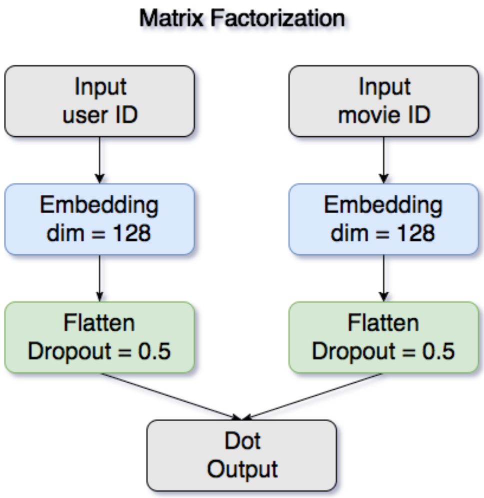
Machine Learning Project 6

Matrix Factorization

- b03902089 資工三 林良翰

Question 1. 3. 5. 使用的模型



Questions

1. 請比較對評分(Rating)有無標準化(Normalize)的差別, 並說明如何標準化.

- Embedding Features: user ID, movie ID.
- latent dimensions: 128
- Validation Ratio: 0.05

RMSE	Not Normalized	Normalized
Epochs	327	74
Training	1.0348	0.7808
Validation	0.8562	0.8492
Kaggle Public	0.8588	0.8519
Kaggle Private	0.8634	0.8561

- 標準化之後成果會比較好, 而且可以大幅減少訓練次數(Epochs).
- Method of Normalization
  - 算出評分R的平均值μ, 標準差σ
  - 把每個評分做標準化:

$$R_n = \frac{(R-\mu)}{\sigma}$$

  - 最後預測出來的結果R_{n-p}再反算回去原來的評分值R_p

$$R_p = R_{n-p} \times \sigma + \mu$$

2. 比較不同的潛在維度(Latent Dimension)的結果.

- Embedding Features: user ID, movie ID.
- Without Dropouts.
- Validation Ratio: 0.05

Latent Dimension	Epochs	Validation RMSE
2	370	0.8803
4	211	0.8733
8	146	0.8636
16	96	0.8589
32	66	0.8535
64	34	0.8528
128	25	0.8470
256	16	0.8465
512	13	0.8438
1024	10	0.8470

- Latent Dimension = 512 的時候結果最佳.

3. 比較有無偏見(Bias)的結果.

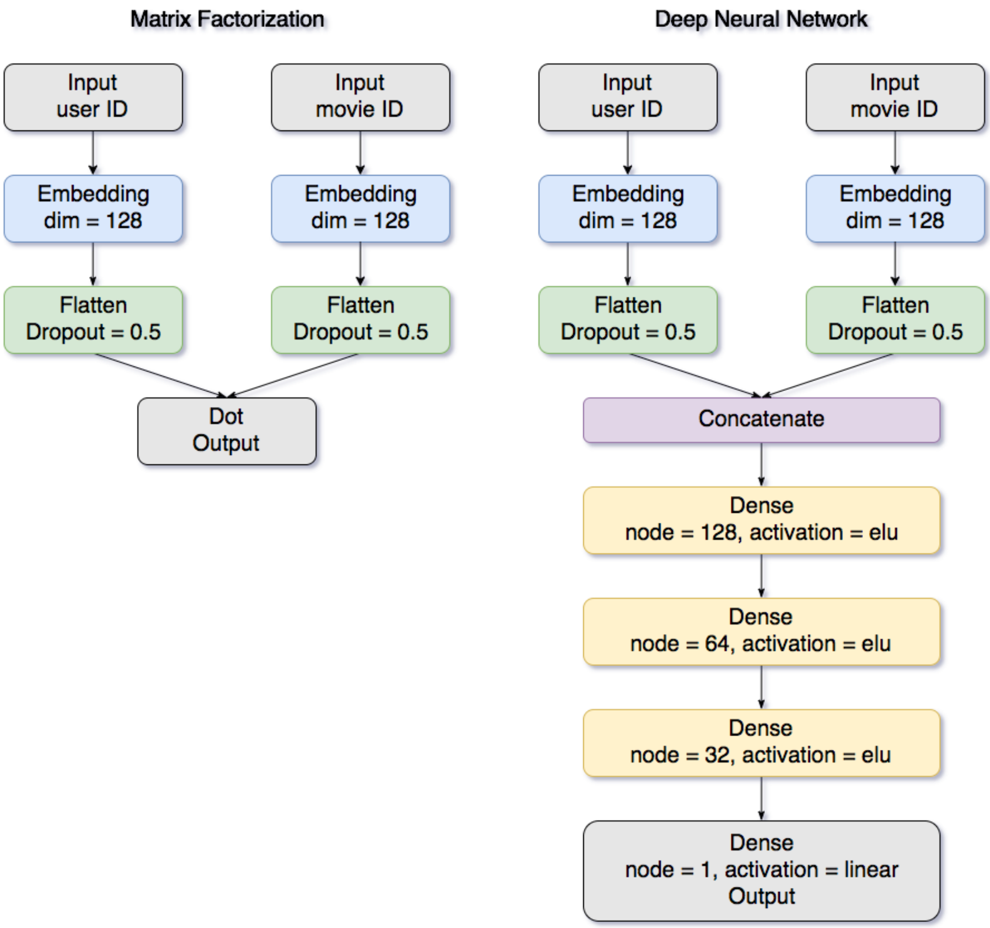
- Embedding Features: user ID, movie ID.
- latent dimensions: 128
- Validation Ratio: 0.05

RMSE	Without Bias	With Bias
Epochs	327	164
Training	1.0348	0.7687
Validation	0.8562	0.8434
Kaggle Public	0.8588	0.8460
Kaggle Private	0.8634	0.8503

- 加了偏見(Bias)之後成果會比較好, 而且可以大幅減少訓練次數(Epochs).

4. 請試著用DNN來解決這個問題, 並且說明實作的方法. 比較MF和DNN的結果, 討論結果的差異.

- MF, DNN模型架構



- 結果比較

RMSE	MF	DNN
Epochs	327	126
Training	1.0348	0.7816
Validation	0.8562	0.8547
Kaggle Public	0.8588	0.8574
Kaggle Private	0.8634	0.8608

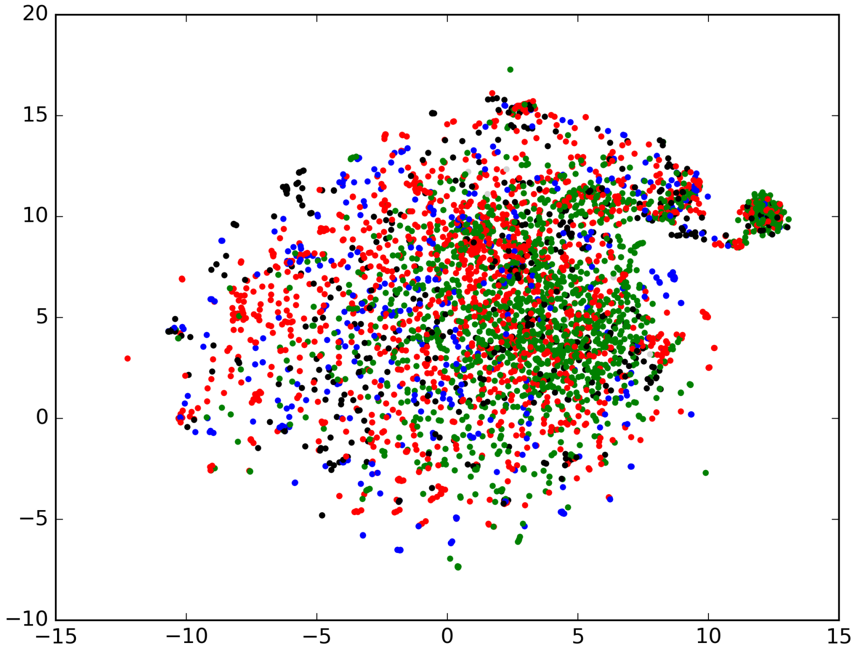
- DNN的結果會比較差一點, 而且DNN不能夠太多層, 否則容易over fitting.

5. 請試著將電影的潛在維度(Embedding)用TSNE降維後，將電影類別當作標記(Label)來作圖

- Usage

  `python3.5 plot_model.py _model_file_ --tsne`

- 使用TSNE降維結果

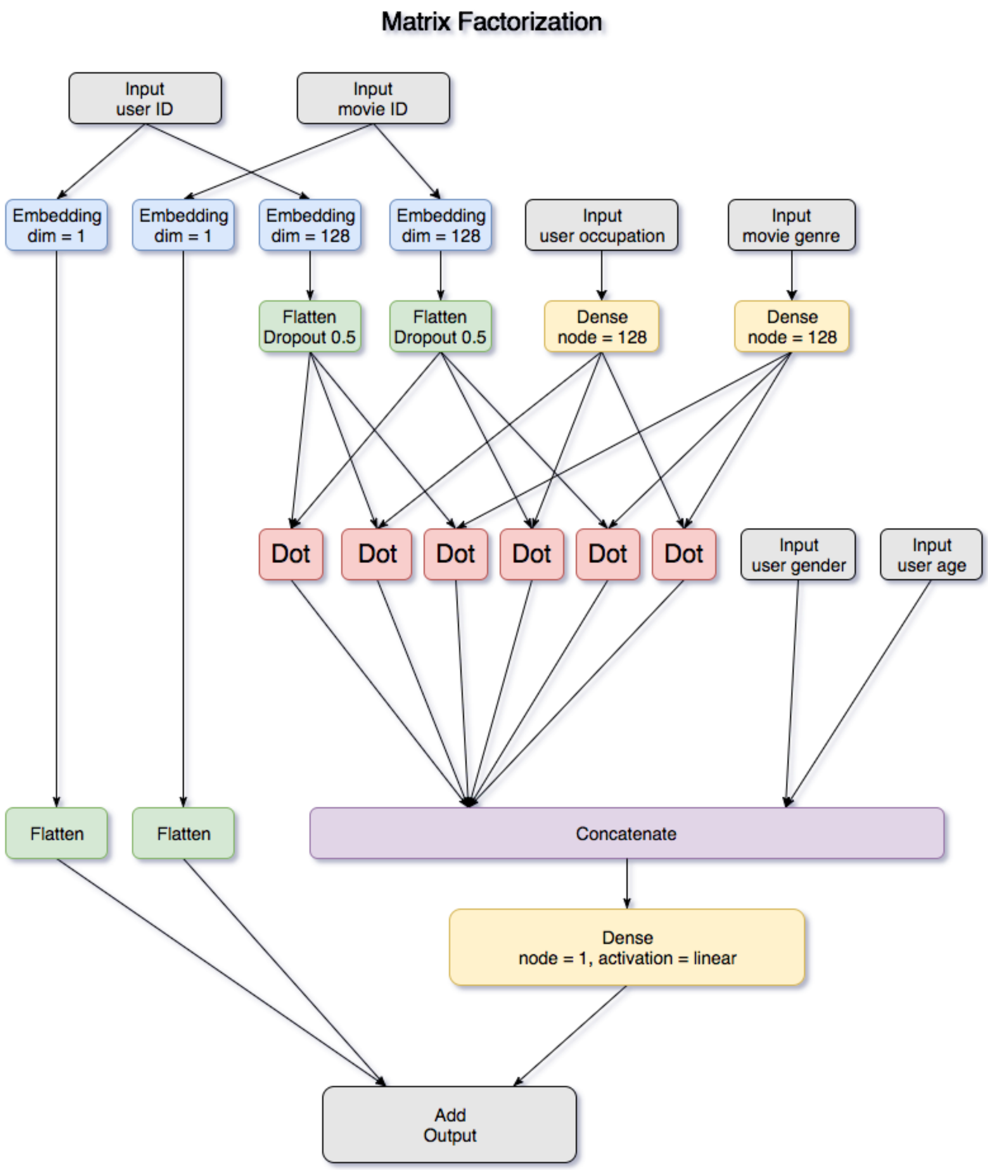


- Legends

Color	Genre
Red	Animation, Children’s, Comedy, Adventure
Green	Romance, Drama, Documentary, Musical
Blue	Fantasy, Action, Sci-Fi, War, Western
Black	Crime, Thriller, Horror, Film-Noir
Gray	Other

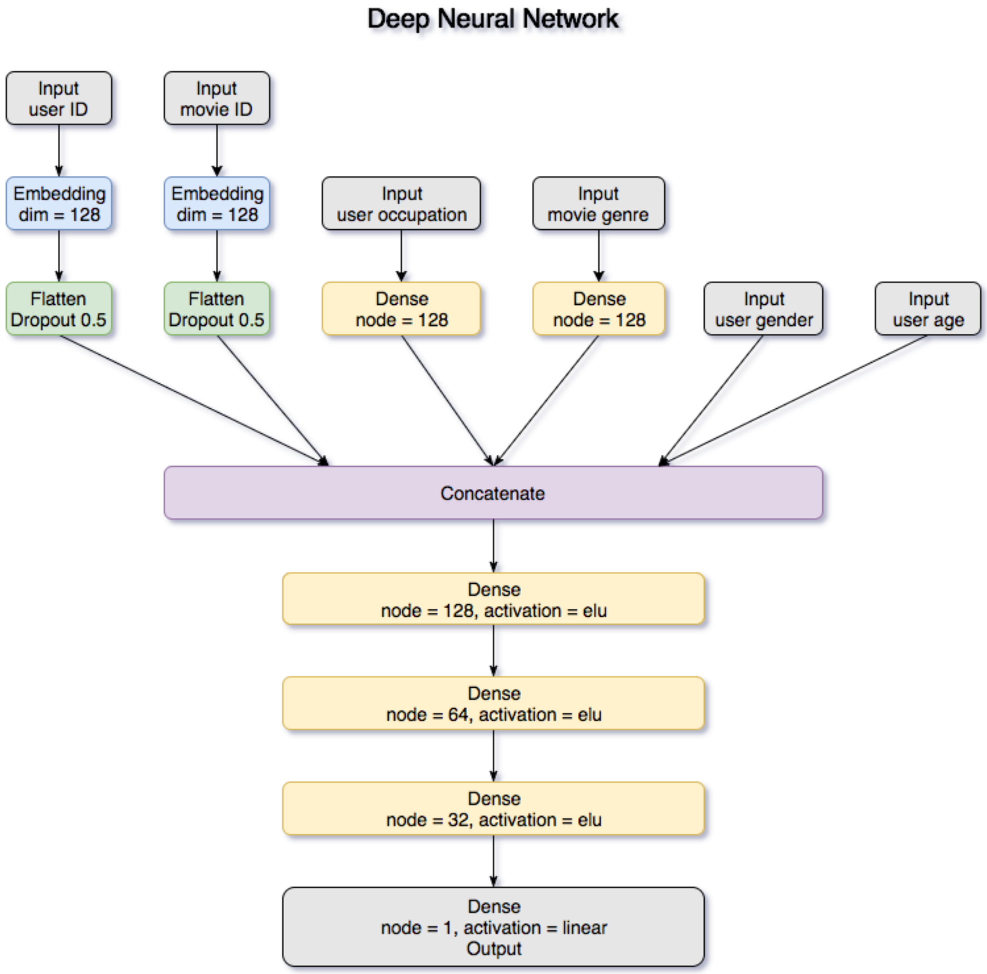
6. 試著使用除了評分以外的特徵(Feature), 並說明你的作法和結果, 結果好壞不會影響評分.

- 把user occupations和movie genres變成one-hot matrices.
    - User Occupations: dim = 21
    - Movie Genres: dim = 18
- 把age除上標準差.
- Latent Dimension: 128
- Validation Ratio: 0.05
- 使用複雜版本的MF模型



- Epochs: 114
- Training RMSE: 0.7319
- Validation RMSE: 0.8400
- Kaggle Public RMSE: 0.8428
- Kaggle Private RMSE: 0.8475

- 使用複雜版本的DNN模型



  - Epochs: 81
  - Training RMSE: 0.7879
  - Validation RMSE: 0.8472
  - Kaggle Public RMSE: 0.8493
  - Kaggle Private RMSE: 0.8551

Strong Baseline Model

- Usage
  `./hw6.sh` `data directory` `prediction file`
- Model
  - `mf_simple_0.844605.h5` : MF, latent dimension=512, bias
- Result (RMSE)
  - Kaggle Public Test: 0.8454
  - Kaggle Private Test: 0.8475

Best Model

- Usage
  `./hw6_best.sh` `data directory` `prediction file`
- Ensemble models
  - `mf_simple_0.842022.h5`
  - `mf_simple_0.844605.h5`
  - `mf_0.843316.h5`
  - `mf_0.839976.h5`
  - `mf_0.839977.h5`
  - `mf_0.841615.h5`
  - `mf_0.840972.h5`
  - `mf_0.841453.h5`
  - `mf_0.841394.h5`
- Result (RMSE)
  - Kaggle Public Test: 0.8320
  - Kaggle Private Test: 0.8365