With a Little Help from My Friends: Generating Personalized Book Recommendations Using Data Extracted from a Social Website

Conference Paper · September 2011

DOI: 10.1109/WI-IAT.2011.9 · Source: IEEE Xplore

CITATIONS
14

READS
537

2 authors:

Maria Soledad Pera
Boise State University
113 PUBLICATIONS **887** CITATIONS

SEE PROFILE

Yiu-kai Ng
Brigham Young University - Provo Main Campus
97 PUBLICATIONS **1,590** CITATIONS

SEE PROFILE

With a Little Help From My Friends: Generating Personalized Book Recommendations Using Data Extracted from a Social Website

Maria Soledad Pera Yiu-Kai Ng
Computer Science Department
Brigham Young University
Provo, Utah, U.S.A.

Abstract

With the large amount of books available nowadays, users are overwhelmed with choices when they attempt to find books of interest. While existing book recommendation systems, which are based on either collaborative filtering, content-based, or hybrid methods, suggest books (among the millions available) that might be appealing to the users, their recommendations are not personalized enough to meet users' expectations due to their collective assumption on group preference and/or exact content matching, which is a failure. To address this problem, we have developed $PReF$*, a Personalized Recommender that relies on Friendships established by users on a social website, such as LibraryThing, to make book recommendations* tailored *to individual users. In selecting books to be recommended to a user* U*, who is interested in a book* B*,* $PReF$ *(i) considers books belonged to* U*'s friends, (ii) applies* word-correlation factors *to disclose books* similar *in contents to* B*, (iii) depends on the ratings given to books by* U*'s friends to identify highly-regarded books, and (iv) determines how reliable individual friends of* U *are in providing books from their own catalogs (that are similar in content to* B*) to be recommended. We have conducted an empirical study and verified that (i) relying on data extracted from social websites improves the effectiveness of book recommenders and (ii)* $PReF$ *outperforms the recommenders employed by Amazon and LibraryThing.*

1 Introduction

In recent years social websites, such as Facebook(.com), Twitter(.com), YouTube(.com), and Delicious(.com), have become increasingly popular [9]. These sites introduce new user-generated data and metadata, such as ratings, social connections, and tags[1], which provide a rich source of information to infer users' interests and preferences. These kinds of information are unique and valuable for making recommendations on books, movies, news articles, etc., which have been examined in [2, 3, 9]. Newly-developed recommenders, such as [9, 16, 22], incorporate data extracted from social websites to increase the quality of tag, news articles, and book recommendations. Book recommenders have been adopted by online shopping companies, social websites, and digital libraries, to name a few, to further facilitate their users' knowledge acquisition process by offering alternative choices (among the millions available) of books they are likely interested in. While suggestions provided by existing book recommenders can introduce users to books that they are not aware of, these recommenders are not personalized enough to achieve their design goals [10]. It is imperative to develop personalized recommenders that provide finer suggestions pertinent to individual users' interests or preferences. To the best of our knowledge, there are no recommendation systems that simultaneously consider *users' relationships*, along with *user-generated* data extracted from a social website, to recommend *books*.

In this paper, we introduce $PReF$, a personalized book recommendation system that depends on friendships established among users in a social website, which is LibraryThing[2] in our case, to generate valuable book recommendations tailored to individual users' interests. $PReF$ locates, among the books bookmarked by U's friends on a social website, the ones that are *similar* in content to a given book B that U is interested in. Hereafter, $PReF$ ranks the candidate books to be recommended by considering not only the *content similarity* between each candidate book CB and B, but also the *ratings* assigned to CB by U's friends, and the *reliability* of each of U's friends.

$PReF$ is an elegant and unique system that relies on (i) *relationships* established between a user and other mem-

[1] Tags are user-defined keywords that describe the content of an item.

[2] LibraryThing(.com) was founded in 2006 for aiding users in cataloging and referencing books. LibraryThing users can rate and review books, add tags to books to describe their contents, and establish friendships, i.e., bi-directional relationships, with other LibraryThing users.

bers of a social website, since as stated in [2], the quality of recommendations given to a user U is improved by considering opinions of other users whom U trusts, (ii) *ratings* provided by users of a social site, which aid in identifying highly-regarded books a user might be interested in, and (iii) *word-correlation factors* [12], which detect books *similar* in content, even if they are described using analogous, but not the same, tags, to generate personalized book recommendations. In addition, $PReF$ can perform the recommendation task with data extracted from *any* social website, provided that users' relationships, book tags, and book ratings can be obtained from the site.

We have conducted an empirical study using data extracted from LibraryThing to validate the effectiveness of personalized book recommendations made by $PReF$. The study has verified $PReF$ is significantly more effective than (the recommenders used at) Amazon and LibraryThing in recommending books that individual users are interested in.

The remaining of this paper is organized as follows. In Section 2, we discuss existing (book) recommendation systems. In Section 3, we detail the design of $PReF$. In Section 4, we present the results of the empirical study conducted for assessing the performance of $PReF$. In Section 5, we give a conclusion.

2 Related Work

Machine learning, information retrieval, natural language processing, and probabilistic models have been adopted for developing systems that recommend (web) documents [8], songs [4], and movies [13], to name a few.

Content-based and *collaborative filtering* are two well-known approaches for making recommendations [18]. The former creates a profile to capture items of interest to a user U using words, phrases, or features, whereas the later identifies the group of people who have similar preferences as U's and recommends items to U that the group is interested in. Recent publications [7, 18] present various hybrid approaches that exploit the benefits of using both content-based and collaborative filtering methods to improve the quality of recommendations. An in-depth discussion of various content-based, collaborative filtering, and hybrid recommendation systems can be found in [1].

There exist a number of book recommendation systems [15, 18, 23]. Amazon's recommender [15] suggests to a user, who is interested in an item I, items that match the purchase patterns of other users who have purchased I. Yang et al. [23] rely on a collaborative filtering approach with ranking, which considers users' preferences on library resources extracted from their access logs to recommend library materials. This approach overcomes the problem that arises due to the lack of initial information to perform the recommendation task. Park and Chang [18] create a user-profile P based on individual and group behavior, such as clicks and shopping habits, compute the Euclidean distance between P and each product profile, and recommend products for which their Euclidean distances are closest to P. For additional references on book recommenders see [14].

While (to the best of our knowledge) none of the existing book recommenders considers data extracted from social websites to make personalized recommendations, which $PReF$ does, the recommenders in [9, 16, 20, 22] employ data extracted from social sites to suggest items other than books. Wang et al. [22] consider a news posting, along with the comments made by users on the posting, to generate a list of recommended news articles for a particular news thread, whereas Guy et al. [9] develop a personalized recommendation system on social items (such as blogs posts and bookmarks), which relies on the relationships between people, items, and tags. Liu et al. [16] and Shepitsen et al. [20] develop different approaches for generating personalized tag recommendations. In accomplishing the task, the authors in [16] combine collaborative information extracted from social tagging systems, such as Delicious, and the users' personalized tag preferences, whereas the authors in [20] apply a hierarchical agglomerative clustering algorithm to identify users' individual interests. Unlike $PReF$, none of the approaches in [16, 20, 22] rely on friendships established on social websites or tag similarity matching in performing the recommendation task.

3 Our Proposed Book Recommender

In this section, we present our book recommender, $PReF$, which employs data extracted from LibraryThing to generate personalized book recommendations. LibraryThing is an innovative, well-designed, and highly popular social application that was set up solely for cataloging books [21]. As of March 7, 2011, LibraryThing archives 5,943,819 unique records (on books), and approximately 1,296,535 users have added more than 73.6 million tags to different book records at LibraryThing, according to the Zeitgeist Overview (librarything.com/zeitgeist) which provides official statistical data of LibraryThing. Each LibraryThing user U has a *personal catalog* that includes books (s)he owns or is interested in. In addition, U can assign tags to books included in his/her catalog, which serve as personalized identifiers of the contents of the books. To indicate how highly regarded a book B in the catalog is, U assigns a *rating* to B, which is a numerical value between '1' and '5', such that '5' is the highest and '1' is the lowest. Moreover, U has a *profile* which includes a list of other LibraryThing users who were explicitly chosen by U to be his/her friends. In LibraryThing, each book B is associated with (i) a *tag cloud*, which is a global visual representation of tags (and their frequencies) assigned to B by LibraryThing users who

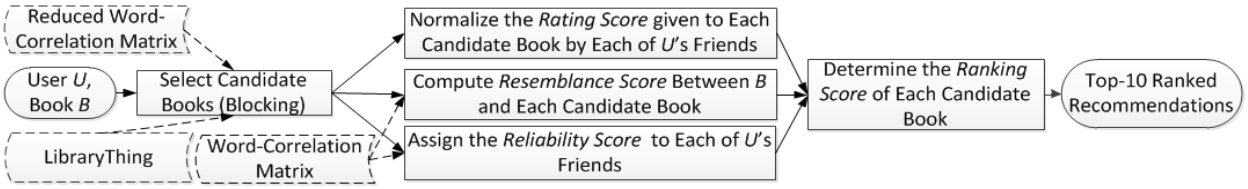

Figure 1: Processing steps of the proposed book recommender, $PReF$

include B in their catalogs, and (ii) a *global rating*, which *averages* the ratings assigned to B by LibraryThing users.

Given a LibraryThing user, denoted LT_User, and a book, denoted $Source\_Bk$, which has been added by LT_User to his/her personal catalog or browsed by LT_User on LibraryThing, $PReF$ identifies LT_User's friends and determines the set of books, denoted $Candidate\_Set$, among those included in the personal catalogs of LT_User's friends that are similar to $Source\_Bk$ (as detailed in Section 3.2). Hereafter, $PReF$ computes the *ranking score* of each book CB in $Canditate\_Set$ (as defined in Section 3.3) and the top-N (≥ 1) ranked books are recommended to LT_User. The overall process of $PReF$ is illustrated in Figure 1.

3.1 Word Correlation Factors

$PReF$ relies on the pre-computed word-correlation factors in the word-correlation matrix [12] to determine the similarity among (the content of) books using their corresponding sets of tags. *Word-correlation factors* were generated using a set of approximately 880,000 Wikipedia documents (wikipedia.org), and each correlation factor indicates the *degree of similarity* of the two corresponding words[3] based on their (i) *frequency of co-occurrence* and (ii) *relative distances* in each Wikipedia document. Wikipedia documents were chosen for constructing the word-correlation matrix, since they were written by more than 89,000 authors (i) with different writing styles, (ii) using various terminologies that cover a wide range of topics, and (iii) with diverse word usage and content. Compared with synonyms/related words compiled by WordNet (wordnet.princeton.edu) in which pairs of words are not assigned similarity weights, word-correlation factors provide a more sophisticated measure of word similarity.

3.2 Selecting Candidate Books

As the number of books in the personal catalog of each LT_User's friend can be large, which can be in the thousands, it is not practical to compare each book with $Source\_Bk$ to identify the ones to be recommended to LT_User, since the comparisons significantly prolong the processing time. To minimize the number of comparisons, $PReF$ applies a *blocking strategy*[4] on the books posted under the personal catalogs of LT_User's friends to yield the subset of books (which is relatively small in size), denoted $Candidate\_Set$, considered for recommendation. At least one of the tags of each book in $Candidate\_Set$ *matches exactly* or is *highly similar* to one of the tags of $Source\_Bk$ assigned by LT_User. As books in $Candidate\_Set$ and $Source\_Bk$ share the same (or analogous) tags, $PReF$ expects books in $Candidate\_Set$ to be similar in content (to a certain degree) to $Source\_Bk$. In case when LT_User does not assign any personal tags to $Source\_Bk$, $PReF$ relies on the top-3 tags, i.e., the tags with the highest frequency of occurrence, in the tag cloud of $Source\_Bk$ to perform the blocking task. The top-3 tags are chosen, since we have observed that LibraryThing users assign, on the average, *three* tags to each book in their personal catalogs.

To identify highly similar tags, $PReF$ employs a *reduced* version of the word-correlation matrix (introduced in Section 3.1) which contains 13% of the most frequently-occurring words (based on their frequencies of occurrence in the Wikipedia documents), and for the remaining 87% of the less-frequently-occurring words only the exact-matched correlation factor, i.e., 1.0, is used. By adopting a reduced word-correlation matrix, instead of the word-correlation matrix, in determining similar books, the overall processing time can be significantly reduced without affecting the accuracy [19].

3.3 Ranking LibraryThing Books

$PReF$ ranks each book CB in $Candidate\_Set$ to prioritize them for recommendations using (i) the *degree of resemblance* of CB and $Source\_Bk$ (in Section 3.3.1), (ii) the *rating score* assigned to CB by each friend of LT_User, who includes CB in his/her personal catalog (in Section 3.3.2), and (iii) the relative *degree of reliability* of each of LT_User's friends (in Section 3.3.3).

[3]Words in the Wikipedia documents were *stemmed* (i.e., reduced to their grammatical roots) after all the *stopwords*, such as articles, conjunctions, and prepositions, which do not play a significant role in representing the content of a document, were removed. From now on, unless stated otherwise, (key)words/tags refer to nonstop, stemmed words.

[4]A *blocking strategy* [11] is a *filtering* technique which reduces the potentially very large number of comparisons to be made among records [5].

3.3.1 Similarity Among Books

To determine the (content) similarity between $Source\_Bk$ and CB, $PReF$ computes their *degree of resemblance* by adding the word-correlation factors between each tag in the tag cloud (provided by LibraryThing) of $Source\_Bk$ and CB, respectively using the word-correlation matrix introduced in Section 3.1, instead of using the reduced word-correlation matrix employed in Section 3.2, since the former provides a *more accurate* similarity measure between (tags representing) $Source\_Bk$ and CB than using the reduced matrix. The *degree of resemblance*, denoted $Resem$, between $Source\_Bk$ and CB is defined as

$$Resem(Source\_Bk, CB) = \frac{\sum_{i=1}^{n} Min\{\sum_{j=1}^{m} wcf(Source\_Bk_i, CB_j), 1\} \times freq_i}{\sum_{i=1}^{n} freq_i} \quad (1)$$

where n (m, respectively) is the number of distinct tags in (the tag cloud of) $Source\_Bk$ (CB, respectively), $Source\_Bk_i$ (CB_j, respectively) is a tag in the tag cloud of $Source\_Bk$ (CB, respectively), $wcf(Source\_Bk_i, CB_j)$ is the correlation factor of $Source\_Bk_i$ and CB_j in the word-correlation matrix, and $freq_i$ denotes the number of times $Source\_Bk_i$ is assigned to $Source\_Bk$ as specified in the tag cloud of $Source\_Bk$. We normalize $Resem(Source\_Bk, CB)$, so that the computed degree of resemblance is in the [0, 1] range, by dividing the accumulated correlation factors by the sum of the frequencies of occurrence of each tag assigned to $Source\_Bk$.

The Min function in Equation 1 imposes a constraint on summing up the word-correlation factors of tags representing $Source\_Bk$ and CB. Even if a tag in the tag cloud of CB (i) matches exactly one of the tags in the tag cloud of $Source\_Bk$ and (ii) is similar to some of the remaining tags describing $Source\_Bk$, which yields a value greater than 1.0, i.e., the word-correlation factor of an exact match, $PReF$ limits the sum of their similarity measure to 1.0. This constraint ensures that if CB contains a dominant tag T in its tag cloud, i.e., T is highly similar to a few tags in the tag cloud of $Source\_Bk$, T alone cannot significantly impact the resemblance value of $Source\_Bk$ and CB, i.e., "one" does not represent "all". Tags assigned to CB that are similar to most of the tags of $Source\_Bk$ should yield a higher degree of resemblance of $Source\_Bk$ and CB than tags assigned to CB that are similar to only one dominant tag representing $Source\_Bk$.

3.3.2 Book Ratings

Among the books in the personal catalog of a LibraryThing user U, U might like some books more than the others, which is natural. In recommending books, $PReF$ considers the *rating* assigned to a book CB in $Candidate\_Set$ by a friend of LT_User, denoted LT_Pal[5], that should reflect the degree to which LT_Pal is interested in CB. $PReF$ suggests to LT_User books given high ratings scores by his/her friends, since these books are treated as more appealing to LT_User than books which are given lower ratings. $PReF$ normalizes the rating given by LT_Pal to CB so that its value is in the range [0, 1] as follows:

$$Rate(CB, LT\_Pal) = \frac{Rating\_CB}{5} \quad (2)$$

where $Rating\_CB$ is the rating score given to CB by LT_Pal, and '5' is the normalization factor, i.e., the highest possible rating score that can be assigned to CB.

Note that not every LibraryThing user assigns a rating to each book in his/her personal catalog. Should LT_Pal not provide a rating for CB, $PReF$ considers the collective opinion of LibraryThing users and computes $Rate(CB, LT\_Pal)$ using the *average*, i.e., global, *rating* assigned to CB by LibraryThing users as $Rating\_CB$.

3.3.3 Reliability of Friends in Book Recommendations

LibraryThing friends of LT_User might include in their catalogs books on various categories, such as religion, politics, fiction, or science, and it is expected that books in certain categories might be more predominant than others in the personal catalogs of LT_User's friends. Thus, not all (the books included in the catalogs) of LT_User's friends should be given the same "weight" for book recommendation, since recommendations provided by friends who include in their catalog a significant number of books in the same category as, i.e., similar in content to, $Source\_Bk$ are more reliable than recommendations provided by friends less familiar with the category of $Source\_Bk$. $PReF$ measures the *degree of reliability* of a friend of LT_User, i.e., LT_Pal, in recommending books that are similar (in content) to $Source\_Bk$ as follows:

$$Rel(Source\_Bk, LT\_Pal) = \frac{\sum_{i=1}^{m} Min\left\{\sum_{j=1}^{n} wcf(Source\_Bk_i, LT\_Pal_j), 1\right\}}{m} \quad (3)$$

where m (n, respectively) is the number of distinct tags assigned to $Source\_Bk$ by LT_User (LT_Pal to books in his/her personal catalog, respectively), $Source\_Bk_i$ (LT_Pal_j, respectively) is a tag assigned by LT_User to $Source\_Bk$ (LT_Pal in describing books in his/her personal catalog, respectively), and $wcf(Source\_Bk_i, LT\_Pal_j)$ is the correlation factor in the word-correlation matrix between $Source\_Bk_i$ and LT_Pal_j. In Equation 3, m is the normalization factor that scales the corresponding *degree of reliability* in a [0, 1] range.

[5]From now on LT_Pal refers to a friend of LT_User who includes a given book (in $Candidate\_Set$) in his/her catalog.

3.3.4 Recommendations

Having determined (i) the *degree of resemblance* between $Source\_Bk$ and each book CB in $Candidate\_Set$, (ii) the *rate* score assigned to CB by each of LT_User's friends, and (iii) the *degree of reliability* of each friend of LT_User, $PReF$ computes the *ranking score* of CB, denoted $Rank(CB)$, as follows:

$$\text{Rank(CB)} = \frac{ArgMax_{LT\_Pal_i \in Pal_{CB}}\{Rel(Source\_Bk, LT\_Pal_i) \times Resem(Source\_Bk, CB) + Rate(CB, LT\_Pal_i)}{1 - Min\{Resem(Source\_Bk, CB), Rate(CB, LT\_Pal_i)\}}\} \quad (4)$$

where Pal_{CB} is the group of LT_USer's friends who include CB in their personal catalogs, and LT_Pal_i is the i^{th} LT_Pal in Pal_{CB}.

The *ArgMax* function in Equation 4 ensures that the highest ranking score of CB, among the ones computed for each of LT_User's friends, is considered during the recommendation process, which guarantees that no duplicate books are recommended to LT_User.

By combining the *resemblance* and *rate* scores (as defined in Equations 1 and 2, respectively) using the *Stanford Certainty Factor* (SCF) [17], $PReF$ measures the relative *appealing* value of CB (in $Candidate\_Set$), which is high only when both the resemblance and rate scores are high, since SCF is monotonically increasing (decreasing) function. Furthermore, by employing the Joint Product [17] in Equation 4, $PReF$ *adjusts* the computed appealing value of CB based on the *reliability* of a friend of LT_User in recommending books for $Source\_Bk$.

The Top-10 *ranked* books are recommended to LT_User, which follows the number of recommendations presented by LibraryThing to its users.

Example 1 Consider the book "Emma" by Jane Austen and a LibraryThing user, Soleenusa, who is one of the independent appraisers of $PReF$ interested in "Emma". Based on the books included in the personal catalogs of Soleenusa's LibraryThing friends, $PReF$ suggests 10 books that might also be of interest to Soleenusa. As shown in Table 1, except for the 10^{th} recommended book, Soleenusa marks all the books as closely related to "Emma" (in bold). Note that Books 1 to 9 are also written by Jane Austen and are in the same subject area of "Emma", which is a classical novel. Furthermore, Books 7 and 8 include two popular Jane Austen's novels along with contextual and source materials, a wide range of interpretations, and bibliographical information. Compared with the books recommended by Amazon and LibraryThing for "Emma", only 2 and 5 of the recommendations generated by Amazon and LibraryThing, respectively are regarded as closely related by Soleenusa (as shown in Table 1). The remaining recommended books, such as "The Odyssey", are considered non-relevant recommendations for "Emma" by Soleenusa. □

4 Experimental Results

In this section, we first introduce the data and metrics in Sections 4.1 and 4.2, respectively which were used for assessing the performance of $PReF$. Thereafter, we detail the results of the empirical study conducted for evaluating $PReF$, in addition to comparing its performance with other existing book recommenders in Section 4.3.

4.1 Experimental Data

To analyze the performance of $PReF$, we rely on data extracted from LibraryThing that contain personal information of a group of independent appraisers who are LibraryThing users, which include (i) (tags and ratings of) books in their personal catalogs, (ii) lists of their friends, and (iii) (tags and ratings of) books posted under their friends' personal catalogs. In addition, the extracted data include the tag cloud and the global rating score of each book listed in (i) and (iii) above.

To the best of our knowledge, there is no existing dataset for assessing the performance of personalized book recommenders, and thus we rely on independent appraisers who manually examined the relatedness of each one of the top-10 recommendations generated by $PReF$ with respect to each book in their personal catalogs, yielding a set of 100 books, denoted $Test\_Books$, used in our empirical study.

4.2 Evaluation Metrics

To evaluate the effectiveness of $PReF$ in generating personalized book recommendations, we apply two well-known information retrieval metrics, the (overall) *Precision at K* and *Mean Reciprocal Rank* [6].

The $P@K$ value quantifies the top-K ranked recommended books for a particular book in $Test\_Books$, which measures the overall user's satisfaction with the top-K recommendations (generated by $PReF$).

$$P@K = \frac{\sum_{i=1}^{N} \frac{Number\_of\_Related\_Recommendations_i}{K}}{N} \quad (5)$$

where K is the (pre-defined) number of book recommendations to be considered, N is the number books in $Test\_Books$, i is a book in $Test\_Books$, and $Number\_of\_Related\_Recommendations_i$ is the number of recommendations out of K that are evaluated as *related* to book i by a particular appraiser who owns i. Note that in our study, we set K = 1, 5, and 10, to evaluate the

Rank	*PReF*	Amazon	LibraryThing
1	**Mansfield Park**	**Sense and Sensibility**	**Northanger Abbey**
2	**Sense and Sensibility**	Little Women	**Lady Susan/Sandition/The Watsons**
3	**Persuasion**	Oliver Twist	**Mansfield Park**
4	**Northanger Abbey**	**Pride and Prejudice**	Villette
5	**Pride and Prejudice**	Tess of the D'Ubervilles	Jane Eyre
6	**The Oxford Illustrated Jane Austen (Six Volume Set)**	The Sonnets and A Lovers Complaint	Wuthering Heights
7	**Emma (Norton Critical Edition)**	The Odyssey	The Tenant of Wildfell Hall
8	**Pride and Prejudice (Norton Critical Edition)**	Alices Adventures in Wonderland	**Vanity Fair**
9	**Minor Works of Jane Austen**	A Christmas Carol	Tess of the D'Ubervilles
10	A Town Like Alice	Jane Eyre	**Middlemarch**

Table 1: Recommendations generated by $PReF$, Amazon, and LibraryThing, respectively in response to the book "Emma", by Jane Austen

relatedness of the recommendations positioned at the top, $middle$, and $overall$ in the ranking, respectively. Since, as stated in Section 3.3.4, we only evaluated the top-10 recommendations generated by a book recommendation system, its $P@10$ score is the same as its $accuracy$ score, a well-known metric in information retrieval [6].

The *Mean Reciprocal Rank* (MRR) of the ranked book recommendations generated by $PReF$ is the averaged sum of the ranking values for the recommendations computed for each book in $Test\_Books$ such that each ranking value is either the reciprocal of the ranking position of the *first* related recommendation among the top-10 recommendations, if there is any, or 0, otherwise.

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{r_i} \tag{6}$$

where r_i is the (position in the) rank of the *first related* recommendation with respect to book i in $Test\_Books$, if it exists, and N and i are as defined in Equation 5.

While the $P@10$ measures the overall user's satisfaction of the recommendations created by $PReF$, $P@K$ and MRR evaluate the ranking strategy of $PReF$, since the higher related recommendations are ranked, the higher their corresponding $P@K$ and MRR scores should be.

To further assess the efficiency of our personalized book recommender, we employ the *imp* metric [20], which is a widely-used evaluation method that measures the level of improvement of a personalized approach when compared to a baseline, i.e., non-personalized, approach in ranking relevant recommended resources, i.e., books in our case. The overall *ranking improvement* of a personalization recommender is calculated by averaging the improvement for all the books in $Test\_Books$ as follows:

$$imp = \frac{\sum_{i=1}^{N}\frac{1}{rp_i} - \frac{1}{rb_i}}{N} \tag{7}$$

where N and i are as defined in Equation 5, and rp_i (rb_i, respectively) is the position in the ranking of the $first$ relevant recommendation as determined by a personalized (baseline, respectively) recommender.

The $higher$ the imp score is, the $better$ the ranking strategy adopted by a recommender is, i.e., imp shows the effectiveness of a personalization technique at moving "good" (book) recommendations to the top of the list [20].

4.3 Performance Evaluation and Comparisons

In this section, we present the experimental results achieved by $PReF$ and compare its performance with the recommendation systems of Amazon and LibraryThing[6], which are two well-known, commercial book recommenders. While the recommender of Amazon has been introduced in Section 2, the recommendation system of LibraryThing (i) compares books in a user's personal catalog with thousands of books in other users' catalogs, (ii) considers common tags assigned to (the tag clouds of) books, and (iii) identifies books with similar Library of Congress Subject Heading and/or Classification to provide a list of books a user might be interested in. (See http://www.librarything.com/wiki/index.php /Automatic_recommendations).

In comparing $PReF$ with Amazon and LibraryThing, we rely on the same group of independent appraisers (as discussed in Section 4.1) who determine which one of the top-10 books recommended by $PReF$, Amazon, and LibraryThing, respectively for each book B in $Test\_Books$ is related to B. Note that since $PReF$ is based on the premise that a user U tends to trust recommendations made by his/her friends, books recommended by $PReF$ to U are books in the personal catalogs of U's friends, whereas books recommended by Amazon (LibraryThing, respectively) are extracted from the entire collection of books available at Amazon (LibraryThing, respectively).

[6]From now on, unless stated otherwise, whenever we mention Amazon (LibraryThing, respectively), we mean Amazon's (LibraryThing's, respectively) book recommender.

Figure 2: $Precision@K$ (K = 1, 5, and 10) scores on the (top-10) recommendations achieved by $PReF$, Amazon, and LibraryThing for the books in $Test\_Books$

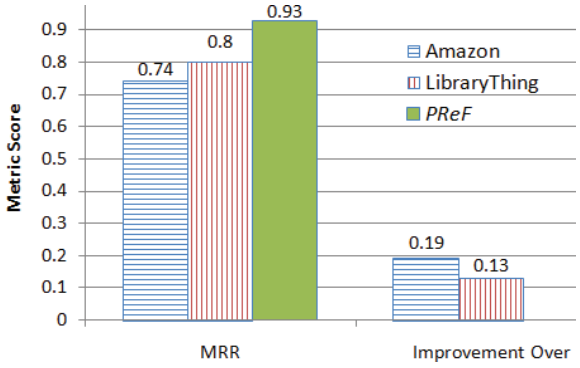

Figure 3: MRR and Imp scores based on (un)related books recommended by $PReF$, Amazon, and LibraryThing for the books in $Test\_Books$

4.3.1 Assessment

To assess the overall performance of $PReF$ (Amazon and LibraryThing, respectively) we have computed the $P@K$ scores on the top-10 book recommendations generated by $PReF$, Amazon, and LibraryThing, respectively for each book B in $Test\_Books$, based on the books labeled as (not) $related$ to B by each independent appraiser. As shown in Figure 2, the $P@1$ score of $PReF$, which is 0.90, indicates that among the 90 out of 100 books in $Test\_Books$, their first recommended books generated by $PReF$, i.e., the books with the highest ranking score, were treated as $related$. A high $P@1$ score implies that the ranking strategy of $PReF$ is highly effective in presenting first books that users are interested in. On the other hand, the $P@1$ scores achieved by Amazon and LibraryThing on the top-10 recommendations generated for books in $Test\_Books$ are 0.63 and 0.77, respectively, which are at least 13% lower compared with $PReF$'s $P@1$ score.

As previously stated, $P@5$ measures the overall user satisfaction with respect to the top-5 recommended books. Figure 2 shows that the $P@5$ score of $PReF$ is at least 21% higher than the $P@5$ scores of Amazon and LibraryThing. The outcome demonstrates that $PReF$, in general, positions higher in the list of recommendations books that are relevant to a particular user than Amazon and LibraryThing, respectively. The $P@10$ scores of $PReF$, Amazon, and LibraryThing are 0.78, 0.53, and 0.48, respectively, as shown in Figure 2. Based on the $P@10$ values, on the average, close to 8 out of the 10 books recommended by $PReF$ are perceived as related recommendations, as opposed to the five recommended by Amazon and LibraryThing.

Besides the $P@K$ scores, we have also assessed the performance of $PReF$ (Amazon and LibraryThing, respectively) based on the MRR metric. As shown in Figure 3, the MRR scores computed for $PReF$, Amazon, and LibraryThing are 0.93, 0.74, and 0.80, respectively, which reflect that while on the average users of $PReF$ are required to browse through the top ($\cong \frac{1}{0.93}$ = 1.07) generated recommendations before locating one that is related to a book that (s)he owns or is examining, Amazon's and LibraryThing's users, on the other hand, scan through at least one ($\cong \frac{1}{0.74}$ = 1.35 and $\cong \frac{1}{0.8}$ = 1.25, respectively) recommended book before identifying one that is appealing to them.

Lastly, we have computed the imp score of $PReF$ over Amazon and LibraryThing. As shown in Figure 3 $PReF$ achieves a 19% (13%, respectively) improvement over Amazon (LibraryThing, respectively) in generating books recommendations relevant, i.e., appealing, to users.

4.3.2 Observations

It is worth mentioning that $PReF$ always presents to users ten recommendations for each given book, as opposed to Amazon and LibraryThing, which occasionally generate less than ten recommendations, the expected number of recommendations. Furthermore, at least one of the top-10 recommendations generated by $PReF$ for each book in $Test\_Books$ is treated as $related$ to the corresponding book by the appraisers. However, Amazon (LibraryThing, respectively) generated either (i) no recommendations at all or (ii) no related recommendations for 8 (23, respectively) books in $Test\_Books$.

As shown in Figures 2 and 3, $PReF$ is more effective than Amazon or LibraryThing in recommending books that satisfy the personal interest of a user, which illustrates that considering (i) data extracted from a social website along with (ii) personal interactions of a user in a social environment enriches the effectiveness of book recommendations.

5 Conclusions

It is an unpleasant experience for book enthusiasts to acquire books and later discover that the books do not appeal

to their "tastes". In addition, it is difficult for book enthusiasts to keep track of new books published on a regular basis due to their number. Existing book recommenders, such as the one employed by LibraryThing, aid users in identifying books of interests. These recommenders, however, present the same recommendations to users that share the same profile information or common interests and hence are inadequate, since the suggestions do not often meet individual users' preferences. To address this problem, we have developed a personalized book recommender, called $PReF$. $PReF$ relies on (i) online *connections*, i.e., friendships, established among users at a social website, (ii) the existence of user-generated *tags* and *ratings*, and (iii) word-correlation factors, i.e., word-similarity measures, to generate book recommendations $tailored$ to the interests of an individual user. Unlike recommenders that rely on the "wisdom of crowds" to make recommendations, $PReF$ considers only interests shared among a user U and members of U's "inner circle", which yields valuable recommendations for U. In addition, $PReF$ is not limited by an exact match constraint and thus identifies books similar in contents, even if they do not share any common tags, which enriches the set of candidate books to be recommended.

We have conducted an empirical study using data extracted from LibraryThing to assess the effectiveness of book recommendations generated by $PReF$ and compare (the performance of) $PReF$ with two well-known recommenders, i.e., the ones employed by Amazon and LibraryThing. The study has verified that $PReF$ outperforms the recommenders adopted by Amazon and LibraryThing in generating personalized books recommendations.

While $PReF$ is currently designed for recommending books, we intend to extend $PReF$ so that it can recommend items in various domains, such as songs and movies, provided that data describing items of interest and friendships among users are available on one or more social websites.

References

[1] G. Adomavicius and A. Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE TKDE*, 17(6):734–749, 2005.

[2] R. Andersen, C. Borgs, J. Chayes, U. Feige, A. Flaxman, A. Kalai, V. Mirrokni, and M. Tennenholtz. Trust-based Recommendation Systems: an Axiomatic Approach. In *WWW*, pages 199–208, 2008.

[3] D. Carmel, N. Zwerdling, I. Guy, S. Ofek-Koifman, N. Har'el, I. Ronen, E. Uziel, S. Yogev, and S. Chernov. Personalized Social Search Based on the User's Social Network. In *CIKM*, pages 1227–1236, 2009.

[4] H. Chen and A. Chen. A Music Recommendation System Based on Music Data Grouping and User Interests. In *CIKM*, pages 231–238, 2001.

[5] P. Christen. Automatic Record Linkage Using Seeded Nearest Neighbor and Support Vector Machine Classification. In *SIGKDD*, pages 151–159, 2008.

[6] W. Croft, D. Metzler, and T. Strohman. *Search Engines: Information Retrieval in Practice*. Addison Wesley, 2010.

[7] L. de Campos, J. Fernandez-Luna, J. Huete, and M. Rueda-Morales. Combining Content-Based and Collaborative Recommendations: A Hybrid Approach based on Bayesian Networks. *Approximate Reasoning*, 51(7):785–799, 2010.

[8] Z. Guan, C. Wang, J. Bu, C. Chen, K. Yang, D. Cai, and X. He. Document Recommendation in Social Tagging Services. In *WWW*, pages 391–400, 2010.

[9] I. Guy, N. Zwerdling, I. Ronen, D. Carmel, and E. Uziel. Social Media Recommendation Based on People and Tags. In *SIGIR*, pages 194–201, 2010.

[10] J. Jung, K. Kim, H. Lee, and S. Park. Are You Satisfied with Your Recommendation Service?: Discovering Social Networks for Personalized Mobile Services. In *KES-AMSTA*, pages 567–573, 2008.

[11] R. Kelley. Blocking Considerations for Record Linkage Under Conditions of Uncertainty. In *Social Statistics Section*, pages 602–605, 1984.

[12] J. Koberstein and Y.-K. Ng. Using Word Clusters to Detect Similar Web Documents. In *KSEM*, pages 215–228, 2006.

[13] Y. Koren. Factorization Meets the Neighborhood: a Multifaceted Collaborative Filtering Model. In *SIGKDD*, pages 426–434, 2008.

[14] H. Li, Y. Gu, and S. Koul. Review of Digital Library Book Recommendation Models. Available at SSRN. http://ssrn.com/abstract=1513415, 2009.

[15] G. Linden, B. Smith, and J. York. Amazon.com Recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[16] L. Liu, B. Fang, and W. Zhang. Speak the Same Language with Your Friends: Augmenting Tag Recommenders with Social Relations. In *HT*, pages 45–50, 2010.

[17] G. Luger. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*, 5^{th} *Ed*. Addison Wesley, 2005.

[18] Y. Park and K. Chang. Individual and Group Behavior-based Customer Profile Model for Personalized Product Recommendation. *Expert Systems with Applications*, 36(2):1932–1939, 2009.

[19] M. Pera, W. Lund, and Y.-K. Ng. A Sophisticated Library Search Strategy Using Folksonomies and Similarity Matches. *JASIST*, 60(7):1392–1406, 2009.

[20] A. Shepitsen, J. Gemmel, B. Mobasher, and R. Burke. Personalized Recommendations in Social Tagging Systems Using Hierarchical Clustering. In *RecSys*, pages 259–266, 2008.

[21] F. Smith. LibraryThing. *Reference Reviews*, 21(8):5–6, 2007.

[22] J. Wang, Q. Li, and Y. Chen. User Comments for News Recommendation in Social Media. In *SIGIR*, pages 881–882, 2010.

[23] C. Yang, B. Wei, J. Wu, Y. Zhang, and L. Zhang. CARES: a Ranking-oriented CADAL Recommender system. In *JCDL*, pages 203–212, 2009.