From Recommendation to Curation

When the system becomes your personal docent

Nevena Dragovic[*]
MarkMonitor
Boise, Idaho, USA
nevena.dragovic@markmonitor.com

Ion Madrazo Azpiazu
People & Information Research Team
Boise State University
Boise, Idaho, USA
ionmadrazo@boisestate.edu

Maria Soledad Pera
People & Information Research Team
Boise State University
Boise, Idaho, USA
solepera@boisestate.edu

ABSTRACT

Curation is the act of selecting, organizing, and presenting content. Some applications emulate this process by turning users into curators, while others use recommenders to select items, seldom achieving the focus or selectivity of human curators. We bridge this gap with a recommendation strategy that more closely mimics the objectives of human curators. We consider multiple data sources to enhance the recommendation process, as well as the quality and diversity of the provided suggestions. Further, we pair each suggestion with an explanation that showcases why a book was recommended with the aim of easing the decision making process for the user. Empirical studies using Social Book Search data demonstrate the effectiveness of the proposed methodology.

CCS CONCEPTS

• **Information systems → Recommender systems**;

KEYWORDS

Curation, Decision-making, Time series, Personalization, Diversity

ACM Reference Format:
Nevena Dragovic, Ion Madrazo Azpiazu, and Maria Soledad Pera. 2018. From Recommendation to Curation: When the system becomes your personal docent. In *Proceedings of Joint Workshop on Interfaces and Human Decision Making for Recommender Systems (IntRS) 2018 (IntRS Workshop).* ACM, 8 pages.

1 INTRODUCTION

Recommenders have been studied for decades. Regardless of the domain, they influence businesses' success and users' satisfaction. From a commercial standpoint, recommenders enable companies to advertise items to potential buyers. From a user perspective, they enhance users' experience by easing identification of items of interests while addressing information overload concerns. The degrees of personalization recommenders offer, however, can be hindered by their limited ability to provide diverse enough suggestions, restricting users' exposure to new, prospective items of interest. This occurs because common recommendation strategies rely on community data, thus suggesting the same items to similar users, which can be vague and impersonal.

Inspired by the results of the work conducted by Willemsen et al. [45], who demonstrated that "diverse, small item sets are just as satisfying and less effortful to choose from than Top-N recommendations," we argue that emulating the *curation* process—the act of selecting, organizing, and presenting content [10]—to suggest a small set of diverse items could lead to an enriched recommendation process. In this paper, we present the algorithmic foundation to make this possible, and facilitate future user studies. A number of applications, e.g. Pinterest, let the user be the decision maker: they offer individualized selections and then allow the user play the role of a curator in choosing appealing items. However, given the ability of a recommender to collect and examine large amounts of data about users and items, the system itself can get to know users better—their interests and behaviors— and act as a curator.

Due to scope limitations, we use **books** as a case study and focus our research efforts on the techniques that lead to *QBook*, a curated book recommender (Figure 1). *QBook* does not only find books that are appealing to a user, but also presents a meaningful set of suggestions with corresponding explanations that pertain to the various preferences of the user. *QBook* takes the role of the curator upon itself; make connections between suggested items and the reasons for their selection; and enriches the recommendation process by addressing issues that affect these systems: **(1)** Using *historical data*, we capture suitable candidate items for each user; **(2)** Understanding *items' metadata*, we access potentially relevant items that otherwise might be ignored by solely relying on ratings; **(3)** Considering *user reviews*, we infer which item features each user cares about and their degree of importance; **(4)** Examining *experts' reviews*, we ensure overall quality of books to be recommended; **(5)** Exploring the *type of data sources a user favors*, we learn about the user and understand why he could be interested in an item;**(6)** Analyzing users' change in *genre preferences over time*, we better identify the current reading interests of individual users.

QBook can be seen as an *interpretable diversification strategy* for recommendation, where evidence factors from (1)-(6) are combined using a diversification technique adapted to each user's interest. Further, explanations on why each book was selected are provided so that the user can better select the book he is interested in.

With this work, we improve research related to recommenders by combining traditional approaches with novel preference matching methods into a single strategy that offers suggestions containing information related to the interests of each user. Moreover, we explicitly undertake *diversity* and *personalization*–key aspects as common collaborative filtering algorithms are known to not propagate users' preferences on diversity into their recommendations [8]—by exploring user reviews and time analysis to understand change of reading preference in time. To assess the effectivenesses of *QBook* we conduct experiments measuring the utility of individual components as well as comparing the recommendation quality of the system as a whole with respect to state-of-the-art systems.

[*]Work conducted while the author was a student at Boise State.

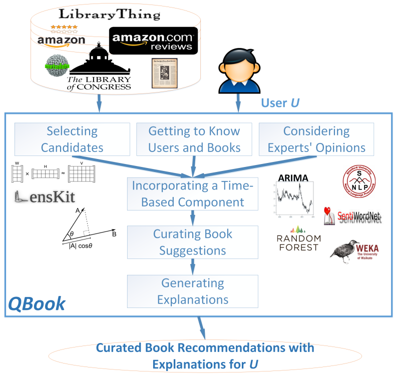
LibraryThing
amazon
amazon.com reviews
The LIBRARY of CONGRESS
User U
Selecting Candidates
Getting to Know Users and Books
Considering Experts' Opinions
LensKit
Incorporating a Time-Based Component
ARIMA
NLP
SentiWordNet
Curating Book Suggestions
RANDOM FOREST
WEKA
Generating Explanations
QBook
Curated Book Recommendations with Explanations for U

Figure 1: Overview of *QBook*

2 RELATED WORK

We discuss work pertaining to book recommenders, as well as the use of explanations and curation for recommendation purposes. **Recommenders & Books**. A number of recommenders have been designed to generate suggestions that help users select suitable books to read [4, 17, 34]. They are based on purchasing or rating patterns, click-through data, content/tag analysis, or ontologies.

Some book recommenders are tailored towards specific group of users: By emulating the readers' advisory service, the authors in [34] describe the process of recommending books for K-12 students, based on the topics, contents, and literary elements that appeal to each individual user, whereas K3Rec [35] uses information about grade levels, contents, illustrations, and topics together with length and writing style, to generate suggestions for emergent readers. Garrido and Illarri [14] rely on content-based data for making book suggestions. Their proposed TMR [14] examines items' descriptions and reviews and relies on lexical and semantic resources to infer users' preferences. However, TMR can only work if descriptions and reviews are available, unlike *QBook*, for which these are only two of the multiple data points considered in the recommendation process. The authors in [4] present a strategy based on graph analysis and PageRank that exploits clicks, purchasing patterns, and book metadata. This strategy is constrained to the existence of a priori pairwise similarity between items, e.g. "similar products", which is not a requirement for *QBook*. The authors in [38] highlight the importance of book recommenders as library services, and thus propose a fuzzy analytical hierarchy process based on a priori rules mining that depends upon the existence of book-loan histories. The empirical analysis presented in [38] is based on a limited and private dataset, which constrains the task of verifying its applicability on large-scale benchmark datasets. More importantly, the proposed strategy is contingent on book-loan historical information that due to privacy concerns libraries rarely, if at all, make available.

Recommenders & Explanations. An ongoing challenge faced by recommenders is to get users to trust them, as they still operate as "black boxes" [18]. A powerful way to build successful relationships between users and recommenders is by providing information on how each system works and why items are suggested [39]. This can be accomplished by including explanations that justify the suggestions, which are known to provide transparency and enhance trust on the system [39]. Unfortunately, justifying the reasons why an item has been suggested to a user is not an easy task [16]. Recent works in this area focus on determining how different types of explanations influence users while making decisions [16]. Among the most common strategies we should highlight those based on exploring previous activity of a user [5], information collected from user reviews [46], and content-based tag cloud explanations. The goal of *QBook* is to provide explanations that reveal reasoning and data behind the recommendation process and contain other users' and experts' (objective) opinions on item characteristics that are of a specific interest for each individual user. Many researchers consider sentiment-based explanations as more effective, trustworthy, and persuasive than the ones that capture relationship between previous activity of the user and suggested items [9]. We, however, follow the premise presented in [12] and do not include sentiment in order to make *QBook* look unbiased from users' perspectives, i.e., we do not select feature descriptions based on their polarity.

Recommenders & Curation. Few research works focus on simulating the curation process for recommendation purposes [21, 22, 37]. In [21], the authors discussed the development of a news application that learns from users' interactions with the system while they swipe through provided news articles and like them. In this research, the authors use social networks and users' browsing history to create and recommend crowd curated content, but using users as curators. The work conducted by Saaya et al. [37], on the other hand, relies on a content-based strategy that considers information authors collect from users' profiles, which are then managed by the users themselves. The most recent study conducted by Kislyuk at al. [22], combines a user-based curation method along with a traditional collaborative filtering strategy to improve Pinterest's recommendation process. The aforementioned alternatives simply let users organize suggested content based on their personal preferences, since all three studies treat users as curators. However, no recommendation system takes the role of the curator upon itself. We take a different approach and allow the system to take the curator role using existing user and item data.

3 THE ANATOMY OF QBOOK

Each step of *QBook*'s recommendation process addresses a particular research problem on its own: Can item metadata complement the lack of historical data (and vice versa)? How can a time component influence recommendation systems?, Can experts' opinions align with readers' preferences?, Are users' features of interests a valuable asset to a recommendation process?, How does curation work as a part of a recommendation process?, Can personalized explanation aid users in selecting relevant suggestions?.

3.1 Selecting Candidate Books

To initiate the recommendation process (described in Algorithm 1), *QBook* identifies a set of books *CB* from a given collection to be curated for a user *U*. The two strategies considered for candidate selection (i.e., matrix factorization and content-based filtering) complement each other and ensure diversity among candidate books. While the former examines users' rating patterns, the latter focuses on books characteristics and does not require user-generated data. Moreover, by considering rating patterns and content, novelty of the recommendations increases, as users are exposed to a variety of books to chose from. We anticipate *QBook* to handle data sparsity and cold start in this step, since even if candidate books do not

Algorithm 1 The Recommendation Process of *QBook*

```
Input: AB-Archived set of books, RS-Set of reviews, ER-Set of expert Reviews, K-# of recommendations, RF-Trained Random Forest
Terms: $RS_U$-Reviews in RS written by U, $RS_b$-Reviews in RS for b, $ER_b$-Reviews in ER for b, $P_U$-Set of books read by U
CandidateSet, Recommendations = empty set
Count=0
for each user U do
    UPref=Ranked list of preferred literary elements using $RS_U$
    CandidateSet=b ∈ $AB$ with $r_{U,b}$ > 3 OR $Cr_{U,b}$ > 3
    for each book $b$ in CandidateSet do
        BPref=Ranked list of preferred literary elements using $RS_b$
        Sim= Similarity(UPref and BPref)
        sWNr=Polarity($ER_b$, SentiWordNet)
        sWNs=Polarity(lastSentence, $ER_b$, SentiWordNet)
        cNLPr=Polarity($ER_b$, CoreNLP)
        cNLPs=Polarity(lastSentence, $ER_b$, CoreNLP)
        $A_{\vec{U,b}}$=< $r_{U,b}$, $Cr_{U,b}$, Sim, sWNr, sWNs, cNLPr, cNLPr>
        AppealScore=GenerateRanking($RF$, $A_{\vec{U,b}}$)
        Recommendations=Recommendations+<b, AppealScore>
    end for
    if $U$ is active then
        DP=Identify most-correlated data point for $U$
        GenrePref=GenreDistribution(ARIMA, $P_U$)
        Recommendations=Sort(Recommendations, DP, GenrePref, $K$)
    else
        Recommendations=Sort(Recommendations)
    end if
    for each b in Recommendations do
        if Count++ <= $K$ then
            print "b + Explanation(b, $ER_b$, $RS_b$, UPref, DP)"
        end if
    end for
end for
```

have (sufficient) ratings assigned to them, they might still have tag descriptions that can help the recommender determine if they are indeed potentially of interest to a user and vice-versa.

Matrix Factorization. To take advantage of U's historical data, *QBook* adopts a strategy based on Matrix factorization [24]. Specifically, it uses LensKit's [3] FunkSVD for candidate generation and includes in *CB* any book b for which its predicted rating for U ($r_{U,b}$) is above 3–ensuring the potential appeal of b to U.

Content Analysis. Content-based filtering methodologies create suggestions by comparing items' characteristics and users' preferences. Available content representations (e.g., metadata) are used to describe items, as well as users' profiles based on items users favored in the past [36]. *QBooks* uses tags, which capture books' content from diverse users' perspectives. Thereafter, it applies Lenskit's implementation of the content-based algorithm [1] (based on the *Vector Space Model* and *TF-IDF* weighting scheme), and includes in *CB* any book b for which its similarity with respect to U's content preferences ($Cr_{U,b}$) is 3 or above.

3.2 Getting to Know Users and Books

QBook aims to provide U with a set of appealing, personalized suggestions based on information he values. *QBook* examines reviews written by U and identifies the set of literary elements (features) that he cares the most about[1]. Thereafter, it determines the degree to which each book in *CB* addresses U's features of interest. To identify features of interest to U, *QBook* performs semantic analysis on reviews and considers the frequency of occurrence of terms U employs in his reviews. By adopting the set of literary elements and the extraction strategy described in [34], *QBook* explores features that characterize book content, such as *character* descriptions or

[1]If U does not have reviews, then the most popular user features are treated as U's features of importance.

Table 1: Sample of terms associated with literary features

Literary Element	Sample of Related Terms
characters	stereotypes, detailed, distant, dramatic
pace	fast, slow, leisurely, breakneck, compelling
storyline	action-oriented, character-centered
tone	happy, light, uplifting, dark, ironic, funny
writing style	austere, candid, classic, colorful
frame	descriptive, minimal, bleak, light-hearted

writing style. As defined in [34], each literary element (i.e., feature) is associated with a set of terms used to describe that element, since different words can be used to express similar book elements. A sample of literary elements and related terms is shown in Table 1.

QBook computes the overall frequency of occurrence of each feature mentioned by U by normalizing the occurrence of the feature based on the number of reviews written by U. This score captures the importance (i.e., weight) of each particular feature for U.

On the same manner, *QBook* examines reviews available for b following the process defined for identifying features of interest to U, in order to gain a deeper understanding of the literary elements that are often used to describe b. This is done by analyzing the subjective opinions of all users who read and reviewed b.

QBook leverages U's preferences in the recommendation process by calculating the degree of similarity between U's feature preferences and b's most-discussed features, as $Sim(U,b) = \frac{\vec{UV} \cdot \vec{BV}}{\left\|\vec{UV}\right\| \times \left\|\vec{BV}\right\|}$, where $\vec{UV} = \langle W_{F_{U,1}}, \ldots, W_{F_{U,n}} \rangle$ and $\vec{BV} = \langle W_{F_{b,1}}, \ldots, W_{F_{b,m}} \rangle$ are vector representations associated with feature discussions of U and b, n and m are numbers of distinct features describing U and b, respectively, and $W_{F_{U,i}}$ and $W_{F_{b,i}}$ capture the weight, i.e., degree of importance, of the i^{th} feature for U and b, based on their normalized frequencies of occurrence (in reviews).

By using $Sim(U,b)$, *QBook* triggers the generation of personalized suggestions, as it captures all features of interests for U and compares them with the most-discussed features of b to determine how likely b is relevant to U.

3.3 Considering Experts' Opinions

To further analyze b, *QBook* takes advantage of experts' reviews in order to consider unbiased and objective opinions as another data point in its recommendation process. Unlike the polarity-neutral strategy adopted to identify user/item features of interest, in the case of experts we explicitly examine the polarity of their opinions. By doing so *Qbook* leverages expert knowledge to ensure that recommended books are of *good quality*. *QBook* explores publicly available book critiques to determine experts' opinions on candidate books by performing semantic analysis to examine which books experts valued more. *QBook* examines *ER*, the set of expert reviews available for b, from two complementary perspectives: it captures sentiment at a word and sentence levels using two popular sentiment analysis tools. By involving experts' reviews in the recommendation process, *QBook* can help overcome the data sparsity issue, since some books do not have sufficient user-generated data, but have professional critiques which provide valuable information.

Sentiment at Word Level. SentiWordNet [13] is a lexical resource for opinion mining that assigns a sentiment score to each

WordNet synset. Using SentiWordNet, *QBook* determines *ER*'s overall sentiment, denoted $sWNr$, by calculating an average score based on the sentiment of each word in *ER*. Based on the study described in [33], and our own analysis, we observe that reviewers often summarize their overall thoughts in the last sentence of their review. For this reason, *QBook* also analyses the sentiment of the last sentence in each review in *ER* and calculates its overage score, denoted $sWNs$, to ensure the real tone of the review is captured.

Sentiment at Sentence Level. In some cases, the polarity of a word on its own does not properly capture the intended polarity of a sentence. Thus, *QBook* uses *CoreNLP* [28] which builds up a representation of whole sentences based on their structure. *QBook* applies *CoreNLP*' parser to extract sentences from *ER* and calculates a sentiment score for each respective sentence. These scores are combined into a single (average) score, denoted $cNLPr$, which captures the overall sentiment of *ER* based on the sentiment of its individual sentences. Similar to the data points extracted at word level, *QBook* also considers the average sentiment of the last sentence in each review in *ER*, denoted $cNLPs$.

3.4 Incorporating a Time-Based Component

To better serve their stakeholders, recommenders must predict readers' interest at any given time. Users' preferences, however, tend to evolve, which is why it is crucial to consider a time component to create suitable suggestions [44]. *QBook* examines *genre*, which is a category of literary composition, determined by literary technique, tone, content, or even length, from a time-sensitive standpoint. Including this component provides the likelihood of reader(s) interest in each genre based on its occurrences at a specific point in the past, not only the most recent or the most frequently read one. *QBook* uses a genre prediction strategy[2] that examines a genre distribution and applies a time series analysis model, Auto-Regressive Integrated Moving Average (*ARIMA*) [32]. In doing so, *QBook* can discover different areas of U's reading preferences along with U's degree of interest on each of them.

Predicting genre preference to inform the recommendation process involves examining genres read by U. We first obtain the genre distribution among the books read by U during continuous periods of time and estimate a significance score for each genre g_n or U at a specific time period t: *GenreImportance*=$\frac{|g_{n,t,b}|}{|G_t|}$, where G_t is the set of books read in t $|g_{n,t}|$ is the frequency of occurrence of a specific genre among books in G_t, and $|G_t|$ is a size of G_t.

Since changes in reading activity between fixed and known periods of time are not constant, *QBook* applies non-seasonal *ARIMA* models. By doing this, *QBook* is able to determine a model tailored to each genre distribution to predict its importance for U in real time based on its previous occurrences. *ARIMA* forecasting (i.e., temporal prediction) model uses a specific genre distribution to predict the likelihood of future occurrences of that genre based on its importance in previous periods of time. This is why our strategy conducts a search over possible *ARIMA* models that capture user preference and selects the one with a best fit for a specific genre distribution in time for U—the one that best describes the pattern of the time series and explains how the past affects the future.

[2]We first discussed the benefits of explicitly considering changes in user preferences over time in [11].

Using ARIMA and genre information about books read by U, *QBook* can estimate the likelihood of occurrence of a given genre g_n at time frame T_W, i.e., the recommendation time in our case. This information is used to determine the degree to which U is interested in reading each genre and subsequently the number of books in each genre that should be recommended to satisfy U's varied interests (see Section 3.5). For example, with the described time series genre prediction strategy, *QBook* is able to prioritize the recommendation of fantasy books for U (a genre U recently started reading more) over comedy books (a genre known to be favored by U in the past), even if proportionally U read more comedy than fantasy books. The described prediction approach provides an additional data point to further personalize the recommendation process.

3.5 Curating Book Suggestions

The last step of *QBook*'s recommendation process focuses on curating *CB* to generate *top-K* suggestions tailored to U. In this step, *QBook*'s goal is to emulate the curation process (as defined in [6]) and become a personal docent that understands U and provides relevant books to read that appeal to his diverse, yet unique, preferences. To do so, *QBook* simultaneously considers different data points and builds a model that creates a single score that quantifies the degree to which U prefers $b \in CB$. For model generation, *QBook* adopts the Random Forest[3] algorithm [7].

As part of the curation process, *QBook* represents $b \in CB$ as a vector $\vec{A_{U,b}} =< r_{U,b}, Cr_{U,b}, Sim(U,b), sWNr, sWNs, cNLPr, cNLPr >$ which captures the degree of appeal of b for U from multiple perspectives and is used as an input instance to the trained Random Forest to generate the corresponding ranking score for b. Note that unlike traditional recommenders that train a model for all the users in the community, in *QBook* a random forest model is trained per user. This allows the model to specifically learn each user's interests similar to what a personal docent would do.

Reading Activity. Reading activity varies among users, influencing *QBook*'s ability to manipulate ranked candidate books for curation. For *non-active* readers–who rate less than 35 books- the lack of available information can hinder the process of further personalizing suggestions. In this case, *QBook* generates the *top-K* recommendations for U by simply ordering the predictions scores obtained using the trained Random Forest model on books in *CB*.

For *active readers* (who read at least 35 books[4]), it is important to identify what motivates their reading selections, which can vary among different readers. For example, some users are biased by experts opinions while others by the preferences of similar-minded individuals. *QBook* explicitly considers these individual biases in the book selection process for each active reader, leading to further personalize suggestions. If U is an *active* reader, then *QBook* captures correlations among different data points involved in the process of creating $\vec{A_{U,b}}$ for U. *QBook* uses Pearson correlation to indicate the extent to which variables fluctuate together (as illustrated in Figure 2). By exploring U past rating behavior, *QBook* can determine the data point that has the most influence on U in

[3]Empirically verified that Random Forests are best suited for our curation task;analysis omitted due to page limitations.

[4]Analysis of recent statistics on average number of books read by Americans on a yearly basis along with examination of rating distributions on development data influenced our threshold selection for experimental purposes.

the process of ratings books, i.e., which data point yield the highest correlated value with respect to U's ratings. This data point is treated as the most important one, in terms of biasing U's decision making process. *QBook* further re-orders the scores computed for each book in *CB* based on the score of the most influential data point and thus provides *top-K* suggestions further tailored for U.

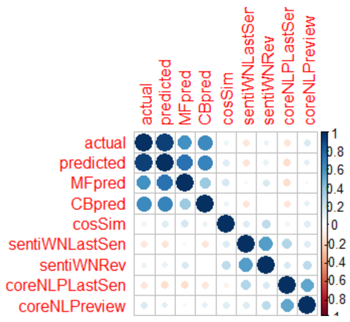

Figure 2: Correlation among data points in *QEval*; "actual" is the rating assigned by a user, "predicted" is the one estimated by *QBook*, color denotes correlation, and the size of the bubbles captures correlation strength.

Genre Preference. In the case of active readers, the final step of *QBook* for curating their suggestions involves explicitly considering U's genre preferences. This is accomplished using the strategy described in Section 3.4. To determine the number of representative candidates from each genre that should be part of the final set of books presented to U, *QBook* relies on the genre preference distribution calculated using ARIMA time series analysis and the process discussed in Section 3.4. In doing so, *QBook* can account for the degree to which U will be likely interested in reading each particular genre at the moment the recommendation is generated.

The final list of *top-K* suggestions for U are generated by considering not only ranking scores and user bias, but also by ensuring that the genre distribution of books among the K suggested match the genre distribution uniquely expected for U.

By performing this curation step, *QBook* looks for diversity among the suggestions by including books from all different areas of users' interests and improves personalization by ordering suggestions based on a specific feature for U. Consequently, *QBook* enables U to choose books from the exhibit tailored solely to him in order to satisfy his reading needs in a given time.

Generating Explanations. In order to be a curator, *QBook* can not only recommend books to read without justifying why the suggestions were generated. To do so, *QBook* pairs each recommendation with an explanation enabling U to make the most informed and fastest decisions in terms of selecting a single item among the suggested ones. To generate explanations, *QBook* uses archived book reviews, experts' reviews, and the set of steps taken to generate curated recommendations and provides U with personalized and valuable information.

QBook creates explanations for a curated book b_c by extracting sentences in reviews pertaining to b_c that refer to the most important literary element of interest to U. Note that if there are multiple sentence describing the same feature, *QBook* arbitrarily selects one to be shown to U. More importantly, *QBook* does not emphasize the sentiment of the features, since *QBook*'s intent is not to make U like one option more than another, but to save time on identifying information important for him in the process of choosing the book to read. Along with the other users' opinions, *QBook* includes in its explanations experts' opinions on the book's quality, as described in Section 3.3. In other words, *QBook* includes in the corresponding explanations a sentence from experts' reviews that also reference users' top feature of interest. This way, U is provided with objective opinions by extracting sentences from experts' reviews pertaining to the feature of U. This increases U's trust in *QBook*, since U can read unbiased opinions that help him determine if he would like to read a particular recommendation or not. For its final step for explanation generation, *QBook* looks into the steps taken in curating each book suggestion. For example, if b_c was selected based on U's rating history, then corresponding explanation includes a simple sentence of the form b_c *was chosen since it has been highly rated by users with similar rating patterns to yours*. If, instead, experts' opinion had a strong influence in the curation process, then *QBook* makes sure that the user is aware of it.

The explanation paired with b_c includes three sentences specifically selected for U. While we pick 3 for simplicity purposes, the number of sentences included in the explanation can be adjusted. By providing *personalized explanations*, *QBook* is able to *tell a story* to U about how each suggestion is related him, which increases users' trust in the system, as well as the system's transparency [40]. Unlike the majority of existing strategies, *QBook* does not act like a "black box" to the user since it provides information regarding the selection and curation of the final set of books that are suggested. Therefore, with this step *QBook* acts as a personal docent for U.

4 EVALUATION

In this section, we discuss the results of the studies conducted to validate *QBook*'s performance and design methodology.

4.1 Framework

Dataset. To the best of our knowledge, there is no benchmark that can be used for evaluating the performance of a curation recommendation systems. Instead we use resources from Social Book Search (SBS) Suggestion Track [2], which consists of 2.7 million book titles along with user reviews and ratings, each with combined book metadata from Amazon.com and LibraryThing. We complement this collection with (i) overlapping catalog records from the Library of Congress and (ii) experts' reviews from known book critiques' websites, such as NPR and Editorial Reviews SBS. We called this enhanced SBS dataset *QData*.

We split *QData* in three parts: 80% of the users were used for training, denoted *QTrain*, 10% for development, denoted *QDevel*, and the remaining 10% for evaluation, denoted *QEval*. To ensure a representative distribution for development and evaluation purposes, we first clustered users from *QData* based on number of books read. Thereafter, we created *QTrain*, *QDevel*, *QEval* by randomly selecting the same percentage of users from clusters to "simulate" real representation of *QData* in each partition.

Metrics. For recommendation validation, we used the well-known *NDCG* and *RMSE*. We also considered:

$Coverage = \frac{|K \cap R \cap A|}{|K \cap R|}$, K is the set of books of the collection known to a given user, R is the set of relevant books to a user and A

is the set of recommended books. This metric captures how many of the items from the dataset are being recommended to all users who get recommendations [15].

Novelty=$\frac{|(R \bigcap A)-K|}{|R \bigcap A|}$, where K, R, and A are defined as in Coverage. This metric captures how different a recommendation is with respect to what the user has already seen along with the relevance of the recommended item [41].

Serendipity, which measures how surprising the recommendations are to a user; computed as in [15].

4.2 Results & Discussion

Temporal Analysis. Since time-based genre prediction and its influence in the recommendation process is a novel strategy, we evaluate it in isolation to demonstrate its effectiveness. To do so, we used a disjoint set of 1,214 randomly-selected users from the dataset introduced earlier in the section. We used *KL-Divergence*, which measures how well a distribution q generated by a prediction strategy approximates a distribution p, the ground truth, i.e., distribution of genres read by a user over a period of time. We also used *accuracy*, a binary strategy that reflects if the predicted genres correspond to the ones read by a user over a given period of time. In establishing the ground truth for evaluation purposes, we adopted the well-known N-1 strategy: the genre of the books rated by a user U in N-1 time frames are used for training U's genre prediction model, whereas the genre of the books rated by U in the N^{th} time frame are treated as "relevant". As a **baseline** for this initial assessment, we use a naive strategy that defines the importance of each genre for U on the current, i.e., N, time frame based on the genre distribution across the previous N-1 time frames.

As shown in Table 3, for N=11 KL divergence scores indicate that genre distribution predicted using time-series better approximates to the ground truth thus leading to better performance. Furthermore, the probability of occurrence of each considered genre is closer to the real values when the time component is included in the prediction process. We observed differences among users who read different number of distinct genres. For users who read only one to two genres, the time-based prediction strategy does not perform better than the baseline. However, if a user reads three or more genres, our time-based genre prediction strategy outperforms the baseline in both metrics. This is not surprising, given that it is not hard to determine area(s) of interest for a user who constantly reads only one or two book genres, which is why the baseline performs as good as time-based prediction strategy. Given that users that read 3 or more genres represent 91% of the users in the dataset used in the remaining of the experiments presented in this section, the proposed strategy provides significant improvements in predicting preferred genre for the vast majority of readers.

Overall Performance. We evaluate the individual strategies that contribute to *QBook*'s recommendation process and analyze how each of them influences the generation of book suggestions.

Table 2: Aims of explanations in a recommender system

Aim	Definition
Effectiveness	Help users make good decisions
Efficiency	Help users make decisions faster
Persuasiveness	Convince users to try or buy
Satisfaction	Increase the ease of usability or enjoyment
Scrutability	Allow users to tell the system it is wrong
Transparency	Explain how the system works
Trust	Increase users' confidence in the system

Table 3: Influence of genre preference change over time on the recommendation process;'*' significant for p<0.001 t-test

Prediction Strategy	KL	Accuracy
Without Time Series	0.663	0.826
With Time Series	**0.623***	**0.870***
Without Time Series (3+ genres)	0.720	0.810
With Time Series (3+ genres)	**0.660***	**0.857***

We create *top*-7[5] recommendations for each user in *QDevel* using the individual strategies defined in Section 3 and evaluate the effectiveness of these recommendations based on NDCG[6].

As shown in Figure 3, matrix factorization and content based approaches are similarly effective in generating suggestions. However, when combined they slightly increase the value of NDCG. This improvement is statistically significant (p < 0.001; t-test), which means that, in general, users get more relevant recommendations when both strategies are considered in-tandem. This can be explained with the fact that these two methodologies complement each other. Furthermore, we can see that the similarity between literary features of interest to a user and literary features most often used to describe a book, has a positive influence on the recommendation process as it increases NDCG by 2.5 % when explicitly considered as part of the recommendation process. This is anticipated, since user-generated reviews hold a lot of information that can allow us to gain knowledge about each user and personalize suggestions.

The most reliable data points, which not only achieve relatively high NDCG but also are widely applicable and do not depend on individual users, are the four strategies that analyze sentiment of expert reviews. These strategies rely on information frequently available and thus are applicable to the majority of books examined by *QBook*. Based on Figure 3, we can see that data points calculated using sentence level sentiment analysis provide slightly better recommendations compared to the ones generated using word level sentiment analysis. Even though the individual strategies perform relatively well, we cannot assume that each data point can be calculated for every single book. *QBook*'s improvements in terms of NDCG can be justified with its ability to: (i) simultaneously consider multiple data points, (ii) include genre-prediction strategy, and (iii) more completely analyze different perspectives of user-book relations to provide recommendations even when some of the data points are unavailable. This is demonstrated based on the fact that NDCG of *QBook* is statistically significant with respect to the NDCG reported for the individual strategies (for p < 0.001).

[5] K is set to 7, based on a study presented in [30], where authors argue that a number of objects an average human can hold in working memory is 7 ± 2.

[6] $QDevel$ and $QEval$ yield comparable scores, indicating consistence in performance regardless of the data use for experimentation and no overfitting.

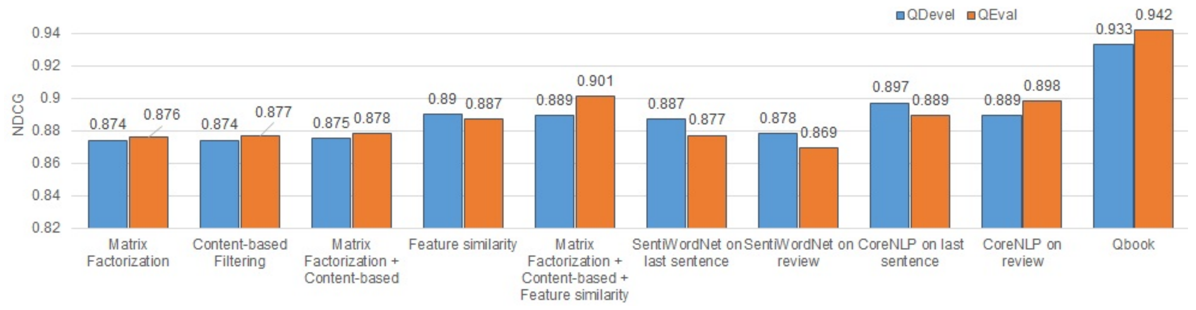

Figure 3: Performance evaluation of individual recommendation strategies considered by *QBook* on *QDevel* and *QEval.*

To further showcase the validity of *Qbook*'s design methodology, we compare its performance with two baselines: SVD (matrix factorization) and CB (content-based). For their respective implementations we rely on LensKit. The significant ($p < 0.01$) NDCG improvement of *QBook*, with respect to SVD (0.874) and CB (0.856), demonstrates that, in general, recommendations provided by *QBook* are preferred over the ones provided by the baselines, which either consider ratings patterns or content, but not both.

Recommendation Explanations. There is no gold-standard to evaluate explanations offline. Thus, following the strategy in [16, 39] we conducted an initial qualitative evaluation for demonstrating the usefulness of the **explanations** generated by *QBook*. We rely on the criteria introduced in [39] and summarized in Table 2, which outlines the "characteristics" of good explanations for recommenders. *QBook* achieves *five* out of the seven criteria expected of explanations generated by recommenders.By suggesting curated books which are described based on users' preferred features of interest, showcasing opinions of other users on those features and describing curation steps, *QBook* addresses *transparency*. *QBook* inspires *trust* on its users, since it does not consider the sentiment connotation of the features to determine if they should be included in an explanation. Instead, *QBook* provides unbiased recommendations and explanations; which can increase users' confidence as they know *QBook* offers a real depiction of each suggested book. Users are also able to make good and *fast* decisions, in terms of selecting books among the suggested ones, since based on provided explanations they can infer which books match their current preferences. With this, *QBook* increases its *effectiveness*. Given that users' overall *satisfaction* with a recommender is related to the perceived quality of its recommendations and explanations [16], *QBook* users can appreciate not having to spend more time researching books with characteristics important to them.

As per the study in [39] and assessments on several explanation-generation strategies [19, 31, 42, 46], we can report that, on average, only two (out of seven) criteria are satisfied. The only strategy that is comparable to *QBook*'s is the one discussed in [46], which addresses five of the criteria. However, this strategy involves sentiment in the generation of the explanations, as opposed to *QBook* which makes unbiased decisions when identifying users' features of preference and selecting which sentences to use to describe these features.

Common Recommendation Issues. We showcase *QBook*'s ability to address popular recommendation issues based on RMSE, in addition to adopting the evaluation framework presented in [27] to simulate online evaluation using offline metrics: coverage, serendipity and novelty. Based on the results of our analysis, we observe that the performance of *QBook* is consistent, regardless of the presence or absence of data points used in the recommendation process. *QBook*'s **RMSE** (Table 4) indicates that its recommendation strategy can successfully predict users' degree of preference for books. *QBook*'s **coverage** score (0.92), highlights that *QBook* considers a vast number of diverse books as potential recommendation, as opposed to popular ones. The **novelty** score (0.73) depicts that a user is provided with suggestions that differ from what he already saw. This characteristic of *QBook*, together with relatively high **serendipity** (0.68), indicates that new and unexpected, yet relevant, suggestions are generated.

State-of-the-art. We compare *QBook* with other book recommenders (optimal parameters were empirically defined).

LDAMF [29] harnesses the information in review text by fitting an LDA model on the review text.

CTR [43] uses a one-class collaborative filter strategy. Even though it is not specifically created for books, we consider it as it exploits metadata comparable for that of books.

HFT combines reviews with ratings [29] and models the ratings using a matrix factorization model to link the stochastic topic distribution in review text and the latent vector in the ratings.

SVD++ [23] refers to a matrix factorization model which makes use of implicit feedback information.

URRP [20], is a Bayesian model that combines collaborative and content-based filtering to learn user rating and review preferences.

'Free Lunch' [26] leverages clusters based on information that is present in the user-item matrix, but not directly exploited during matrix factorization.

RMR [25], which combines baselines by using the information of both ratings and reviews.

In Table 4 we summarize the results of the evaluation conducted using *QEval* in Table 4 in terms of RMSE. *QBook* outperforms existing state-of-the-art book recommenders considered in this study in terms of predicting the degree of which a user would like to read each recommended book. The difference on RMSE computed for *QBook* with respect to aforementioned state-of-the-art book recommenders are statistically significant with $p < 0.001$.

The prediction power of *QBook* is evidenced by its ability to achieve lowest RMSE among state-of-the-art approaches. When analyzing the performance of different strategies in more detail, we can see that Matrix Factorization strategies perform better, as in the case of Free Lunch (with and without clustering) and SVD++.

However, *QBook* goes beyond matrix factorization by using a content based approach as well as involving different perspectives, including other users' and experts' reviews.

Table 4: *QBook* vs. state-of-the-art recommenders

Strategy	RMSE	Strategy	RMSE
QBook	**0.795**		
RMR	1.055	Free Lunch	0.933
LDAMF	1.053	Free Lunch w/ Clustering	0.825
CTR	1.052	SVD++	0.908
HFT	1.066	URRP	1.104

5 CONCLUSIONS & FUTURE WORK

We presented *QBook*, a book recommender that acts as a curator by showcasing tailored book selections that meet the reading needs of individual users.As part of its recommendation process, *QBooks* examines different areas of user interest, not only the most dominant or recent ones, as well as varied data points. In doing so, *QBook* can yield a diverse set of suggestions, each paired with an explanation, to provide a user not only with reasons why a book was included in the curated list of recommendations but also how each recommendation was selected, with the objective of enhancing trust and transparency towards the user.

We conducted a number of offline experiments to validate the performance of *QBook* using a popular dataset. We also demonstrated the importance of considering diverse data sources, beyond ratings or content, to enhance the recommendation process.

With this work, we set the algorithmic foundations that will allow us to conduct in-depth online experiments in the future, in order to quantify the usability of *QBook*, the value of its explanations, and the degree of which its curation strategy can enrich the recommendation process from a user's perspective. Given the domain-independent nature of our strategies, we plan to validate *QBook* on datasets other than books to demonstrate its applicability on domains other than books. Our goal is to go one step further and enable our personal curator to generate suggestions in multiple domains, based on a complete virtual footprint available for a user.

REFERENCES

[1] Content based recommendation system. Available at: http://eugenelin89.github.io/recommender_content_based/.
[2] INEX Amazon/LibraryThing book corpus. http://social-book-search.humanities.uva.nl/data/ALT_Nondisclosure_Agreements.html. Accessed: 2016-02-07.
[3] LensKit open-source tools for recommender systems. https://lenskit.org.
[4] C. Benkoussas, A. Ollagnier, and P. Bellot. Book recommendation using information retrieval methods and graph analysis. In *CLEF*. CEUR, 2015.
[5] R. Blanco, D. Ceccarelli, C. Lucchese, R. Perego, and F. Silvestri. You should read this! let me explain you why: explaining news recommendations to users. In *ACM CIKM*, pages 1995–1999. ACM, 2012.
[6] C. Borrelli. Everybody's a curator. chicago tribune. https://goo.gl/hpTF3Q. Accessed: 2015-12-06.
[7] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
[8] S. Channamsetty and M. D. Ekstrand. Recommender response to diversity and popularity bias in user profiles. In *AAAI FLAIRS*, pages 657–660, 2017.
[9] L. Chen and F. Wang. Sentiment-enhanced explanation of product recommendations. In *WWW*, pages 239–240. ACM, 2014.
[10] Dictionary. Oxford: Oxford university press, 1989.
[11] N. Dragovic and M. S. Pera. Genre prediction to inform the recommendation process. *Proceedings of the Poster Track of the 10th ACM Conference on Recommender Systems*, 2016.
[12] N. Dragovic and M. S. Pera. Exploiting reviews to generate personalized and justified recommendations to guide users' selection. In *AAAI FLAIRS*, pages 661–664, 2017.
[13] A. Esuli and F. Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *LREC*, volume 6, pages 417–422, 2006.
[14] A. L. Garrido and S. Ilarri. Tmr: a semantic recommender system using topic maps on the items' descriptions. In *ESWC*, pages 213–217. Springer, 2014.
[15] M. Ge, C. Delgado-Battenfeld, and D. Jannach. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *ACM RecSys*, pages 257–260. ACM, 2010.
[16] F. Gedikli, D. Jannach, and M. Ge. How should i explain? a comparison of different explanation types for recommender systems. *International Journal of Human-Computer Studies*, 72(4):367–382, 2014.
[17] S. Givon and V. Lavrenko. Predicting social-tags for cold start book recommendations. In *ACM RecSys*, pages 333–336. ACM, 2009.
[18] J. L. Herlocker, J. A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *ACM CSCW*, pages 241–250. ACM, 2000.
[19] A. Hernando, J. Bobadilla, F. Ortega, and A. GutiéRrez. Trees for explaining recommendations made through collaborative filtering. *Information Sciences*, 239:1–17, 2013.
[20] M. Jiang, D. Song, L. Liao, and F. Zhu. A bayesian recommender model for user rating and review profiling. *Tsinghua Science & Technology*, 20(6):634–643, 2015.
[21] G. Kazai, D. Clarke, I. Yusof, and M. Venanzi. A personalised reader for crowd curated content. In *ACM RecSys*, pages 325–326. ACM, 2015.
[22] D. Kislyuk, Y. Liu, D. Liu, E. Tzeng, and Y. Jing. Human curation and convnets: Powering item-to-item recommendations on pinterest. *arXiv preprint arXiv:1511.04003*, 2015.
[23] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *ACM SIGKDD*, pages 426–434. ACM, 2008.
[24] Y. Koren, R. Bell, C. Volinsky, et al. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.
[25] G. Ling, M. R. Lyu, and I. King. Ratings meet reviews, a combined approach to recommend. In *ACM RecSys*, pages 105–112. ACM, 2014.
[26] B. Loni, A. Said, M. Larson, and A. Hanjalic. 'free lunch'enhancement for collaborative filtering with factorization machines. In *ACM RecSys*, pages 281–284. ACM, 2014.
[27] A. Maksai, F. Garcin, and B. Faltings. Predicting online performance of news recommender systems through richer evaluation metrics. In *ACM RecSys*, pages 179–186. ACM, 2015.
[28] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60, 2014.
[29] J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *ACM RecSys*, pages 165–172. ACM, 2013.
[30] G. A. Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63(2):81, 1956.
[31] J. Misztal and B. Indurkhya. Explaining contextual recommendations: Interaction design study and prototype implementation. In *IntRS@RecSys*, pages 13–20, 2015.
[32] R. Nau. Introduction to arima models.duke university. http://people.duke.edu/~rnau/411arim.htm. Accessed: 2016-05-06.
[33] B. Ohana and B. Tierney. Sentiment classification of reviews using sentiwordnet. *9th IT & T Conference*, 2009.
[34] M. S. Pera and Y.-K. Ng. Automating readers' advisory to make book recommendations for k-12 readers. In *ACM RecSys*, pages 9–16. ACM, 2014.
[35] M. S. Pera and Y.-K. Ng. Analyzing book-related features to recommend books for emergent readers. In *ACM HT*, pages 221–230. ACM, 2015.
[36] F. Ricci, L. Rokach, and B. Shapira. *Introduction to recommender systems handbook*. Springer, 2011.
[37] Z. Saaya, R. Rafter, M. Schaal, and B. Smyth. The curated web: a recommendation challenge. In *ACM RecSys*, pages 101–104. ACM, 2013.
[38] Y. Teng, L. Zhang, Y. Tian, and X. Li. A novel fahp based book recommendation method by fusing apriori rule mining. In *ISKE*, pages 237–243. IEEE, 2015.
[39] N. Tintarev and J. Masthoff. A survey of explanations in recommender systems. In *IEEE ICDEW*, pages 801–810, 2007.
[40] N. Tintarev and J. Masthoff. Evaluating recommender explanations: problems experienced and lessons learned for the evaluation of adaptive systems. In *UCDEAS Workshop associated with UMAP*. CEUR-WS, 2009.
[41] S. Vargas and P. Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *ACM RecSys*, pages 109–116. ACM, 2011.
[42] J. Vig, S. Sen, and J. Riedl. Tagsplanations: explaining recommendations using tags. In *IUI*, pages 47–56. ACM, 2009.
[43] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *TACM SIGKDD*, pages 448–456. ACM, 2011.
[44] J. Wang and Y. Zhang. Opportunity model for e-commerce recommendation: right product; right time. In *ACM SIGIR*, pages 303–312, 2013.
[45] M. C. Willemsen, M. P. Graus, and B. P. Knijnenburg. Understanding the role of latent feature diversification on choice difficulty and satisfaction. *UMUAI*, 26(4):347–389, 2016.
[46] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, and S. Ma. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *ACM SIGIR*, pages 83–92. ACM, 2014.