

Recommender Systems: Case Studies

Radek Pelánek

2014

Case Studies: Note

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

Case Studies

- Game Recommendations
- Amazon
- YouTube
- Google News
- 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

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)

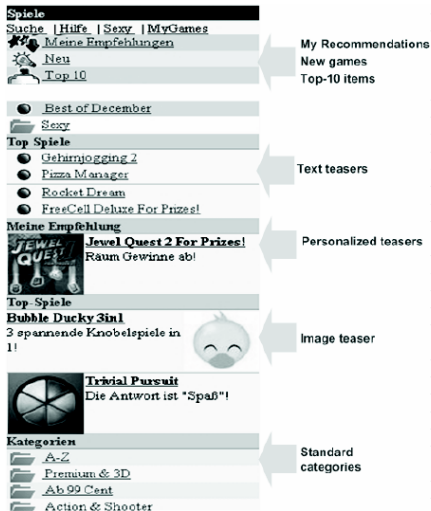


Figure 1: Catalog navigation and categories

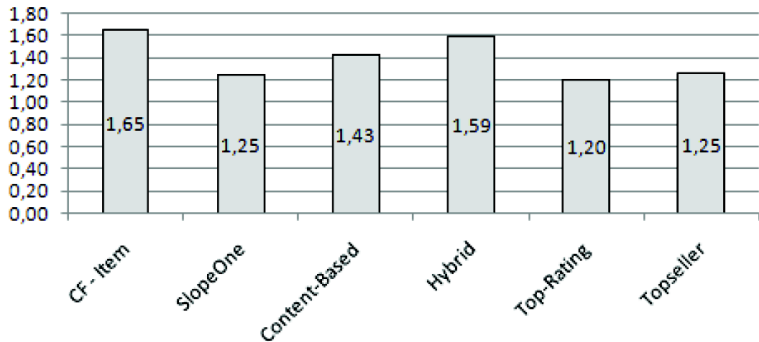


Figure 2: Average number of item detail views per “My Recommendations” visits

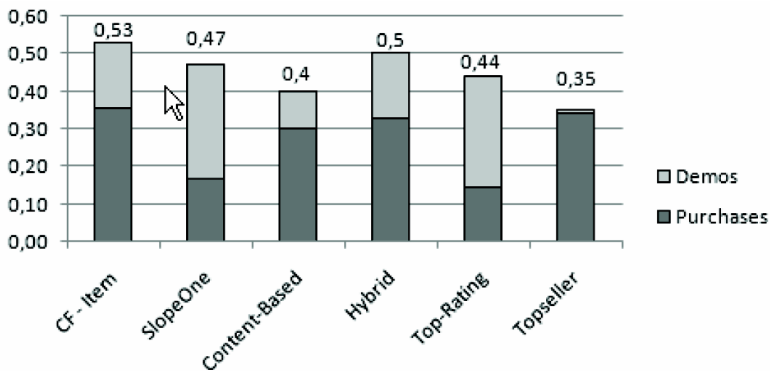


Figure 3: Average number of downloads per “My Recommendations” visit

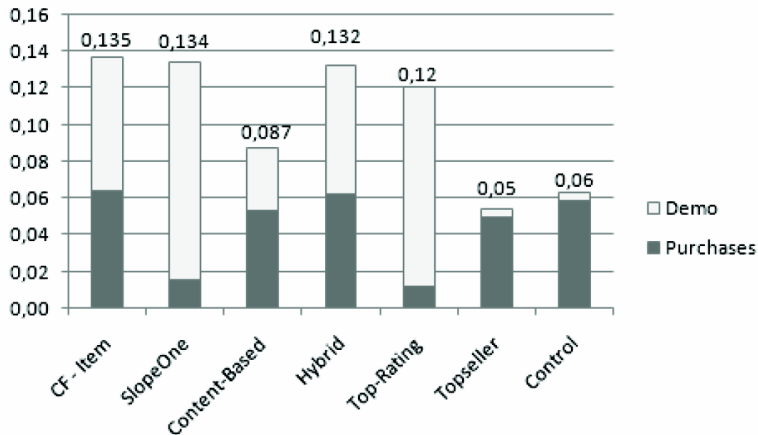


Figure 4: Average number of game purchases and demo downloads in post-sales situation.

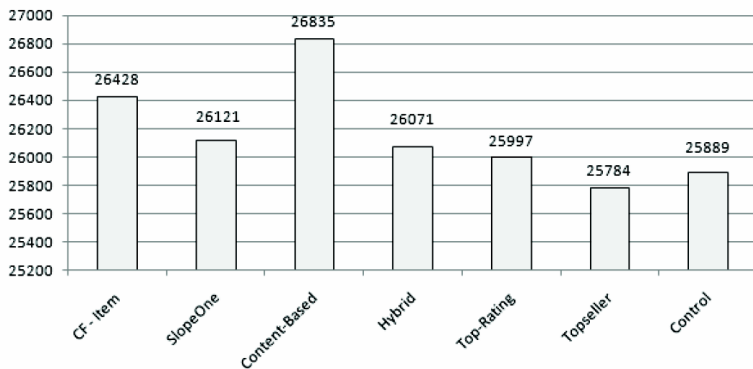


Figure 5: Total number of non-free game downloads.

- *Amazon.com recommendations: item-to-item collaborative filtering* (2003)
- introduced item-item CF and methods for delivery of recommendations (common today)
- discussion of scalability issues, approaches to deal with it (clustering, offline processing)
- no evaluation or technical details

- *The YouTube video recommendation system* (2010)
 - description of system design (e.g., related videos)
 - evaluation, application (weak)
- *The impact of YouTube recommendation system on video views* (2010)
 - 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

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-visitation count (within given time period, e.g. 24 hours)
- $f(v_i, v_j)$ – normalization, “global popularity”, e.g.,
 $f(v_i, v_j) = c_i \cdot c_j$ (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

$$C_1(S) = \cup_{v_i \in S} R_i$$

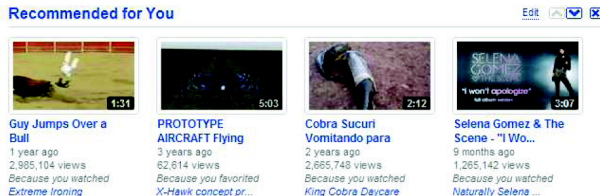
$$C_n(S) = \cup_{v_i \in C_{n-1}} R_i$$

Ranking

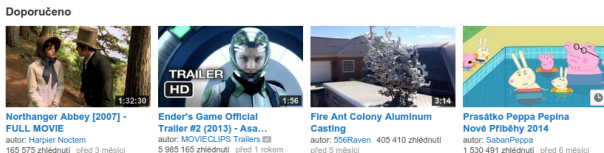
- 1 video quality
 - “global stats”
 - total views, ratings, commenting, sharing, ...
- 2 user specificity
 - properties of the seed video
 - user watch history
- 3 diversification
 - balance between relevancy and diversity
 - limit on number of videos from the same author, same seed video

User Interface

screenshot in the paper:



screenshot from current application:

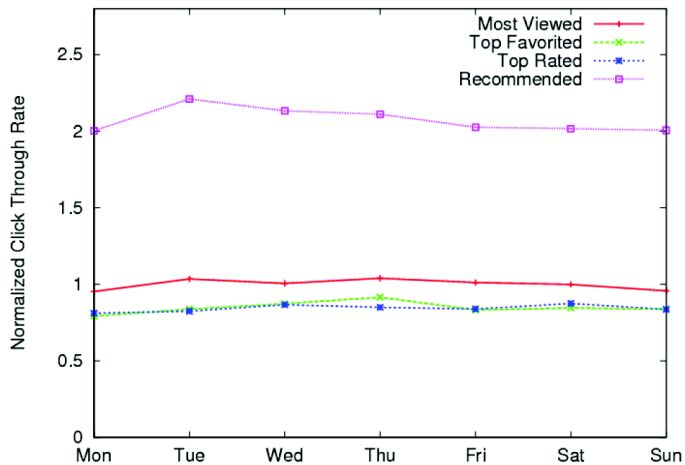


System Implementation

“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



Google News Personalization: Scalable Online Collaborative Filtering (2007)

specific aspects:

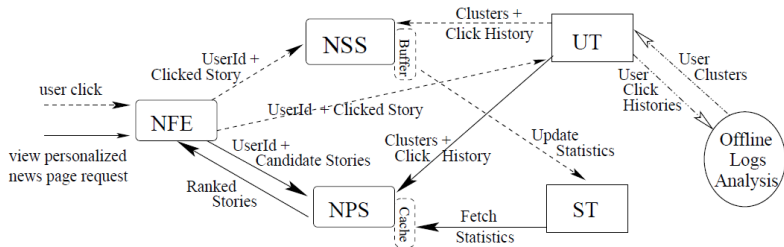
- short time span of items
- scale, timing requirements

Google News: Algorithms

- collaborative filtering using MinHash clustering
- probabilistic latent semantic indexing
- covisitation counts

MapReduce implementations

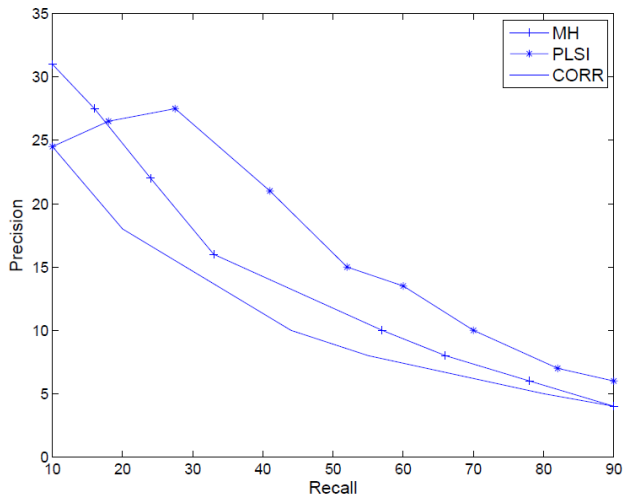
System Setup



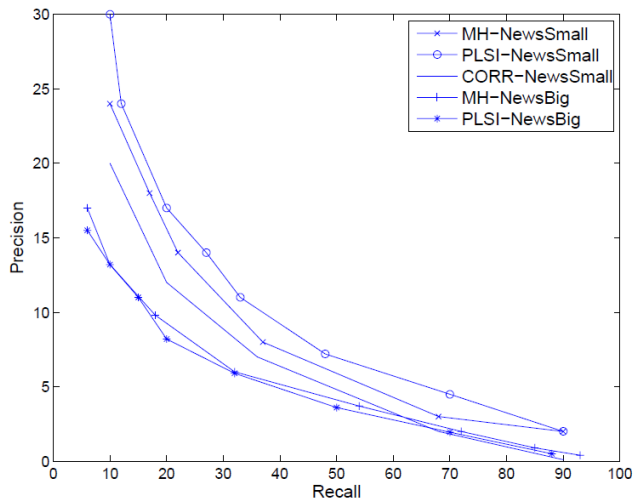
Evaluation

- 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



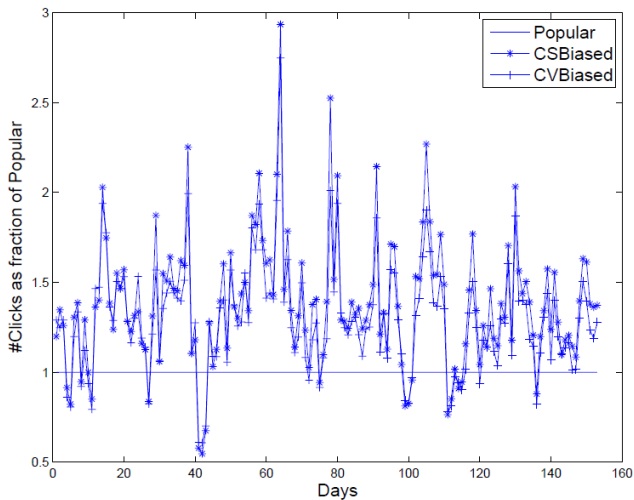
Evaluation



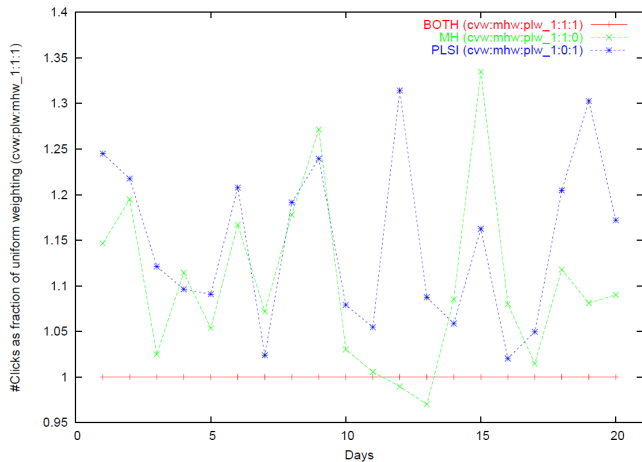
Evaluation on Life Traffic

- 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



Evaluation



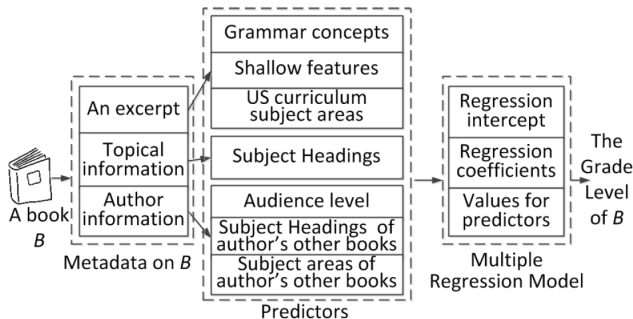
Book Recommendations for Children

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

books for children:

- focus on text difficulty
- less ratings available

Readability Analysis



Evaluation of Readability Analysis

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

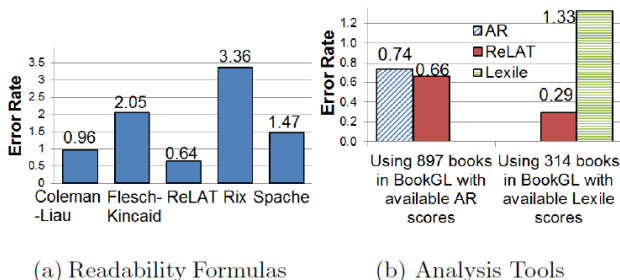


Figure 2: Performance evaluation of ReLAT

Book Recommender

- 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

$$CSim(B, P) = \max_{P_B \in P} \frac{\sum_{i=1}^n VB_i \times VP_{B_i}}{\sqrt{\sum_{i=1}^n VB_i^2} \times \sqrt{\sum_{i=1}^n VP_{B_i}^2}} \quad (3)$$

where B and P_B are represented as n -dimensional vectors $VB = \langle VB_1, \dots, VB_n \rangle$ and $VP_B = \langle VP_{B_1}, \dots, VP_{B_n} \rangle$, respectively, n is the number of distinct words in the descriptions of B and P_B , and VB_i (VP_{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

Condition	Weight Assignment
$B_i \in B$ and $P_{B_i} \in P_B$	$V_{B_i} = tf_{B_i, B} \times idf_{B_i}$ and $V_{P_{B_i}} = tf_{P_{B_i}, P_B} \times idf_{P_{B_i}}$
$B_i \in B$ and $P_{B_i} \notin P_B$	$V_{B_i} = tf_{B_i, B} \times idf_{B_i}$ and $V_{P_{B_i}} = \frac{\sum_{c \in HS_{B_i}} tf_{c, P_B} \times idf_c}{ HS_{B_i} }$
$B_i \notin B$ and $P_{B_i} \in P_B$	$V_{B_i} = \frac{\sum_{c \in HS_{P_{B_i}}} tf_{c, B} \times idf_c}{ HS_{P_{B_i}} }$ and $V_{P_{B_i}} = tf_{P_{B_i}, P_B} \times idf_{P_{B_i}}$

Readership Similarity

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

$$RSim(B, P) = \max_{P_B \in P} \left(1 - \frac{\min(|S_B - S_{\cap}|, |S_{P_B} - S_{\cap}|)}{\min(|S_B - S_{\cap}|, |S_{P_B} - S_{\cap}|) + |S_{\cap}|} \right)$$

Rank Aggregation

- combine ranking from content and readership similarity
- Borda Count voting scheme
- simple scheme to combine ranked 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)

Evaluation

