“I’m looking for something like ...”: Combining Narratives and Example Items for Narrative-driven Book Recommendation

Toine Bogers
Science, Policy and Information Studies
Department of Communication & Psychology
Aalborg University Copenhagen
Denmark
toine@hum.aau.dk

Marijn Koolen
Department of Digital Infrastructure
Humanities Cluster
Royal Netherlands Academy of Arts and Sciences
Netherlands
marijn.koolen@di.huc.knaw.nl

ABSTRACT

Current-generation recommendation algorithms are often focused on generic ratings prediction and item ranking tasks based on a user’s past preferences. However, many recommendations are more complex with specific criteria and constraints on which items are relevant. This paper focuses on a particular type of complex recommendation needs: narrative-driven recommendation, where users describe their needs in short narratives, often with one or more example items that fit that need, against a background of historical preferences that may not be spelled out in the narrative, but do play a role in their considerations. Previous work has shown that numerous examples of such complex needs exist on the Web, yet current-generation systems offer limited to no support for these needs. In this paer, we focus on narrative-driven book recommendation in the context of LibraryThing users posting recommendation requests in the discussion forums. We propose several new algorithms that take advantage of these narratives and example items as well as hybrid systems, the majority of which significantly outperform classic collaborative filtering. We show that narrative-driven recommendation is indeed a complex scenario that requires further study. Our findings have consequences for system design and development not only in the book domain, but also in other domains where users express focused recommendation needs, such as movies, television, games, and music.

KEYWORDS

Narrative-driven recommendation, book recommendation, example-driven recommendation

ACM Reference Format:
Toine Bogers and Marijn Koolen. 2018. “I’m looking for something like ...”: Combining Narratives and Example Items for Narrative-driven Book Recommendation. In *Proceedings of Knowledge-aware and Conversational Recommender Systems (KaRS) Workshop 2018 (co-located with RecSys 2018).* ACM, New York, NY, USA, 9 pages.

1 INTRODUCTION

Research on recommendation algorithms for ratings prediction and item ranking has resulted in a better understanding of these tasks and an array of algorithms with state-of-the-art performance. However, not all recommendation needs can be solved by providing a ‘generic’ set of recommendations based on a user’s past preferences. In many cases, recommendation is a more complex problem and often just a single stage in a user’s more complex background need. These needs can place a variety of constraints on which recommendations are interesting and appropriate to the user—and they are unlikely to be satisfied well by the current generation of ratings prediction and item ranking algorithms.

Relatively little research has been done on these complex recommendation needs and how much of a problem they still pose for current-generation recommender systems. In 2017, Kang et al. [20] analyzed the composition of such complex needs by exploring how people use natural language to ask for movie recommendations using a chatbot. They coded 498 natural-language interactions with the chatbot, revealing a complex mix of objective, subjective and navigational aspects of the users’ recommendation needs. Kang et al. argue that many of these recommendation needs are hard to support using current systems and data sources. This argument was echoed in a similar study by Bogers and Koolen [4], who analyzed a set of 974 complex book requests from the LibraryThing (LT) forums. They argue that these are examples of *narrative-driven recommendation* (NDR), which they define as a scenario that combines (1) a narrative description of the desired aspects of relevant items provided by the user, and (2) user preference information, either in the form of a transaction log or of a user-provided *mini-profile* containing positive and/or negative examples of other items. Analysis of these 974 requests revealed them to be similar in many ways to the movie recommendation needs collected by Kang et al. [20]. Bogers and Koolen estimate that over 25,000 of such complex requests exist in the LT forums alone, with an order of magnitude more requests available all across the Web for a variety of domains.

However, neither Kang et al. [20] nor Bogers and Koolen [4] propose any recommendation algorithms to address such needs and provide focused recommendations. To the best of our knowledge, no algorithms exist that can tackle these types of complex recommendation needs. We attempt to remedy this by building on the work by Bogers and Koolen with a first attempt at designing different NDR algorithms that incorporate both the textual narratives as well as the example books and authors provided by the users. Our experiments show that most of our algorithms outperform state-of-the-art collaborative filtering (CF). Hybrid recommenders that combine complementary algorithms based on narratives and examples provide a significant performance boost over our individual algorithms. Nevertheless, recommendation accuracy in general suggests there is still room for improvement with many difficult subproblems that require further study.

Knowledge-aware and Conversational Recommender Systems (KaRS) Workshop 2018 (co-located with RecSys 2018), October 7, 2018, Vancouver, Canada.

Because of the relative novelty of our specific book recommendation scenario, we start by describing it and our methodology in greater detail in the next section. Sections 3 and 4 respectively describe algorithms that utilize the textual narratives and example items to produce relevant book recommendations. Our efforts at hybrid recommendation are described in Section 5. We discuss related work in Section 6 and conclude in Section 7.

2 METHODOLOGY

Bogers and Koolen [4] introduced the notion of Narrative-Driven Recommendation (NDR) and examined the prevalence, composition, and complexity of such needs on the LibraryThing (LT) forums.[1] They argued that narrative recommendation needs have elements of both recommendation and information retrieval (IR). Our work in this paper builds on this notion and uses the requests from the LT forum as an experimental testbed for NDR. LT is a social cataloging website that allows its users to add books to their profile and tag, rate and review them. The LT forums support users in discussing their reading experiences as well as offer a venue for requesting book recommendations.

Figure 1 illustrates our complex NDR scenario on the LT forums. Users can post *requests* for book recommendations to a forum discussion group, describing their recommendation need containing both a *narrative* description of the kinds of books they want to read and examples of (un)suitable) books and/or authors. These examples form a so-called *mini-profile*, a temporary representations of that user's current preferences. Other users can reply to these posts and come with suggestions for relevant books or authors to read and explore. Many of the users interacting in these discussion threads have their own LT profile of their book preferences. On LT, these books are represented with different types of metadata and user-generated content. We wish to explore the role these user preferences and book representations can play in generating accurate book recommendations.

We expect that tailored recommendations based on individual requests will provide better recommendations than generating a list of book recommendations using a state-of-the-art CF algorithm based on a user's entire profile.

2.1 Datasets

To develop and evaluate our NDR algorithms, we use the Amazon/LibraryThing (A/LT) dataset. The A/LT dataset has been used in the INEX Interactive Track [3] and the Social Book Search Lab [21] and has 2.8M book records consisting of both traditional book metadata as well as user-generated content (UGC) harvested from Amazon (user reviews and ratings) and LT (user tags), as visualized in the bottom-left corner of Figure 1. The same book (or *work*) can have many different editions, each of them with a different ISBN. LT uses its own internal 'work ID' to map books to all associated ISBNs. The book records in the A/LT collection use ISBNs as identifiers, so to avoid including duplicates in our recommendation lists, we mapped these ISBNs to the LT book identifiers using a mapping file provided by LT. Not all ISBNs could be mapped to a general work on LT, so for all intents and purposes we are recommending books from a collection of 2,654,082 book records, corresponding to 1,904,950 distinct works.

To enable recommendations based on the example authors in mini-profiles, we assigned unique author IDs to each author in the A/LT collection. This was necessary, because the A/LT dataset has inconsistent author naming and no author identifiers. We matched duplicate authors by converting their name strings to lowercase and removing punctuation and then assigned IDs. This reduced the list of authors from 968,703 author strings to 916,479 unique author IDs. We also checked for matches between names with and without middle names and/or initial and collapsed them into the same author ID. While not perfect—`John Smith` and `John D. Smith` need not refer to the same person—inspection on a random sample of 100 automatic matches revealed an precision of 89%. Applying this rule further reduced the number of unique authors from 916,479 to 849,578. We then matched all example authors in the mini-profiles to their A/LT author IDs.

In addition to the A/LT dataset, we also use a crawl of LT user profiles with cataloging dates, tags, and ratings. This crawl was conducted in 2012 and 2013 and contains over 66K LT user profiles, with over 29M cataloging transactions and 4.5M distinct books. The LT user profiles dataset has 4,409,399 ratings by 38,174 users, representing less than 15% of the total number of cataloging transactions. Over 42% of users have assigned no ratings at all, and the 58% who have, rate only a small fraction of the books in their catalog. We therefore ignore the ratings and treat each transaction as implicit positive feedback on user preferences.

2.2 Experimental setup

We use the same subset of 974 narrative requests from the LT forums collected by Bogers and Koolen [4] as a starting point for our experiments. These are requests where a LT user started a discussion thread requesting book recommendations and at least one other member posted a reply with suggestions. In our evaluation, we excluded all requests of those users not present in our LT user profile crawl. This resulted in a set of 331 requests, of which 298 have $\geq$ 1 example book with a mean (median) of 2.6 (2) books, and 121 have $\geq$ 1 example author with a mean (median) of 2.0 (1) authors. All requests have at least one example book or author. These examples make up the mini-profile for each request.

When recommending based on these example books and authors, we do not distinguish between positive and negative examples. Instead, we consider all example books as equally relevant sources of recommendation; we are aware of the bias this may introduce. Sentiment analysis could conceivably be used to determine whether examples are cast in a positive or negative light. However, this would also add another layer of complexity and thereby make it harder to tease apart the accuracy of our recommendation algorithms from our example sentiment detection component.

We split the 331 requests into a random 50-50 train/test split with 166 training requests and 165 test requests. Default train/test splits may have more training items, but none of our proposed algorithms have more than one parameter to be optimized. In addition, the recommendation narratives show great variety in terms of narrative length as well as example counts and popularity; a larger test set would allow for better analysis of our algorithms.

[1] Available at https://www.librarything.com/groups.

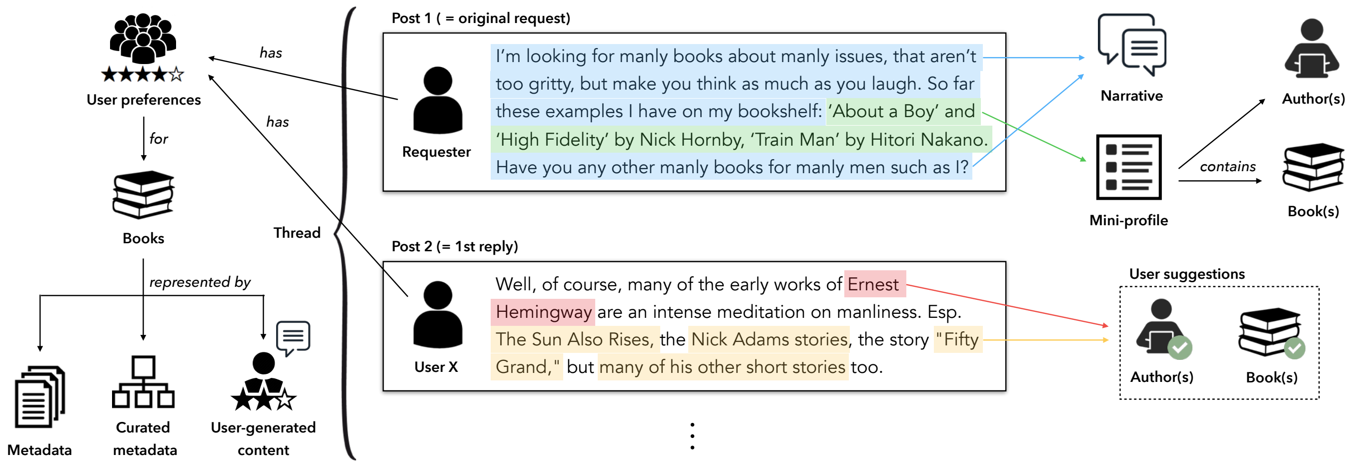
User preferences
has
has
for
Books
represented by
Metadata
Curated metadata
User-generated content
Thread
Post 1 (= original request)
Requester
I'm looking for manly books about manly issues, that aren't too gritty, but make you think as much as you laugh. So far these examples I have on my bookshelf: 'About a Boy' and 'High Fidelity' by Nick Hornby, 'Train Man' by Hitori Nakano. Have you any other manly books for manly men such as I?
Post 2 (= 1st reply)
User X
Well, of course, many of the early works of Ernest Hemingway are an intense meditation on manliness. Esp. The Sun Also Rises, the Nick Adams stories, the story "Fifty Grand," but many of his other short stories too.
Narrative
Mini-profile
contains
Author(s)
Book(s)
User suggestions
Author(s)
Book(s)

Figure 1: A schematic illustration of the narrative-driven book recommendation scenario on LibraryThing.

The requests were posted in the period 2006–2012, with some users posting multiple requests during this period. At the time of these requests, the same user will have different user profiles. This introduces a challenge in training CF models, as their different profiles should not be represented as different users. For a user u posting request r_1 at time t_1 with profile p_1 and r_2 at time t_2 with profile p_2, we need to either update the model in between requests, or use only the profile at p_1. There are some users with multiple requests in our dataset, so to keep our baseline setup simple for training, we represent as user u with multiple requests at t_1 and t_2 by their profile at time t_1. For 53 out of 331 requests, the pre-request profile ended up being based on the earliest request.

We realize that our evaluation uses a relatively low number of requests and only a single train-test split, but wish to emphasize that NDR is different from generic ratings prediction and top-k recommendation, and instead a hitherto unaddressed problem. In our scenario, we are dealing with recommendation needs with a strong IR element, where evaluation using 50-100 topics is common.

2.3 Evaluation

The book requests can be evaluated from different perspectives, representing different recommendation tasks. First, there are the user profiles with all the books that each user added to their personal catalog, with a cataloging date that can be used to determine which books were cataloged *before* a user posted their request (= *pre-request cataloged items*) and which books were cataloged afterwards (= *post-request cataloged items*). For a typical recommendation scenario, the former represent the user's profile, the latter the books to recommend. The request also receives replies from other users with *suggestions* of books relevant to the request. Other users often report having consulted the requester's profile to target their suggestions, although they sometimes suggest books the requester already cataloged. These suggestions represent a recommendation task more focused on the recommendation need. A third perspective is the intersection of the *suggestions* and the *post-request cataloged items*. We refer to these as *post-request cataloged suggestions* (PCS). They represent focused and successful recommendations.

For NDR, the recommendation needs are central, so we use the *suggestions* as relevant recommendations, with the PCS as the most relevant ones. Post-request cataloged items that are *not* suggested in the request thread are considered outside the focus of the recommendation need. We therefore adopted the relevance grading used in CLEF 2016 Social Book Search Lab [21] and consider *suggestions* relevant with relevance value $rv = 1$ and PCS as relevant with $rv = 8$. To enable baseline CF and content-based filtering (CBF) approaches, the training data for a user are the *pre-request catalogued* items of that user prior to their request.

We assume this is a high-precision task, so we focus on top-N measures. We are using graded relevance judgements, so we use nDCG@10 as our main metric and report MRR to provide further insights into the rank where users can find the first relevant item. All reported statistical significance tests are one-tailed bootstrap with 100,000 re-samples at levels $p < 0.05$ and $p < 0.01$.

2.4 Baseline

As argued in Section 1, narrative-driven book recommendation offers a unique and complex recommendation scenario. Generic book recommendation can straightforwardly be addressed using state-of-the-art CF algorithms, such as matrix factorization. However, applying CF to the problem of NDR is unlikely to provide many relevant, on-topic recommendations. To provide evidence for this assertion, we compare our narrative- and example-driven algorithms to a CF baseline.

We used the LightFM toolkit[2] by Kula [26] to train a baseline CF model. LightFM uses Stochastic Gradient Descent for training, with four different loss functions: Logistic, Weighted Approximate-Rank Pairwise (WARP, [41]), Bayesian Personalised Ranking (BPR, [35]) and k-th order statistic loss (k-OS WARP, [42]). Since most users do not rate books on LT, we assume the presence of an book in a user's catalog as implicit positive feedback. We optimized for top-N recommendation with $N = 10$ and evaluated after every 10th epoch up to 100 epochs. Optimal performance was achieved with WARP running for 50 epochs and 300 factors. The narrative requests focus

[2]Available at https://github.com/lyst/lightfm.

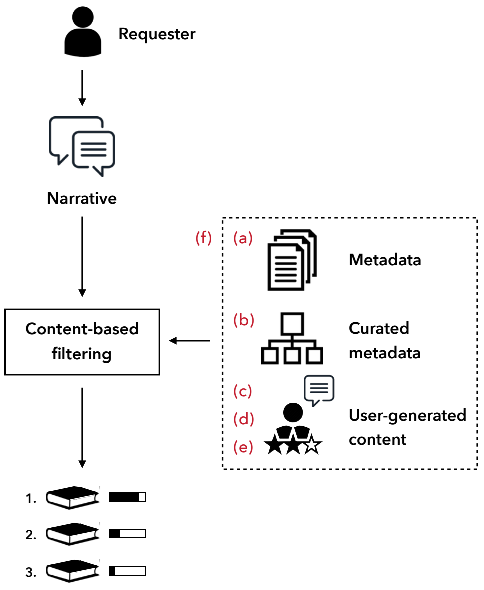

Figure 2: A schematic illustration of our NDR algorithm. The requester's narrative is used as a temporary representation of the user's profile. It is treated as a query and matched against the different book representations; books are then recommended in order of topical relevance. Six different book representations are available for matching: (a) traditional metadata, (b) curated metadata, (c) tags, (d) reviews, (e) user-generated content (= tags and reviews combined), and (f) all representations combined.

the user's recommendation need on specific criteria. We expect that basing recommendations on the entire user profile leads to poor performance. However, given the modest number of examples in the mini-profiles, we expect CF to have little signal to work with, akin to the cold-start problem, resulting in poor performance.

For the CBF approaches, we used the language modeling approach as implemented in the Indri toolkit. Optimal retrieval parameter values[3] for the different versions of the A/LT collection are based on earlier experiments that share a similar setup in that the narrative was also used as a query [5, 6].

3 NARRATIVE-DRIVEN RECOMMENDATION

One source of information about the user's recommendation need is the *narrative* posted to the LT forums. As shown earlier in Figure 1, each request consists of a narrative and, as part of that narrative, a set of example books and authors. In this section, we propose a NDR algorithm that uses the textual representation of the narrative to identify and recommend related books. Figure 2 shows a schematic illustration of our approach.

For each request, we take the narrative description and use CBF to match it against the collection of 2.6 million books. Here, the requester's narrative is treated as a temporary representation of their user profile. Each book in the A/LT collection is represented using different types of metadata fields and matched against the narratives using retrieval algorithms. In our experiments, we distinguish between six categories of metadata representations. One category of metadata (a) contains the core metadata, such as title, author, and publisher. Curated metadata (b), contains index terms and Dewey codes form another category, while Tags (c) are the set of distinct tags assigned to each book by the LT users. The Amazon reviews (d) form another collection representation; the combination of reviews and tags (e) represents the UGC associated with each book. Finally, combining all metadata fields (f) is the sixth collection representation.

[3]An overview of these parameter values can be found at http://anon.ymiz.ed/url.

Table 1: Results for NDR on the test set ($N = 165$) using NDCG@10 and MRR as evaluation metrics, calculated on graded relevance judgments 0-1-8. Best runs are printed in bold. Significant differences indicated for $p < 0.05$ (better $^{\triangle}$ or worse $^{\triangledown}$) and $p < 0.01$ (better $^{\blacktriangle}$ or worse $^{\blacktriangledown}$) as compared to the CF baseline.

Collection	nDCG@10	MRR
(a) Metadata	0.024	0.059$^{\triangle}$
(b) Curated metadata	0.026	0.052
(c) Tags	0.048$^{\blacktriangle}$	0.111$^{\blacktriangle}$
(d) Reviews	0.067$^{\blacktriangle}$	0.164$^{\blacktriangle}$
(e) Reviews + Tags	**0.074**$^{\blacktriangle}$	**0.181**$^{\blacktriangle}$
(f) All fields	**0.074**$^{\blacktriangle}$	0.180$^{\blacktriangle}$
Collaborative filtering	0.017	0.031

3.1 Results & Analysis

Table 1 shows the results of the narrative-based recommendation algorithm. Differences are tested for statistical significance using one-tailed bootstrap with 100,000 resamples. CBF based on narratives outperforms CF, which is not surprising, given the focused nature of the task. CF will recommend items from similar users, but not necessarily within the confines of the information need. Within the CBF approaches, user-generated content clearly outperforms other metadata, possibly because it reflects the language of requesters better than e.g. curated metadata. Reviews perform significantly better than tags. The combination of reviews and tags is also significantly better than tags alone, but not than reviews alone. Reviews use a broader vocabulary than tags to match narrative requests, and maybe cover book aspects that are both relevant in reviewing and in searching. We found no correlation between the performance of different NDR approaches and the length of narratives or the number of examples. Furthermore, we looked at the popularity of recommended items, in terms of how many LT users cataloged them. CF tends to recommend items from the more popular end of the spectrum than the suggestions (including the successful suggestions) and the examples, while CBF tends to recommend more items from the long tail, especially when using curated metadata. Here another advantage of UGC is revealed, as it favors more popular books compared to curated metadata, in that popular books have more UGC than obscure books, so have a higher probability of being retrieved, but favors more obscure books compared to CF in that it penalizes the topic drift in the huge amounts of UGC of popular books.

4 EXAMPLE-DRIVEN RECOMMENDATION

The second source of information about the user's recommendation need is the *mini-profile* they provide, i.e., the example books and/or authors they mention as relevant to their current need. We proposed two algorithms that each uses one of the example types as input: (1) example-driven recommendation using similar books (EDR-1), and (2) example-driven recommendation using similar authors (EDR-2). Figure 3 shows a schematic illustration of both algorithms.

Both algorithms can use different representations of the books and authors, corresponding to the six metadata groups introduced in Section 3: (a) traditional metadata, (b) curated metadata, (c) tags, (d) reviews, (e) UGC (= tags and reviews combined), and (f) all representations combined. In addition, we used the user-item matrix to calculate book-to-book similarities as option (g). Sections 4.1 and 4.2 describe the two algorithms in more details as well as their effectiveness compared to traditional CF and NDR.

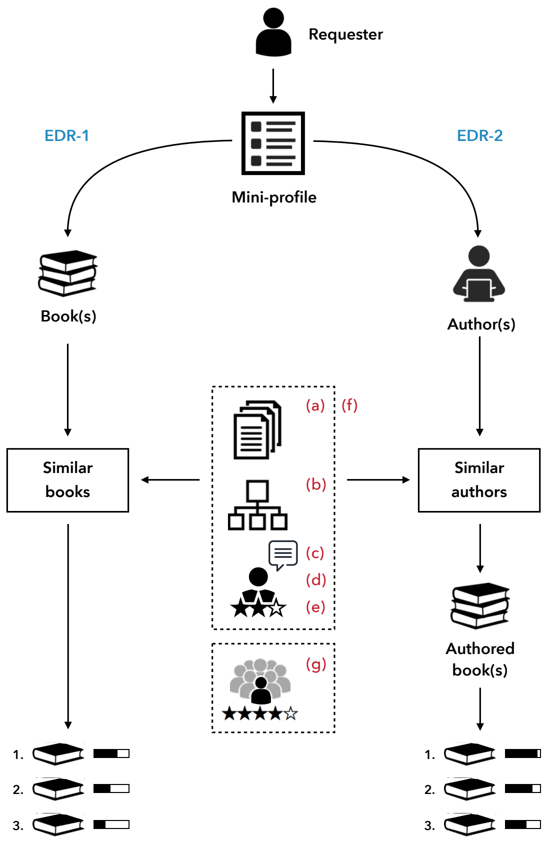

Figure 3: A schematic illustration of two variants of example-based book recommendation. EDR-1 identifies books similar to those in the mini-profile using either (a) traditional metadata, (b) curated metadata, (c) tags, (d) reviews, (e) reviews+tags, and (f) all representations combined, and (g) user preferences. EDR-2 identifies authors similar to the ones mentioned in the mini-profile and ranks their authored books by author-author similarity.

4.1 Recommendation using example books

Our first example-driven algorithm EDR-1 uses the example books in the mini-profiles and is visualized on the left-hand side of Figure 3. As mentioned in Section 2, we do not distinguish between positive and negative examples, but consider all example books as relevant sources of recommendation. The EDR-1 algorithm consists of two basic steps: (1) identifying unseen books that are similar to each of the example books, and (2) merging these sets of similar books to get a single ranked list of book recommendations.

(1) Identifying similar books. In the first step of EDR-1 for each example book mentioned in a mini-profile, we can identify similar books using one of two data sources: metadata representations of each book, or item-to-item similarities calculated on the user-book matrix. These roughly correspond to the difference between CBF and CF. With regard to metadata-based matching, the metadata representations of each example book are matched against the 2.6 million book representations in the A/LT collection. Matching can be done for each of the six representation collections (a)-(f) introduced in Section 2. Each example book representation is converted into an Indri-safe query and matched against each of the collection's book representations, producing ranked lists of book recommendations. Since the collection is indexed at the level of individual ISBNs, the same book can be represented by multiple XML representations, one for each ISBN. In each case, we pick the representation that contains the largest amount of text as the canonical representation of a book. Not every LT work could be mapped to an ISBN identifier; 52 out of 740 (7.0%) of all of our mini-profile example books could not be mapped. This means that for requests without mappable books no recommendations could be generated.

Another issue is that for the richer collections, such as (f) with all metadata fields combined, these book representations quickly become too long for efficient document retrieval. For instance, the median word count of book records for the (f) representation is 27,988 words, while the longest representation contains 210,790 words. Query-document matching in IR toolkits like Indri is optimized for relatively short queries, so to speed up matching against 2.6 million book representations, we capped each book representation at an empirically determined upper limit of the 500 most frequent words in each representation.

To identify similar books based on user preferences we calculated item-to-item similarities from the user-book matrix using the CF model trained with LightFm, as described in Section 2.4.

(2) Merging sets of similar books. Of the 298 requests with example books, 120 requests (40.3%) contain only a single example book. In these cases the ranked list of relevant book representations directly corresponds to the list of book recommendations after being mapped back from ISBNs to LT works. The remaining 59.7% contain multiple example books, which means they yield multiple results lists per request, with many books occurring on several lists. We merge these lists using the CombSUM fusion method introduced by Fox and Shaw [15], which has been shown to be effective for fusing recommendation runs [7]. CombSUM sums the normalized

similarity scores for the same book across different results lists, corresponding to a linear combination with equal weights. Before applying CombSUM, we score-normalize each results list according to the formula $sim_{norm} = \frac{sim_{original} - sim_{min}}{sim_{max} - sim_{min}}$, where *original*, *max*, *min*, and *norm* represent the original, highest, lowest and normalized similarity scores respectively.

One parameter that could influence recommendation accuracy is the number of similar books to include from each results list. Including a larger number of results—the top 500 vs. the top 10—is likely to result in more books occurring in multiple lists. Another perspective is to see this as the number of nearest (book) neighbors to recommend. We denote this parameter as k and optimized its value on our training set of 166 requests, separately for each of our seven collection representations (a)-(g). We varied the value of k in steps of 50 up to $k = 1000$. The line chart on the left-hand side of Figure 4 shows how varying k influences recommendation performance. Overall, performance tends to be fairly stable with regard to k. The optimal values of k for EDR-1 are included in Table 2.

4.2 Recommendation using example authors

Our second example-driven algorithm EDR-2 uses the example authors in requests, and is shown on the right-hand side of Figure 3. It consists of three stages: (1) identifying authors that are similar to each of the example authors; (2) merging these sets of similar authors to get a single ranked list of similar authors; and (3) replacing each similar author by their authored books and merging these to get a single ranked list of book recommendations.

(1) Identifying similar authors. In the first stage of EDR-2 for each example author mentioned in a mini-profile, we identify similar authors by matching their metadata representations against those of all authors in the A/LT collection. To make this possible, we re-indexed the A/LT collection at an author-level by aggregating all of an author's book representations into a single author-level representation. For each book, we again took the associated ISBN with the largest amount of text. These book representations were then aggregated into a single author representation, one for each of the six metadata representations (a)-(f) introduced in Section 2. This resulted in an author-centric collection containing 849,578 author representations. Each example author's representation is converted into an Indri-safe query and matched against each of the collection's author representations, producing a ranked list of author recommendations. As with EDR-1, we capped each author representation at a limit of the 500 most frequent words.

(2) Merging sets of similar authors. The 121 requests with example authors contained an average of 2.0 author per request. As for EDR-1, we used CombSUM fusion to merge normalized author lists belonging to the same request. We also examined the influence of the number of similar authors k to include from each results list; the results of this optimization can be found in Because there are fewer authors than books in the A/LT collection and because expanding authors to their authored books results in a much larger number of recommended books, we varied k for EDR-2 in smaller step sizes. The line chart on the right-hand side of Figure 4 shows how k influences performance. There are more pronounced peaks for EDR-2 than for EDR-1, but , performance tends to be fairly stable with regard to k. The optimal values of k for EDR-2 are also included in Table 2.

(3) Inserting & merging authored books. To arrive at lists of recommended books, we expanded each recommended author by their authored books and assigned each book the similarity score of that author. Because of errors in mapping ISBNs to work IDs, some books occurred multiple times for the same request; we again summed all of a book's individual scores to produce a single list of recommended books, capped at 1000 results.

4.3 Results & Analysis

Table 2 shows the results of EDR-1 and EDR-2, with scores over all test topics and over only those that have example books and example authors respectively. EDR-1 is less effective than NDR but more than EDR-2, with UGC again more effective than other metadata. This is possibly due to requests having fewer authors than books or that (especially prolific) authors introduce more topic drift. EDR-2 is more effective than CF for requests with example authors when using UGC.

5 HYBRID RECOMMENDATION

In this section we look at hybrid systems that combine NDR and EDR. More specifically, we apply results fusion, where the outputs of different recommendation algorithms are combined into a single results list, which is a form of weighted hybridization according to the hybrid recommendation taxonomy by Burke [8]. This results fusion is done by first score-normalizing the results of the individual systems and then re-ranking results using a weighted combination of the prediction scores, i.e., a linear combination or weighted CombSUM [15]. We range λ weights of runs between 0.1 and 0.9 where combined run weights sum to 1.

Table 3 contains the results of the hybrid recommenders, which we compare against the best single approach from the previous sections, i.e. the NDR (g) system which uses UGC. We only report the best performing weights. We discuss the results with a per-request analysis to explain the differences. Combining NDR and EDR-1 (f) results in a significant improvement in nDCG@10, although not in MRR. The narratives and examples complement each other as representations of the information need. The combination increases the number of requests with non-zero scores from 55 to 68, and also improves more request scores than it hurts.

Combining NDR with EDR-1 with user preferences ((g)) strongly improves precision for 20 requests, but slightly hurts it for 35 requests, which is why scores are higher, but not significantly so. Again, user preferences are good for early precision, but beyond that the signal gets so weak that popular items pollute the ranking. The combination with EDR-2 (f) run leads to only 17 requests having a different score, but significantly hurts performance. The similar author recommendations appear to cause topic drift. Combining all three approaches leads to small but statistically insignificant improvements, mainly reducing the number of requests scoring zero. Combining EDR-1 and EDR-2 leads to small improvements over the individual EDR approaches, again, mainly by reducing the number of zero-scoring requests.

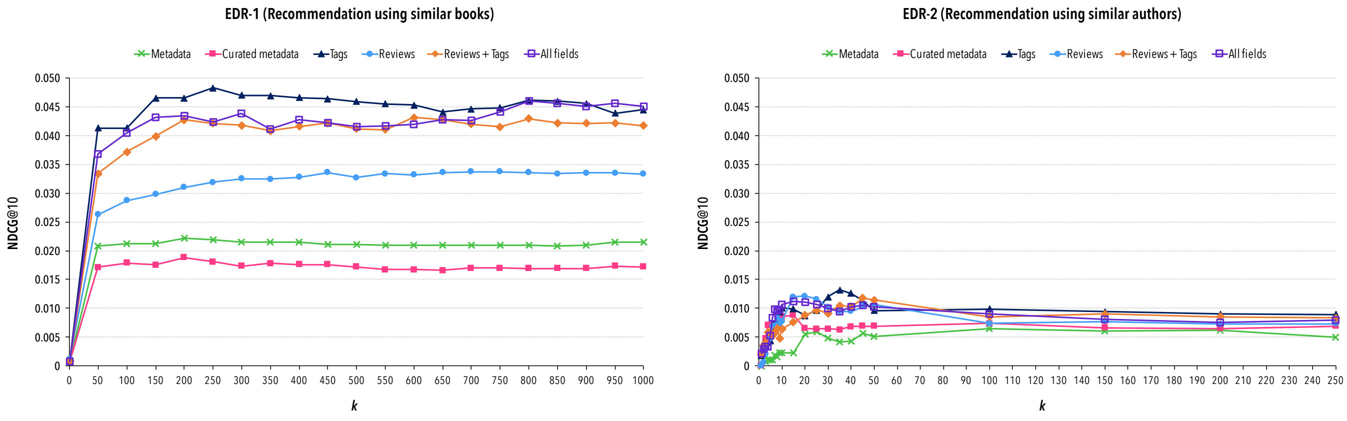

Figure 4: Effect of the number of similar books/authors k on recommendation accuracy on the training set. The figure on the left shows the influence of k on EDR-1; the figure on the right shows the influence of k on EDR-2.

Table 2: Results for the EDR-1 and EDR-2 algorithms on the test set (N = 165) using nDCG@10 and MRR as evaluation metrics, calculated on graded relevance judgments 0-1-8. Scores are shown both for all requests and the request sets that contain example books/authors. The column labeled k contain the optimized number of similar books/authors used in the recommendations. Best runs for each algorithm are printed in bold. Significant differences indicated for $p < 0.05$ (better $^{\triangle}$ or worse $^{\triangledown}$) and $p < 0.01$ (better $^{\blacktriangle}$ or worse $^{\blacktriangledown}$) as compared to the CF baseline.

	EDR-1					EDR-2				
		All requests		Example books			All requests		Example authors	
Collection	k	**nDCG@10**	**MRR**	**nDCG@10**	**MRR**	k	**nDCG@10**	**MRR**	**nDCG@10**	**MRR**
(a) Metadata	200	0.009	0.027	0.010	0.029	100	0.003$^{\triangledown}$	0.016	0.009	0.045
(b) Curated metadata	250	0.018	0.035	0.020	0.039	15	0.004$^{\triangledown}$	0.017	0.012	0.047
(c) Tags	400	0.045$^{\blacktriangle}$	0.091$^{\blacktriangle}$	0.050$^{\blacktriangle}$	0.100$^{\blacktriangle}$	30	0.005$^{\triangledown}$	0.017	0.014	0.049$^{\triangle}$
(d) Reviews	200	0.033$^{\triangle}$	0.079$^{\blacktriangle}$	0.036	0.086$^{\blacktriangle}$	15	0.006	0.016	0.016	0.045
(e) Reviews + Tags	200	0.044$^{\blacktriangle}$	0.107$^{\blacktriangle}$	0.048$^{\blacktriangle}$	0.117$^{\blacktriangle}$	45	0.006$^{\triangledown}$	0.017	0.017$^{\triangle}$	0.050$^{\triangle}$
(f) All fields	800	**0.046**$^{\blacktriangle}$	**0.113**$^{\blacktriangle}$	**0.051**$^{\blacktriangle}$	**0.123**$^{\blacktriangle}$	15	0.006	0.022	**0.018**	**0.062**$^{\triangle}$
(g) User preferences	10	0.040$^{\blacktriangle}$	0.101$^{\blacktriangle}$	0.047$^{\triangle}$	0.108$^{\blacktriangle}$	-	-	-	-	-
Collaborative filtering	-	0.017	0.031	0.019	0.035	-	**0.017**	**0.031**	0.007	0.020

Overall, the hybrids exploit the complementarity of the data sources, leading to more requests with relevant recommendations, although it is hard to improve over the competitive NDR baseline. However, our experiments do show that combining recommendation based on the narrative and the examples can lead to significantly improved performance.

6 RELATED WORK

Our work on book recommendation is far from the first: numerous book recommenders have been proposed over the past years. Most of the related work has focused on exploring and comparing CF, CBF and graph-based algorithms for generic book ranking [10, 27, 31, 33, 39, 40]. At least two shared tasks have focused on book recommendation: the Semantic Web Evaluation Challenge [12] and the Social Book Search lab [21].

Our focus on expressing recommendation needs through textual narratives is reminiscent of the work by Adomavicius et al. [1, 2] on query-driven recommendation. They proposed REQUEST, a structured query language for customizing recommendations, which can be used to tailor recommendation needs beyond the traditional "give me items I would like" task. However, it does not allow for purely textual representations of recommendation needs. Hariri et al. [18] introduced a query-driven, context-aware recommender system that provides recommendations based on a user's preferences and can be adapted to a given context that represents the short-term interests or needs of a user. Drenner et al. [14] perform movie linking between a movie recommendation Web site and a movie-oriented discussion forum. Through automatic detection and an interactive component, the system recognizes references to movies in the forum and adds recommendation data to the forums and conversation threads to movie pages.

Narrative descriptions of needs and interests are typical of conversational recommendation, where one person describes the kind of items they like and what they would be interested in, while

Table 3: Results for hybrid recommendation based on weighted fusion of narrative-driven recommendation (NDR), example-driven recommendation based on books (EDR-1) and authors (EDR-2), with $\lambda_{1,2,3}$ determining fusion weights. Calculated on graded relevance judgments 0-1-8 on the test set ($N = 165$). Best runs are printed in bold. Significant differences indicated for $p < 0.05$ (better $^{\triangle}$ or worse $^{\triangledown}$) and $p < 0.01$ (better $^{\blacktriangle}$ or worse $^{\blacktriangledown}$) as compared to the best NDR run (e) with reviews and tags combined.

Collection	$\lambda_{1,2,3}$	nDCG@10	MRR
Best single run: NDR (e)	-	0.074	0.181
EDR-1 (f) + EDR-1 (g)	.9/.1	0.051$^{\blacktriangledown}$	0.117$^{\blacktriangledown}$
EDR-1 (f) + EDR-2 (f)	.8/.2	0.047$^{\blacktriangledown}$	0.117$^{\blacktriangledown}$
NDR (e) + EDR-1 (f)	.9/.1	**0.088**$^{\triangle}$	0.177
NDR (e) + EDR-1 (g)	.9/.1	**0.079**	0.193
NDR (e) + EDR-2 (f)	.9/.1	0.065$^{\triangledown}$	0.162
NDR (e) + EDR-1 (f) + EDR-1 (g)	.8/.1/.1	0.081	0.200
NDR (e) + EDR-1 (f) + EDR-2 (f)	.8/.1/.1	0.082	0.170

others provide suggestions and explanations. [20] analysed 498 movie recommendation conversations with a chatbot and found that such conversations elicit many complex aspects of recommendation needs. As such, NDR is related to conversational and critiquing-based recommender systems, which aim to elicit more information about a user's recommendation needs through interaction and dialog [9, 11, 29, 30, 34, 38].

The narrative requests we use for recommendation also bear similarities to online product reviews in terms of their composition and complexity. Reviews cover different aspects of a product with the author describing likes and dislikes. The overall review score of a product can be seen as personal ranking of the importance of aspects. O'Mahony and Smyth [32] investigated ways of recommending reviews that offer contrasting views to help users make informed choices. Dong et al. [13] extracted topical and sentiment information from reviews to identify the most informative reviews.

Our work on example-driven recommendation also has a few parallels with earlier work. For instance, Liu et al. [28] use seed items for cold-start recommendation, which is similar to our use of mini-profiles. Schnabel et al. [36] studied shortlists as an interface component for recommender systems, where users iteratively build up a list of candidate items to consume, and found that they support the user's decision process, and the elicited implicit feedback can increase recommendation quality. The main difference with the NDR scenario is that the items in our mini-profiles are not candidate items for final recommendation, but examples to illustrate the recommendation need. However, the interface and system design proposed by Schnabel et al. [36] would support both scenarios. Our LT ratings data was sparse with many of the transactions lacking ratings. The work by Sharma et al. [37] suggests a possible way of remedying this by modeling the mini-profile as a set of ratings by using any ratings available and predicting the missing ratings in the set. Sharma et al. asked Movielens users to give a single rating for sets of items (movies) that they had previously rated individually and investigated techniques for predicting item ratings based on the user's set ratings and ratings of other items not in the set.

Our focus on complex recommendation mirrors the increased focus in IR on complex search tasks and how best to support them [17, 24]. Some of these tasks are located on the–not necessarily clear—edge between search and recommendation [23]. Complex narratives are commonly used in IR evaluation and test collection building to guide assessors [19]. They have been also used in interactive IR [16] to study how users perform complex search tasks. The Social Book Search campaigns at INEX [22] and CLEF [21] found complex, narrative-focused information needs to be common in online book discussion forums, such as GoodReads and LibraryThing. We build on their work in this paper. However, Koolen et al. [25] found that the narratives written to assess artificially constructed topics for IR evaluation are of a different nature than the narratives that users write when asking peers for recommendations. In addition, the choices that users make are very different from traditional relevance judgments used in IR.

7 DISCUSSION & CONCLUSIONS

In this paper we explored techniques to address narrative-driven recommendation needs. This is a novel, complex scenario in which users have specific recommendation needs that they express in a request containing both a natural language description as well as a mini-profile containing relevant example of books and authors. The combination of these complex user needs and the richness and heterogeneity of the item descriptions poses many challenges for designing recommender systems that support this task.

For the narrative part of the requests, we experimented with CBF techniques on a range of book metadata and user-generated content. We found that the latter is more effective as it more closely matches the vocabulary of the user's narrative and strikes a balance between popularity and specificity. For example-driven recommendation (EDR) on the mini-profiles we experimented with both CBF and CF approaches, and found that example books offer more focus and thereby better recommendations than example authors. Again, UGC is more effective than curated metadata and CF-based item-to-item recommendation. Finally, we looked at hybrid systems that combine NDR and EDR and found that NDR uses the more important signal, but that carefully weighted combinations can lead to significant improvements, especially in providing more requests with at least some relevant recommendations.

Overall we found that NDR is a challenging task that requires multiple data sources and algorithms to solve. Nevertheless, there are several issues that future work should address. One of them is the detection of and differentiation between positive and negative examples using sentiment analysis techniques; currently all examples are treated as positive examples, but this likely affects performance negatively. Other algorithms, such as those from work on conversational recommendation, or graph-based algorithms on multi-partite networks containing user, book, author and tag nodes could perhaps provide a more integrated approach to EDR. Finally, testing our algorithms on other domains, such as movies, games and music would be required to determine the generalizability of our findings.

REFERENCES

[1] Gediminas Adomavicius, Alexander Tuzhilin, and Rong Zheng. 2005. RQL: A Query Language For Recommender Systems. (2005). http://hdl.handle.net/2451/14109

[2] Gediminas Adomavicius, Alexander Tuzhilin, and Rong Zheng. 2010. REQUEST: A Query Language for Customizing Recommendations. *Information Systems Research* 22, 1 (2010).

[3] Thomas Beckers, Norbert Fuhr, Nils Pharo, Ragnar Nordlie, and Khairun Nisa Fachry. 2010. Overview and results of the inex 2009 interactive track. In *International Conference on Theory and Practice of Digital Libraries*. Springer, 409–412.

[4] Toine Bogers and Marijn Koolen. 2017. Defining and Supporting Narrative-driven Recommendation. In *RecSys '17: Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 238–242.

[5] Toine Bogers and Vivien Petras. 2015. Tagging vs. Controlled Vocabulary: Which is More Helpful for Book Search?. In *Proceedings of iConference 2015*. iDEALS.

[6] Toine Bogers and Vivien Petras. 2017. An In-depth Analysis of Tags and Controlled Vocabulary for Book Search. In *Proceedings of iConference 2017*. iDEALS.

[7] Toine Bogers and Antal Van Den Bosch. 2011. Fusing Recommendations for Social Bookmarking Websites. *International Journal of Electronic Commerce* 15, 3 (2011), 31–72.

[8] Robin Burke. 2002. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction* 12, 4 (2002), 331–370.

[9] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. 2016. Towards Conversational Recommender Systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 815–824.

[10] Maarten Clements, Arjen P. de Vries, and Marcel J.T. Reinders. 2010. The Influence of Personalization on Tag Query Length in Social Media Search. *Information Processing & Management* 46, 4 (2010), 403–412.

[11] Jeffrey Dalton, Victor Ajayi, and Richard Main. 2018. Vote Goat: Conversational Movie Recommendation. In *SIGIR '18: Proceedings of the 41st International ACM SIGIR Conference on Research &#38; Development in Information Retrieval*. 1285–1288.

[12] Tommaso Di Noia, Iván Cantador, and Vito Claudio Ostuni. 2014. Linked open data-enabled recommender systems: ESWC 2014 challenge on book recommendation. In *Semantic Web Evaluation Challenge*. Springer, 129–143.

[13] Ruihai Dong, Markus Schaal, Michael P O'Mahony, and Barry Smyth. 2013. Topic Extraction from Online Reviews for Classification and Recommendation. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*. AAAI Press, 1310–1316.

[14] Sara Drenner, Max Harper, Dan Frankowski, John Riedl, and Loren Terveen. 2006. Insert Movie Reference Here: A System to Bridge Conversation and Item-oriented Web Sites. In *CHI '06: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 951–954.

[15] Edward A. Fox and Joseph A. Shaw. 1994. Combination of Multiple Searches. In *TREC-2 Working Notes*. 243–252.

[16] Maria Gäde, Mark Michael Hall, Hugo C. Huurdeman, Jaap Kamps, Marijn Koolen, Mette Skov, Toine Bogers, and David Walsh. 2016. Overview of the SBS 2016 Interactive Track. In *Working Notes of CLEF 2016 - Conference and Labs of the Evaluation forum, Évora, Portugal, 5-8 September, 2016*. 1024–1038.

[17] Maria Gäde, Mark Michael Hall, Hugo C. Huurdeman, Jaap Kamps, Marijn Koolen, Mette Skov, Elaine Toms, and David Walsh. 2015. First Workshop on Supporting Complex Search Tasks. In *Proceedings of the First International Workshop on Supporting Complex Search Tasks co-located with the 37th European Conference on Information Retrieval (ECIR 2015), Vienna, Austria, March 29, 2015.*

[18] Negar Hariri, Bamshad Mobasher, and Robin Burke. 2013. Query-driven Context-aware Recommendation. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 9–16.

[19] Donna Harman and Ellen M. Voorhees. 2006. TREC: An Overview. *ARIST* 40, 1 (2006), 113–155.

[20] Jie Kang, Kyle Condiff, Shuo Chang, Joseph A. Konstan, Loren Terveen, and F. Maxwell Harper. 2017. Understanding How People Use Natural Language to Ask for Recommendations. In *RecSys '17: Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 229–237.

[21] Marijn Koolen, Toine Bogers, Maria Gäde, Mark M. Hall, Iris Hendrickx, Hugo C. Huurdeman, Jaap Kamps, Mette Skov, Suzan Verberne, and David Walsh. 2016. Overview of the CLEF 2016 Social Book Search Lab. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 7th International Conference of the CLEF Association, CLEF 2016, Évora, Portugal, September 5-8, 2016, Proceedings*. 351–370.

[22] Marijn Koolen, Toine Bogers, Jaap Kamps, Gabriella Kazai, and Michael Preminger. 2014. Overview of the INEX 2014 Social Book Search Track. In *Working Notes for CLEF 2014 Conference, Sheffield, UK, September 15-18, 2014*. 462–479.

[23] Marijn Koolen, Toine Bogers, Jaap Kamps, and Antal van den Bosch. 2015. Looking for Books in Social Media: An Analysis of Complex Search Requests. In *ECIR '15: Proceedings of the 37th European Conference on Information Retrieval (Lecture Notes in Computer Science)*, Vol. 9022. Springer, 184–196.

[24] Marijn Koolen, Jaap Kamps, Toine Bogers, Nicholas J. Belkin, Diane Kelly, and Emine Yilmaz. 2017. Current Research in Supporting Complex Search Tasks. In *Proceedings of the Second Workshop on Supporting Complex Search Tasks co-located with the ACM SIGIR Conference on Human Information Interaction & Retrieval (CHIIR 2017), Oslo, Norway, March 11, 2017*. 1–4.

[25] Marijn Koolen, Jaap Kamps, and Gabriella Kazai. 2012. Social Book Search: Comparing Topical Relevance Judgements and Book Suggestions for Evaluation. In *CIKM '12: Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, Xue-wen Chen, Guy Lebanon, Haixun Wang, and Mohammed J. Zaki (Eds.). ACM, 185–194.

[26] Maciej Kula. 2015. Metadata Embeddings for User and Item Cold-start Recommendations. In *Proceedings of the 2nd Workshop on New Trends on Content-Based Recommender Systems co-located with 9th ACM Conference on Recommender Systems (RecSys 2015), Vienna, Austria, September 16-20, 2015. (CEUR Workshop Proceedings)*, Toine Bogers and Marijn Koolen (Eds.), Vol. 1448. CEUR-WS.org, 14–21.

[27] Dong Kun. 2012. Research of personalized book recommender system of university library based on collaborative filter. *Data Analysis and Knowledge Discovery* 11 (2012), 44–47.

[28] Qi Liu, Biao Xiang, Enhong Chen, Yong Ge, Hui Xiong, Tengfei Bao, and Yi Zheng. 2012. Influential Seed Items Recommendation. In *RecSys '12: Proceedings of the Sixth ACM Conference on Recommender Systems*. ACM, 245–248.

[29] Tariq Mahmood and Francesco Ricci. 2009. Improving Recommender Systems with Adaptive Conversational Strategies. In *Proceedings of the 20th ACM Conference on Hypertext and Hypermedia*. ACM, 73–82.

[30] Lorraine McGinty and James Reilly. 2011. On the Evolution of Critiquing Recommenders. In *Recommender Systems Handbook*, Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor (Eds.). Springer, 419–453.

[31] Raymond J. Mooney and Loriene Roy. 2000. Content-based Book Recommending using Learning for Text Categorization. In *DL '00: Proceedings of the Fifth ACM Conference on Digital Libraries*. 195–204.

[32] Michael P O'Mahony and Barry Smyth. 2010. A Classification-based Review Recommender. *Knowledge-Based Systems* 23, 4 (2010), 323–329.

[33] Maria Soledad Pera and Yiu-Kai Ng. 2012. BReK12: a book recommender for K-12 users. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1037–1038.

[34] Rachael Rafter and Barry Smyth. 2005. Conversational Collaborative Recommendation–An Experimental Analysis. *Artificial Intelligence Review* 24, 3-4 (2005), 301–318.

[35] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[36] Tobias Schnabel, Paul N. Bennett, Susan T. Dumais, and Thorsten Joachims. 2016. Using Shortlists to Support Decision Making and Improve Recommender System Performance. In *WWW '16: Proceedings of the 25th International Conference on World Wide Web*. 987–997.

[37] Mohit Sharma, F. Maxwell Harper, and George Karypis. 2017. Learning from Sets of Items in Recommender Systems. In *eKNOW 2017: Proceedings of the Ninth International Conference on Information, Process, and Knowledge Management*. 59–64.

[38] Yueming Sun and Yi Zhang. 2018. Conversational Recommender System. In *SIGIR '18: Proceedings of the 41st International ACM SIGIR Conference on Research &#38; Development in Information Retrieval*. 235–244.

[39] Anand Shanker Tewari, Abhay Kumar, and Asim Gopal Barman. 2014. Book recommendation system based on combine features of content based filtering, collaborative filtering and association rule mining. In *Advance Computing Conference (IACC), 2014 IEEE International*. IEEE, 500–503.

[40] Paula Cristina Vaz, David Martins de Matos, Bruno Martins, and Pavel Calado. 2012. Improving a hybrid literary book recommendation system through author ranking. In *Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries*. ACM, 387–388.

[41] Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, Vol. 11. 2764–2770.

[42] Jason Weston, Hector Yee, and Ron J Weiss. 2013. Learning to rank recommendations with the k-order statistic loss. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 245–248.