學號：B03901165 系級： 電機四 姓名：謝世暐

1. (1%)請比較有無 normalize(rating)的差別。並說明如何 normalize.
(collaborator: b03901161 楊耀程)

我使用的 normalize 方式是將 ratings 減掉平均值再除以標準差，實作上我做了 MF 和 NN 兩種 model，下表是兩者在有無 normalize 的 validation rmse loss 比較 (validation 為 10%的訓練資料，shuffle 方式固定，也就是 random seed 固定以便比較，latent dimension 選用每個 model 實驗中最好的 dimension):

	有 Normalize	無 Normalize
Matrix Factorization	0.86350	0.85266
DNN	0.84865	0.84783

有點意外的是 Normalize 在兩個 model 都沒有變好反而變差，上表的結果都是有經過 fine-tuned 的最終結果。最後在 kaggle 上最好的也是 DNN rating 沒有經過 normalize 的版本 (rmse:0.84578)。

2. (1%)比較不同的 latent dimension 的結果。
(collaborator: b03901161 楊耀程)

在兩種 model 上，我都比較三種不同 latent dimension 帶來的結果，分別為 32、64、128。以下結果一樣是 validation 的 rmse loss，且為 rating 沒有經過 normalize 的情況，和上面不同的是為節省時間，以下為沒有經過 fine-tuned 結果:

Dimension	32	64	128
Matrix Factorization	0.8545	0.85304	0.85429
DNN	0.8526	0.85811	0.86391

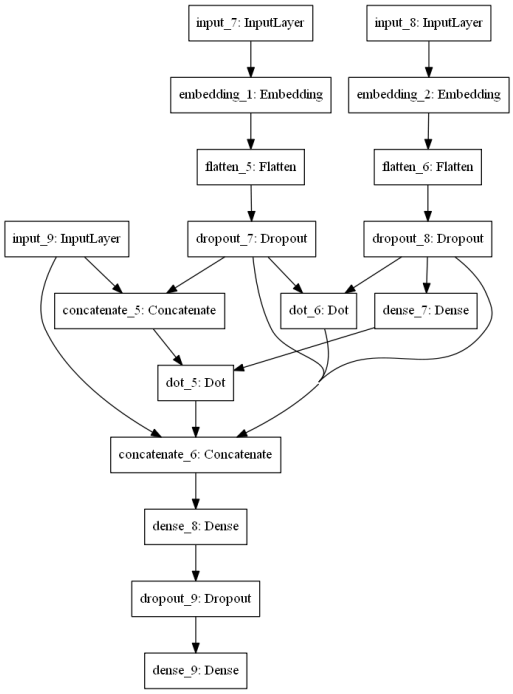
在我的實驗中，MF 最好的 dimension 是 64 維，DNN 最好的維度是 32 維，因此最後提交的 DNN 也是使用 32 維來做 embedding。

3. (1%)比較有無 bias 的結果。
(collaborator: b03901161 楊耀程)

由於 DNN 本身無法限制它會不會去學加 bias，這題我僅在 MF 上做有無 bias 比較。沒有 bias 的部分是將 movie 和 user 的 bias 一併去除，僅留下內積的部分，兩者其餘部分皆相同，皆用 64 維 embed，皆無 Normalize 和 fine-tuned。很意外的是，有 bias 的如上面的結果 loss 是 0.85304，沒有 bias 的 loss 最後是 0.85327，幾乎跟有 bias 差不多。原本以為 bias 的影響應該會很大，但實驗結果卻沒有想像中差異大。在實驗過程中，發現不加 bias 的 training loss 下降的速度相當慢，原因可能是 model 沒那麼複雜比較不容易 overfit，也因此在最後得到不錯的成績。

4. (1%)請試著用 DNN 來解決這個問題，並且說明實做的方法(方法不限)。並比較 MF 和 NN 的結果，討論結果的差異。
(collaborator: b03901161 楊耀程)

我使用的 DNN 方法如下，除了 train.csv 之外，我也使用了 users.csv 的資料作幫助，詳細如何使用留到 Bonus 解說，這邊主要解釋 model 架構。我的 DNN input 共有三個，第一個(input7)是 userID，第二個(input8)是 movieID，第三個(input_9)是 userID 對應的 feature(來自 users.csv)。假設 embed 過後的 userId 和 movieId 分別為 u 和 m，user feature 為 u_f，則整體架構就是將 u、m、u_f、u·m、$(\text{concat}(u, u_f))^T Am$全部 concatenate 起來，再經過兩層 Dense 得到 output，其中 A 是一個矩陣，其元素為訓練出來的(以 Dense 實現)(架構如下圖)。MF 和 NN 的 Optimizer 皆使用 Adamax，early stopping 的 patience 分別為 3 和 2，皆不用 normalize，出來的結果如第一題所示，DNN loss 約比 MF 好 0.005。



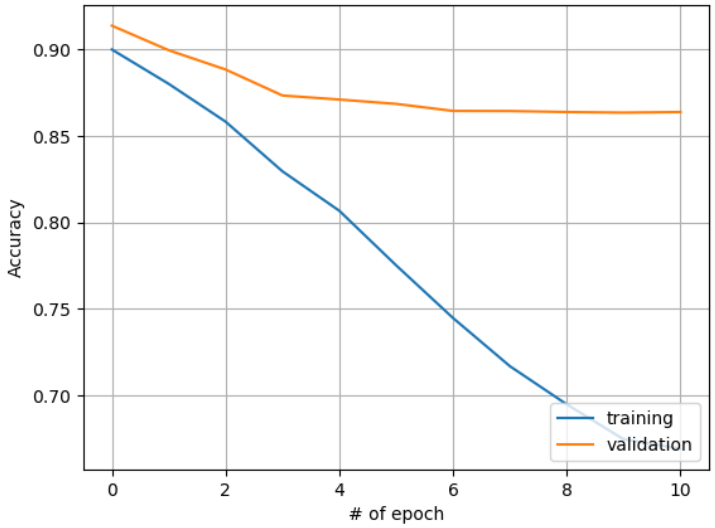


Fig.1. NN 的架構與訓練過程 (MF 置於下頁 Fig.3)

5. (1%)請試著將 movie 的 embedding 用 tsne 降維後，將 movie category 當作 label 來作圖。
(collaborator: b03901161 楊耀程)

如下圖所示，紅色代表 Children、Animation、Musical，米色代表 Film-Noir、War、Crime，藍色則是 Thriller、Horror。其餘類別的電影由於不知道要如何分就沒有包含於下圖。Movie 的 Embedding 是用 kaggle 上最好的 model (DNN) output 出來的。圖上可以觀察到，米色主要分布在左上角，藍色偏向左下角，而紅色則相對靠右上，如此可依稀分開三個類別。

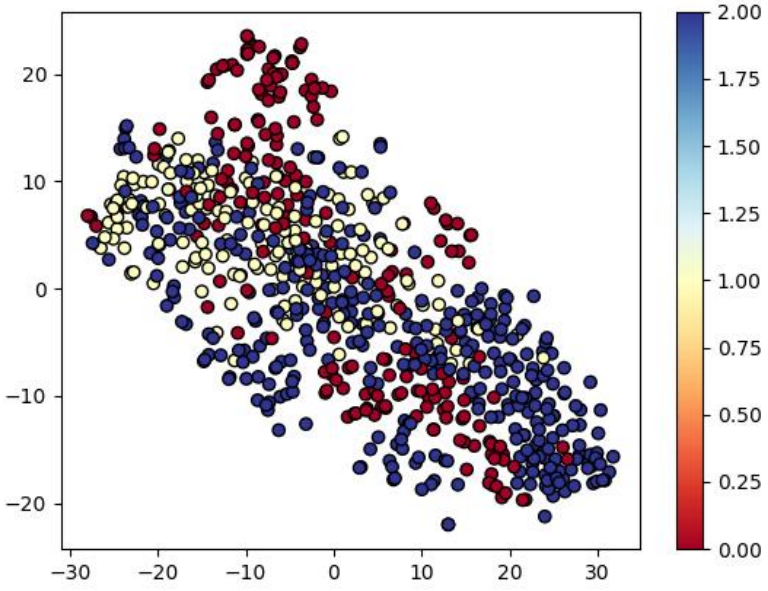


Fig.2. TSNE Result

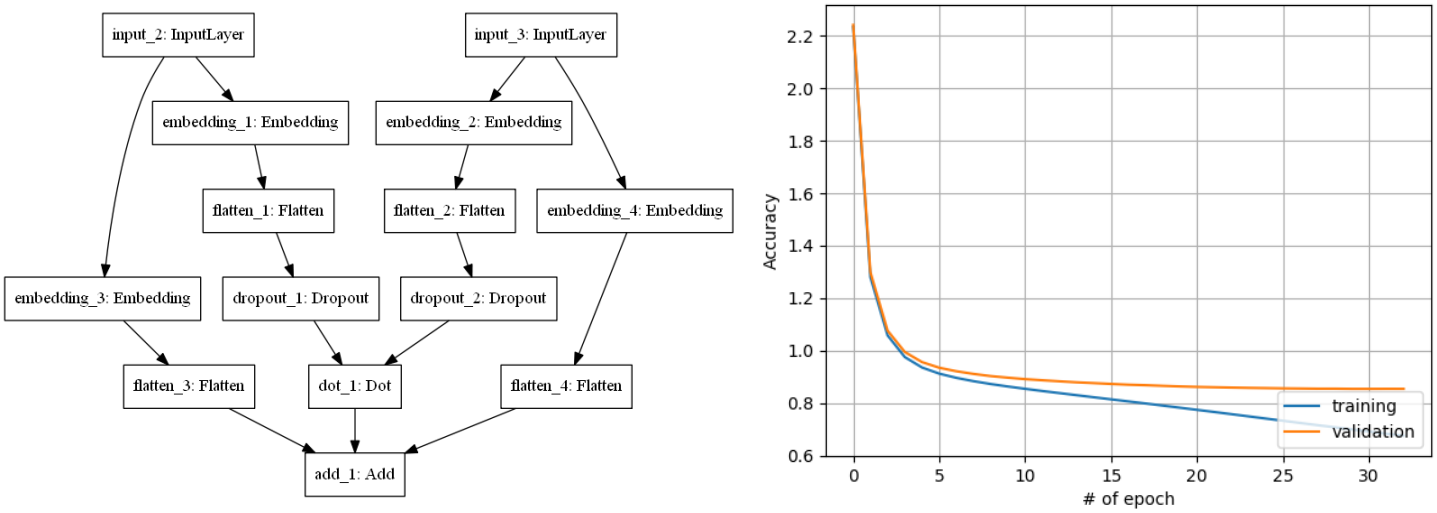


Fig.3. MF 的架構和訓練過程

6. (BONUS)(1%)試著使用除了 rating 以外的 feature, 並說明你的作法和結果，結果好壞不會影響評分。

   (collaborator: b03901161 楊耀程)

   我的 DNN 使用了 users.csv 做輔助(movies.csv 無使用，MF 則兩者皆無使用)，我將性別、年齡、職業、郵遞區號分別先 normalize (性別先轉為 0 和 1)，normalize 方式一樣是減掉平均再除以標準差，這些資料的維度最後為 (batch_size,4)。在 DNN 中，使用的地方在之前提到的$(\text{concat}(u, u_f))^T Am$中，同時本身也會直接和其他資料直接 concatenate 餵進 Dense 中。使用這些 feature 在最後也確實有讓我在 kaggle 上從 0.85x 進步到 0.84578。