Results

Written by: Itay Levinas
On the paper: Rianne van den Berg, Thomas N. Kipf, Max Welling, Graph Convolutional Matrix Completion (2017)
Results are on the new dataset of Book-Crossing

Datasets and results from the paper

The table below summarizes the given datasets and their properties:

Dataset	Users	Items	Features	Ratings	Density	Rating levels
Flixster	3,000	3,000	Users/Items	26,173	0.0029	0.5, 1, ..., 5
Douban	3,000	3,000	Users	136,891	0.0152	1, 2, ..., 5
YahooMusic	3,000	3,000	Items	5,335	0.0006	1, 2, ..., 100
MovieLens 100K (ML-100K)	943	1,682	Users/Items	100,000	0.0630	1, 2, ..., 5
MovieLens 1M (ML-1M)	6,040	3,706	—	1,000,209	0.0447	1, 2, ..., 5
MovieLens 10M (ML-10M)	69,878	10,677	—	10,000,054	0.0134	0.5, 1, ..., 5

Results are measured using RMSE values. For MovieLens 100K dataset, the option of side-information is tested and does improve the results by a little:

Model	ML-100K + Feat
MC [3]	0.973
IMC [11, 31]	1.653
GMC [12]	0.996
GRALS [25]	0.945
sRGCNN [22]	0.929
GC-MC (Ours)	0.910
GC-MC+Feat	**0.905**

Model	Flixster	Douban	YahooMusic
GRALS	1.313/1.245	0.833	38.0
sRGCNN	1.179/0.926	0.801	22.4
GC-MC	**0.941/0.917**	**0.734**	**20.5**

Note: Another result from the paper that I didn't check was a cold-start analysis.

The new dataset

Book-Crossing dataset is a dataset of book ratings, which contains 278,858 users (anonymized but with demographic information) providing 1,149,780 ratings (explicit / implicit) about 271,379 books.

From the whole raw dataset, I had to do two main things to get the final dataset on which I implemented the method:

- User and item features: Some books were removed because they didn't match the format I used (for example, a book name with symbols that weren't able to be unicoded). Some Users were mentioning ages such as 200, so I bounded the ages between 2 and 100.
  The final node features are age for users, author name and year of publication for books.

- Ratings and density: I removed the rating 0 (implicit rating), leaving only the ratings of 1-10. After that, I kept only the users that rated more than 0.00005 of the (valid) books. The final file has only those users and rated books.

To sum up, the data I used contains 3,565 users, 75,711 books and 132,021 ratings (density of approximately 0.0004).

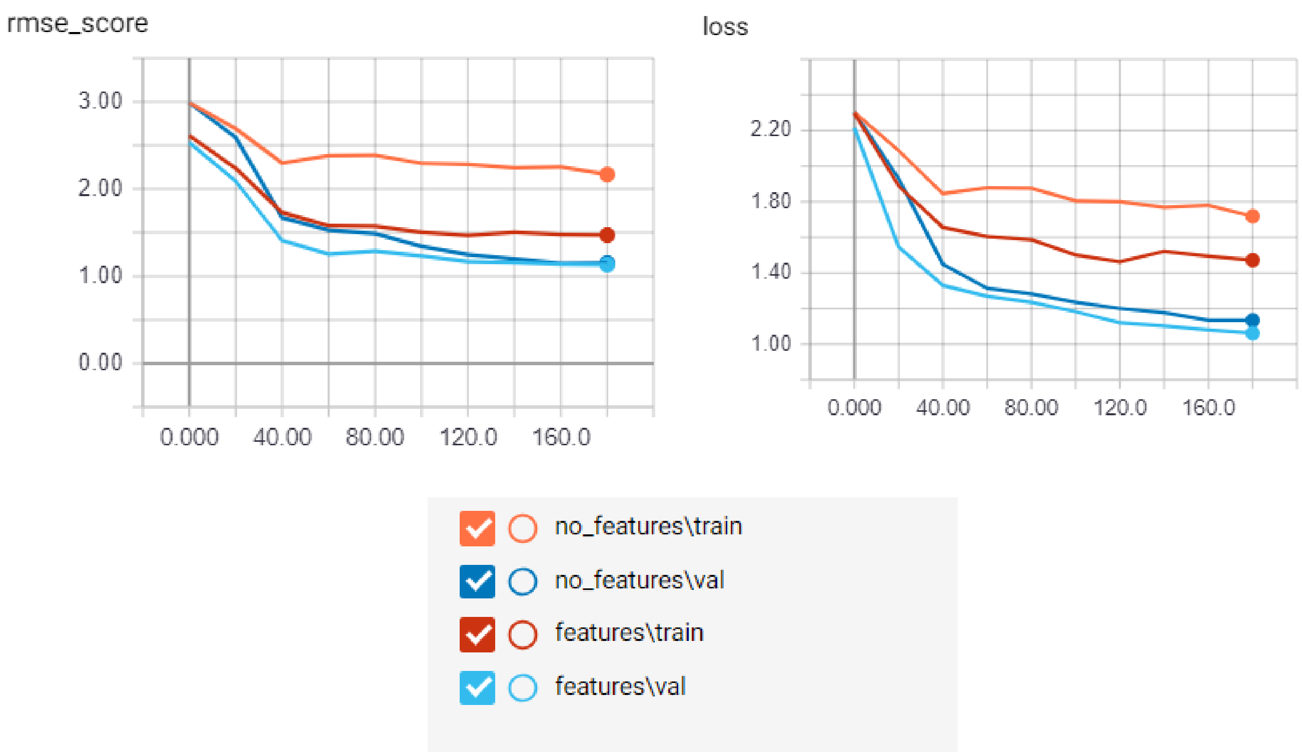
SETTINGS:

Accumulation – stack
Dropout 0.7
Epochs 200
Hidden features (if side-features are added) – 10
Hidden layers - [500, 75]
learning rate - 0.01
Normalization – left
Number of base functions - 2

Results

The method was implemented on the books' dataset in two versions: One with hidden features (age for users, author and year of publication for items) and one without them.

The data was split into 90% training and 10% test. Of the training set, 20% was taken as validation (but the model still used it for training). The results are presented below (cropped from the TensorFlow summary):



On the test set:

Without side information: test loss: 2.0581605, test RMSE: 2.3416162.

With side information: test loss: 1.7695448, test RMSE: 1.5815072.

Clearly, the using side-information improves the results. However, the training time takes much longer using it. Another observation is that the parameters I chose (not wisely but as a guess) cause an overfitting.

Since I haven't checked other models, I can't conclude anything about the performance of this model on the dataset. The only thing I can conclude is that this model can learn, since the loss decreases over the epochs.