Tinderbook: Fall In Love with Culture

Enrico Palumbo, Alberto Buzio, Andrea Gaiardo, Giuseppe Rizzo, Raphael Troncy, Elena Baralis

ESWC 2019, Portoroz, Slovenia

linksfoundation.com
Book Recommendation

Key concepts

• 2,210,000 new books are published every year

• **Hard to find a good book to read**, most readers typically give up on a book in the early chapters

• **Recommender Systems help in finding a good book** → several existing websites (GoodReads, LibraryThing, WhichBook…)
Collaborative filtering

Most of the existing book recommender systems are based on collaborative filtering

“People who buy X, often buy Y”

Problem:

- Collaborative filtering suffers the **cold start problem**, i.e. it does not work well when little is known about the preferences of the user.
- **Long onboarding procedures**: ask lot of information about the user, mandatory login, ask to rate many books, etc...
## Competing systems

### Comparison of existing book recommender systems

| RECOMM. APPROACH | MIN # BOOKS FOR RECOMM. | FEEDBACK MODE | USER PROFILING | USER EXPERIENCE |
|------------------|-------------------------|---------------|----------------|-----------------|
| **goodreads**    | collaborative filtering 20 | rating        | ✓              | ✓               |
|                  | collaborative filtering 1 | x             | x              | ✓               |
| **LibraryThing** | collaborative filtering 10 | rating        | ✓              | x               |
|                  | content-based filtering 0 | x             | x              | ✓               |
| **BookBub**      | content-based filtering 0 | like          | ✓              | ✓               |
| **YOURNEXTREAD.COM** | collaborative filtering 1 | like & dislike | X              | ✓               |
| **Readgeek**     | collaborative filtering 2 | rating        | ✓              | ✓               |
| **TINDERBOOK**   | Hybrid filtering 1       | like & dislike| X              | ✓               |

- **MANDATORY LOGIN**: user data, favourite genres
- **INFO REQUIRED**: book liked, book ratings & liked, book tags
- **FAST & EASY ONBOARDING**: ✓
- **WEB**: ✓
- **MOBILE (OPTIMIZED)**: ✓
TinderBook

Goal

• **Accurate** recommendations given just **one book** that the **user** likes

• **No login**, no additional information about the user
Onboarding

Usage Session

**Temperature** parameter governs the trade-off between showing most popular vs random books

**Popularity** of the book is defined as the fraction of positive feedback (ratings r ≥ 8) obtained by the book in the LibraryThing dataset

- Low Temperature → very popular books
- High Temperature → less popular books
Recommendations

Usage Session

User receives **five recommended books** based on the onboarding choice

- **Positive feedback** → Thumbs Up or Right Swipe

- **Negative feedback** → Thumbs Down or Left Swipe

- To read DBpedia **book abstract** → Info button
Graphical User Interface
Architecture

Jane

User Interface

TINDERBOOK
Fall in love with culture
Can't find a good book?
Tell us a single book you like and you will get book recommendations.

MongoDB

Seed
Discard
Feedback

API

ρ (i_j, i_k)

Book metadata

Book cover

DBpedia

Model

Jane

User Interface

TINDERBOOK
Fall in love with culture
Can't find a good book?
Tell us a single book you like and you will get book recommendations.

MongoDB

Seed
Discard
Feedback

API

ρ (i_j, i_k)

Book metadata

Book cover

DBpedia

Model
Approach

How to generate accurate recommendation with only one book?

- Entity2rec $^{[1]}$ is a knowledge graph embedding algorithm that has reached SOTA performance
- Knowledge Graphs are ideal for hybrid RSs
- Research has shown that KG embeddings can be effectively used for recommendations $^{[1]}$ $^{[2]}$ $^{[3]}$ $^{[4]}$
- However, in entity2rec as in $^{[1]}$, you need information about the user

- Need to extend entity2rec to address cold-start problem
Item-to-Item Recommendation

**Definition**

\[ \rho (i, j, i, k) \]

- **Seed**
- **Ranking function**
- **Items**

Graph showing books: "Animal Farm" by George Orwell, "Fahrenheit 451" by Ray Bradbury, "Lord of the Flies" by William Golding, and "Brave New World" by Aldous Huxley.
Entity2rec

**Item-item recommendation**

Property-specific embeddings using node2vec \[^5\]

Final ranking function

$$\rho_{\text{dct:subject}}(i, j, k) = \text{avg}(\rho_p(i, j, k))$$

$$\rho_{\text{dbo:author}}(i, j, k) = \text{avg}(\rho_p(i, j, k))$$
Offline evaluation

Experimental setup

Datasets

- **LibraryThing**: 7,112 users, 37,231 books and 626,000 book ratings ranging from 1 to 10
  
  Training set: 70%  
  Validation set: 10%  
  Test set: 20%

- **LibraryThing + DBpedia**: LibraryThing books have been mapped to DBpedia entities [6]

  6,789 users (95.46%)  
  9,926 books (26.66%)  
  410,199 ratings (65.53%)

Metrics

- **P@5**: fraction of books in top-5 recommendations that are relevant to the user (rel-in-top-5 / 5)
- **R@5**: fraction of relevant books for the user that are in the top-5 recommendations (rel-in-top-5 / all-rel)
- **SER@5**: like P@5, but top 5 popular books are not counted. Measures non-obvious accurate recommendations
- **NOV@5**: how little known (in terms of popularity) are the recommended books. Does not matter accuracy.
## Offline evaluation

### Results

| System                | P@5    | R@5    | SER@5   | NOV@5 |
|-----------------------|--------|--------|---------|--------|
| entity2rec            | 0.0549 | 0.0508 | 0.0514  | 11.099 |
| ItemKNN               | 0.0484 | 0.0472 | 0.0463  | 12.2   |
| RDF2Vec (content-based)| 0.0315 | 0.0288 | 0.0311  | 13.913 |
| MostPop               | 0.0343 | 0.0256 | 0.007   | 8.4525 |
Online evaluation

Metrics

- Completeness
- Discard
- Dropout rate
- Seed popularity
- Recommendation time (or latency)
Online evaluation

Setup

- Time span: two weeks
- Estimate a number of >100 users
- Two different configurations of temperature parameter: governs the degree of randomness in the books that are presented to the user in the onboarding phase
  - low temperature = more popular books
  - high temperature = more randomness

|                  | All  | T = 0.3 (1st week) | T = 1. (2nd week) |
|------------------|------|--------------------|-------------------|
| Tot. # seeds     | 470  | 358                | 112               |
| Tot. # feedback  | 1936 | 1495               | 441               |
| Tot. # discarded books | 3668 | 2263               | 1405              |
Online evaluation

**Temperature**

- $T = 0.3$ more than 90% of the seed books are concentrated in the top 10% in terms of popularity.

- $T = 1$. The popularity bias, although still strong, decreases.
## Online evaluation

### Results

|                  | $T = 0.3$                      | $T = 1.$                      | p value      | significant |
|------------------|-------------------------------|-------------------------------|--------------|-------------|
| P@5              | 0.497368 ± 0.026381           | 0.495833 ± 0.052701           | 9.79E-01     | no          |
| SER@5            | 0.417105 ± 0.024892           | 0.437500 ± 0.047382           | 7.07E-01     | no          |
| NOV@5            | 8.315443 ± 0.176832           | 10.095039 ± 0.347261          | 2.30E-05     | yes         |
| completeness     | 0.903947 ± 0.018229           | 0.937500 ± 0.025108           | 2.86E-01     | no          |
| discard          | 6.321229 ± 0.663185           | 12.544643 ± 2.070238          | 2.09E-03     | yes         |
| dropout          | 0.131285 ± 0.019150           | 0.178571 ± 0.039930           | 2.45E-01     | no          |
| seed_pop         | 0.002626 ± 0.000060           | 0.000835 ± 0.000086           | 2.74E-48     | yes         |
Summary

- TinderBook is a book recommender system that provides recommendations given just one book that the user likes.

- It is deeply rooted in semantic technologies:
  - It presents an extension of a state-of-the-art KG embeddings algorithm to the cold start scenario.
  - It is based on DBpedia data.

- It has received good feedback from the public saying that it’s fun to use and recommendations are accurate.

- Application is online at [http://www.tinderbook.it](http://www.tinderbook.it) and data collection is ongoing.
Lessons learned and discussion

• The use of semantic technologies have allowed us to:
  • Rely on a knowledge graph embeddings algorithm to create a hybrid recommender system in a cold start scenario
  • Use DBpedia data to describe books (“Info” panel, “title”, “author”). Linked Open Data allowed us to build an application quickly, without creating an ad-hoc curated database. Semantic Web technologies integrate seamlessly with web applications.

• At the same time…:
  • We have used an existing mapping from LibraryThing to DBpedia. If someone wanted to replicate this approach on a different dataset, she would have to take care of the mapping, which is time consuming
  • Mapping books to DBpedia has produced a huge data loss (26% of the original catalog)
  • DBpedia information is not always accurate. For thumbnails, we had to rely on Google to have more accurate results.

• To foster the development of SW-based application, effort needs to be focused on improving data quality and in making available to the public as many datasets mapping as possible
References

[1] Palumbo, E., Rizzo, G., Troncy, R.: Entity2rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation. In: 11th International Conference on Recommender Systems (RecSys). pp. 32–36 (2017)

[2] Palumbo, E., Rizzo, G., Troncy, R., Baralis, E., Osella, M., Ferro, E.: Knowledge graph embeddings with node2vec for item recommendation. In: European Semantic Web Conference (ESWC), Demo Track. pp. 117–120 (2018)

[3] Palumbo, E., Rizzo, G., Troncy, R., Baralis, E., Osella, M., Ferro, E.: Translational models for item recommendation. In: European Semantic Web Conference (ESWC) Satellite Events. pp. 478–490 (2018)

[4] Rosati, J., Ristoski, P., Di Noia, T., Leone, R.d., Paulheim, H.: RDF graph embeddings for content-based recommender systems. In: CEUR workshop proceedings. vol. 1673, pp. 23–30 (2016)

[5] Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: 22nd International Conference on Knowledge Discovery and Data Mining (SIGKDD). pp. 855–864 (2016)

[6] Di Noia, T., Ostuni, V.C., Tomeo, P., Di Sciascio, E.: Sprank: Semantic path-based ranking for top-n recommendations using linked open data. ACM Transactions on Intelligent Systems and Technology (TIST) 8(1) (2016)
http://www.tinderbook.it

https://github.com/D2KLab/entity2rec

https://www.slideshare.net/EnricoPalumbo2

Contacts

Enrico Palumbo - Rec Sys - enrico.palumbo@linksfoundation.com

Alberto Buzio - User Interface - alberto.buzio@linksfoundation.com