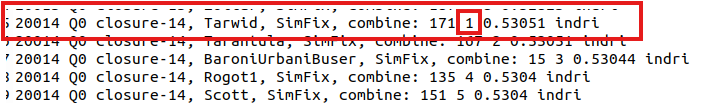
Closure-14的原始資料combine編號171是只有在top5找到。

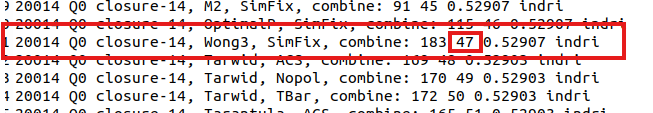


而有其他的結果是更好的，如combine編號183



但是套用模型的輸出結果，171的順位排在了第1個，而183的排在47，而且分數差別也很小。





目前想到的原因，可能是因為我的正確lable是以有沒有升成correct patch，來做表示的，如果有的bug-id都沒有正確的label，會不會就是因為這樣所以造成模型結果訓練不佳。

因此將其改成只保留有產生正確ptach的bug\_id資料來訓練，會不會比較合適?

上次提到的問題，就是感覺好像是把整個資料都拿來訓練的問題。

因為我的想法是所有的組合方法，即47個SBFL與4個APR的組合，共有188種組合，所以目的就是要找出這個bug\_id哪個組合是比較好的，就跟學長的LTR資料一樣的準備方式。