Building a Book Recommender
Cécile Logé (ceciloge@stanford.edu) & Alexander Yoffe (ayoffe@stanford.edu) / CS 229 Fall 2019

**MOTIVATION & DATASET**

Goodreads is a social platform where users can discuss and rate books on a scale from 1 to 5. We want to build a Book Recommender and find an efficient way to predict book ratings.

**Dataset:**
- 8,000 books (8) snippet, genre, # of pages, year, authors, title, book cover
- 2m ratings from 15,000 users (I) (average of 140 ratings per user)
- 71% of the books in the dataset are behind 95% of the user ratings. We will define the Tail (T) as the other 29% (2298 books)

The dataset is split between train (65% + 15% for cross validation) and test (20%).

**EVALUATION METRICS**

Building a Book Recommender can be divided into three core goals each evaluated by a key metric:

1. Predict a user’s ratings on books they haven’t read yet (RMSE)
2. Surface a ranked list of top k books for each user (nDCC)
3. Help users discover relevant items (DivScore / all books but train set):

   \[
   \text{DivScore} = \sum_{u \in \text{test}} \sum_{i \in \text{all}} \frac{1}{M^2} \sum_{w \in \text{snippet}} \text{idf}_w \times \text{tf}_i
   \]

where \(w\) is the actual rating at our predicted rank \(i\), and \(n\) the actual rating at the actual rank \(i\).

**NEURAL NETWORKS**

Neural Networks were used to predict the average rating of a book based on the following input: Cover Image, Author, Title, Genre, Year published, # of pages

- **CNN:** Convolutional Neural Network for a model based on cover images
- **MLP:** Multi-Layer Perceptron used to handle 3 discrete (Categorical) variables and 2 continuous (Numerical) variables
- **Mixed Model:** concatenated NN outputs combining both the CNN and MLP

**RESULTS**

<table>
<thead>
<tr>
<th>Model</th>
<th>Set</th>
<th>RMSE</th>
<th>nDCC</th>
<th>nDCC Median</th>
<th>% nDCC + 1</th>
<th>Diverge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Popularity</td>
<td>Test</td>
<td>0.999</td>
<td>0.864</td>
<td>0.864</td>
<td>1.0%</td>
<td>0</td>
</tr>
<tr>
<td>MLP</td>
<td>Test</td>
<td>1.243</td>
<td>0.916</td>
<td>0.819</td>
<td>1.0%</td>
<td>0.259</td>
</tr>
<tr>
<td>TF-IDF</td>
<td>Test</td>
<td>0.589</td>
<td>0.998</td>
<td>0.999</td>
<td>0.2%</td>
<td>0.349</td>
</tr>
<tr>
<td>Matrix Fact.</td>
<td>Test</td>
<td>0.654</td>
<td>0.900</td>
<td>0.909</td>
<td>1.1%</td>
<td>0.319</td>
</tr>
</tbody>
</table>

**DISCUSSION & FUTURE WORK**

Matrix Factorization turned out to be the best model to predict individual ratings (RMSE) and obtain the most ideal ranking (highest nDGC with satisfying distribution among users). Div10 = 0.189 implies a more balanced model than the Popularity Baseline, but below the Tail proportion of our dataset (0.29).

**REFERENCES:**

[4] pymage2search
Link to our Poster Video

Complete link: https://www.dropbox.com/s/egydg2dsir3o956/Poster%20CS229%20Book%20Reco.mov?dl=0