Recommender Systems: Practical Aspects, Case Studies

Radek Pelánek

2018
This Lecture

- “practical aspects”: attacks, context, shared accounts, ...
- case studies, illustrations of application
- illustration of different evaluation approaches
- specific requirements for particular domains

focus on “ideas”, quick discussion (consult cited papers for technical details)
Focus on Ideas

even simple implementation often brings most of the advantage
potential inspiration for projects, for example:

- taking context into account
- highlighting specific aspects of each domain
- specific techniques used in case studies
- analysis of data, visualizations
- evaluation
Attacks on Recommender System

- Why?
- What type of recommender systems?
- How?
- Countermeasures?
Attacks

susceptible to attacks: collaborative filtering

reasons for attack:

- make the system worse (unusable)
- influence rating (recommendations) of a particular item
  - push attacks – improve rating of “my” items
  - nuke attacks – decrease rating of “opponent’s” items
Fig. 2  Simplified system database showing authentic user profiles and a number of attack profiles inserted. In this example, user $h$ is seeking a prediction for item 7, which is the subject of a product nuke attack.
Types of Attacks

more knowledge about system → more efficient attack

random attack generate profiles with random values  
(preferably with some typical ratings)
average attack effective attack on memory-based systems  
(average ratings → many neighbors)
bandwagon attack high rating for “blockbusters”, random  
values for others
segment attack insert ratings only for items from specific  
segment
special nuke attacks love/hate attack, reverse bandwagon
Fig. 3 Prediction shift (left) and hit ratio (right) for product push attacks mounted against the user-based collaborative recommendation algorithm. Hit ratio results relate to a 10% attack size.
Countermeasures

- more robust techniques: model based techniques (latent factors), additional information
- increasing injection costs: Captcha, limited number of accounts for single IP address
- automated attack detection
Attacks and Educational Systems

- cheating \sim false rating
  example: Problem Solving Tutor, Binary crossword
- gaming the system – using hints as solutions

can have similar consequences as attacks
Cheating Using Page Source Code
Context Aware Recommendations

taking context into account – improving recommendations

- when relevant?
- what kind of context?
context:

- **physical** – location, time
- **environmental** – weather, light, sound
- **personal** – health, mood, schedule, activity
- **social** – who is in room, group activity
- **system** – network traffic, status of printers
Context – Applications

- tourism, visitor guides
- museum guides
- home computing and entertainment
- social events
Contextualization

- pre- post- filtering
- model based
  - multidimensionality: user $\times$ item $\times$ time $\times$ ...
  - tensor factorization
Context – Specific Example


- social events (meetup.com)
- inherent item cold-start problem
  - short-lived
  - in the future, without “historical data”
- contextual information useful
Contextual Models

social groups, social interaction
content textual description of events, TF-IDF
location location of events attended
time time of events attended
Figure 1: Geographical densities of two users.
Context: Time

(a) Distribution per day.

(b) Distribution per hour.
Learning, Evaluation

- machine learning feature weights (Coordinate Ascent)
- historical data, train-test set division
- ranking metric: normalized discounted cumulative gain (NDCG)
Shared Accounts


typical example: family sharing single account

Is this a problem? Why?
Shared Accounts


typical example: family sharing single account

Is this a problem? Why?

- dominance: recommendations dominated by one user
- generality: too general items, not directly relevant for individual users
- presentation
Shared Account: Evaluation

- hard to get “ground truth” data
- log data insufficient

How to study and evaluate?
Shared Account: Evaluation

- hard to get “ground truth” data
- log data insufficient

How to study and evaluate?

- artificial shared accounts – mix of two accounts
- not completely realistic, but “ground truth” now available
- combination of real data and simulation
## Shared Account: Example

Table 3: Example of user 562 suffering from sharing an account with user 4385.

<table>
<thead>
<tr>
<th>user ID</th>
<th>562</th>
<th>4385</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{au} = DAMIB-COVER$ (p=0.75)</td>
<td>The Silence of the Lambs, Fargo, Schindler's List, A Nightmare on Elm Street, Halloween: H20, Pulp Fiction, Shakespeare in Love, The Shining, The Exorcist, Sleepy Hollow</td>
<td></td>
</tr>
</tbody>
</table>
Case Studies: Note

- recommender systems widely commercially applied
- nearly no studies about “business value” and details of applications (trade secrets)
Case Studies

- Game Recommendations
- App Recommendations
- YouTube
- Google News
- Yahoo! Music Recommendations
- Book Recommendations for Children
Personalized Game Recommendations

- Recommender Systems - An Introduction book, chapter 8
  Personalized game recommendations on the mobile internet

- A case study on the effectiveness of recommendations in the mobile internet, Jannach, Hegelich, Conference on Recommender systems, 2009
Personalized Game Recommendations

setting:

- mobile Internet portal, telecommunications provider in Germany
- catalog of games (nonpersonalized in the original version):
  - manually edited lists
  - direct links – teasers (text, image)
  - predefined categories (e.g., Action&Shooter, From 99 Cents)
  - postsales recommendations
Figure 1: Catalog navigation and categories
Personalized Game Recommendations

personalization:
  - new “My Recommendations” link
  - choice of teasers
  - order of games in categories
  - choice of postsales recommendations
Algorithms

- nonpersonalized:
  - top rating
  - top selling

- personalized:
  - item-based collaborative filtering (CF)
  - Slope One (simple CF algorithm)
  - content-based method (using TF-IDF, item descriptions, cosine similarity)
  - hybrid algorithm (< 8 ratings: content, ≥ 8 ratings: CF)
Figure 2: Average number of item detail views per “My Recommendations” visits
Figure 3: Average number of downloads per “My Recommendations” visit
Figure 4: Average number of game purchases and demo downloads in post-sales situation.
Figure 5: Total number of non-free game downloads.
App Recommendations

App recommendation: a contest between satisfaction and temptation (2013)

- one-shot consumption (books, movies) vs continuous consumption (apps)
- impact on alternative (closely similar) apps, e.g., weather forecast
- when to recommend alternative apps?
Figure 2: Three scenarios of failed recommendation. The solid arrow means the user downloads the app while the dashed arrow indicates the particular app is recommended to the user.
Actual Value, Tempting Value

- actual value – “real satisfactory value of the app after it is used”
- tempting value – “estimated satisfactory value” (based on description, screenshots, ...)

computed based on historical data: users with installed App $i$ who view description of App $j$ and decide to (not) install $j$
Figure 5: Actual-tempting difference with regarding to app category. Note that negative value means the app’s actual value is smaller than its tempting value and vice versa.
Recommendations, Evaluation

- AT model, combination with content-based, collaborative filtering
- evaluation using historical data
- relative precision, recall
YouTube

- *The YouTube video recommendation system* (2010)
  - description of system design (e.g., related videos)
  - analysis of data from YouTube
- *Video suggestion and discovery for YouTube: taking random walks through the view graph* (2008)
  - algorithm description, based on view graph traversal
  - use of context, predicting watch times
YouTube: Challenges

compare to movies (Netflix) or books (Amazon)

- poor meta-data
- many items, relatively short
- short life cycle
- short and noisy interactions
Input Data

- content data
  - raw video streams
  - metadata (title, description, ...)
- user activity data
  - explicit: rating, liking, subscribing, ...
  - implicit: watch, long watch

in all cases quite noisy
Related Videos

goal: for a video \( v \) find set of related videos

*relatedness score* for two videos \( v_i, v_j \):

\[
 r(v_i, v_j) = \frac{c_{ij}}{f(v_i, v_j)}
\]

- \( c_{ij} \) – co-visititation count (within given time period, e.g. 24 hours)
- \( f(v_i, v_j) \) – normalization, “global popularity”, e.g.,
  \[
  f(v_i, v_j) = c_i \cdot c_j \text{ (view counts)}
  \]

top N selection, minimum score threshold
Generating Recommendation Candidates

- seed set $S$ – watched, liked, added to playlist, ...
- candidate recommendations – related videos to seed set

\[
C_1(S) = \bigcup_{v_i \in S} R_i \\
C_n(S) = \bigcup_{v_i \in C_{n-1}} R_i
\]
Ranking

1. video quality
   - “global stats”
   - total views, ratings, commenting, sharing, ...

2. user specificity
   - properties of the seed video
   - user watch history

3. diversification
   - balance between relevancy and diversity
   - limit on number of videos from the same author, same seed video
User Interface

screenshot in the paper:

screenshot from current application:
System Implementation

“batch-oriented pre-computation approach”

1. data collection
   - user data processed, stored in BigTable
2. recommendation generation
   - MapReduce implementation
3. recommendation serving
   - pre-generated results quickly served to user
Evaluation

![Graph showing normalized click through rate over the week]

- **Most Viewed**
- **Top Favorited**
- **Top Rated**
- **Recommended**

specific aspects:
- short time span of items (high churn)
- scale, timing requirements
Google News: Algorithms

- collaborative filtering using MinHash clustering
- probabilistic latent semantic indexing
- co-visitation counts

MapReduce implementations
Evaluation

- datasets:
  - MovieLens $\sim$ 1000 users; 1700 movies; 54,000 ratings
  - NewsSmall $\sim$ 5000 users; 40,000 items; 370,000 clicks
  - NewsBig $\sim$ 500,000 users, 190,000 items; 10,000,000 clicks

- repeated randomized cross-validation (80% train set, 20% test set)

- metrics: precision, recall
Evaluation
Evaluation
Evaluation on Life Traffic

- large portion of life traffic on Google news
- comparison of two algorithms:
  - each algorithms generates sorted list of items
  - interlace these two lists
  - measure which algorithm gets more clicks
- baseline: “Popular” (age discounted click count)
Evaluation

The graph shows the number of clicks as a fraction of popular items over a period of days. The lines represent different conditions: "Popular", "CSBiased", and "CVBiased". The data fluctuates significantly across the days, indicating variability in click behavior under different conditions.
Evaluation

The graph shows the number of clicks as a fraction of uniform weighting over days. The lines represent different conditions:
- BOTH (cvw:mhw:plw_1:1:1)
- MH (cvw:mhw:plw_1:1:0)
- PLSI (cvw:mhw:plw_1:0:1)

The x-axis represents the number of days, and the y-axis shows the number of clicks as a fraction of uniform weighting.
Music Recommendations


- large dataset (KDD cup 2011): 600 thousand items, 1 million users, 250 million ratings
- multi-typed items: tracks, albums, artists, genres
- taxonomy
- temporal dynamics
Why the peaks?

Figure 1: The distribution of ratings. The approximately discrete nature of the distribution is evident.
Ratings

Why the peaks?
Different widgets used for collecting ratings, including “5 stars” (translated into 0, 30, 50, 70, 90 values)

Figure 1: The distribution of ratings. The approximately discrete nature of the distribution is evident
Figure 2: The distribution of item mean ratings
User Mean Ratings

Figure 3: The distribution of user mean ratings
Item vs user means – why the discrepancy?
Item vs user means – why the discrepancy?

Users with who rate less, rate higher. Long term users are more critical.
Number of Ratings and Mean Rating

Figure 4: Median of user ratings as a function of the number of ratings issued by the user. The vertical lines represent inter-quartile range.
Types of Items

Also the type of rated items differ:

Figure 5: The fraction of ratings the four item types receive as a function of the number of ratings a user gives.
Get to know your data before you start to use it.
Figure 6: Items temporal basis functions $\{f_i(t)\}_{i=1}^{4}$ vs. time since an item’s first rating measured in weeks.
Table 2: Root Mean Squared Error (RMSE) of the evolving model. RMSE reduces while adding model components.
Book Recommendations for Children

What to read next?: making personalized book recommendations for K-12 users (2013)

books for children, specific aspects:

- focus on text difficulty
- less ratings available
Readability Analysis

Diagram:
- **A book** B
- Metadata on B
- An excerpt
- Topical information
- Author information

<table>
<thead>
<tr>
<th>Predictors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grammar concepts</td>
</tr>
<tr>
<td>Shallow features</td>
</tr>
<tr>
<td>US curriculum subject areas</td>
</tr>
<tr>
<td>Subject Headings</td>
</tr>
<tr>
<td>Audience level</td>
</tr>
<tr>
<td>Subject Headings of author’s other books</td>
</tr>
<tr>
<td>Subject areas of author’s other books</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Regression Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression intercept</td>
</tr>
<tr>
<td>Regression coefficients</td>
</tr>
<tr>
<td>Values for predictors</td>
</tr>
</tbody>
</table>

The Grade Level of B
Evaluation of Readability Analysis

dataset: > 2000 books, “gold standard”: publisher-provided grade level

Figure 2: Performance evaluation of ReLAT
1. identifying candidate books (based on readability)
2. content similarity measure
3. readership similarity measure
4. rank aggregation
Content Similarity

- brief descriptions from book-affiliated websites (not the content of book itself)
- cosine similarity, TF-IDF
- word-correlation factor – based on frequencies of co-occurrence and relative distance in Wikipedia documents
Content Similarity – Equations Preview

\[ C\text{Sim}(B, P) = \max_{P_B \in P} \frac{\sum_{i=1}^{n} V B_i \times V P_{B_i}}{\sqrt{\sum_{i=1}^{n} V B_i^2} \times \sqrt{\sum_{i=1}^{n} V P_{B_i}^2}} \quad \text{(3)} \]

where \( B \) and \( P_B \) are represented as \( n \)-dimensional vectors \( V B = <V B_1, ..., V B_n> \) and \( V P_B = <V P_{B_1}, ..., V P_{B_n}> \), respectively, \( n \) is the number of distinct words in the descriptions of \( B \) and \( P_B \), and \( V B_i \) (\( V P_{B_i} \), respectively), which is the weight assigned to word \( B_i \) (\( P_{B_i} \), respectively), is calculated as shown in the equations in Table 2.

Table 2: TF-IDF weighting scheme used in the enhanced cosine similarity measure in Equation 3

<table>
<thead>
<tr>
<th>Condition</th>
<th>Weight Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>( B_i \in B ) and ( P_{B_i} \in P_B )</td>
<td>( V_{B_i} = tf_{B_i,B} \times idf_{B_i} ) and ( V_{P_{B_i}} = tf_{P_{B_i},P_B} \times idf_{P_{B_i}} )</td>
</tr>
<tr>
<td>( B_i \in B ) and ( P_{B_i} \notin P_B )</td>
<td>( V_{B_i} = tf_{B_i,B} \times idf_{B_i} ) and ( V_{P_{B_i}} = \frac{\sum_{c \in HS_{B_i}} tf_c, P_B \times idf_c}{</td>
</tr>
<tr>
<td>( B_i \notin B ) and ( P_{B_i} \in P_B )</td>
<td>( V_{B_i} = \frac{\sum_{c \in HS_{P_{B_i}}} tf_{c,B} \times idf_c}{</td>
</tr>
</tbody>
</table>
Readership Similarity

- collaborative filtering, item-item similarity
- co-occurrence of items bookmarked by users
- Lennon similarity measure

\[ RSim(B, P) = \max_{P_B \in P} \left(1 - \frac{\min(|S_B - S_n|, |S_{P_B} - S_n|)}{\min(|S_B - S_n|, |S_{P_B} - S_n|) + |S_n|}\right) \]
Rank Aggregation

- combine ranking from content and readership similarity
- Borda Count voting scheme
  - simple scheme to combine ranked list
  - points $\sim$ order in a list
Evaluation

- data: BiblioNasium (web page for kids), bookmarked books
- evaluation protocol: five-fold cross validation
- ranking metrics: Precision10, Mean Reciprocal Rank (MRR), Normalized Discounted Cumulative Gain (nDCG)
Summary

illustration of many aspects relevant in development of recommender systems:

- attacks
- context
- groups, shared accounts
- approaches to evaluation
- diversity
- differences between domains (books, movies, news... )