Explainability, Robustness, and Fairness: in Recommender Systems as the Example

Min Zhang, Yiqun Liu, Shaoping Ma

Co-Contributors: Hongyu Lu, Shaoyun Shi, Chong Chen, Yongfeng Zhang

Department of Computer Science & Technology, Tsinghua University, China

z-m@tsinghua.edu.cn
Why this three factors?

- Rising requirements on more powerful AI system —— Explainability and Robustness
  - Know how, and know why
  - Systems work in non-ideal (bad) situations

- Fairness: a companion issue to explainability and Robustness

  Not trivial topics.

  Will take recommender systems as the example in this talk.
Explainability:

Know How, Know Why
For users

• Why did you show this result to me?
  • Recommend this product
  • Show this piece of news

For systems/engineers

• Why does my system give this output?
• Where do they (errors, bonus) come from?
• What factor(s) are the most important one(s)?
Challenges in Generating Explanations

- Factorization Models (e.g. LFM) have achieved significant success
  - The ability to recommend without clear content information
  - High rating prediction accuracy

Latent Factor Models [Koren et al. 2009]

\[
\begin{align*}
\text{User-Item Matrix} & \quad \approx \quad \text{Latent Space} \\
\text{Row} & \quad \text{Column} & \quad \text{Rating} & \quad \text{Rows} & \quad \text{Columns} & \quad \text{Product} \\
\text{User} & \quad \text{Item} & \quad \text{Vector} & \quad \text{Vector} & \quad \text{Matrix} & \quad \text{Rating}
\end{align*}
\]

\[
\hat{R}_{ij} = \langle u_i, v_j \rangle
\]
Challenges in Generating Explanations

- Factorization Models (e.g. LFM) have achieved significant success
  - The ability to recommend without clear content information
  - High rating prediction accuracy
- Factorization Models are hard to explain
  - The latent features make it difficult to explain the recommendation results to users

Can we find some solutions that are both highly accurate and easily explainable?
1. Aspect-level Explainable Recom. 

Users pay attention to different features 

Review Corpus → Sentiment Lexicon → Sentiment Analysis

Items perform well on different features

Battery, OS, Color, Memory, Earphone, Price, Screen, Service, Brand

Recommend


minimize
\[
\frac{\|PQ^T - A\|^2_F}{F} + \lambda_x \|U_1V^T - X\|^2_F + \lambda_y \|U_2V^T - Y\|^2_F
\]
\[
+ \lambda_u (\|U_1\|^2_F + \|U_2\|^2_F) + \lambda_h (\|H_1\|^2_F + \|H_2\|^2_F) + \lambda_v \|V\|^2_F
\]

\[P = [U_1 \ H_1], \ Q = [U_2 \ H_2]\]
Results: explanations do help

- Offline test on Dianping and Yelp

![Graphs showing RMSE for offline test on Dianping and Yelp datasets.]

- Online test on JD.com

<table>
<thead>
<tr>
<th>User Set</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Records</strong></td>
<td><strong>#Record</strong></td>
<td><strong>#Click</strong></td>
<td><strong>#Record</strong></td>
</tr>
<tr>
<td></td>
<td>15,933</td>
<td>691</td>
<td>11,483</td>
</tr>
<tr>
<td>CTR</td>
<td>4.34%</td>
<td></td>
<td>3.22%</td>
</tr>
</tbody>
</table>
2. Review-level Explainable Recommender Systems

Rating (to the item) - An Awesome Movie!

By Jokerz Wild on October 9, 2017
Format: Amazon Video | Verified Purchase
I love Iron Man!

Review (to the item) - Comic book characters... making millions of horrible movies these days.

By TylerVogt3329 on November 14, 2008
Format: DVD
You people these days consider this a good movie? Haha. Who in their right mind believes that a rich playboy can save the world from evil? For good and REAL action check out WWE, ECW, or TNA. For good classic Wrestling... check out WCW and WWF.

Rated usefulness (to the review) - Good solid film

By M-M on July 30, 2013
Format: Amazon Video | Verified Purchase
It turned out to be entertaining and at the end I enjoyed the film. Good special effects, nice story line for "actions" and "comics". The protagonist (Tony Stark) looks natural: arrogant, brash, but at the same time clever, intelligent and ethic. The villain is a little bit overreacting, and annoying, as most of antagonists :) Overall that's a good movie.

8 people found this helpful. Was this review helpful to you? [Yes] [No] Report abuse
Neural Attentional Regression with Reviews

Text processing

Users' Reviews' Attention

Item id embedding

Users' Reviews' Attention

Item id embedding

User Modeling

Item Modeling

Attention-based review pooling

Experimental Analysis: Data & Metric

Datasets:

<table>
<thead>
<tr>
<th></th>
<th>Toys_and_Games</th>
<th>Kindle_Store</th>
<th>Movies_and_TV</th>
<th>Yelp_2017</th>
</tr>
</thead>
<tbody>
<tr>
<td>users</td>
<td>19,412</td>
<td>68,223</td>
<td>123,960</td>
<td>199,445</td>
</tr>
<tr>
<td>items</td>
<td>11,924</td>
<td>61,935</td>
<td>50,052</td>
<td>119,441</td>
</tr>
<tr>
<td>ratings &amp; reviews</td>
<td>167,597</td>
<td>982,619</td>
<td>1,679,533</td>
<td>3.072.129</td>
</tr>
</tbody>
</table>

Evaluation Metric: RMSE

\[
RMSE = \sqrt{\frac{1}{N} \sum_{u,i} (\hat{R}_{u,i} - R_{u,i})^2}
\]

- CF-based Methods
  - PMF, NMF, SVD++
- LDA-based Method
  - HFT
- Deep learning Method
  - DeepCoNN

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>PMF</th>
<th>NMF</th>
<th>SVD++</th>
<th>HFT</th>
<th>DeepCoNN</th>
<th>NARRE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ratings</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Textual Reviews</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Deep Learning</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Review Usefulness</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>
# Model Comparisons

- Performance comparison on four datasets for all methods (RMSE)

<table>
<thead>
<tr>
<th>Method</th>
<th>Toys_and_Games</th>
<th>Kindle_Store</th>
<th>Movies_and_TV</th>
<th>Yelp-2017</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMF</td>
<td>1.3076</td>
<td>0.9914</td>
<td>1.2920</td>
<td>1.3340</td>
</tr>
<tr>
<td>NMF</td>
<td>1.0399</td>
<td>0.9023</td>
<td>1.1125</td>
<td>1.2916</td>
</tr>
<tr>
<td>SVD++</td>
<td>0.8860</td>
<td>0.7928</td>
<td>1.0447</td>
<td>1.1735</td>
</tr>
<tr>
<td>HFT</td>
<td>0.8925</td>
<td>0.7917</td>
<td>1.0291</td>
<td>1.1699</td>
</tr>
<tr>
<td>DeepCoNN</td>
<td>0.8890</td>
<td>0.7875</td>
<td>1.0128</td>
<td>1.1642</td>
</tr>
<tr>
<td>NARRE</td>
<td><strong>0.8769</strong></td>
<td><strong>0.7783</strong></td>
<td><strong>0.9965</strong></td>
<td><strong>1.1559</strong></td>
</tr>
</tbody>
</table>

![Bar charts for RMSE comparisons](chart_images)
Review usefulness (1): in Terms of User Rated

- Case study

<table>
<thead>
<tr>
<th>Item 1</th>
<th>(a_{ij} = 0.1932)</th>
<th>These brushes are great quality for children’s art work. They seem to last well and the bristles stay in place very well even with tough use.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(a_{ij} = 0.1932)</td>
<td>I bought it for my daughter as a gift.</td>
</tr>
<tr>
<td>Item 2</td>
<td>(a_{ij} = 0.2143)</td>
<td>From beginning to end this book is a joy to read. Full of mystery, mayhem, and a bit of magic for good measure. Perfect flow with excellent writing and editing.</td>
</tr>
<tr>
<td></td>
<td>(a_{ij} = 0.0319)</td>
<td>I like reading in my spare time, and I think this book is very suitable for me.</td>
</tr>
</tbody>
</table>

- Ground truth:
  - Top_rated_useful

<table>
<thead>
<tr>
<th></th>
<th>Toys_and_Games</th>
<th>Kindle_Store</th>
<th>Movies_and_TV</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Latest</td>
<td>Random</td>
<td>Length</td>
</tr>
<tr>
<td>Precision@1</td>
<td>0.1487</td>
<td>0.3255</td>
<td>0.2476</td>
</tr>
<tr>
<td>Recall@1</td>
<td>0.0362</td>
<td>0.0952</td>
<td>0.0771</td>
</tr>
<tr>
<td>Precision@10</td>
<td>0.1550</td>
<td>0.2000</td>
<td>0.2316</td>
</tr>
<tr>
<td>Recall@10</td>
<td>0.4367</td>
<td>0.5763</td>
<td>0.6763</td>
</tr>
</tbody>
</table>

Latest, Length: Bias / unfairness
Review usefulness (2)
Crowd-sourcing based Usefulness Labeling

Bias/unfairness in users’ votes:
The Matthew Effect

Ua = 1: not useful at all;
2: somewhat useful;
3: fairly useful;
4: very useful.

<table>
<thead>
<tr>
<th></th>
<th>Precision@1</th>
<th>Precision@5</th>
<th>Precision@10</th>
<th>Recall@1</th>
<th>Recall@5</th>
<th>Recall@10</th>
<th>NDCG@1</th>
<th>NDCG@5</th>
<th>NDCG@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top_Rated_Useful</td>
<td>0.4800</td>
<td>0.4440</td>
<td>0.3610</td>
<td>0.0821</td>
<td>0.3453</td>
<td>0.4953</td>
<td>0.6640</td>
<td>0.6906</td>
<td>0.7076</td>
</tr>
<tr>
<td>NARRE</td>
<td>0.5900**</td>
<td>0.4760**</td>
<td>0.3850**</td>
<td>0.1067**</td>
<td>0.3532**</td>
<td>0.5046**</td>
<td>0.7413**</td>
<td>0.7231**</td>
<td>0.7358**</td>
</tr>
</tbody>
</table>

In the Figure →
A: Ours;
B: Top_rated_useful;

- A is more useful than B
- B is more useful than A
- A and B are almost the same, both useful
- A and B are almost the same, both useless
• Discriminant way or Generalization way?
  • Both
• How to evaluate the effectiveness of explanations directly?
  • Might be a promising way: Personalized way with user behaviors/feedback.
  • Inter-disciplinary research from Computer Sci. and Psychologist
• Explanations vs. bias?
  • Or even Bias introduced by explanations?
How Robust Your System Could Be?
Robustness Issues

- Noise / Error
- Parameter sensitivity
- Unbalanced data
- Missing data
- Security

One of the missing data problems in recommender systems:

New users/items → Cold start problem
Traditional ways to handle cold-start problem

Traditional recommender algorithms:

- Content-Based (CB)
- Collaborative Filtering (CF): based on the feedback info.
  - Better performance generally, but invalid for cold-start

*Use global weights to model content and feedback information.*

*Cold-start CF vectors affect the performance.*
Model 1: **Result Level Attention**

- Attention determine where the results come from.
  - *explainability for the system/engineers*
- History information gets a small weight as long as user/item is cold.
- Cannot distinguish item cold and user cold.
Model 2: Vector Level Attention

- Attention determines which kind of information the user/item vector comes from.
- **The weights of CF and CB adaptively change for each user/item**
- Attentional Content & Collaborate Model (ACCM)

Cold-start: not shown before

In the training process, all users and items have been seen → *No cold samples in training.*

In test procedure, CF vectors of cold users/items are in the untrained status
- usually *random* at a distribution different from trained vectors.

Attention have never seen such a ‘stranger’ and may give wrong weights.
Solution 1: **Attention Set Strategy**

- Lay off a set of data from the training data.
- The training process alternate between two steps:
  - **model update** - update the main stream of the model, which is the black part.
  - **attention update** - Use the laying off “attention set” to train the attention network, which is the blue part.

**Historical feedback information in the laying off “Attention Set” is not used to train the id embedding.**
Solution 2: Cold Sampling Strategy

- A new and effective alternative strategy - “Cold Sampling”.
- Randomly sample some users and items to become cold in each batch.
- The model tends to use all information available.
- Let $c^u \in \{0, 1\}$ denote whether the feedback information of the user is shadowed. $u_{CF} \leftarrow (1 - c^u)u_{CF} + c^u v_r$ is a random vector.
Experiments: Dataset & Baselines

- **ml-100k**
  - **user**: age, gender, occupation; **movie**: year, 19 genres tag

- **WeiboDouban**
  - **user**: gender, 100 tags; **books**: star counts, 100 tags.

### Table 1: Statistics of Evaluation Datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Interaction#</th>
<th>User#</th>
<th>Item#</th>
<th>Sparsity</th>
</tr>
</thead>
<tbody>
<tr>
<td>ML-100k</td>
<td>100,000</td>
<td>943</td>
<td>1,682</td>
<td>93.70%</td>
</tr>
<tr>
<td>WeiboDouban</td>
<td>354,929</td>
<td>5,796</td>
<td>14,468</td>
<td>99.58%</td>
</tr>
</tbody>
</table>

- UserKNN
- ItemKNN
- BiasedMF
- SVD++
- UserItemCB
- NCF
- NFM
- AFM
- Wide&Deep
# Performance

<table>
<thead>
<tr>
<th></th>
<th>ML-100k</th>
<th>WeiboDouban</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Random</td>
<td>Cold¹</td>
</tr>
<tr>
<td>UserKNN</td>
<td>0.9376</td>
<td>1.0498</td>
</tr>
<tr>
<td>ItemKNN</td>
<td>0.9237</td>
<td>1.0431</td>
</tr>
<tr>
<td>BiasedMF</td>
<td>0.9375</td>
<td>1.0186</td>
</tr>
<tr>
<td>SVD++</td>
<td>0.9220</td>
<td>1.0151</td>
</tr>
<tr>
<td>UserItemCB</td>
<td>0.9370</td>
<td>0.9931</td>
</tr>
<tr>
<td>NCF</td>
<td>0.9450</td>
<td>1.1696</td>
</tr>
<tr>
<td>NFM</td>
<td>0.9143</td>
<td>0.9974</td>
</tr>
<tr>
<td>AFM²</td>
<td>0.9274</td>
<td>0.9934</td>
</tr>
<tr>
<td>Wide&amp;Deep</td>
<td>0.9099</td>
<td>0.9966</td>
</tr>
<tr>
<td>RLWS</td>
<td>0.9133</td>
<td>1.0567</td>
</tr>
<tr>
<td>RLAM (Regular Train)</td>
<td>0.9132</td>
<td>1.1574</td>
</tr>
<tr>
<td>RLAM (Attention Set)</td>
<td>0.9123</td>
<td>1.0150</td>
</tr>
<tr>
<td>RLAM (Cold Sampling)</td>
<td>0.9134</td>
<td>0.9951</td>
</tr>
<tr>
<td>VLWS</td>
<td>0.9119</td>
<td>1.0608</td>
</tr>
<tr>
<td>ACCM (Regular Train)</td>
<td>0.9006³</td>
<td>1.0176</td>
</tr>
<tr>
<td>ACCM (Attention Set)</td>
<td>0.9012³</td>
<td>0.9906</td>
</tr>
<tr>
<td>ACCM (Cold Sampling)</td>
<td>0.9027³</td>
<td>0.9776*</td>
</tr>
</tbody>
</table>

- **1.** Randomly item 30% cold and user 30% cold
- **2.** Experiments of AFM on WeiboDouban cannot finish in acceptable time and computing resources
- **3.** The bold values in the same column are not significantly different among themselves (p > 0.01)
- **•** Significantly better than the best baseline (italic ones) with p < 0.01
Different Cold Scenarios

- Note that the RMSE of BiasedMF keeps growing up to a high value in some scenarios.

- To have a clear comparison between other methods, lines of BiasedMF is not drawn entirely in those diagrams.
Attention Visualization

- Samples are from the test set of ML-100k in which randomly 30% of samples are item cold, and 30% of samples are user cold.

Also helps on explainability.
A lot more on robustness

• Robustness is one of the most difficulty issues
  • We/Systems are always working in a non-ideal noisy environment with varies types of bad cases
  • “Happy families are all alike; every unhappy family is unhappy in its own way.” —— Lyev Tolstoy

• Robustness is always connected with explainability.
Fairness Issue Does Matter
“A person’s experience with an information system should irrelevance depend on their personal characteristics (Michael et al. 2018)"
# Does unfairness exist in RS

## For users
- **Model performance** is not consistent across users (Michael et al., 2018)
  - Men receive better recommendations
    - MovieLens 1M, LastFM 1K
  - Old (50+) and young (under 18) receive better recommendations
    - LastFM 360K data

## For items
- Skewed distribution of the item exposure (Leonhardt et al., 2018)
  - Most of items are infrequently recommended or not at all
### How can we measure fairness

#### Difference in treatment of groups

<table>
<thead>
<tr>
<th>User Groups</th>
<th>Male vs. Female, Old vs. Young, ..., Different items</th>
</tr>
</thead>
<tbody>
<tr>
<td>Treatments</td>
<td>Recommendation performance (Average predicted ratings, Estimation errors, etc.) Item Exposure</td>
</tr>
<tr>
<td>Metrics for Difference</td>
<td>Absolute difference in means Kolmogorov-Smirnov statistic Gini coefficient</td>
</tr>
</tbody>
</table>
What causes unfairness: Diversity?

• Diversity?
  • Improve diversity $\rightarrow$ improve the fairness for items
  $\rightarrow$ but decrease fairness for users. (Leonhardt et al. 2018)

Observations:
  High-rated items have been recommended to potential buyers, say the rich users (utility)
  Remaining Low-rated items are recommended to poor users (to improve the overall item diversity)
What causes unfairness: Popularity?

Traditional offline evaluation methods prefer popular items
- Lead Recommender systems to only recommend popular items (unfairness)
- However popularity performs poor in true precision.

(Rocío Cañamares and Pablo Castells, 2018)

![Graph showing nDCG@10 for MovieLens 1M and CM100k datasets.](image)

- Popularity almost optimal
- Avg rating < popularity
- Popularity ~ random
- Avg rating > popularity
What causes unfairness: Quality?

Quality?  Low-quality news, like clickbait, receives more clicks

• Click signal ≠ user actual preference. (Lu et al., 2018)
• Reward systems that recommend more low-quality news (unfairness).

![Graph showing click probability vs. top-K positions]

Figure 5: Click Probability of the news in top-k positions conditioned by the news quality. The low-quality news attracts more clicks.

More than half clicked news is disliked by the user

I. Incorporate fairness into recommendation algorithms

- **Re-ranking**
  - Greedy to choose items to keep the distribution
    (Chen Karako and Putra Manggala, 2018)

- **Tensor-based recommend** (Zhu et al. 2018)
  - Isolate sensitive attributes

- **Optimization function** (Lin et al. 2017)
Performances Comparisons on MoviePilot with Borda Relevance, K = 10

<table>
<thead>
<tr>
<th>Metrics</th>
<th>LM Ranking</th>
<th>Ave Ranking</th>
<th>SPGreedy</th>
<th>EFGreedy</th>
<th>Greedy-LM</th>
<th>Greedy-Var</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prec@K</td>
<td>0.0385</td>
<td>0.0467</td>
<td>0.0003</td>
<td>0.0013</td>
<td><strong>0.0484</strong></td>
<td><strong>0.0488</strong></td>
</tr>
<tr>
<td>Rec@K</td>
<td>0.0762</td>
<td>0.0910</td>
<td>0.0003</td>
<td>0.0020</td>
<td><strong>0.0942</strong></td>
<td><strong>0.0945</strong></td>
</tr>
<tr>
<td>F@K</td>
<td>0.0512</td>
<td>0.0617</td>
<td>0.0003</td>
<td>0.0016</td>
<td><strong>0.0639</strong></td>
<td><strong>0.0644</strong></td>
</tr>
<tr>
<td>NDCG@K</td>
<td>0.2376</td>
<td>0.2450</td>
<td>0.0008</td>
<td>0.0077</td>
<td><strong>0.2507</strong></td>
<td><strong>0.2502</strong></td>
</tr>
</tbody>
</table>

Table 1. Performances of GreedyAlg-Var under different λ on MovieLens with Borda Relevance, |G| = 8, K = 1

<table>
<thead>
<tr>
<th>λ, RG</th>
<th>0</th>
<th>0:1</th>
<th>0:2</th>
<th>0:3</th>
<th>0:4</th>
<th>0:5</th>
<th>0:6</th>
<th>0:7</th>
<th>0:8</th>
<th>0:9</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>NDCG@K</td>
<td>0.0335</td>
<td>0.1577</td>
<td>0.1691</td>
<td>0.1906</td>
<td>0.2003</td>
<td>0.2031</td>
<td>0.2072</td>
<td>0.2091</td>
<td><strong>0.2079</strong></td>
<td><strong>0.2108</strong></td>
<td>0.2079</td>
</tr>
<tr>
<td>λ, DG</td>
<td>0</td>
<td>0:1</td>
<td>0:2</td>
<td>0:3</td>
<td>0:4</td>
<td>0:5</td>
<td>0:6</td>
<td>0:7</td>
<td><strong>0:8</strong></td>
<td><strong>0:9</strong></td>
<td>1</td>
</tr>
<tr>
<td>F@K</td>
<td>0.0292</td>
<td>0.0656</td>
<td>0.0714</td>
<td>0.0848</td>
<td>0.0925</td>
<td>0.0950</td>
<td>0.0959</td>
<td>0.0978</td>
<td><strong>0.0990</strong></td>
<td><strong>0.0983</strong></td>
<td>0.0960</td>
</tr>
<tr>
<td>NDCG@K</td>
<td>0.0767</td>
<td>0.1715</td>
<td>0.1839</td>
<td>0.2076</td>
<td>0.2185</td>
<td>0.2246</td>
<td>0.2218</td>
<td>0.2237</td>
<td><strong>0.2251</strong></td>
<td><strong>0.2248</strong></td>
<td>0.2228</td>
</tr>
</tbody>
</table>
How can we handle it (cont.)

II. Improve evaluation metrics (Lu et al. 2018)

CTR = \frac{\sum Clicked(i)}{\#imp} \frac{\sum Gain(i)}{\#imp} \frac{\sum Pref(i)}{\#imp} \frac{\sum Predicted\_user\_pref(i)}{\#imp}

1. Predict user actual preference
2. Replace click signals

Better correlated with user satisfaction

Summary on Fairness

• Unfairness widely exists in recommender systems.
• We can measure fairness as the difference of treatment for groups.
• There are many reasons: from data, algorithms, and evaluation methods.
• To handle it:
  • We can improve recommendation algorithm.
  • We can improve evaluation methods.
  • Bridge the click signal to actual preference, bridge the online metrics to satisfaction.
Overview

For User/Sys.
Methodology
Evaluation
Bias

Problem def.
Data
Algorithm
Evaluation

Diversity
Popularity
Quality
......

Noise/error
Sensitivity
Security
Unbalance
......

explain ability

robustness

Fairness

......
Contact: Min Zhang  z-m@tsinghua.edu.cn