2014-02-01

Using Online Data Sources to Make Recommendations on Reading Material for K-12 and Advanced Readers

Maria Soledad Pera
Brigham Young University - Provo

Using Online Data Sources to Make Recommendations on Reading Materials for K-12 and Advanced Readers

Maria Soledad Pera

A dissertation submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Yiu-Kai Ng, Chair
David Embley
Christophe Giraud-Carrier
Eric Ringger
Sean Warnick

Department of Computer Science

Brigham Young University

February 2014

ABSTRACT

Using Online Data Sources to Make Recommendations on Reading Materials for K-12 and Advanced Readers

Maria Soledad Pera
Department of Computer Science, BYU
Doctor of Philosophy

Reading is a fundamental skill that each person needs to develop during early childhood and continue to enhance into adulthood. While children/teenagers depend on this skill to advance academically and become educated individuals, adults are expected to acquire a certain level of proficiency in reading so that they can engage in social/civic activities and successfully participate in the workforce. A step towards assisting individuals to become lifelong readers is to provide them adequate reading selections which can cultivate their intellectual and emotional growth. Turning to (web) search engines for such reading choices can be overwhelming, given the huge volume of reading materials offered as a result of a search. An alternative is to rely on reading materials suggested by existing recommendation systems, which unfortunately are not capable of simultaneously matching the *information needs*, *preferences*, and *reading abilities* of individual readers. In this dissertation, we present novel recommendation strategies which identify *appealing* reading materials that the readers can *comprehend*, which in turn can motivate them to read. In accomplishing this task, we have examined used-defined data, in addition to information retrieved/inferred from reputable and freely-accessible online sources. We have incorporated the concept of "*social trust*" when making recommendations for advanced readers and suggested fiction books that match the *reading ability* of individual K-12 readers using our readability-analysis tool for books. Furthermore, we have emulated the *readers' advisory* service offered at school/public libraries in making recommendations for K-12 readers, which can be applied to advanced readers as well.

A major contribution of our work is in the development of *unsupervised* recommendation strategies for advanced readers which suggest reading materials for both entertainment and learning acquisition purposes. Unlike their counterparts, these recommendation strategies are unaffected by the *cold-start* or *long-tail* problems, since they exploit user-defined data (if available) while taking advantage of alternative publicly-available metadata. Our readability-analysis tool is innovative, which can predict the readability-levels of books *on-the-fly*, even in the absence of excerpts from books, a task that cannot be accomplished by any of the well-known readability tools/strategies. Moreover, our *multi-dimensional* recommendation strategy is novel, since it simultaneously analyzes the *reading abilities* of K-12 readers, *which* books readers enjoy, *why* the books are appealing to them, and *what* subject matters the readers favor. Besides assisting K-12 readers, our recommender can be used by parents/teachers/librarians in locating reading materials to be suggested to their (K-12) children/students/patrons.

We have validated the performance of each methodology presented in this dissertation using existing benchmark datasets or datasets we created for the evaluation purpose (which is another contribution we make to the research community). We have also compared the performance of our proposed methodologies with their corresponding baselines and state-of-the-art counterparts, which further verifies the correctness of the proposed methodologies.

Keywords: Recommendation Systems, Readability, K-12, Readers' Advisory

ACKNOWLEDGMENTS

I first want to thank my PhD committee, who were there every step of this journey known as graduate school. I am very thankful for Dr. Embley and Dr. Christophe Giraud-Carrier, their advice and suggestions were invaluable in completing my work. I also appreciate Dr. Ringger's and Dr. Warnick's willingness to serve as my committee members and their feedback to improve my work.

Thanks go also to Jen, Jenny, and Gordon, who let me hang out at the CS Office on the good days and the not so good ones.

My family and my friends, in the US and in Argentina, were my greatest support system, so to all of you all I can say is thank you... you are awesome!

And of course, I want to thank my advisor, Dr. Yiu-Kai Ng, without whom I could not have produced this research. Dr. Ng has been an incredible mentor, who has helped me grow and improve both academically and personally. His support, encouragement, patience, and help in completing this work and throughout my time at BYU are deeply appreciated.

To my Yaya.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Reading is an activity performed on a daily basis: from reading news articles and books to cereal boxes and street signs. According to the National Institute of Child Health and Human Development, “reading is the single most important skill necessary for a happy, productive, and successful life”.[1] Unfortunately, a significant number of children/teenagers are underachieving at school, especially in reading. The 2013 National Assessment of Educational Progress study[2] shows that only 32% of American 4^{th} graders are proficient in reading. Even more troublesome is the claim made by the US National Center for Education Statistics which states that “children who have not developed some basic literacy skills by the time they enter school are 3 to 4 times more likely to drop out in later years” [1]. This disturbing finding is echoed by the 2013 report made by UNESCO (United Nations Educational, Scientific, and Cultural organization) Institute for Statistics which indicates that global illiterate population rate among youth ascends to 123.5 million [54].

Promoting good reading habits among K-12 students is essential, given the enormous influence of reading on students’ development as learners and members of the society [5]. In fact, reading is a skill that is important not only for K-12 readers, but is for “just about anyone” [84]. A federal finding published by USA Today[3] reveals that approximately 32 million adults in the US have difficulty in reading “anything more challenging than a children’s picture book or to understand a medication’s side effects listed on a pill bottle”. UNESCO also reports that, as of 2013, there are 773.5 million adults with rudimentary or no reading skills worldwide [54]. Adults

[1] http://www.ksl.com/?sid=15431484

[2] http://nationsreportcard.gov/reading_math_2013/#/student-groups

[3] http://usatoday30.usatoday.com/news/education/2009-01-08-adult-literacy_N.htm

who cannot read to their children can have a negative impact on their children/teenagers' early development, as "learning to read begins long before a child enters school".[4] Lack of reading proficiency among adults also negatively affects their participation in the workforce. According to an Educational Testing Service assessment,[5] if the current literacy levels in the US are not improved, "by 2030 American workforce will be unequipped and unskilled to work in the demanding global market."

As stated in [9], "reading fiction actually increases people's emotional intelligence ... their accurate awareness of themselves and others, and their ability to create positive relationships with others based on managing their own reactions". Indeed, there is a clear correlation between the academic performance of students and their reading ability [35]. Given that reading affects both the intellectual and emotional growth of individuals, it is indispensable to provide adequate reading selections and encourage good reading habits among individuals, especially at an early age. A step towards achieving this goal is to identify "the right material to the right audience" [152]; however, this is not a trivial task, since the meaning of "right material" can have multiple connotations, depending on each individual and the task the reader is trying to accomplish. For example, conducting research within the academic setting often requires the researcher to identify articles that match the research context, whereas if reading is solely for entertainment, then finding books appealing to a reader becomes a must.

Finding *relevant items*, such as (non-)fiction books or scholarly articles can be challenging. It is a common practice for readers to turn to trusted advisors, i.e., teachers/mentors/colleagues within the academic environment or parents/peers/librarians within a reader's private surroundings, to quest for needed reading materials. It is unrealistic, however, to expect these "advisors" to keep up with the huge amount of materials on diverse topics which are being published on a regular basis. A reader may also turn to tools available at (scientific) digital libraries, such as ACM Portal, or popular book-related websites, such as Amazon.com, to search for reading materials in various domains. These search tools, however, not only are inadequate for conducting per-

[4] http://www.literacymidsouth.org/resources/literacy-statistics/

[5] http://www.ets.org/Media/Education_Topics/pdf/AmericasPerfectStorm.pdf

sonalized/contextual searches [106] based on (im)properly formatted (complex sentenced-based) queries which express the users' information needs [137], but they also present users with an overwhelming number of items to choose from [106]. Recommendation systems, on the other hand, which are tailored towards offering reading materials in different domains and for various purposes are the solution to the problem. These recommenders can assist readers to cope with the *information overload* problem and minimize the *time* and *efforts* imposed on discovering unknown items that address their information needs [133]. Moreover, recommenders are available for readers anytime and anywhere, can be accessible simultaneously by multiple users, and examine hundreds of available resources in making suggestions. Compared with web search tools, recommenders often manage to identify items of interest for their users, which decreases the number of *unproductive* searches [137] that often frustrate users with failed search experiences.

A number of recommenders in the reading domain have been developed over the past decades [50, 57], which are employed by well-known commercial websites, such as Amazon.com, and social bookmarking sites, such as CiteULike.org and GoodReads.com. We have detected, however, many deficiencies in the design methodologies of these recommenders. Regarding the data required to make suggestions, current state-of-the-art methodologies (i) have not eradicated the *cold start*[6] problem [149] and (ii) are constrained by the requirement of large historical data on their users, which include, but not limited to, ratings, personal tags, and purchasing/accessing patterns, that might not be easily assembled. In terms of their applicability, majority of these recommenders are restricted to operate within a social (bookmarking/networking) setting, since they rely on users' bookmarks/ratings/established connections collected on the respective site for which they are developed. Moreover, the design of a number of existing recommenders ignore the fact that users prefer, whenever possible, suggestions made by people they trust [93, 133]. In addition, these recommenders disregard the reading abilities of their users and may not consider their individual preferences, which is a major concern, especially for K-12 readers, since suggesting books

[6] Cold start problem refers to identifying potential suggestions for "new" users for whom no preference on items are available, or suggesting a "new" item for which no review/rating information exists

that are either too easy/difficult to read or involve topics unappealing to the readers could diminish their interest in reading [7].

We have proposed unsupervised, simple, and yet effective methodologies that address the deficiencies of existing recommendation strategies for identifying reading materials to be suggested for readers that satisfy their specific needs, even in the absence of user-defined data. The major design goal of our recommendation strategies is to provide reading choices to users to motivate them to read, which could enrich their learning [69] and recreational [36] experience, and subsequently sharpen their critical thinking and analytical skills [53].

Our recommendation strategies are innovative, which facilitate the task of finding fiction and non-fiction reading materials of interest for users of all ages. The recommendations are specifically important for K-12 readers who are offered reading materials to choose from, which support their lifelong reading habits [36] and could have a significant impact on their future academic and career development [69].

In developing our recommendation strategies, we have explored user-defined data. We realized that, either for pleasure-reading or learning acquisition, advanced readers favor recommendations made by people they know. We have learned that even though tags capture readers' intents in depicting the content of a particular reading material, they are not always publicly and freely accessible. We have also learned that bookmarking sites do not always archive personal ratings and the interpretations of rating scales differ from site to site. For this reason, we have developed recommendation strategies that do not rely on ratings/tags to make recommendations. Unlike the traditional approaches based on exact matching, we have also validated the correctness of relying on similarity-based approaches for content matching. Moreover, we have observed that design methodologies proven successful for making recommendations for general audiences cannot be applied directly to K-12 readers partially due to the limited data on minors shared/archived by bookmarking sites due to privacy restrictions, such as the Children's Online Privacy Protection Act ("COPPA").[7] Instead of relying on user-defined or provided information, we have identified

[7] http://www.coppa.org/coppa.htm

metadata (inferred from data available at public, reliable online sources) that can be exploited to represent the interests of readers, the contents/general traits of reading materials appealing to them, and their reading ability to make recommendations.

We organized the discussion on the methodologies developed for addressing the deficiencies of existing recommenders on reading materials in five Chapters: Chapters 2 and 3 pertain to recommendation systems developed with advanced, i.e., general, readers in mind, and Chapters 4 to 6 focus on K-12 readers. In Chapter 2 (published[8] in Proceedings of 2011 IEEE/WIC/ACM Joint Conference on Web Intelligence [120]) we introduce *PReF*, a recommender that exploits user-defined data, i.e., personal tags, ratings, and connections, to suggest relevant reading materials for entertaining purposes. At the time of developing *PReF*, trust-aware recommenders either required a user U to explicitly indicate a level of trust on other users of a social site or inferred a trust-network for U based on U's similarity with other users [97]. *PReF* is a social, instead of a trust-aware, recommender that analyzes U's connections already established on a social site without requiring user-defined "degrees of trust", which are seldom archived on social sites and impose a burden on the users to provide explicit feedback. Unlike the recommender in [93] that predicts the rating of books, *PReF* is a top-N recommender which, to the best of our knowledge, was the first to incorporate the concept of trust inferred from social connections in the book domain.

Besides pleasure reading, another type of reading involves academic publications relevant to the research work of a reader. In Chapter 3 (published in the Journal of Intelligent Information Systems [124]), we present *PReSA*, which enhances the recommendation strategy introduced in Chapter 2 for suggesting scholarly articles by analyzing *content descriptors* on articles that are readily available on academic social sites besides considering users' connections. The latter captures the social-trust aspect of *PReSA*, which we have validated to be important in the academic domain as well. Using content descriptions, *PReSA* can make suggestions even in the absence of tag representations, which, along with the exclusion of personal ratings, solves the "cold start" problem. *PReSA*, which considers the immediate information needs of a user, instead of making

[8] The version of the paper included in this dissertation differs from the originally published one, since it includes additional discussions.

suggestions based on the user's profile, discovers literature pertinent to the current research interest of a reader, a highly-desirable service offered to researchers. Unlike its counterparts [24, 109], *PReSA* adopts a *content-similarity* matching strategy to increase the number of relevant publications to be suggested.

As previously stated, strategies proven successful for making recommendations for advanced readers cannot always be applied directly to K-12 readers. This is partially due to privacy issues that limit the type of information that can be shared/archived by bookmarking sites for children and teenagers. For example, BiblioNasium.com, which is a safe and secure social (networking) site on books that targets children and teenagers, neither archives (personal) ratings/tags nor shares friendships/connections established on the site. Furthermore, the aforementioned strategies target solely advanced readers who are assumed to have acquired the same or similar reading abilities; however, this assumption does not hold for recommenders that make suggestions for K-12 audiences whose readability levels can be diverse. For this reason, we have explored alternative data sources and other strategies to suggest reading materials for K-12 readers.

In making recommendations for K-12 audiences, we realized that the readability levels of K-12 readers and books should be determined to ensure that the readers can comprehend the reading materials. We have explored well-known readability formulas/tools [19], such as Flesch-Kincaid, Lexile, and AR, and discovered that they (i) require at least a sample text of a book for readability-level analysis, which are often not available due to the copyright law and (ii) offer readability measures for only a small fraction of published books [19]. To handle these deficiencies we present *TRoLL*, a new readability-analysis tool, in Chapter 4 (in submission [48]), which predicts the reading level of a book B without human involvement. *TRoLL* analyzes features inferred from (i) an *excerpt* of B (if it is available), (ii) the *audience* targeted by B, (iii) *subject areas* (defined by the US Curriculum) that are covered in B, (iv) Library of Congress *Subject Headings* assigned to B, and (v) books written by the *author* of B. Analyzing text-based features from book excerpts is preferable, given its direct impact on the degree of difficulty in understanding the content of a book. Unfortunately, only 7.7% of books in the OCLC database, which is a widely-used worldwide library cooperative, are linked to their partial/full content [32]. Making readability-level predictions on-the-fly, even in the absence of text, is the main contribution of *TRoLL*.

Taking advantage of *TRoLL*, we developed a book recommender tailored to K-12 readers, called *BReK12*, which is introduced in Chapter 5 (published in Proceedings of 2013 ACM Conference on Recommender Systems [122]). *BReK12* is unique, since it suggests books that simultaneously match the preferences and reading level of a user. Unlike current state-of-the-art recommenders, which either rely on the availability of users' ratings [133], purchasing patterns [89], or millions of data points to perform the recommendation task [58], *BReK12* generates relevant book suggestions without requiring historical data in the form of ratings or connections established on a social site that are often lacking among K-12 readers. Furthermore, unlike the "one-size-fits-all" strategy employed by recommenders at well-known book-affiliated sites, such as Novelist and Amazon, which makes the same suggestions to users without considering their individual preferences [89], *BReK12* is *personalized*. This is of special significance, given that not all readers in the same grade or age group have the same reading skill and preference [104].

In the latest development of our recommendation systems, we recognized that libraries, which have been established to champion and encourage reading [138], offer the Readers' Advisory (RA) service. RA identifies reading materials of potential interest to individual readers with "the help of knowledgeable and non-judgmental library staff" [138]. Given the correlation between the ultimate goal of RA and the major design goal of our work in suggesting the "right" reading material for the "right" audience, it seemed natural to extend *BReK12* in Chapter 5 by emulating the RA process. In order to fully automate the RA process, we have designed *Rabbit*, the book recommender discussed in Chapter 6 (in submission [125]), which can simultaneously offer the RA service to any number of readers anywhere and anytime, a task that cannot be achieved by traditional RA [159]. *Rabbit* is novel, since it considers the *reading ability* of a reader and the *subjects* that matter the most to the reader, besides examining the *types* of books that are appealing to the reader and the *reasons* behind the interest of the reader in these books. The latter requires the analysis of appeal factors, i.e., literary elements, of a book B that stimulate a reader's subconscious, emotional reaction to B, which impacts the perceptions of the reader on B. Descriptions on the literary elements of books, however, are not easily accessible, which are either determined by professionals on-the-fly or accessed through a paid subscription to RA databases. *Rabbit* automatically generates descriptions of appeal factors that apply to books using a comprehensive taxonomy on literary elements, in addition to applying simple natural language processing techniques that examine the semantic connections between words in book reviews.

Last, we give concluding remarks and address directions for future work in Chapter 7.

We introduce one of our earlier research work on group recommendations in the Appendix of this dissertation. We chose not to include *GroupReM* (published in the Journal of Information Processing and Management [123]), in the body of this dissertation, since *GroupReM* was designed to recommend movies, instead of reading materials. *GroupReM* adopts a content-based strategy, which is similar to the ones employed by the aforementioned recommenders. However, *GroupReM* focuses on satisfying the information needs of a *group* of users (regardless of the group's *size* and the degree of *cohesiveness* among group members), which is unlike the recommenders discussed in Chapters 2 to 6 that are tailored to *individual* users. Given the unsupervised and domain-independent nature of *GroupReM*, its design methodology can be adopted to make recommendations on multimedia items other than movies.

Chapter 2

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

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.

2.1 Introduction

In recent years social websites, such as Facebook(.com), Twitter(.com), YouTube(.com), and Delicious(.com), have become increasingly popular [64]. 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 [10, 30, 64]. Newly-developed recommenders, such as [64, 90, 154], 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 [74]. 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

[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.

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 members of a social website, since as stated in [10], 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* [78], 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 2.2, we discuss existing (book) recommendation systems. In Section 2.3, we detail the design of $PReF$. In Section 2.4, we present the results of the empirical study conducted for assessing the performance of $PReF$. In Section 2.5, we give a conclusion.

2.2 Related Work

Machine learning, information retrieval, natural language processing, and probabilistic models have been adopted for developing systems that recommend (web) documents [61], songs [31], and movies [81], to name a few.

Content-based and *collaborative filtering* are two well-known approaches for making recommendations [117]. 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 [46, 117] 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 [4].

There exist a number of book recommendation systems [89, 117, 156]. Amazon's recommender [89] 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. [156] 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 [117] create a user-profile P based on individual and group behaviour, 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 [88].

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 [64, 90, 143, 154] employ data extracted from social sites to suggest items other than books. Wang et al. [154] 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. [64] 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. [90] and Shepitsen et al. [143] develop different approaches for generating personalized tag recommendations. In accomplishing the task, the authors in [90] combine collaborative information extracted from social tagging systems, such as Delicious, and the users' personalized tag preferences, whereas the authors in [143] apply a hierarchical agglomerative clustering algorithm to identify users' individual interests. Unlike $PReF$, none of the approaches in [90, 143, 154]

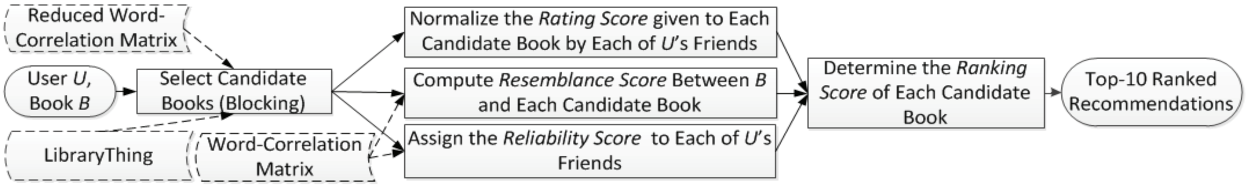

Reduced Word-Correlation Matrix
User *U*, Book *B*
Select Candidate Books (Blocking)
LibraryThing
Word-Correlation Matrix
Normalize the *Rating Score* given to Each Candidate Book by Each of *U*'s Friends
Compute *Resemblance Score* Between *B* and Each Candidate Book
Assign the *Reliability Score* to Each of *U*'s Friends
Determine the *Ranking Score* of Each Candidate Book
Top-10 Ranked Recommendations

Figure 2.1: Processing steps of the proposed book recommender, $PReF$

rely on friendships established on social websites or tag similarity matching in performing the recommendation task.

2.3 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 [146]. 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 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 2.3.2). Hereafter, $PReF$ computes the *ranking score* of each book CB in $Canditate\_Set$ (as defined in Section 2.3.3) and the top-N ($\geq$ 1) ranked books are recommended to LT_User. The overall process of $PReF$ is illustrated in Figure 2.1.

2.3.1 Word Correlation Factors

$PReF$ relies on the pre-computed word-correlation factors in the word-correlation matrix [78] 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.

2.3.2 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

[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* [75] is a *filtering* technique which reduces the potentially very large number of comparisons to be made among records [33].

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 2.3.1) 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 [126].

2.3.3 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 2.3.3), (ii) the *rating score* assigned to CB by each friend of LT_User, who includes CB in his/her personal catalog (in Section 2.3.3), and (iii) the relative *degree of reliability* of each of LT_User's friends (in Section 2.3.3).

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 2.3.1, instead of using the reduced word-correlation matrix employed in Section 2.3.2, 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

$$Resem(Source\_Bk, CB) = \frac{\sum_{i=1}^{n} Min\{\sum_{j=1}^{m} wcf(Source\_Bk_i, CB_j), 1\} \times freq_i}{\sum_{i=1}^{n} freq_i} \tag{2.1}$$

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 2.1 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 $Source\_Bk$.

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:

$$Rate(CB, LT\_Pal) = \frac{Rating\_CB}{5} \tag{2.2}$$

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 $Rating\_CB$.

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

[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.

$Source\_ Bk$. $PReF$ 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:

$$Rel(Source\_Bk, LT\_Pal) = \frac{\sum_{i=1}^{m} Min\left\{\sum_{j=1}^{n} wcf(Source\_Bk_i, LT\_Pal_j), 1\right\}}{m} \tag{2.3}$$

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 2.3, m is the normalization factor that scales the corresponding *degree of reliability* in a [0, 1] range.

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:

$$\text{Rank(CB)} = Max_{LT\_Pal_i \in Pal_{CB}}\{Rel(Source\_Bk, LT\_Pal_i) \times$$

$$\frac{Resem(Source\_Bk, CB) + Rate(CB, LT\_Pal_i)}{1 - Min\{Resem(Source\_Bk, CB), Rate(CB, LT\_Pal_i)\}}\} \tag{2.4}$$

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 *Max* function in Equation 2.4 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 2.1 and 2.2, respectively) using the *Stanford Certainty Factor* (SCF) [91], $PReF$ 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 [91] in Equation 2.4, $PReF$ *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 2.1, 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 2.1). The remaining recommended books, such as “The Odyssey”, are considered non-relevant recommendations for “Emma” by Soleenusa. □

2.4 Experimental Results

In this section, we first introduce the data and metrics in Sections 2.4.1 and 2.4.2, 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 2.4.3.

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 Lover's Complaint	Wuthering Heights
7	**Emma (Norton Critical Edition)**	The Odyssey	The Tenant of Wildfell Hall
8	**Pride and Prejudice (Norton Critical Edition)**	Alice's 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 2.1: Recommendations generated by $PReF$, Amazon, and LibraryThing, respectively in response to the book "Emma", by Jane Austen

2.4.1 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[6] 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.

[6] To conduct the initial empirical evaluation on the performance of $PReF$, we relied on 25 LibraryThing users, randomly selected from the site, that had at least one explicitly established connection with other site members.

2.4.2 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* [42].

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 $PReF$).

$$P@K = \frac{\sum_{i=1}^{N} \frac{Number\_of\_Related\_Recommendations_i}{K}}{N} \tag{2.5}$$

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 relatedness of the recommendations positioned at the top, $middle$, and $overall$ in the ranking, respectively. Since, as stated in Section 2.3.3, 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 [42].

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.

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{r_i} \tag{2.6}$$

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 2.5.

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 [143], 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:

$$imp = \frac{\sum_{i=1}^{N} \frac{1}{rp_i} - \frac{1}{rb_i}}{N} \tag{2.7}$$

where N and i are as defined in Equation 2.5, 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 [143].

2.4.3 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,[7] which are two well-known, commercial book recommenders. While the recommender of Amazon has been introduced in Section 2.2, 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

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

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 2.4.1) 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).

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 2.2, 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 2.2 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 2.2. Based on the $P@10$

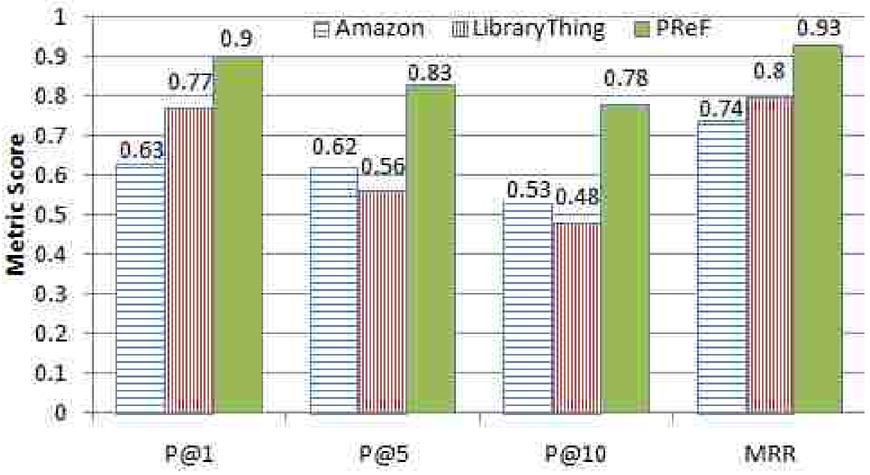


Figure 2.2: $Precision@K$ (K = 1, 5, and 10) scores on the (top-10) recommendations achieved by $PReF$, Amazon, and LibraryThing for the books in $Test\_Books$

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 2.3, 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 2.3 $PReF$ achieves a 25% (16%, respectively) improvement over Amazon (LibraryThing, respectively) in generating books recommendations relevant, i.e., appealing, to users.

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 recom-

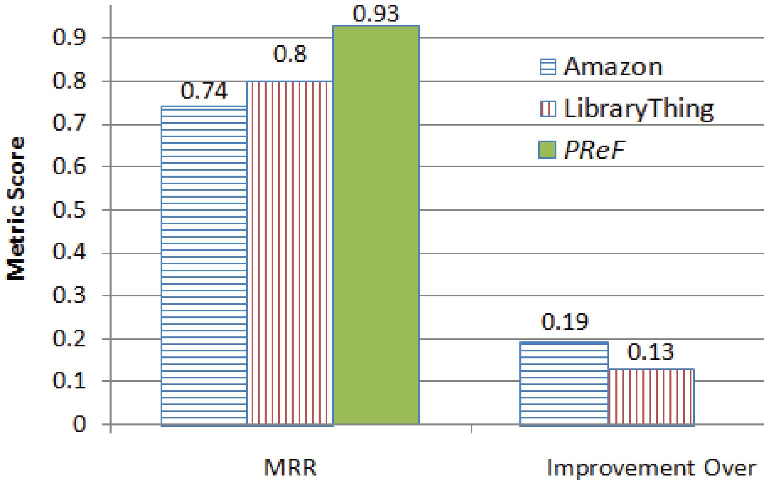


Figure 2.3: MRR and Imp scores based on (un)related books recommended by $PReF$, Amazon, and LibraryThing for the books in $Test\_Books$

mendations, 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 2.2 and 2.3, $PReF$ 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.

2.5 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.

Chapter 3

Exploiting the Wisdom of Social Connections to Make Personalized Recommendations on Scholarly Articles

Abstract:

Existing scholarly publication recommenders were designed to aid researchers, as well as ordinary users, in discovering pertinent literature in diverse academic fields. These recommenders, however, often (i) depend on the availability of users' historical data in the form of ratings or access patterns, (ii) generate recommendations pertaining to users' (articles included in their) profiles, as oppose to their current research interests, or (iii) fail to analyze valuable user-generated data at social sites that can enhance their performance. To address these design issues, we propose $PReSA$, a personalized recommender on scholarly articles. $PReSA$ recommends articles bookmarked by the *connections* of a user U on a social bookmarking site that are not only *similar* in *content* to a target publication P currently of interest to U but are also *popular* among U's connections. $PReSA$ (i) relies on the content-similarity measure to identify potential academic publications to be recommended and (ii) uses only information readily available on popular social bookmarking sites to make recommendations. Empirical studies conducted using data from CiteULike have verified the efficiency and effectiveness of (the *recommendation* and *ranking* strategies of) $PReSA$, which outperforms a number of existing (scholarly publication) recommenders.

3.1 Introduction

Web search tools employed by scientific digital libraries, such as ACM Portal and IEEE Xplore, are designed to retrieve archived publications in diverse technological fields using keyword queries.

These search tools, however, not only are inadequate for performing personalized/contextual searches, but also present users with an overwhelming number of items to choose from [106]. To assist users to cope with the information overload problem and minimize the time and efforts imposed on the users in discovering articles that are appealing, but unknown, to them, a number of publication recommenders have been developed [23, 50].

Non-personalized recommenders on scholarly articles archived at well-known digital libraries adopt a "one-size-fits-all" strategy which suggests the same publications to users without considering any external knowledge about the users [111]. Existing personalized recommenders, on the other hand, either (i) require historical data in the form of ratings or citations [108], which may not be publicly available, or (ii) rely on the exact string-matching approach to compare the content of potential publications to be recommended with their counterparts in users' profiles [109], which excludes articles of interests to their users that are represented using analogous, but different, keywords. Furthermore, most of these recommenders fail to consider their users' current information needs and solely focus on the general interests of the users instead [103], which demonstrates that personalization is not yet fully exploited.

To suggest academic citations relevant to a target publication P of interest to a user U, a common inquire conducted within the academic setting these days, we introduce $PReSA$, a personalized recommender on scholarly publications. Based on the premise that a person often turns to other people whom (s)he trusts when seeking advice [93, 133], $PReSA$ considers the connections explicitly established by U on a social bookmarking site and identifies (candidate) articles to be recommended among the ones bookmarked by U's connections that are similar in *content* to P, but are not necessarily described using the same keywords/tags as the ones assigned to P. To determine which candidate publication CP should be recommended, $PReSA$ applies a ranking strategy based on the weighted linear combination of multiple *content descriptors* of CP, which include its *title*, *abstract*, and *tags*, in addition to the *popularity* of CP among U's connections.

Unlike collaborative-filtering-based recommendation systems, $PReSA$ does not require user-feedback in the form of ratings. Rather, it takes advantage of data, which include bookmarked items, content descriptors on items, and users' connections, readily available on popular sites that archive scholarly publications, such as CiteULike.org. As opposed to employing an exact-matching strategy, $PReSA$ relies on a similarity-matching approach based on *word-correlation factors* [78] applied to the content descriptors of publications to make recommendations. Moreover, $PReSA$, which requires neither domain-specific information nor training (other than to determine the weight of content descriptors and popularity measures), can easily be adopted to suggest items other than publications, provided that the required data, i.e., tags, title, a short description of each item and social interactions, are available.

We have conducted an empirical study using data from CiteULike which validates the efficiency and effectiveness of $PReSA$. The conducted study also demonstrates that $PReSA$ outperforms a number of traditional (social) recommendation systems (on scholarly publications).

The remaining of this paper is organized as follows. In Section 3.2, we discuss existing publication recommenders. In Section 3.3, we detail the design of $PReSA$. In Section 3.4, we present the empirical study conducted to assess the performance of $PReSA$. In Section 3.5, we give a conclusion.

3.2 Related Work

During the past decade, the demand for recommendation/ filtering systems to solve the web information overload problem has become apparent [23]. While a number of recommenders have been developed to suggest a variety of items, such as news articles/blog posts, songs, and movies, only few recommenders are designed for suggesting scholarly publications [23].

The recommender proposed in [16] applies a content filtering approach based on the Vector Space Model and TF-IDF weighting scheme to assign conference paper submissions to program committee members for review. Sugiyama and Kan [148] consider scholarly articles published by an author A, in addition to reference papers cited in work published by A, to recommend publications that appeal to A. Ekstrand et al. [50] measure the influence of a scholarly publication among a web of citations, using algorithms such as HITS and PageRank, to enhance existing content-based and collaborative filtering recommendation approaches. Parra and Brusilovsky [118] introduce two variants of a user-based collaborative filtering method which analyze the information available on existing tagging systems, such as CiteULike, for suggesting scientific articles. Pudhiyaveetil et al. [128] employ a concept-based approach that relies on ACM classification tree and documents accessed by users on CiteSeer to create user profiles and suggest items to the users.

The recommenders in [16, 50, 118, 128, 148] suggest publications that appeal to a user's general interests, as opposed to $PReSA$ which recommends publications relevant to a target research paper currently of interest to the user. Furthermore, unlike $PReSA$, these recommenders rely either on information inferred from a network of citations, the actual content of publications, or user-feedback in the form of ratings, which are not always publicly available [82].

The authors in [24] compare user- and item-based collaborative filtering algorithms designed for recommending scholarly publications. They analyze metadata, such as tags or author(s), available at social bookmarking sites and use well-known fusion strategies, such as CombSUM and CombMNZ, to combine the output lists of recommendations generated by the algorithms. Nascimento et al. [109], on the other hand, represent each publication P as a vector V such that each component in V indicates the weighted frequency of a term in the abstract or title of P and calculate the similarity among publications using the well-known cosine metric. Unlike $PReSA$, the recommenders in [24, 109] do not employ a similarity measure and thus cannot identify publications similar in content that are represented using different keywords/tags which are analogous in meaning.

To make recommendations, the approach in [72] employs an LDA-based model, which determines the research problem and the proposed solution presented in the abstract of a target paper, along with a citation graph of academic publications.The recommender in [72], like $PReSA$, makes recommendations based on a target paper P. However, while the former suggest publications matching either the problem or the solution presented in P, the latter suggests publications matching the diverse subject areas addressed in P.

Bellogin et al. [18] compare the performance of content-based/collaborative filtering recommenders with their proposed social recommender using data from Last.FM. The latter applies the traditional collaborative filtering strategy but replaces the set of nearest neighbors of a user U by users who are explicitly connected to U. $PubRec$ [121] also considers articles bookmarked by connections of a user U on CiteULike to make recommendations relevant to a scholarly article P of interest to U. Unlike $PubRec$, $PReSA$ does not require the *ratings* of publications to generate recommendations and takes full advantages of a variety of content descriptors, other than CiteULike tags, to more precisely compute the degree of resemblance between P and a publication to be recommended. Besides the recommenders in [18, 121], recommenders such as the ones proposed in [27, 77, 79] also incorporate explicit relationships established among social site users to enhance the recommendation process [27, 77, 79]. However, to the best of our knowledge, none of the existing recommenders on *scholarly publications* (with the exception of $PubRec$) is based on the premise that people "tend to rely more on recommendations from people they trust than online systems which generate recommendations based on anonymous similar people" [133]. Indeed, $PReSA$ considers *explicit relationships* among users on a social site as part of its recommendation process.

3.3 Our Proposed Recommender

In this section, we detail the design of $PReSA$ which relies on data extracted from CiteULike to make personalized recommendations of scholarly articles. Given a CiteULike user $Cusr$, $PReSA$ first identifies $Cusr$'s *connections* (as discussed in Section 3.3.1). Using word-correlation factors (introduced in Section 3.3.2), $PReSA$ determines the set of publications, denoted $CandidateP$, among the ones in the personal libraries of $Cusr$'s connections that are similar in content (to a certain degree) to a target publication P in which $Cusr$ is interested (as detailed in Section 3.3.3). Hereafter, $PReSA$ recommends to $Cusr$ the 10 publications in $CandidateP$ with the highest

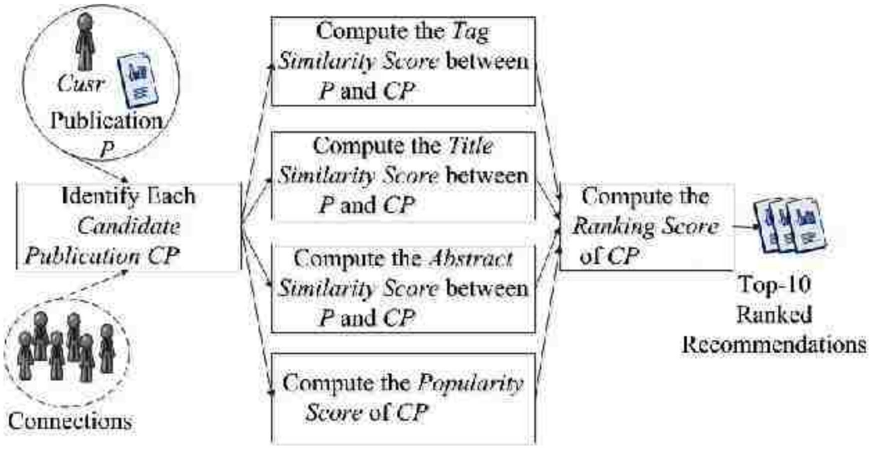

Cusr
Publication
P
Identify Each
Candidate
Publication CP
Connections
Compute the Tag Similarity Score between P and CP
Compute the Title Similarity Score between P and CP
Compute the Abstract Similarity Score between P and CP
Compute the Popularity Score of CP
Compute the Ranking Score of CP
Top-10
Ranked
Recommendations

Figure 3.1: The architecture of the proposed personalized recommender on scholarly articles, $PReSA$

ranking scores (computed in Section 3.3.4). The recommendation process of $PReSA$ is illustrated in Figure 3.1.

3.3.1 CiteULike

CiteULike, which is one of the leading social sites established for managing and sharing bibliographic references, includes 6,462,237 indexed articles (as of November 24, 2012) and allows its users to organize, post, and search publication information. A CiteULike user's *personal library* includes a number of bibliographic references bookmarked by the user. Besides bookmarking publications and maintaining their metadata, CiteULike users can add *personal comments*, *ratings*, and *tags* to publications in their personal libraries [23]. Each publication P indexed in CiteULike can be assigned a list of tags by the CiteULike users who have bookmarked P. The list is used by $PReSA$ to infer the tag cloud of P, which is a global visual representation of the *tags* assigned to P, including their *frequencies*.

CiteULike offers its users an infrastructure to establish *explicit* communication channels with other CiteULike users. Explicitly-connected users, called *connections* in CiteULike, can exchange private messages and share bibliographic references of interest with one another.[1]

[1] See wiki.citeulike.org/index.php/Social_Features for all the social features offered by CiteULike.

3.3.2 Word Correlation Factors

$PReSA$ relies on the pre-computed word-correlation factors (bounded between 0 and 1) in the word-correlation matrix [78] to determine the *similarity* between any two tags/keywords. The word-correlation factors are used for identifying *candidate publications* to be considered for recommendation (as detailed in Section 3.3.3) and determining the *degrees of resemblance* of candidate publications to a given publication based on their tag clouds/titles/abstracts (as discussed in Section 3.3.4, 3.3.4, and 3.3.4, respectively).

Word-correlation factors were created using a set of approximately 880,000 Wikipedia documents (wikipedia.org). Each correlation factor indicates the degree of similarity of the two corresponding words[2] 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 with different writing styles, and the documents cover a wide range of topics with diverse word usage and contents. Compared with the sets of synonyms/related words compiled by the well-known 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.

3.3.3 Selecting Candidate Publications

As previously stated, since $PReSA$ is based on the premise that a user tends to trust recommendations made by other users with whom (s)he has an explicit connection, scholarly publications recommended by $PReSA$ to each user $Cusr$ are publications chosen from the personal libraries of $Cusr$'s connections on CiteULike. As the number of publications bookmarked by each of $Cusr$'s connections can be large, i.e., in the thousands, it is inefficient to compare each publication with P, a publication of interest to $Cusr$, to identify the ones similar to P to be recommended to $Cusr$. This computation can significantly prolong the recommendation process of $PReSA$. To minimize

[2] Words in the Wikipedia documents were $stemmed$ after all the $stopwords$ were removed. From now on, unless stated otherwise, (key)words/tags refer to *non-stop*, *stemmed* (key)words/tags.

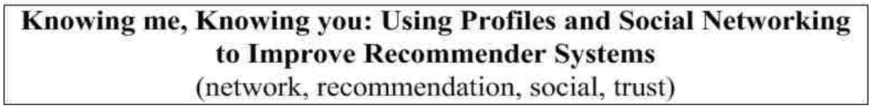
the number of comparisons and thus reduce the processing time required in generating recommendations, $PReSA$ applies a $blocking$ strategy[3] on articles included in the personal libraries of $Cusr$'s connections to generate a subset of articles (excluding P), denoted $CandidateP$, to be considered for recommendation. Each publication in $CandidateP$ contains at least one tag *exactly matching* or *highly similar* to one of the personal tags of P assigned by $Cusr$. As publications in $CandidateP$ and P share same (or analogous) tags, it is anticipated that they are similar in content to a certain degree and address the *same* or *similar* topic. If $Cusr$ has not assigned any personal tags to P, $PReSA$ relies on the tags in the tag cloud of P to perform the blocking task.

To identify highly similar tags, $PReSA$ employs a reduced version of the word-correlation matrix (introduced in Section 3.3.2) 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 version of the word-correlation matrix to determine potentially similar publications, the overall processing time of $PReSA$ is significantly reduced without affecting its accuracy [121].

Example 2 Consider the scholarly publication SP_A entitled "Knowing me, Knowing you: Using Profiles and Social Networking to Improve Recommender Systems" that has been bookmarked by $Cusr_A$, a real CiteULike user. (The CiteULike user name of $Cusr_A$ is not used due to the privacy policy established by CiteULike.) The title of SP_A and the tags assigned to SP_A by $Cusr_A$ (as appeared in CiteULike) are shown in Figure 3.2.

$PReSA$ first identifies $CandidateSP_A$, i.e., the set of candidate publications considered to be recommended to $Cusr_A$ based on his/her interest in SP_A. Applying the *blocking strategy*, $PReSA$ decreases the number of publications to be considered from 1,221, which is the total number of distinct publications bookmarked by $Cusr_A$'s connections, to 199, which yield $CandidateSP_A$. Figure 3.3 shows (a few of) the publications bookmarked by $Cusr_A$'s connections, along with the personal tags assigned by the connections who include the corresponding

[3] A blocking strategy is a $filtering$ technique that reduces the potentially very large number of comparisons to be made among records, i.e., publications in CiteULike in our case.



Knowing me, Knowing you: Using Profiles and Social Networking to Improve Recommender Systems
(network, recommendation, social, trust)

Figure 3.2: A publication, SP_A, bookmarked by $Cusr_A$, a CiteULike user, along with the tags assigned to SP_A by $Cusr_A$

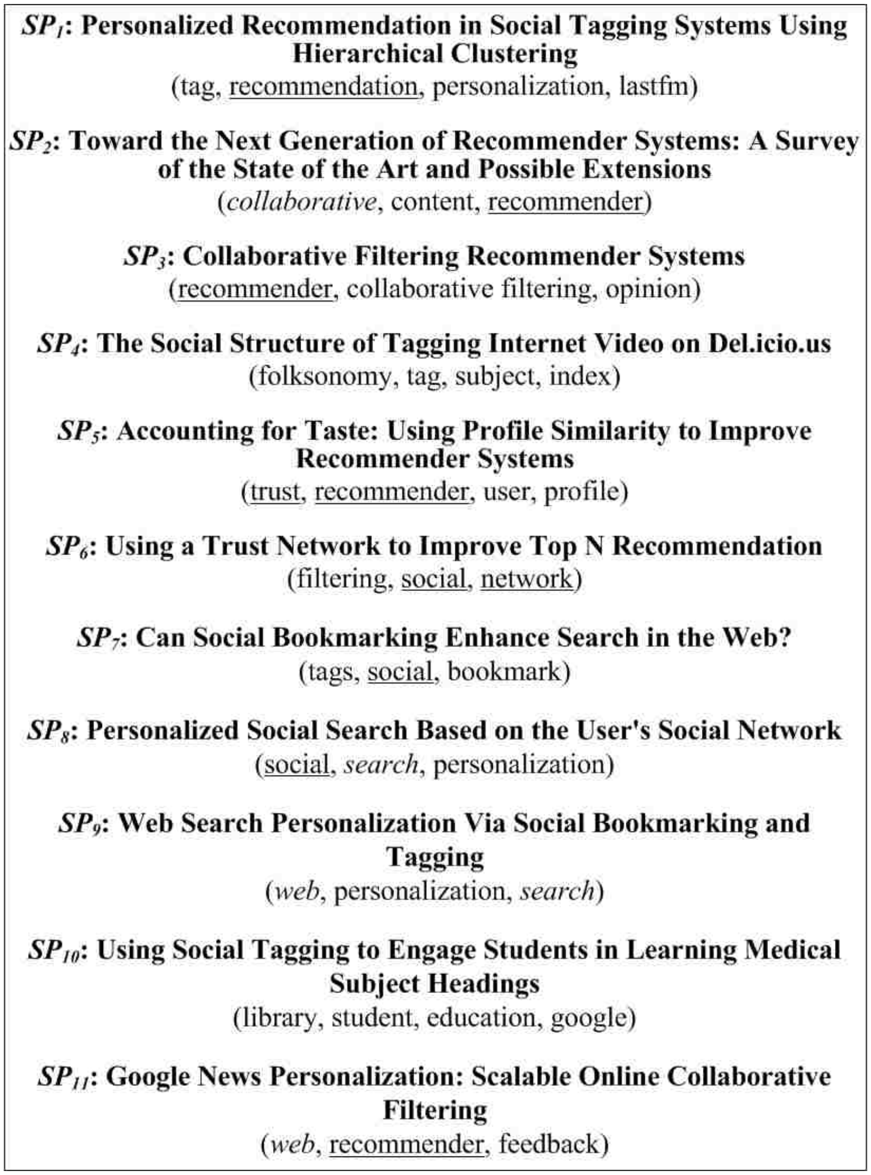
publications in their personal libraries. A subset of the publications in $CandidateSP_A$ is shown in Figure 3.4. □

3.3.4 Ranking of Scholarly Publications

$PReSA$ ranks each publication CP in $CandidateP$ to prioritize them for recommendation using the (i) *degree of similarity* between P and CP according the tags in their corresponding tag clouds (calculated in Section 3.3.4), (ii) *title similarity* between P and CP (determined in Section 3.3.4), (iii) *abstract similarity* between P and CP (computed in Section 3.3.4), and (iv) *number of $Cusr$'s connections* who include CP in their personal libraries (discussed in Section 3.3.4). Note that (i), (ii), and (iii) capture the *content similarity* between P and CP, whereas (iv) reflects the *popularity* of CP.

Tag Similarity of P and CP

To determine the $similarity$ between the tags assigned to P and CP, denoted $Sim(P, CP)$, $PReSA$ adds the word-correlation factors between each tag in the tag cloud of P and CP in CiteULike, respectively. Tags in the tag cloud of P (CP, respectively) are considered, since these tags provide a more comprehensive description of (the content of) P (CP, respectively), as opposed to the personal tags assigned to P (CP, respectively), which only reflect the personal opinion of, and vocabulary used by, $Cusr$ (one of $Cusr$'s connections, respectively) in describing (the content of) P (CP, respectively). The word-correlation matrix introduced in Section 3.3.2 is used, instead of the reduced word-correlation matrix used in Section 3.3.3 for blocking, since the former provides



SP_1: Personalized Recommendation in Social Tagging Systems Using Hierarchical Clustering
(tag, recommendation, personalization, lastfm)

SP_2: Toward the Next Generation of Recommender Systems: A Survey of the State of the Art and Possible Extensions
(*collaborative*, content, recommender)

SP_3: Collaborative Filtering Recommender Systems
(recommender, collaborative filtering, opinion)

SP_4: The Social Structure of Tagging Internet Video on Del.icio.us
(folksonomy, tag, subject, index)

SP_5: Accounting for Taste: Using Profile Similarity to Improve Recommender Systems
(trust, recommender, user, profile)

SP_6: Using a Trust Network to Improve Top N Recommendation
(filtering, social, network)

SP_7: Can Social Bookmarking Enhance Search in the Web?
(tags, social, bookmark)

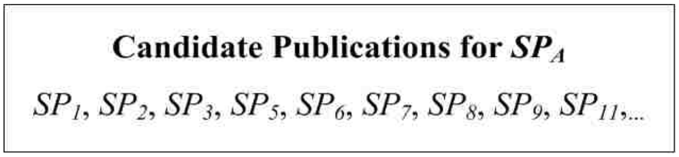
SP_8: Personalized Social Search Based on the User's Social Network
(social, *search*, personalization)

SP_9: Web Search Personalization Via Social Bookmarking and Tagging
(*web*, personalization, *search*)

SP_{10}: Using Social Tagging to Engage Students in Learning Medical Subject Headings
(library, student, education, google)

SP_{11}: Google News Personalization: Scalable Online Collaborative Filtering
(*web*, recommender, feedback)

Figure 3.3: A subset of the publications bookmarked by $Cusr_A$'s connections, along with their corresponding sets of *personal tags*. Tags *exactly matching* (*highly similar* to, respectively) the ones shown in Figure 3.2 are underlined (italicized, respectively)



Candidate Publications for SP_A

SP_1, SP_2, SP_3, SP_5, SP_6, SP_7, SP_8, SP_9, SP_{11},...

Figure 3.4: Some publications in $CandidateSP_A$

a more accurate overall similarity measure between (all the tags representing) P and CP than the latter.

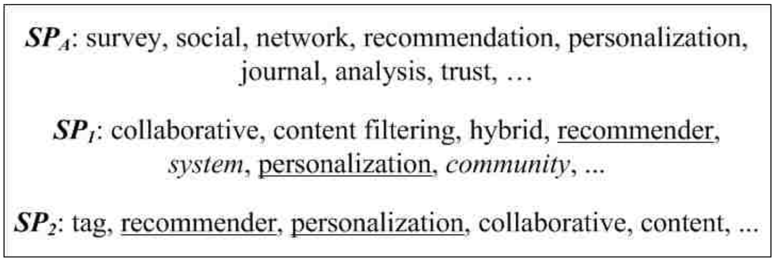
$$Sim(P, CP) = \sum_{i=1}^{n} Min\{\sum_{j=1}^{m} wcf(P_i, CP_j), 1\} \times freq_{P_i} \tag{3.1}$$

where n (m, respectively) is the number of distinct tags in the tag cloud of P (CP, respectively), P_i (CP_j, respectively) is a tag in the tag cloud of P (CP, respectively), $wcf(P_i,\ CP_j)$ is the correlation factor of P_i and CP_j in the word-correlation matrix, and $freq_{P_i}$ denotes the number of times P_i is assigned to P as specified in the tag cloud of P.

$Freq_{P_i}$ in Equation 3.1, which indicates the number of CiteULike users who have chosen P_i to tag P, reflects the *relative degree of significance* of P_i in representing the content of P. The *larger* $freq_{P_i}$ is, the *more significant* P_i is in representing the content of P, which is based on the assumption that a content-indicator tag is more likely chosen by users to identify the content/topic of its corresponding publication.

The Min function in Equation 3.1 imposes a constraint on adding the word-correlation factors of tags assigned to P and CP. Even if a tag in the tag cloud of P (i) matches exactly *one* of the tags and (ii) is similar to *some* of the remaining tags in the tag cloud of CP, which yields a value greater than 1.0, i.e., the word-correlation factor of an *exact* match, $PReSA$ limits the sum of their similarity measures to 1.0. This constraint ensures that if P contains a *dominant* tag T in its tag cloud, i.e., T is highly similar to a few tags in the tag cloud of CP, T alone cannot significantly impact the similarity value between P and CP, i.e., "one does not represent all." Tags in the tag cloud of P that are similar to most of the tags in the tag cloud of CP should yield a higher degree of similarity of P and CP than the existence of only one dominant tag in P.

Example 3 Consider a snapshot of the tags in the tag clouds of SP_A, SP_1, and SP_2 as shown in Figure 3.5. The tag clouds of both SP_1 and SP_2 include two tags that exactly match their corresponding tags in the tag cloud of SP_A, i.e., "recommender" and "personalization," and thus should be treated as equally similar to SP_A using the *exact-matching* approach. By employing the



***SP**_A*: survey, social, network, recommendation, personalization, journal, analysis, trust, …

***SP**_1*: collaborative, content filtering, hybrid, recommender, *system*, personalization, *community*, ...

***SP**_2*: tag, recommender, personalization, collaborative, content, ...

Figure 3.5: A snapshot of the (inferred) *tag clouds* of SP_A, SP_1, and SP_2 in Figures 3.2 and 3.3, respectively such that the tags of SP_1 and SP_2 that exactly-match (are highly similar to, respectively) the tags describing SP_A are underlined (italicized, respectively)

word-correlation factors, however, $PReSA$ is able to determine a more accurate degree of similarity among the publications. The degree of similarity (computed using Equation 3.1) between SP_A and SP_1 is *3.8*, whereas the similarity between SP_A and SP_2 is only *2.9*, which accurately reflects that SP_1, a publication that discusses "the use of social networking data to enhance recommenders," is more similar (in content) to SP_A than SP_2, which presents several strategies for "improving the recommendation task." □

Title Similarity of P and CP

As stated in [161], the title of a publication captures its content, i.e., its subject matter. It deems appropriate to consider the degree of resemblance between P and CP partially based on (the similarity of) their respective titles.

$PReSA$ computes $Sim(T_P, T_{CP})$, which is the similarity between the titles T_P of P and T_{CP} of CP, using Equation 3.1, where n (m, respectively) is the number of distinct keywords in the title of P (CP, respectively), P_i (CP_j, respectively) is a keyword in the title of P (CP, respectively), $wcf(P_i, CP_j)$ is the correlation factor of P_i and CP_j in the word-correlation matrix, and $freq_{P_i}$ denotes the number of times P_i appears in the title of P.

Abstract Similarity of P and CP

Besides the titles of publications, abstracts are publicly available as metadata on publications [109]. Unlike the tag cloud of P which characterizes the corresponding users' personal preferences, in terms of using chosen keywords, to describe P, the abstract of P is a brief summary of the content of P created by its author(s), which provides its readers a quick overview of P [161].

Along with the tag and title similarity between P and CP, $PReSA$ considers the abstract similarity of P and CP to partially determine the degree of relevance of CP with respect to P. As defined for $Sim(T_P, T_{CP})$, the similarity between the abstracts A_P of P and A_{CP} of CP, $Sim(A_P, A_{CP})$, is computed using Equation 3.1.

Popularity of Scholarly Publications

In addition to computing the content similarity between P and CP, $PReSA$ further considers the *popularity* of CP, which is determined by the number of $Cusr$'s connections who include CP in their personal libraries.

Publications that attract the attention of $Cusr$'s connections are more likely bookmarked in the personal libraries of $Cusr$'s connections. $PReSA$ considers the fact that publications frequently-bookmarked by $Cusr$'s connections may also be of interest to $Cusr$, since $Cusr$ and his/her connections share common interests to a certain degree. While solely relying on the popularity of an item in making recommendation leads to less personalized recommendations (which does not apply to $PReSA$), Adomavicius and Kwon [3] claim that the accuracy of recommendations can be enhanced by considering the popularity of the item, along with other measures, during the recommendation process.

$PReSA$ computes the *popularity score* of CP, which is ranged between 1 and the total number of connections of $Cusr$ on CiteULike, as another factor to be considered for its recommendation to $Cusr$.

$$Popularity(CP) = \sum_{Ccon \in Connections_{Cusr}} Bookmarked(Ccon, CP) \tag{3.2}$$

where $Connections_{Cusr}$ is the set of $Cusr$'s connections and $Bookmarked(Ccon, CP)$ is "1" if $Ccon$ includes CP in his/her personal library, and is "0", otherwise.

Ranking Score

After computing the (i) degree of *content similarity* between P and each CP in $CandidateP$ based on their respective tag clouds, titles, and abstracts and (ii) the $popularity$ score of CP, $PReSA$ calculates the $ranking$ score of CP, denoted $Rank(CP)$, by employing the well-known *weighted linear combination strategy* [109].

$$Rank(CP) = \sum_{c=1}^{N} w_c \times nScore_c(CP) \tag{3.3}$$

where N is set to 4, which is the number of distinct measures to be considered in positioning CP in the ranking of recommended articles and are the four scores computed for CP in Sections 3.3.4-3.3.4, i.e., $Sim(P, CP)$, $Sim(T_P, T_{CP})$, $Sim(A_P, A_{CP})$, and $Popularity(CP)$, $nScore_c(CP)$ is the *normalized* score calculated by the c^{th} $(1 \leq \text{c} \leq 4)$ measure for CP, and w_c is the weight of the c^{th} measure.

In computing $Rank(CP)$, it is necessary to scale the original scores determined by each of the measures in Sections 3.3.4-3.3.4, respectively into a *common range*, which can be achieved by applying Equation 3.4 so that each normalized score is within the range [0, 1], a common range [24].

$$nScore_c(CP) = \frac{Score_c(CP) - Score_c^{min}}{Score_c^{max} - Score_c^{min}} \tag{3.4}$$

where $Score_c(CP)$ is the score of CP computed using the c^{th} measure, and $Score_c^{max}$ ($Score_c^{min}$, respectively) is the maximum (minimum, respectively) value computed using measure c on publications in $CandidateP$.

Rather than treating each measure computed for CP as *equally significant* in determining the overall ranking score of CP, $PReSA$ considers the degree of *importance*, i.e., *weight*, of each individual measure defined in Sections 3.3.4-3.3.4, respectively (as shown in Equation 3.3). To define these weights, $PReSA$ relies on SVM^{rank}, which is the implementation of RankSVM[4] [73]

[4] The implementation of SVM^{rank} is available at cs.cornell.edu/people/tj/svm_light/svm_ rank.html.

based on the Support Vector Machine (SVM) classification method. As detailed in [17], SVM^{rank} solves a maximum-margin optimization problem by finding a hyperplane, defined as W = <w_1, w_2, w_3, w_4>, which is the vector of weights associated with each of the four measures computed by $PReSA$. The hyperplane provides an ideal separation of candidate publications into relevant and non-relevant. Each training instance I, which represents an article A (not) to be recommended to a CiteULike user U based on U's interest in a target publication and is used as an input to the RankSVM learning process, is a vector of the form I= <m_1, m_2, m_3, m_4, b>, where m_1, m_2, m_3, and m_4 are the measures computed for A in Sections 3.3.4-3.3.4, respectively and b is either "1" or "0" which indicates whether A is a relevant or non-relevant recommendation for U.

We rely on RankSVM to establish the weight of each of the four measures (i.e., w_c in Equation 3.3), since RankSVM (i) is publicly available, as opposed to other proprietary methods, (ii) requires a very short, one-time training processing step,[5] and (iii) is highly accurate [17].

Having calculated the $ranking$ score of each scholarly article in $CandidateP$ using Equation 3.3, $PReSA$ recommends the top-10 ranked publications to $Cusr$.

Example 4 $PReSA$ computes (i) the *tag*, *title*, and *abstract similarity* between each one of the 199 candidate publications CP_A and SP_A (as shown in Figure 3.2 and mentioned in Example 1) and (ii) the *popularity* of CP_A among $Cusr_A$'s connections. Table 3.1 displays the normalized scores, i.e., Tag Sim(ilarity), Title Sim(ilarity), Abs(tract) Sim(ilarity) of the publications explicitly shown in Figure 3.4 with respect to SP_A, along with their normalized Popularity scores.

Using the weighted linear combination strategy (as defined in Equation 3.3), $PReSA$ identifies the most relevant publications (based on their respective $Rank$ scores) to SP_A. The ranking positions of the articles suggested by $PReSA$ (i.e., publications explicitly shown in Figure 3.4) for $Cusr_A$ based on $Cusr_A$'s interest in SP_A are displayed in Table 3.1. The publications suggested by $PReSA$ are not only (to a certain degree) *similar* (in content) to SP_A (as determined by both manual examination and their relatively high tag, title, and abstract similarity with respect to SP_A),

[5] We have empirically established that, on the average, it takes 4 seconds to train the RankSVM using 11,000 training instances to determine the weight of each measure employed by $PReSA$ for ranking candidate publications. The training instances do not overlap with the dataset described in Section 3.4.1, which is used to assess the overall performance of $PReSA$.

Publication	Tag Sim	Title Sim	Abs Sim	Popularity	Rank Score	Rank Order
SP_1	0.54	0.48	0.39	0.60	4.21	4
SP_2	0.62	0.49	0.52	0.33	4.07	5
SP_3	0.38	0.36	0.41	0.33	2.95	6
SP_5	**0.86**	**0.93**	**0.72**	**0.66**	**6.13**	**1**
SP_6	0.81	0.71	0.65	0.66	5.82	2
SP_7	0.23	0.14	0.19	0.33	2.00	8
SP_8	0.08	0.32	0.10	0.25	1.08	9
SP_9	0.13	0.09	0.22	0.66	2.46	7
SP_{11}	0.57	0.00	0.41	0.80	4.80	3

Table 3.1: Normalized scores computed by $PReSA$ between each candidate publication explicitly listed in Figure 3.4 and SP_A shown in Figure 3.2, along with the ranking position of each candidate publication

but they are also *popular* (based on the frequency in which they were bookmarked) to a certain degree among $Cusr_A$'s connections.

As shown in Table 3.1, publications SP_5 and SP_6 are positioned first and second, respectively, in the list of publications ordered by $PReSA$ with respect to SP_A. SP_5 and SP_6, which detail different strategies that consider user profiles and data extracted from social websites to make recommendations, are closely related to the topic covered in SP_A, which introduces an approach that "relies on data extracted from social networking environments to enhance the performance of recommenders." SP_{11} is also related (to a certain degree) to SP_A, since SP_{11} considers "information included in user profiles and user activities to perform the recommendation task". The remaining publications, as shown in Table 3.1, are also related to SP_A, but to a lesser degree. While SP_1, SP_2, and SP_3 describe "alternative approaches adopted for generating recommendations", SP_7, SP_8, and SP_9 consider "data extracted from social environments to enhance web searches". □

3.3.5 Observations

$PReSA$ recommends the top-10 ranked publications to its users, since users often view up to the first 10 generated results [68]. Besides using data extracted from CiteULike to make recommendations, as mentioned earlier, $PReSA$ can perform the recommendation task on publications using collaborative data and user connections extracted from *any* social website developed for managing academic references.

Instead of relying on a single descriptor in performing the recommendation task, $PReSA$ considers in-tandem multiple measures to determine the (i) publications that are to be recommended to a user based on a particular scholarly article of interest to the user and (ii) order in which the selected publications should be positioned in the ranking. $PReSA$ takes into account the *strength* of each measure defined in Sections 3.3.4-3.3.4, respectively, regardless of the actual (high or low) value of a particular measure, to position a publication in the ranking. Note that the *popularity* and *tag similarity* scores computed by $PReSA$ are always non-zero, since publications in $CandidateP$ (i) must be bookmarked by at least one of $Cusr$'s connections and (ii) include at least a tag (assigned by at least one of $Cusr$'s connections) in their respective tag clouds exactly-matched with, or highly similar to, a tag assigned to P.

Example 5 Consider publications SP_1 and SP_{11} for which their respective tag, title, abstract, and popularity scores are shown in Table 3.1. The (normalized) *tag/abstract* similarity and *popularity* scores of SP_1 are lower than the respective ones computed for SP_{11}. The scores indicate that SP_{11} is more similar in content to SP_A (based on their respective tag clouds and abstracts) and more popular among $Cusr_A$'s connections than SP_1. The *title* similarity score of SP_1, however, is larger than its counterpart for SP_{11}, which is zero. Since $PReSA$ considers multiple descriptors to determine the ranking of publications to be recommended to $Cusr_A$, even though the keywords in the title of SP_{11} are less similar to the ones in the title of SP_A than the keywords in the title of SP_1, $PReSA$ positions SP_{11} higher in the ranking than SP_1. $\Box$

3.4 Experimental Results

In this section, we first introduce the dataset, evaluation protocol, and metrics used for assessing the performance of $PReSA$ (in Sections 3.4.1, 3.4.2, and 3.4.3, respectively). Hereafter, we detail the empirical study conducted for evaluating the effectiveness and efficiency of $PReSA$ and compare its performance with existing (scholarly publication) recommenders (in Section 3.4.4).

Dataset	
Number of Distinct CiteULike Users	261
Number of Distinct Publications	107,161
Number of Publications Bookmarked by Active Users	34,569
Number of Distinct Tags Assigned to Publications	39,858
Number of Distinct Personal Tags Created by CiteULike Users	22,926
Average Number of Connections Per Active User	4.2
Average Number of Personal Tags Per Publication	3.8
Average Number of Distinct Tags (in the Inferred Tag Cloud) Per Publication	6.8

Table 3.2: Statistical data on the dataset used for evaluating the performance of $PReSA$

3.4.1 Dataset

To the best of our knowledge, there is no publicly available dataset for evaluating the performance of a recommender on scholarly publications that includes explicit connections established among users on a social bookmarking site. For this reason, we constructed a dataset using data extracted from CiteULike. We randomly selected a subset of users, who have explicit connections and recently bookmarked articles on CiteULike, and call them "active users." For each active user U, we determined the connections of U and retrieved the personal tags of each article posted in U's personal library (under the personal libraries of U's connections, respectively). The dataset also includes the set of tags (along with their frequencies of occurrence) in the (inferred) tag cloud of each scholarly publication bookmarked by either an active user or his/her connections. The resultant dataset includes 261 CiteULike users, who are either active users or their connections, and 107,161 distinct scholarly publications in the personal libraries of the active users and their connections. (Detailed information on the dataset is shown in Table 3.2.) Since, as previously stated, $PReSA$ generates personalized recommendations for a user based on his/her interest on a particular publication, we evaluate $PReSA$ based on its recommendations generated for each one of the 34,569 user-publication pairs, denoted UP_pairs, in the constructed dataset.

3.4.2 Evaluation Protocol

We assess the overall performance of $PReSA$, using the metrics to be introduced in Section 3.4.3, on the recommendations generated by $PReSA$ for each user(U)-publication(P) pair in UP_pairs. As a ground truth, i.e., to determine whether a recommendation R generated by $PReSA$ for a U-P pair is relevant, we depend on the publications bookmarked by U on CiteULike. R is *relevant* if it is included in U's personal library (excluding P) and is *non-relevant*, otherwise, which is a commonly-employed protocol for evaluating a recommender system [18, 23, 133].

Since only publications in a user's personal library are considered *relevant*, it is not possible to account for potentially relevant publications the user has not bookmarked. Thus, the results of the conducted empirical study (as presented in Section 3.4.4) are underestimated, which is a well-known limitation of the evaluation protocol applied to assess recommender systems [18, 133]. As this limitation affects all the evaluated recommenders, i.e., $PReSA$ and a number of recommendation strategies employed to suggest scholarly publications as discussed in Section 3.4.4, the measures computed in the empirical study are consistent for comparison purposes [18].

3.4.3 Metrics

We treat $PReSA$ as a content-retrieval system which, instead of identifying relevant ranked items in response to a query, recommends to its users a list of top-10 ranked publications relevant to a publication, which is a popular evaluation strategy [18, 23]. We apply *Precision@K* ($P@K$), *Mean Reciprocal Rank* (MRR), and *Normalized Discounted Cumulative Gain* ($nDCG$) to evaluate the performance of $PReSA$.

$P@K$ measures the *relevance* of the top-K ranked recommendations generated by $PReSA$ for each user-publication pair in UP_pairs. We consider $P@1$ and $P@10$ to evaluate the degree of relevance of each $first$ recommendation made by $PReSA$ and the $overall$ effectiveness of the recommendations generated by $PReSA$, respectively.

MRR is the averaged sum of the ranking values of recommendations generated by $PReSA$ for each user-publication pair in UP_pairs such that each ranking value is either the *reciprocal* of the ranking position of the *first* relevant recommendation among each set of top-10 recommendations, if it exists, or 0, otherwise.

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{r_i} \tag{3.5}$$

where r_i is the ranking position of the *first relevant* recommendation among the top-10 with respect to the i^{th} $(1 \leq i \leq N)$ user-publication pair in UP_pairs, if it exists, and N is the total number of user-publication pairs in UP_pairs.

$P@K$ does not evaluate the relative *ranking* of all the relevant publications, whereas MRR solely focuses on the (average) ranking position of the *first* relevant publication recommended by $PReSA$. We compute the $nDCG$ to determine the overall ranking performance of $PReSA$. $nDCG_{10}$ (defined in Equation 3.6 for assessing the top-10 publications recommended by $PReSA$) *penalizes* relevant publications ranked *lower* in the list of recommendations. The penalization is based on a relevance reduction, which is logarithmically proportional to the position of each relevant publication in a ranked list (as shown in Equation 3.7). The *higher* the $nDCG_{10}$ score is, the *better* the ranking strategy adopted by a recommender system RS is, since relevant recommendations generated by RS are positioned high.

$$nDCG_{10} = \frac{1}{N}\sum_{i=1}^{N}\frac{DCG_{10,i}}{IDCG_{10,i}} \tag{3.6}$$

where N and i are as defined in Equation 3.5,

$$DCG_{10,i} = \sum_{j=1}^{10}\frac{(2^{rel_j} - 1)}{log_2(1+j)} \tag{3.7}$$

where rel_j is the binary relevant judgment of the publication at the j^{th} $(1 \leq j \leq 10)$ ranking position and is assigned a value of “1” if the publication is *relevant* and is “0”, otherwise. Furthermore,

$IDCG_{10,i}$ (in Equation 3.6) is the best possible $DCG_{10,i}$ value for the recommendations generated by $PReSA$ for the i^{th} user-publication pair.[6]

3.4.4 Performance Evaluation

In this section, we first verify the correctness of $PReSA$ using multiple evidences for identifying scholarly publications to be recommended (in Section 3.4.4) and demonstrate the efficiency of $PReSA$ (in Section 3.4.4) in suggesting publications. Thereafter, we compare the performance of $PReSA$ with a number of existing recommenders (in Section 3.4.4) and offer our insights on the performance of each evaluated recommender (in Section 3.4.4).

Effectiveness of $PReSA$

As stated in Section 3.3.4, $PReSA$ relies on a number of *content similarity* scores (computed using word-correlation factors) and the *popularity* score of a given publication to generate recommendations. To verify the correctness of the strategy employed by $PReSA$ to suggest publications, we first compared alternative implementations of $PReSA$ using each user(U)-publication(P) pair in UP_pairs.

The first alternative implementation of $PReSA$, denoted *Exact*, considers the tag description of P and a candidate publication CP and uses the $Dice$ coefficient [42] on exactly-matched tags in the tag cloud of CP and P to determine their degree of resemblance. Unlike *Exact*, WCF, the second alternative implementation of $PReSA$, relies on the word-correlation factors and considers analogous, besides exactly-matched, tags. WCF computes the Sim score of each candidate publication with respect to P as defined in Equation 3.1. We consider further alternative implementations which, besides using the tag similarity among publications, incorporate in-tandem the title similarity, i.e., TT, the title and abstract similarity, i.e., TTA, and the title and abstract similarity along with the popularity, i.e., $TTAP$, respectively. The aforementioned alternatives employ an

[6] $IDCG_{10,i}$ is computed as $DCG_{10,i}$ using an *ideal* ranking such that the recommendations are arranged in descending order of their relevant judgment scores in the ranking.

unweighted linear combination strategy (as the one shown in Equation 3.3 without the weights) on the computed normalized scores.

As shown in Figure 3.6, based on the results conducted using UP_pairs, WCF improves the accuracy of recommendations generated by $Exact$ as indicated by the increase in (average) $nDCG_{10}$ achieved by WCF over $Exact$, which indicates that relaxing the exact-matching constraint by adopting word-correlation factors enhances the effectiveness of the recommendations. Moreover, the constant improvement on the (average) $nDCG_{10}$ achieved by considering the tag, $title$, and $abstract$ similarity scores demonstrates the importance of relying on various content descriptors on publications to suggest scholarly articles of interest to a user. Furthermore, the increase in (average) $nDCG_{10}$ achieved by $TTAP$ shows that by using the *popularity* measure, along with the similarity of diverse content descriptors, the quality of the generated recommendations is further enhanced.

Figure 3.6 also illustrates the benefit of considering the *weight* (presented in Section 3.3.4) of each individual score. $PReSA$ achieves more than 2% (statistically significant) increase on $nDCG_{10}$ over $TTAP$, since the latter treats each similarity/popularity score equally important in generating recommendations and thus may impose a bias towards recommending publications for which only one of the computed scores is considerably larger than the remaining scores.

We assess the correctness of applying the *blocking strategy* introduced in Section 3.3.3 as part of the recommendation process of $PReSA$, which minimizes the number of comparisons required to make recommendations. As shown in Figure 3.6, $PReSA\_NB$, which does not apply the blocking strategy, increases the (average) $nDCG_{10}$ achieved by $PReSA$ by 1%. The average time required for $PReSA$ to generate each set of top-10 recommendations, i.e., 567 milliseconds, is more than 3 times *faster* than the average time required, i.e., 2,109 milliseconds, when no blocking strategy is applied. As the cost (in time) outweighs the benefit, in terms of improvements in accuracy, we consider the 1% increase in $nDCG_{10}$ relatively unimportant.

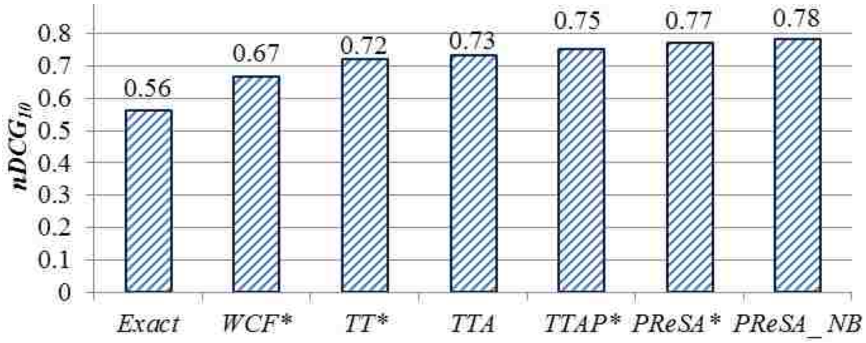


Figure 3.6: $nDCG_{10}$ scores achieved by alternative implementations of $PReSA$ using UP_pairs, where an implementation marked with "*" achieves a statistically significant difference in $nDCG_{10}$ with respect to the implementation shown to its immediate left

An alternative implementation of $PReSA$ that achieves a difference in $nDCG_{10}$ statistically significant, as determined using a Wilcoxon Rank Sum Test ($p < 0.05$), with respect to the implementation shown immediately to its left in Figure 3.6, is marked with "*".

Efficiency of $PReSA$

To assess the efficiency of $PReSA$, we have examined the processing time required by $PReSA$ to recommend scholarly publications for each user-publication pair in UP_pairs. The *average* time it takes $PReSA$ to make recommendations for a pair in UP_pairs is 567 milliseconds, as mentioned in Section 3.4.4. Figure 3.7 shows the time it takes to recommend publications for each pair in UP_pairs with respect to the number of candidate publications to be considered by $PReSA$ for the corresponding pair. Even when the number of candidate publications is in the thousands, $PReSA$ makes recommendations in between 2 to 5 seconds, which demonstrates the scalability of $PReSA$.

Comparing $PReSA$ with Other Recommenders

To further demonstrate and verify the effectiveness of $PReSA$, we compare its performance with two well-known, widely-adopted recommender systems, $SocialRecommender$ (SR) [18] and $TagVectorSimilarity$ (TVS) [61], in addition to three state-of-the-art recommenders, $L\text{-}Cosine$ (Cos) [109], $Fusion$ [24], and $PubRec$ [121]. TVS, which represents each publication as a

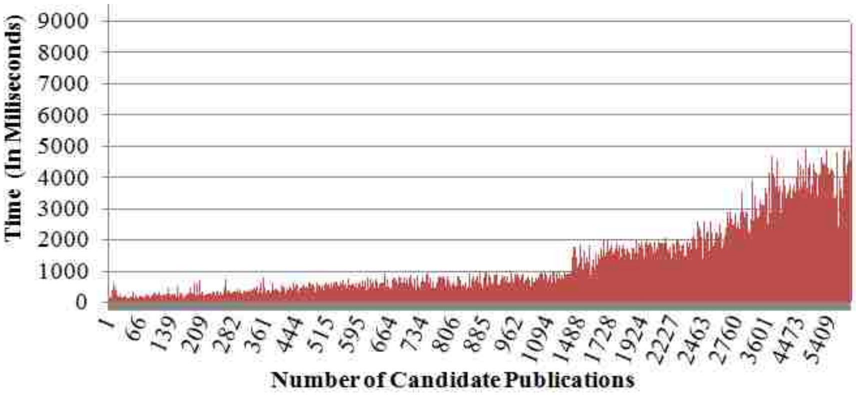


Figure 3.7: Processing time of $PReSA$ required to generate recommendations relative to the number of candidate articles considered by $PReSA$ for the corresponding user-publication pair in UP_pairs

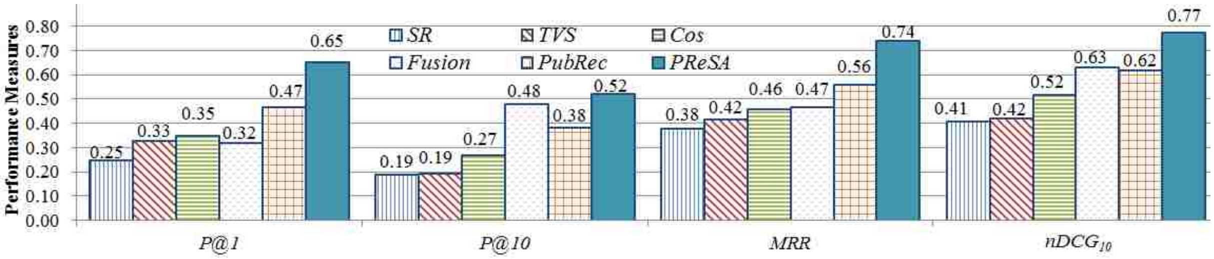


Figure 3.8: P@1, P@10, MRR, and $nDCG_{10}$ scores of SR, TVS, Cos, $Fusion$, $PubRec$, and $PReSA$, respectively

TF-IDF tag profile vector, computes the cosine similarity among tag vector representations to determine the publications to be recommended to a user. SR, Cos, $Fusion$, and $PubRec$ have been briefly introduced in Section 3.2.

Given that $PReSA$, SR, and $PubRec$ recommend publications included in the personal libraries of active users' connections, we restricted the publications to be considered for recommendation by TVS, Cos, and $Fusion$ to the ones bookmarked by active users' connections. We determined the *relevance* of each recommendation made by either SR, TVS, Cos, $Fusion$, or $PubRec$, for each user-publication pair in UP_pairs according to the evaluation protocol detailed in Section 3.4.2.

As shown in Figure 3.8[7], the average $P@1$ score achieved by SR, TVS, Cos, $Fusion$, or $PubRec$ is lower than the average $P@1$ score achieved by $PReSA$. The latter indicates that for 65% of the user-publication pairs in UP_pairs, the first publication recommended by $PReSA$ is relevant, as opposed to at most 47% relevant publications recommended first when considering SR, TVS, Cos, $Fusion$, or $PubRec$. In addition, the $P@10$ value, as shown in Figure 3.8, reflects that more than half of the publications recommended by $PReSA$ are relevant, which doubles the average $P@10$ achieved by SR, TVS, and Cos, respectively.

Figure 3.8 also includes the MRR scores of SR, TVS, Cos, $Fusion$, $PubRec$, and $PReSA$ which demonstrates that while on the average $PReSA$ and $PubRec$ users are required to browse through less than (top) *two* ($\cong \frac{1}{0.74}$ = 1.35, $\cong \frac{1}{0.57}$ = 1.75, respectively) recommended publications before locating one that is relevant to a scholarly publication that (s)he is interested in, users relying on SR, TVS, Cos, or $Fusion$, respectively are required to scan through close to *three* ($\cong \frac{1}{0.38}$ = 2.63) and at least *two* ($\cong \frac{1}{0.42}$= 2.38, $\cong \frac{1}{0.46}$= 2.17, $\cong \frac{1}{0.47}$= 2.12, respectively) recommended publications before locating one that is of interest. In addition, the $nDCG_{10}$ score of $PReSA$ is at least 14% higher than the $nDCG_{10}$ score computed for SR, TVS, Cos, $Fusion$ or $PubRec$. A higher $nDCG_{10}$ value indicates that $PReSA$ is more effective than SR, TVS, Cos, $Fusion$, and $PubRec$ in ranking *higher* in the list of recommended scholarly articles that are relevant.

Discussion

TVS, Cos, $Fusion$, $PubRec$, and $PReSA$ outperform SR, regardless of the performance metric considered, which is anticipated, since the Pearson Coefficient [42] employed for identifying "similar-minded" connections of a particular user U is "0" when U and the corresponding connection C do not include matching publications in their libraries, which in turn causes none of the publications bookmarked by C to be eligible for recommendation. Furthermore, SR relies on explicit user-feedback, i.e., ratings, which may not always be available [82]. $PReSA$ outperforms

[7] Unless stated otherwise, the improvements in $P@1$, $P@10$, MRR, and $nDCG_{10}$ achieved by $PReSA$ with respect the other recommenders considered for comparison purpose are statistically significant.

TVS and Cos in terms of $P@1$, $P@10$, MRR, and $nDCG_{10}$, since $PReSA$ (i) determines the degree of *content similarity* among scholarly articles using word-correlation factors, instead of the *exact-matching* constraint imposed on publications that is required by TVS and Cos and (ii) relies on both the *content* (captured by tags and keywords in titles and abstracts of publications) and *popularity* of scholarly publications to make recommendations, which enhances the quality and ranking of the recommended publications, as opposed to TVS (Cos, respectively) that solely considers tags (keywords in titles and abstracts, respectively) to represent the content of publications to generate recommendations.

Even though $Fusion$ and $PReSA$ consider tags to represent the contents of publications and the keywords in their respective abstracts and titles to perform the recommendation task, $PReSA$ relaxes the exact-matching constraint imposed by $Fusion$ and treats publications represented with analogous, but not the same, tags/keywords as similar, which explains why $PReSA$ outperforms $Fusion$ in making recommendations. $Fusion$ also considers the similarity between any two publications based on the overlap in users that have bookmarked both items in a social site. This strategy depends on detailed information of publications that have been bookmarked by *each* individual user of a social bookmarking site, which is not always publicly available. Furthermore, due to the underlying design of different social bookmarking sites, these information can be difficult to compile.

Unlike $PubRec$, which simply analyzes the content of publications to be recommended based on the tags available on CiteULike, $PReSA$ relies on content descriptors on publications defined by their corresponding author(s) and users of a social bookmarking site. Subsequently, $PReSA$ is able to more effectively capture the content of publications and better estimate their resemblance with respect to a given publication of interest to a user, which in turn enhances the overall quality of the recommendations.

3.4.5 Applying PReSA to Other Domains

In developing PReSA, we have relied on easily-accessible online data sources. PReSA, which is not domain-specific, can easily be extended to suggest items other than scholarly publications.

PReSA partially depends on the existence of users' connections to make recommendations. Similar to CiteULike, majority of popular social bookmarking sites offer their users an option to establish online relationships with other users/members of the corresponding website. For example, Netflix.com allows its users to connect with their Facebook friends to access movies they have watched in the past, whereas Delicious.com offers the functionality for users to network with other users and access their bookmarked URLs. Furthermore, LibraryThing.com users can explicitly befriend other users of the site and explore the books they have read/bookmarked. Given the widely-available connection information on social bookmarking sites, they can be used to identify *candidate items* (other than academic publications) to be recommended, in addition to computing their popularity scores using the approaches described in Sections 3.3.3 and 3.3.4, respectively. Moreover, most of the well-known social bookmarking sites, such as MovieLens.org, LibraryThing.com, and Delicious.com, to name a few, archive tag-based descriptions of items. With the existence of these resources, one can apply the *tag-based filtering* and *similarity* strategies employed by PReSA (as defined in Sections 3.3.3 and 3.3.4) to recommend items such as movies, books, and URLs, respectively. Furthermore, PReSA considers the title and a brief description (i.e., abstract) of each publication to analyze the similarity of publications based on information offered by professionals (i.e., authors), as opposed to personal user descriptions captured by tags. This type of metadata is easily accessible from online sources, such as Amazon.com, RottenTomatoes.com, and Last.FM, making it possible to apply the content matching strategies defined on Sections 3.3.4 and 3.3.4 to suggest items such as books, movies, and songs, respectively.

3.5 Conclusions

Researchers, as well as ordinary users, often turn to scholarly publication recommenders to locate pertinent literature in diverse academic fields. These recommenders, however, adopt the "one-size-fits-all" strategy or suggest articles matching the general interests of its users, which fail to consider users' specific needs on citations relevant to a target publication, a research task performed in the academic setting on a regular basis. To address these issues, we have introduced $PReSA$, a personalized recommender on scholarly articles. $PReSA$, which focuses on the interest of a user U on a target publication P, is based on the premise that U treasures recommendations made by people with whom U has an explicit connection, a design methodology that differs from existing recommenders which suggest items inferred from users unknown to U. With that in mind, $PReSA$ suggests to U publications which have been bookmarked by his/her connections that are *popular* (among U's connections) and *similar* (in content) to P.

$PReSA$ requires neither supervision (other than to determine the weight of content descriptors and popularity measures) nor domain-specific information to make recommendations. Moreover, $PReSA$ does not rely on the availability of ratings on publications provided by its users. Instead, $PReSA$ depends on data, such as bookmarked items or content descriptors on items, readily available on social sites. Furthermore, $PReSA$ relaxes the exact-matching constraint imposed by existing recommenders to determine suggested items by using pre-computed *word-correlation factors* on tags/keywords of archived scholarly articles and the ones that capture the content of a target publication.

To assess the performance of $PReSA$, we have conducted an empirical study using data from CiteULike. Experimental results have verified (i) the efficiency and effectiveness of (the *recommendation* and *ranking* strategies of) $PReSA$ and (ii) the superiority of $PReSA$, in terms of performance, over a number of recommenders (on scholarly articles).

The current design of PReSA analyzes users' personal opinions, in terms of user-defined keywords (i.e., tags), in describing the subject matter of a publication, in addition to considering brief descriptions and titles provided by the corresponding authors of the publication, to make recommendations. Keywords that correctly capture the topic(s) of a publication, however, may not always be available on the tag cloud or in the title/abstract of the publication. As part of our future research work, we plan to consider existing models, such as LDA, which can depict the topic of a publication using its entire content and incorporate its inferred information in the PReSA's recommendation process, which could further improve the correctness of its generated suggestions.

The quality and quantity of relevant suggestions made by PReSA could also be enhanced by analyzing publications beyond the ones that have been bookmarked by users' respective connections. We intend to extend the current design methodology of PReSA by including references in candidate publications, besides the candidate publications themselves, which have been bookmarked by users' connections.

Chapter 4

A Readability Level Prediction Tool for K-12 Books

Abstract:

The readability level of a book is a useful measure for children and teenagers (teachers, parents, and librarians, respectively) to identify reading materials suitable for themselves (their K-12 readers, respectively). Unfortunately, majority of published books are assigned a readability level range, such as K-3, instead of a single readability level for their intended readers by professionals, which is not useful to the end-users who look for books at a particular grade level. This leads to the development of readability formulas/analysis tools. These formulas/tools, however, require at least an excerpt of a book to estimate its readability level, which is a severe constraint due to copyright laws that often prohibit book content from being made publicly accessible. To alleviate the text constraint imposed on readability analysis on books, we have developed TRoLL, which relies heavily on metadata of books that is publicly and readily accessible from reputable book-affiliated online sources, besides using snippets of books, if they are available, to predict the readability level of books. Based on a multi-dimensional regression analysis, TRoLL determines the grade level of any book instantly, even without a sample text in the book, which is its uniqueness. Furthermore, TRoLL is a significant contribution to the educational community, since its computed book readability levels can (i) enrich K-12 readers' book selections and thus can enhance their reading for learning experience, and (ii) aid parents, teachers, and librarians in locating reading materials suitable for their K-12 readers, which can be a time-consuming and frustrating task that does not always yield a quality outcome. Empirical studies conducted using a large set of K-12

books have verified the prediction accuracy of TRoLL and demonstrated its superiority over existing well-known readability formulas/analysis tools.

4.1 Introduction

As reading is an essential skill [135], which can have significant impact on a youth's educational and future career development, it is imperative to encourage children to read and learn starting from an early age. Reading for learning, however, cannot take place unless readers can accurately and efficiently decode, i.e., comprehend, the words in a text [112]. During the last century, educators and researchers have dedicated resources to develop readability assessment tools/formulas which quantify the degree of difficulty in understanding a text [19, 52].

Traditional readability formulas, such as Flesch-Kincaid (Reading Ease) [76], simply perform a one-dimensional analysis on a text based on shallow features, such as the average number of syllables per word (words per sentence, respectively), the average sentence length, and vocabulary lists, which might not precisely capture the complexity of a text.[1] More recently-developed readability formulas have gone beyond shallow features and rely on natural language processing tools to examine complex linguistic features on a text [52]. All of these formulas, however, require a given (snippet of a) text in order to determine its readability level (i.e., grade level), which is a constraint if applied to books, since even an excerpt of a book is not always freely accessible due to copyright laws. The same constraint affects Lexile Framework [145] and Advantage-TASA Open Standard for Readability (ATOS) [139], two widely-used readability analysis tools these days specifically developed for analyzing the readability level of books.

To address the deficiencies of the designs of existing readability formulas/analysis algorithms, we propose a tool for regression analysis of literacy levels, denoted TRoLL, which considers metadata of books publicly accessible from reputable online sources, in addition to snapshots of books only if they are available, to predict the grade level of any book. To determine the grade

[1] Davison and Kantor [45] claim that "nonsense text" can be classified as easy-to-read by traditional readability formulas if it contains frequently-used, short words organized into brief sentences.

level of a book Bk, TRoLL extracts from well-known sources, such as WorldCat.org, (i) an excerpt from Bk (if it is available online) to analyze various shallow features, determine its subject area established by the US Curriculum, and examine different grammatical concepts in Bk; (ii) the subject headings assigned to Bk; and (iii) the targeted audience level of Bk and its (first) author,[2] besides the subject headings and audience levels of books written by the author.

TRoLL predicts the grade level of K-12 books. The grade levels can serve as a guideline for young readers to select books by themselves, which is a valuable, and often overlooked, tool, since "when students choose books that match their interests and level of reading achievement, they gain a sense of independence and commitment and they are more likely to complete, understand, and enjoy the book they are reading" [92]. The grade levels predicted by TRoLL on (non-)fictional books and textbooks can also be used as a guidance for parents, teachers, and librarians in locating reading materials suitable for their K-12 readers.

TRoLL is unique, since it can predict the grade level of a book instantly, even if its sample text is unavailable online. TRoLL performs a multi-dimensional analysis on the metadata/content of books and their authors to accurately predict the readability level of books. Unlike other readability formulas/tools, such as Lexile, which predict the difficulty of a text based on their own readability-level scales, TRoLL predicts the grade level of a book, a measure preferred by teachers/librarians, given that grade levels are easy to understand and use when communicating with students/patrons [132].

The main contribution of TRoLL is in its development as a tool that can determine the grade level of books on-the-fly, requiring $solely$ on publicly available information on books and without involving human experts. This task cannot be accomplished by existing text-based readability formulas nor the popular Lexile or ATOS that offer readability measures for only a small fraction of published books and require direct involvement from their developers in order to generate the readability level of books that have yet to be analyzed [19]. As a by-product of our work, we have created a dataset consisting of more than 18,000 books with their respective grade level ranges

[2] We have empirically verified that by considering only the *first* author of a K-12 book, the processing time of TRoLL is minimized without affecting its accuracy in predicting the grade level of the book.

defined by their corresponding publishers. Given the difficulty in obtaining large-scale datasets on books for training/testing a grade-level prediction tool on books [151], the constructed dataset is an asset to the research community.

The remainder of this paper is organized as follows. In Section 4.2, we discuss existing readability formulas/analysis tools. In Section 4.3, we detail the design methodology of TRoLL. In Section 4.4, we present the results of the empirical studies conducted to assess the performance of TRoLL and compare its performance with existing popular readability formulas/analysis tools. In Section 4.5, we give a concluding remark and directions for future work.

4.2 Related Work

For almost a century, readability formulas/analysis tools have been developed to determine the readability level or degree of difficulty of a text, resulting in hundreds of them [153]. Traditional formulas, including Flesch-Kincaid [76] and Gunning Fog (Index) [62], are based on shallow features. These formulas, however, only provide a rough estimation of the difficulty of a text and thus are not always reliable [19, 52]. Lexile [145] and ATOS [139], two well-known readability analysis tools, are based upon traditional readability features. While the former compares words in a text with 600 million words in the Lexile corpus to establish the semantic difficulty (i.e., word frequency) and syntactic complexity (i.e., sentence length) of the text, the latter considers word length, sentence length, and grade level of words, in addition to book length, i.e., word count, when it is applied to books.

Besides the formulas/tools listed above, new readability analysis approaches based on linguistic features have been developed [38, 59, 65, 140]. Coh-Metrix [59] uses lexicons, part-of-speech classifiers, latent semantic analysis, and syntactic parsers, to name a few, to determine the difficulty of a text, which is influenced by cohesion relations and language/discourse characteristics. Collins-Thompson and Callan [38] combine multiple statistical language models, which capture patterns of word usage in different grade levels, using a Naïve Bayes classifier to estimate the most probable grade level of a text. Schwarm and Ostendorf [140] apply support vector machines on various features extracted from statistical language models, along with shallow features and features derived from analyzing the syntactic structure of texts, to determine the readability level of a text T. Heilman et al. [65] consider lexical and grammatical features derived from syntactic structures to analyze the difficulty of T. (For a detailed discussion on commonly-used features for assessing the readability of a text, see [52].)

Qumsiyeh and Ng [129] and Ma et al. [94] have recently developed their own readability assessment tools. ReadAid [129] performs an in-depth analysis beyond exploring the lexicographical and syntactical structures of an excerpt of a book by considering the authors of the book along with topic(s) covered in the book. Besides examining text-based features, SVM-Ranker [94] considers visually-oriented features (such as the average font size and ratio of annotated image rectangle area to page area) and adopts a rank-based strategy, as opposed to the commonly-employed classification/regression approaches, to determine the grade level of a book.

ReadAid [129], ATOS [139], and SVM-Ranker [94], along with the aforementioned readability formulas, either partially or fully depend on the availability of at least a sample of a $text$ to compute its grade level, which is a severe constraint, since text in a book is not always freely accessible, either online or in a hard copy, due to copyright laws. TRoLL bypasses this constraint by using publicly available metadata on books to accomplish its task.

4.3 Our Readability Analysis Tool

To alleviate the reliance of existing readability formulas/analysis tools on the text of a book, and to improve upon one-dimensional approaches towards determining readability levels of books, we introduce TRoLL, a sophisticated readability analysis tool that can operate without book content, i.e., sample text. Given a unique identifier of a book Bk, which is either its ISBN or its title and (first) author, TRoLL either retrieves the pre-computed readability level of Bk, if it has already been determined by TRoLL, or calculates its readability level on the fly using a multiple linear regression model which analyzes publicly accessible information on Bk that are offered by professional providers, who are either government or educational agents, and can be extracted on-

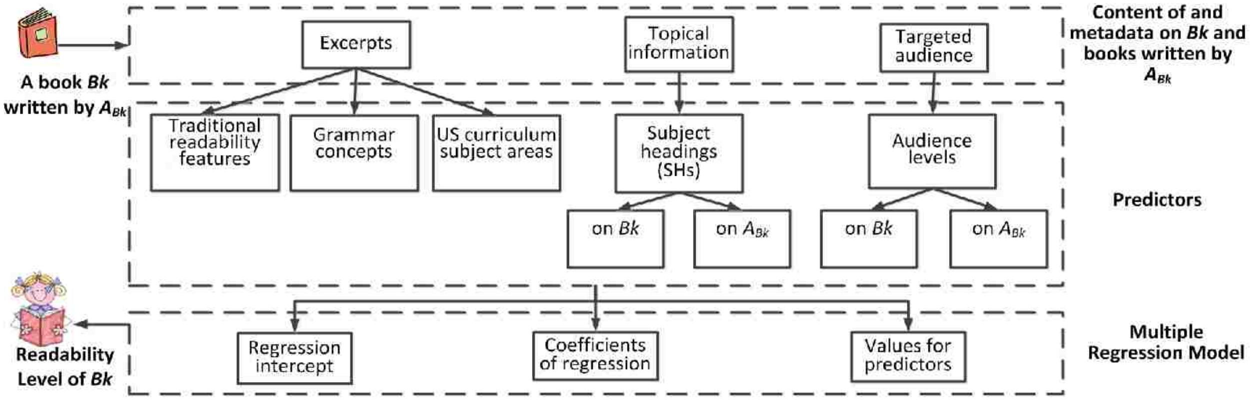

A book *Bk* written by A_{Bk}
Excerpts
Topical information
Targeted audience
Content of and metadata on *Bk* and books written by A_{Bk}
Traditional readability features
Grammar concepts
US curriculum subject areas
Subject headings (SHs)
Audience levels
on *Bk*
on A_{Bk}
on *Bk*
on A_{Bk}
Predictors
Readability Level of *Bk*
Regression intercept
Coefficients of regression
Values for predictors
Multiple Regression Model

Figure 4.1: An overview of the readability level prediction process of TRoLL

line. Examples of such providers include the Library of Congress,[3] the Online Computer Library Center (OCLC),[4] and Open Library.[5] These freely accessible information sources often include metadata, such as subject headings assigned to Bk, and occasionally include the target audience and/or the partial/full text of Bk. The overall readability prediction process of TRoLL is depicted in Figure 4.1.

4.3.1 Multiple Regression Analysis

To predict the readability level of a book Bk, TRoLL employs multiple linear regression analysis [155], which is a classical statistical technique for building estimation models [150]. The analysis accounts for the influence of multiple contributing factors, which are derived from metadata and/or content of Bk, to estimate the readability level of Bk using the following equation:

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n \tag{4.1}$$

where y is the dependent variable, which is the predicted readability level of Bk, β_0 is the intercept parameter, $\beta_1, \ldots, \beta_n$ are the coefficients of regression, X_i $(1 \leq i \leq n)$ is an independent variable (predictor), and n is the number of predictors in the regression analysis [155].

[3] http://www.loc.gov

[4] http://www.worldcat.org

[5] http://www.openlibrary.org

In Equation 4.1, each unknown parameter, i.e., the intercept and coefficients of regression, which is required to predict the readability level of a book by TRoLL, is estimated through a one-time training process using the Ordinary Least Squares method [155] and the $BookRL\text{-}RA$ training dataset (introduced in Section 4.4.1). Each book b in $BookRL\text{-}RA$ is represented as a vector of the form $<b_1, \ldots, b_{54}, r>$, where b_i is the (value of the) the i^{th} predictor ($1 \leq i \leq 54$) computed for b, and r is the *target*, i.e., the known readability level for b in our case. (The fifty-four predictors included in the regression model of TRoLL are defined in Sections 4.3.2, 4.3.3, and 4.3.4, whereas the target readability level of a book is determined by its publisher and included in $BookRL\text{-}RA$.) Since publishers usually suggest a $range$ of readability levels for each of their published books, such as grades 3-6, TRoLL considers the *average* grade level of the range as the *target* grade level of a book to avoid any bias by assigning books to their lowest or highest grade levels in the ranges during the regression training.

The Ordinary Least Squares method calculates the *residual* of each book b in $BookRL\text{-}RA$, which is the difference between the target readability level of b and the readability level of b predicted using the (values of the) predictors in the vector representation of b and Equation 4.1. Unknown parameters are estimated by minimizing the sum of squared distances between residuals of books in $BookRL\text{-}RA$.

4.3.2 Analyzing Book Content

According to [32], only 7.7% of books in the OCLC database, which is a worldwide library cooperative that offers services to improve access to the world's information, are linked to their partial or full content. We found similar results among the 7,142 books in the $BookRL\text{-}RA$ dataset: only 5% of them include their partial or full content. Despite the low percentage of books with available content online, TRoLL utilizes the content of a book, if it is available, in predicting the readability level of the book.

Available online content of a book is either a *snippet* of less than five pages of the book, a *preview* of one or more of its chapters, or its full text [32]. The analysis of book content is the basis for a number of TRoLL predictors, which rely on (i) textual features considered by traditional readability formulas, (ii) the grammar of its content, or (iii) subject areas addressed in the book. When calculating the values of these predictors, we only consider from the first up till the last sentence that includes the first $2,500^{th}$ characters[6] of the content of a book in order to improve the efficiency of TRoLL. We detail the analysis of the content of a book below.

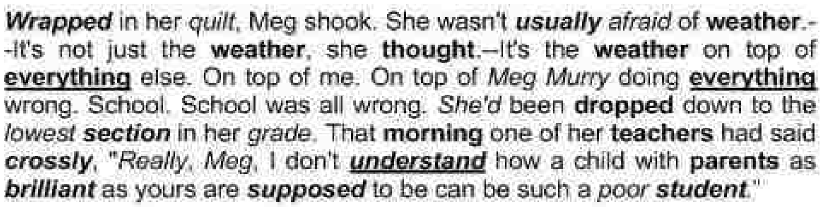
Predictors Based on Features Used by Traditional Readability Formulas

Existing widely-accepted readability formulas, such as Flesch-Kincaid [76], Coleman-Liau (Index) [37], Spache (Readability Index) [147], Gunning Fog [62], and SMOG (Index) [102], seek to combine, through a mathematical formula, several textual features to compute the readability level of a text. We do not use any of these readability formulas as a TRoLL predictor, since there is no consent on which readability formula is the *most* accurate. Instead, we consider the features based on *vocabulary* and the *count of syllables* that are commonly used by traditional readability formulas as predictors so that TRoLL is not biased towards any particular readability formula.

TRoLL considers seven traditional *textual features* used in readability formulas: the count of (i) long words (with more than six letters), (ii) sentences, (iii) total words, (iv) letters, (v) syllables, (vi) words with three or more syllables, and (vii) unique unfamiliar words [147]. Since the length of the text, i.e., the total number of characters, available online is different for each book, we normalize these counts to the length of the text.

Example 6 Consider the book "A Wrinkle in Time," denoted Bk_1, written by Madeleine L'Engle, which tells the story of a 14-year-old, Meg Murry, who lives a normal life until she enters a science fiction/fantasy world in which she goes on adventures. Its publisher suggests the target readers for the book to be in grades 5-7. Based on the first twenty pages of Bk_1's text that are publicly available (a sample of which is shown in Figure 4.2), TRoLL analyzes a snippet with the first 2,639 characters (including the last sentence with the $2,500^{th}$ character) and calculates (the values

[6] The number of characters examined by TRoLL corresponds to the average number of words, i.e., 300 words, often examined by well-known readability formulas, which include Flesch-Kincaid, Fry, and Lexile, to determine the readability level of a text [55].



Wrapped in her *quilt*, Meg shook. She wasn't ***usually*** *afraid* of **weather**.--It's not just the **weather**, she **thought**.–It's the **weather** on top of **everything** else. On top of me. On top of *Meg Murry* doing **everything** wrong. School. School was all wrong. *She'd* been **dropped** down to the *lowest* ***section*** in her *grade*. That **morning** one of her **teachers** had said ***crossly***, "*Really*, *Meg*, I don't ***understand*** how a child with **parents** as ***brilliant*** as yours are ***supposed*** to be can be such a *poor* ***student***."

Figure 4.2: A sample of the text in "A Wrinkle in Time" in which long words are in **bold**, unfamiliar words (determined by traditional readability formulas) are *italicized*, and words with three or more syllables are underlined

of) the following predictors: Count of long words = $\frac{81}{2{,}639}$ = 0.031, Count of sentences = $\frac{39}{2{,}639}$ = 0.015, Count of total words = $\frac{475}{2639}$ = 0.032, Count of letters = $\frac{2{,}018}{2{,}639}$ = 0.765, Count of syllables = $\frac{635}{2{,}639}$ = 0.241, Count of words with three or more syllables = $\frac{29}{2{,}639}$ = 0.011, Count of unique unfamiliar words = $\frac{86}{2{,}639}$ = 0.033. □

Grammar Predictors on Book Content

TRoLL examines grammatical constructions, as defined by the US curriculum and shown in Table 4.1, to compute the values of grammar predictors. These predictors reflect the *complexity* of the (i) writing style, (ii) organization of the sentences, and (iii) grammatical constructs found in a text. The analysis of the grammar of textual content in a book Bk is somewhat more profound, due to advances in natural language processing, such as the Stanford NLP Parser [47], than the analysis used in Flesch-Kincaid [76], Coleman-Liau [37], Spache [147], Gunning Fog [62], SMOG [102], and other readability formulas.

There are two types of predictors created using grammatical constructions: simple and parse-tree. For *simple* grammatical concepts (listed in Table 4.1), which are easily measured, TRoLL simply *counts* their occurrences per sentence in the text of a book Bk. When a grammatical concept is *more difficult* to find and count, TRoLL employs the Stanford Parser [47] to parse the text into *parse trees*. Hereafter, TRoLL counts the occurrences of a grammatical structure per *parse tree* and normalizes the frequency of occurrence of the grammatical structures so that they are comparable regardless of the length of the text.

Table 4.1: List of predictors used by TRoLL

Predictors Based on Content (37)			
Predictors Based on Traditional Text Features (7)			
Count of long words	Count of sentences	Count of total words	Count of letters
Count of syllables	Count of words with three or more syllables	Count of unique unfamiliar words	
Predictors Based on Grammatical Constructions (29)			
Simple		Parse-tree Based	
Common prefixes (un-, re-,pre-, in-, de-, dis-)	Personal pronouns (him, her, it)	Adverbial phrases	Interrogative sentences
Conjunctions (and, but, or)	Plural words	Adverbs	Model verbs of deduction
Conjunctive adverbs (however, therefore, on the other hand)	Possessive nouns	Comparatives and superlatives	Participles
Contractions	Prepositions	Consecutive verbs	Past progressive tense
Determiners	Suffixes (-er, -ment, -able, -ness, -ly, -ful, -less, -tion, -ight, -ite, -ate)	Dependent clauses	Past tense
Irregular vowel combinations, spelling, phonetics (boot, soil, trout)	Syncategorematic words (like, as, to, if, all)	First conditional form	Prepositional phrases
		Future tense	Present perfect tense
		Independent clauses	Present progressive tense
			Quantifiers
Content-based subject area predictor (1)			
Predictors Based on Topical Information (13)			
Total count of subject headings			
Frequency distribution predictors: mean, median, lowerBound, upperBound (4)			
Frequency distribution predictors within one standard deviation: SD_mean, SD_median, SD_lowerBound, SD_upperBound (4)			
Number of previously encountered subject headings			
Ratio of previously encountered subject headings			
Ratio of previously encountered subject headings assigned to books written by an author			
Median of readability levels paired with subject headings assigned to books written by an author			
Predictors based on Targeted Audience (4)			
Book audience level			
Average author's audience level			
Minimum author's audience level			
Maximum author's audience level			

The grammatical predictors offer an in-depth analysis on the grammar of the textual content of Bk, which are valuable to the regression analysis conducted by TRoLL.

The Subject Area Predictor on Book Content

TRoLL takes advantage of the mapping established by the US curriculum between *subject areas* and *grade levels* and exposes the subject area covered in a book to predict its readability level. A *subject area* is a specific topic specified in the US curriculum that is taught to students at a particular grade in the US public school system. For example, multiplication is taught at the 3^{rd} grade, whereas geometry at the 10^{th}. TRoLL pre-defines *fifty-five* distinct subject areas to be considered. These subject areas (and their corresponding grade levels) were inferred from the K-12 curriculum posted under Elkhart Community School website,[7] and each book is assigned a subject area by TRoLL using Equation 4.3 defined below.

To determine the subject area of a book Bk, TRoLL first analyzes (an excerpt of) its content by using a Latent Dirichlet Allocation (LDA) model [22], which is a generative probabilistic model that represents documents as random mixtures over ($latent$) $topics$ such that each $topic$ is characterized by a distribution over $words$. To train a LDA model, we pre-defined the number of latent topics to be fifty-five, which match the number of subject areas considered by TRoLL, and applied JGibbLDA,[8] a Java implementation of LDA, on 5,500 training documents randomly chosen from Wikipedia.org.[9] Note that stopwords in the documents were removed and the remaining words were reduced to their grammatical root using the well-known Porter stemmer. During the training process, the LDA model estimates the probability distribution of *words* in latent topics (topics in documents, respectively). To accomplish this task, we adopted Gibbs sampling [60], a general method applied for probabilistic inference when direct sampling is difficult, which iteratively analyzes the set of training documents to estimate the *probability* of a word w given a (latent) topic

[7] www.elkhart.k12.in.us/3_staff/curric/pdf/1eng.pdf

[8] http://jgibblda.sourceforge.net/

[9] The training documents are uniformly distributed among the 55 pre-defined subject areas, i.e., 100 documents per subject area, and were retrieved by using a keyword query on each subject area SA on Wikipedia so that the top-ranked retrieved Wikipedia page P_{SA}, along with the pages linked from P_{SA}, are treated as documents related to SA.

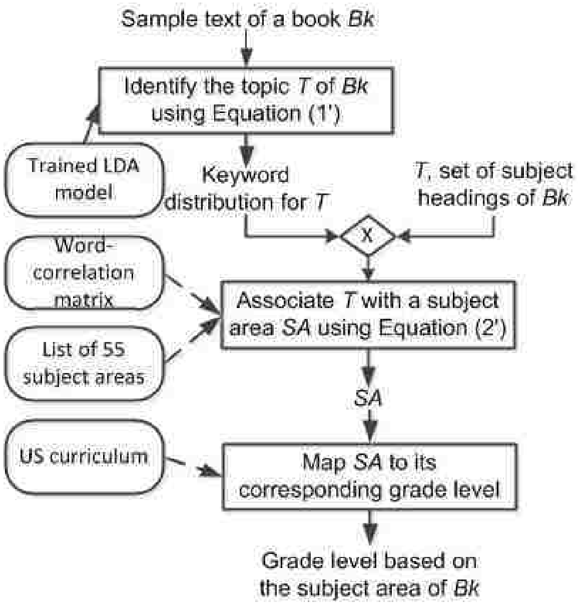

Sample text of a book Bk
Identify the topic T of Bk using Equation (1')
Trained LDA model
Keyword distribution for T
T, set of subject headings of Bk
X
Word-correlation matrix
Associate T with a subject area SA using Equation (2')
List of 55 subject areas
SA
US curriculum
Map SA to its corresponding grade level
Grade level based on the subject area of Bk

Figure 4.3: Determining the subject area and grade level of a book Bk using its content

t (t given a document, respectively). The sampling method is efficient and has been successfully used for obtaining good approximations for LDA [71].

As shown in Figure 4.3, given an excerpt of Bk, denoted Bk_e, TRoLL uses the trained LDA model and Equation 4.2 to identify the (latent) topic covered in Bk_e. Each potential latent topic of Bk_e is associated with a probability value which indicates its likelihood in describing Bk_e. Thereafter, the topic T with the *highest* probability with respect to Bk_e is treated as the latent *topic* of Bk.

$$\begin{aligned} Topic(BK_e) &= \text{argmax}_{T \in LT}\ P(T|BK_e) \\ &= \text{argmax}_{T \in LT} \sum_{i=1}^{|BK_e|} P(w_i|T) \end{aligned} \tag{4.2}$$

where LT is the set of fifty-five pre-defined latent topics considered by the trained LDA model, $P(T|BK_e)$ is the probability of T given BK_e, $|BK_e|$ is the number of distinct non-stop, stemmed words in BK_e, w_i is the i^{th} word in BK_e, and $P(w_i|T)$ is the probability of w_i given T as determined by the trained LDA model.

Table 4.2: A sample of the fifty-five subject areas considered by TRoLL, along with their corresponding grades

Subject Area	Grade	Subject Area	Grade
Shapes	K	Geography	8
Addition	2	Mental disorders	9
Cultures	5	European history	11
Agriculture	6	Statistics	12

Having identified T covered in Bk_e using the trained LDA model, TRoLL applies Equation 4.3 to compute the subject area score (*SAS*) between T and each one of the fifty-five subject areas considered by TRoLL, which captures the *degree of resemblance* between (words in) T and (words in) the corresponding subject area. The subject area SA with the $highest$ computed SAS is treated as the subject area of Bk based on its similarity with T and is assigned to Bk. Hereafter, the grade level associated with SA, which is determined by the mapping between US curriculum subject areas and grade levels employed by TRoLL (a portion of which is shown in Table 4.2), becomes the value of the *content-based subject area* predictor of Bk.

$$\begin{aligned} Subject(T) &= \text{argmax}_{SA \in S}\ SAS(T, SA) \\ &= \text{argmax}_{SA \in S} \frac{1}{|T|} \sum_{i=1}^{|T|} P(w_i|T) \times \frac{1}{SA_n} \sum_{j=1}^{|SA|} wcf(k_j, w_i) \end{aligned} \quad (4.3)$$

where S is the set of fifty-five subject areas, $|T|$ ($|SA|$, respectively) is the number of keywords in T (SA, respectively), T, w_i and $P(w_i|T)$ are as defined in Equation 4.2, k_j is the j^{th} word in SA, $wcf(k_j, w_i)$ is the *word-correlation factor* of k_j and w_i specified in the pre-defined word-correlation matrix [78], and SA_n is the number of words in SA that have a non-zero wcf score with respect to words that define T.

Word-correlation factors in the correlation matrix, which is introduced in [78], reflect the degree of similarity between any two non-stop, stemmed words based on their (i) *frequencies* of co-occurrence and (ii) relative *distances* in a set of approximately 880,000 Wiki-pedia.org documents written by more than 89,000 authors that cover a wide variety of topics. Compared with synonyms/related words compiled by WordNet[10] in which pairs of words are not assigned similarity weights, word-correlation factors offer a more sophisticated measure of word similarity.

Example 7 Consider the book "The scorpions of Zahir," denoted Bk_2, written by Chris Brodien-Jones, which tells the story of a young girl who travels to the Moroccan desert with her family on a quest to save the ancient city of Zahir. Using Equations 4.2 and 4.3, TRoLL identifies "Cultures" as the subject area of Bk_2, which is taught in the 5^{th} grade (see Table 4.2). Consequently, "5" is the value of the *subject area predictor* of Bk_2, which correlates with the publisher's grade level range for Bk_2, which is 5 and up. □

4.3.3 Analyzing Topical Information Metadata

In this section, we discuss the analysis of the metadata of a book Bk based on its topical information, which are subject headings assigned to Bk by professional catalogers who are certified by the Library of Congress or other book cataloging organizations. A *subject heading* is a set of *keywords* used by librarians to categorize and index books according to their themes. Subject headings take on several forms [105], which include the *inverted form*, e.g., "Trolls, Green," the *natural language form*, e.g., "Green Trolls," and the *subdivision form*, e.g., "Fantasy—Mythical Creatures—Trolls—Green." Each component in a subdivision form is treated as a subject heading, whereas subject headings in inverted and natural language forms are each treated as a *single* subject heading. We discuss the predictors derived from subject headings of Bk and the ones derived using the subject headings of books written by the author of Bk below.

Book Subject Heading Predictors

To compute the predictors derived from the subject headings of a book Bk, TRoLL examines (i) their total count, (ii) their associated grade levels, and (iii) their rate of occurrence.

- *Total Count of Subject Headings*. TRoLL uses the *count* of subject headings assigned to Bk as a predictor in Equation 4.1, since books that are *more difficult* to comprehend are

[10] Wordnet.princeton.edu

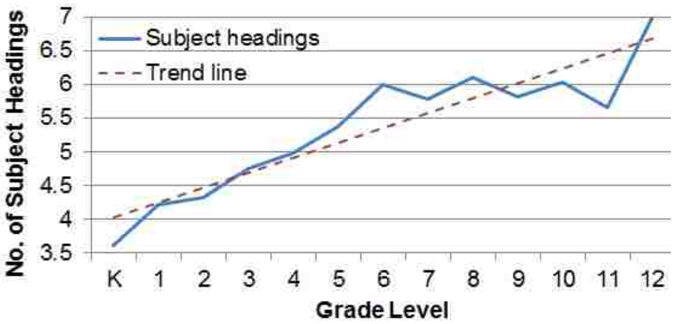
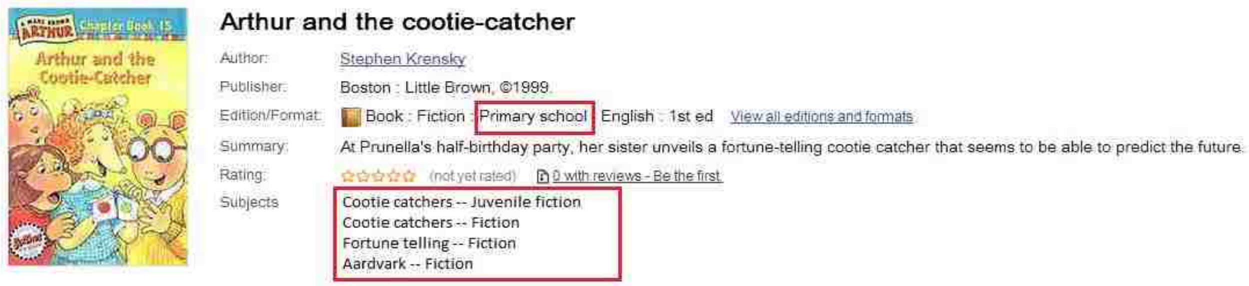


Figure 4.4: The number of subject headings assigned to books versus their readability levels determined by AR

often assigned *more* subject headings. We have empirically verified this claim by counting the number of subject headings assigned to each one of the 5,718 randomly chosen books (available at ARbookfind.com) with its readability levels determined by Accelerated Reader (AR). The mappings between the number of subject headings and grade levels are depicted in Figure 4.4. The trend line in Figure 4.4 has a positive slope of about $\frac{2}{9}$, which demonstrates that books of high readability levels are assigned, on average, more subject headings than books of lower reading levels.

Example 8 Consider *"Arthur and the Cootie Catcher,"* denoted Bk_3, which is a book written by Stephen Krensky and included in the Arthur the Aardvark children's series. Bk_3 was assigned *"aardvark," "cootie catchers," "fiction," "fortune telling,"* and *"juvenile fiction"* as its subject headings. (A portion of the OCLC record for Bk_3, which includes its subject headings, is shown in Figure 4.5.) Five (the number of subject headings) is the value of the $count$ for the *Subject Heading* predictor of Bk_3, one of the predictors used in Equation 4.1 for predicting the readability level of Bk_3. □

- *Subject Headings and Grade Levels.* Besides using the $count$ of subject headings, TRoLL considers the subject headings of Bk that are *previously encountered* in books with a known readability level (range) recommended by their respective publishers. A previously encountered subject heading is a heading observed during the one-time mapping process of TRoLL,


A Marc Brown ARTHUR Chapter Book 15
Arthur and the Cootie-Catcher

Arthur and the cootie-catcher

Author:	Stephen Krensky
Publisher:	Boston : Little Brown, ©1999
Edition/Format:	Book : Fiction : Primary school : English : 1st ed View all editions and formats
Summary:	At Prunella's half-birthday party, her sister unveils a fortune-telling cootie catcher that seems to be able to predict the future.
Rating:	☆☆☆☆☆ (not yet rated) 0 with reviews - Be the first.
Subjects	Cootie catchers -- Juvenile fiction Cootie catchers -- Fiction Fortune telling -- Fiction Aardvark -- Fiction

Figure 4.5: A portion of the OCLC record for the book "Arthur and the Cootie Catchers," available at http://www.worldcat.org/title/arthur-and-the-cootie-catcher/oclc/40444058/

which paired subject headings assigned to each of the 8,737 books in the $BookRL$-SH dataset (introduced in Section 4.4.1) with the readability level range of the corresponding book determined by its publisher. To account for the possibility that a subject heading, SH, is paired with many books and therefore many readability levels, TRoLL considers all readability levels paired with SH as a frequency distribution, D. An analysis of the *mean*, *median*, *lower bound*, and *upper bound* readability levels in D yield four predictors, which are called *frequency distribution predictors* (FDP). Additionally, in order to reduce the effect of outlier readability levels in D, TRoLL further considers the *mean*, *median*, *lower bound*, and *upper bound* of the readability levels within one *standard deviation* of the mean of D, which generate another four predictors based on the mapping between subject headings and grade levels. The value of each of these eight predictors is calculated as

$$FDP_{m_i}(Bk) = \frac{\sum_{j=1}^{|V|} m_i(D_j)}{|V|} \tag{4.4}$$

where m_i is either *mean, median, lowerBound, upperBound, SD_mean, SD_median, SD_lowerBound*, or *SD_upperBound*, V is the set of all the subject headings assigned to Bk that have been previously encountered, D_j is the frequency distribution corresponding to a subject heading $SH_j \in V$, and $m_i(D_j)$ is the application of m_i to D_j.

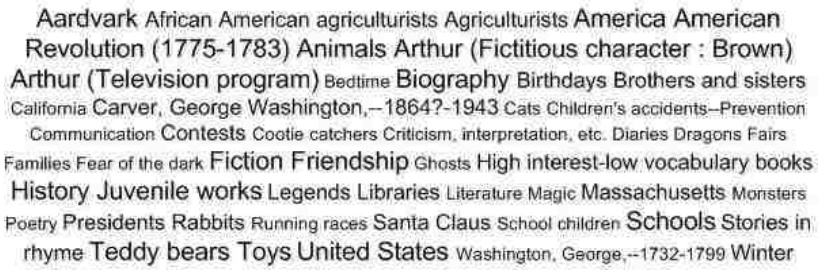
Example 9 To illustrate how the *mean* frequency distribution predictor is calculated, let's consider the three subject headings, i.e., $\{$*aardvark, fiction, juvenile fiction*$\}$ = V (out of the five total), assigned to Bk_3 (in Example 8) which have been previously encountered. According to Equation 4.4

$$FDP_{mean} = \frac{mean(D_{aardvark})+mean(D_{fiction})+mean(D_{juvenile\ fiction})}{3}$$

We observe that *nine* of the books used in the one-time mapping process described above were assigned the subject heading *"aardvark"*. The *mean* of the readability levels of the corresponding nine books, which are established by their publishers, are $D_{aardvark}$ = <0, 0, 0, 1.5, 1.6, 1.6, 2.2, 2.3, 2.5>. Based on this distribution, $mean(D_{aardvark})$ = 1.3. In the same manner, TRoLL examines the readability level distribution for *"fiction"* and *"juvenile fiction"* to compute $mean(D_{fiction})$ = 3.81 and $mean(D_{juvenile\ fiction})$ = 3.10. Subsequently, $FDP_{mean} = \frac{1.3\ +\ 3.81\ +\ 3.10}{3}$ = 2.74, is the value of one of eight *frequency distribution predictors* based on the mean metric. □

- *Common Subject Headings*. Besides considering the mapping of subject headings to their grade levels, TRoLL also counts *commonly occurred* subject headings of Bk. If a subject heading was previously encountered during the mapping process when 38,315 subject headings (assigned to the books in $BookRL\text{-}SH$) were examined, it is considered a *commonly occurred* subject heading. We conjecture that commonly occurred subject headings are assigned to books with lower readability levels, since books for lower readability levels cover less advanced, specific topics. The predictors created by using *commonly occurred* subject headings are (i) the number of *previously encountered subject headings* assigned to Bk and (ii) the *ratio* of previously encountered subject headings to the total number of subject headings assigned to Bk.

Example 10 Consider Bk_3 in Example 8. Since the subject headings *"aardvark," "fiction,"* and *"juvenile fiction"* have been previously encountered, whereas the others have not, 3 and $\frac{3}{5}$ are the values of the *number of previously encountered subject headings* and the *ratio of previously encountered subject headings* predictors, respectively. □



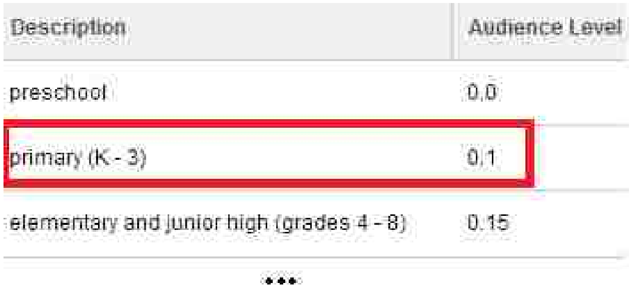
Aardvark African American agriculturists Agriculturists America American Revolution (1775-1783) Animals Arthur (Fictitious character : Brown) Arthur (Television program) Bedtime Biography Birthdays Brothers and sisters California Carver, George Washington,--1864?-1943 Cats Children's accidents--Prevention Communication Contests Cootie catchers Criticism, interpretation, etc. Diaries Dragons Fairs Families Fear of the dark Fiction Friendship Ghosts High interest-low vocabulary books History Juvenile works Legends Libraries Literature Magic Massachusetts Monsters Poetry Presidents Rabbits Running races Santa Claus School children Schools Stories in rhyme Teddy bears Toys United States Washington, George,--1732-1799 Winter

Figure 4.6: Subject headings assigned to books written by Stephen Krensky, available at http://www.worldcat.org/wcidentities/lccn-n79-109188

Author's Subject Headings Predictors

The subject headings assigned to books (including Bk) written by A_{Bk}, who is the author of Bk, are analyzed in the same manner as the subject headings assigned to Bk (as discussed in Section 4.3.3). The analysis of *commonly occurred* subject headings assigned to books written by A_{Bk} is captured in one predictor, which is the *ratio* of the number of previously encountered subject headings to the total number of subject headings assigned to books written by A_{Bk}. FDP_{median}, which is based on the subject headings assigned to all the books written by A_{Bk}, is established as another predictor. The *median* readability level was employed, since medians are *less* influenced by outliers, which often decrease the accuracy of a frequency distribution predictor. Note that only the *median*, instead of all eight of the frequency distribution predictors defined in Section 4.3.3 is considered for A_{Bk}, since subject headings assigned to books written by A_{Bk} are not always directly related to Bk, even though A_{Bk} often writes books at a particular readability level.

Example 11 Consider Stephen Krensky, the author of Bk_3 in Example 8. The books written by the author have been assigned *fifty* subject headings, which are shown in Figure 4.6. The ratio of previously encountered subject headings of books written by Krensky, $\frac{30}{50}$, is the value of the predictor for the author based on the previously encountered subject headings, whereas FDP_{median} for Krensky, another TRoLL predictor, is calculated to be 2.6. □



Description	Audience Level
preschool	0.0
primary (K - 3)	0.1
elementary and junior high (grades 4 - 8)	0.15

•••

Figure 4.7: The OCLC mapping between the targeted readers and their corresponding audience levels is available at http://www.oclc.org/research/activities/audience.html

4.3.4 Analyzing Targeted Audience Metadata

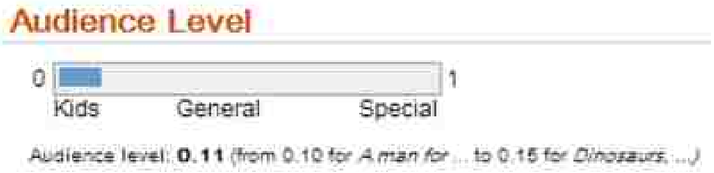
TRoLL also considers the audiences targeted by books and their corresponding authors in predicting the readability level of books.

The Book Audience Level Predictor

For each book in its database, OCLC provides an *audience level*, which is a numerical value between 0 and 1 that indicates "the type of reader believed to be interested in a particular book" and is publicly available at OCLC.[11] We have observed that there is a $correlation$, which is not a direct relationship, between the audience level of a book Bk and its readability level, which is expected, since authors often write at the reading comprehension level of their respective audiences [44]. The audience level of Bk is the value of the *book audience level* predictor used by TRoLL.

Example 12 Consider Bk_3 in Example 8 again. As depicted in the OCLC record for Bk_3 and shown in Figure 4.5, Bk_3 is aimed towards primary school readers with its audience level score being 0.1, which is the value of the corresponding book audience level predictor as specified in the mapping between targeted audiences and audience levels provided by OCLC and as shown in Figure 4.7. □

[11] http://www.oclc.org/research/activities/audience.html



Audience level: **0.11** (from 0.10 for *A man for* ... to 0.15 for *Dinosaurs, ...*)

Figure 4.8: The audience level defined by OCLC for the author Stephen Krensky, available at http://www.worldcat.org/wcidentities/lccn-n79-109188

The Author's Audience Level Predictor

Besides the audience level of Bk, OCLC also defines the audience level of its author A_{Bk} as the *average* of the audience levels of the books written by A_{Bk}, including Bk. In addition, OCLC provides the *minimum* (*maximum*, respectively) audience level of books A_{Bk} has written. Based on these three audience level scores determined by OCLC, we define three other audience level predictors: the *average, lowest* (minimum), and *highest* (maximum) audience levels of A_{Bk}, which refer to the comprehensive levels of the audience targeted by books written by A_{Bk}.

Example 13 Consider Stephen Krensky who is the author of Bk_3 in Example 8. As depicted in the audience level record in OCLC and shown in Figure 4.8, the *average, lowest*, and *highest* audience levels for Stephen Krensky are 0.11, 0.10, and 0.15, respectively, which are the values of the corresponding three audience level predictors. □

As illustrated in Figure 4.8, the audience level does not directly matches the grade level of an author. Instead, the audience level simply reflects the groups of readers targeted by an author at various levels (from values of 0 for $kids$ to 1 for *advanced readers*).

4.3.5 The Predicted Readability Level of a Book

It is possible that some of the fifty-four predictors defined in Equation 4.1 for predicting the readability level of a book Bk cannot be calculated, since their corresponding metadata or content may be missing. Hence, TRoLL defines a number of regression models, which are the variances of the one shown in Equation 4.1, that analyze diverse combinations of available predictors. Based on the distinct subsets of predictors that can be applied to books in the $BookRL$-RA dataset, there are 107 trained regression models used by TRoLL for predicting the readability levels of books.

With the calculated value of each of the predictors pertinent to Bk, TRoLL selects, among the trained regression models, the one that includes the most (values of) predictors available for Bk and that excludes any predictor not applicable to Bk to compute the readability level of Bk.

Example 14 Based on the information available online for Bk_1 as presented in Example 6, 52 predictors are applicable to Bk_1. Using the corresponding regression model for Bk_1 the grade level of Bk_1 predicted by TRoLL is 6.8, which falls within the grade-level range, i.e., 5 to 7, defined for the book by its publisher. TRoLL also examines the 23 predictors applicable to Bk_3 as presented in Example 8 and predicts 0.98 as the readability level for Bk_3 using the corresponding regression model for Bk_3, which correlates with the readability level, i.e., 1.0, defined by the publisher of Bk_3. □

4.4 Experimental Results

In this section, we first introduce the dataset and metric used for assessing the performance of TRoLL (in Sections 4.4.1 and 4.4.2, respectively). Thereafter, we present the results of the empirical studies conducted for evaluating the effectiveness of TRoLL in grade level prediction and compare its prediction accuracy with existing widely-used readability formulas/analysis tools (in Section 4.4.3).

4.4.1 The Dataset

To the best of our knowledge, there is no existing benchmark dataset that can be used for assessing the performance of readability-level prediction tools on books. For this reason, we constructed our own dataset, $BookRL$, using data extracted from CLCD.com, a website established to assist teachers, parents, and librarians in choosing books for K-12 readers, Young Adults Book Central (Yabookscentral.com), Young Adults Library Service Association (ala.org/yalsa), ARbookfind.com,

Table 4.3: Sources of books used for creating $BookRL$

Online Sources	**Number of Books**	**Online Sources**	**Number of Books**
ARbookfind	4,037	Penguin	600
Bookadventure	1,017	Simon & Schuster	388
CLCD	6,667	YABC	3,038
Lexile	2,154	Yalsa	226
		Total	18,127

Lexile.com, and reputable publishers' websites. (See Table 4.3 for the source websites and their numbers of books included in $BookRL$.) $BookRL$ consists of 18,127 books distributed among the K-12 grade levels with their ranges determined by their publishers. Due to the lack of common consensus among researchers on the most accurate existing readability prediction tool [19], we consider publisher-provided grade levels as the "gold-standard," since they are defined by human experts.

It is an easier task for a publisher to provide a range of grade levels for a book than a single readability level, since the latter requires precision, whereas the former an intelligent estimate. These human-assessed ranges of readability levels of books are adopted as the gold standard, which is applied to assess the performance of TRoLL and the readability formulas/analysis tools considered in our empirical study.

Among the 18,127 books in $BookRL$, 40% of the books, denoted $BookRL$-RA, were utilized to train the *regression analysis* model of TRoLL (as introduced in Section 4.3.1). Another 47% of the books, denoted $BookRL$-SH, were employed by TRoLL to perform a one-time mapping between *subject headings* and readability levels (as discussed in Section 4.3.3), and the remaining 13% of books in $BookRL$, denoted $BookRL$-$Test$, was used for assessing the performance of TRoLL and a number of well-known readability formulas/analysis tools. All the subsets of $BookRL$ are disjoint.

4.4.2 Metrics

To assess the performance of TRoLL and other widely-used readability formulas/analysis tools, we compute their (absolute) error rates (ER) [42], each of which is the averaged *absolute differences* between the *expected* and the *predicted* grade levels of the books in $BookRL$-$Test$ determined by the corresponding formula/tool.

$$ER = \frac{1}{|BookRL - Test|} \sum_{B \in BookRL - Test} |PR(B) - GL(B)| \tag{4.5}$$

where $|BookRL$-$Test|$ is the number of books in $BookRL$-$Test$, $GL(B)$ is the grade level of a book B in $BookRL$-$Test$ predicted by a readability formula/analysis tool, and $PR(B)$ is either the $lower$ or $upper$ bound of the grade level range of B determined by its publisher, whichever is closest to $GL(B)$, which reflects the *closeness* of the predicted grade level to the grade level range of B.

We have also applied the *Wilcoxon signed-rank test*, which is a non-parametric test based on the differences between pairwise samples [42], to determine the *statistical significance* of the error rates on grade-level prediction obtained by TROLL with respect to their counterparts obtained by various readability formulas/analysis tools.

4.4.3 Performance Evaluation

In this section, we (i) analyze the prediction accuracy of (various groups of) TRoLL's predictors, (ii) verify the correctness of using content and/or metadata for readability-level prediction, and (iii) compare the performance of TRoLL with other readability analysis formulas/tools.

Analyzing TRoLL's Predictors

TRoLL uses up to fifty-four predictors to determine the readability level of a book. As shown in Figure 4.1, these predictors can be grouped into seven categories according to the type of information on books and authors considered, which include traditional readability features, grammar

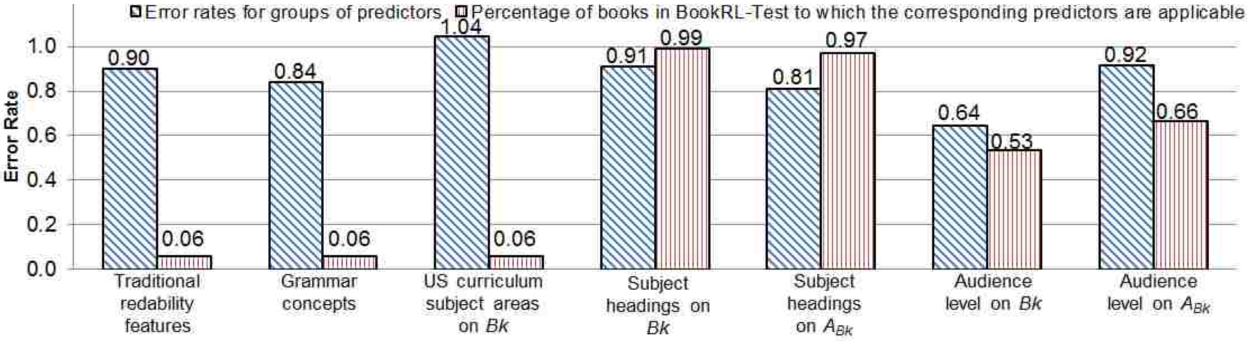


Figure 4.9: An analysis of the performance of (group of) predictors considered by TRoLL based on each book Bk and its author A_{Bk} in $BookRL$-$Test$

concepts, subject areas, subject headings, and audience levels. Figure 4.9 shows the error rates obtained by each of the groups of predictors with the fraction of books in $BookRL$-$Test$ to which the corresponding group of predictors is applicable. Based on the compiled results, we draw the following observations:

- The predictor on the *audience level* of a book provided by OCLC achieves the $lowest$ error rate in readability-level prediction. This is anticipated, since there is a high correlation between the readability level of a book and its targeted audience, even though there is no direct mapping between an audience level and a readability level. Unfortunately, the OCLC's audience level for a book is not always available. For example, only 53% of the books in $BookRL$-$Test$ are assigned an audience level. The same applies to the *audience level of an author* provided by OCLC, from where only 66% of book authors in $BookRL$-$Test$ are assigned an audience level.

- The *subject area* predictor receives the $highest$ error rate, since books for emergent readers tend to include more pictures than text and these non-textual contents are not utilized by TRoLL to identify US curriculum subject areas covered in books. However, this predictor is a suitable indicator of the readability levels of books targeting more advanced readers. Using $BookRL$-$Test$, we have empirically verified that this predictor yields at most a 0.14 error rate in analyzing the readability levels of books in the 5^{th} to 8^{th} grade levels.

- The most *reliable* predictors, which do not only achieve relatively low error rates but also are widely applicable, are the two groups that analyze *subject headings*. These groups of predictors rely on information frequently available for books and thus are applicable to the majority of books examined by TRoLL. As shown in Figure 4.9, predictors based on subject headings are applicable to at least 97% of the books in $BookRL\text{-}Test$.
- The group of predictors based on *traditional readability features* and *grammar concepts* are effective; however, these predictors are computed on excerpts of books, which are seldom available. For example, only 6% of the books in $BookRL\text{-}Test$ come with their corresponding excerpts.

Validating the Accuracy of Using Content, Topical Information, and Targeted Audiences in Predicting Readability Levels

As discussed in Section 4.3, TRoLL examines two major types of information to determine the readability levels of books: content (if it is available) and metadata of books. We have validated the prediction accuracy of TRoLL when distinct set of predictors based on content and/or metadata are considered using $BookRL\text{-}Test$.

- *Using content-based information.* The low error rate, which is 0.53, achieved by considering only the content-based predictors (as shown in Figure 4.10) is anticipated, since book content is a *reliable* source of information which has direct impact on the degree of *difficulty* in understanding the content of a book, even if only an excerpt of the book is available for analysis. The error rate obtained by using content-based predictors is based on the 127 books with content in $BookRL\text{-}Test$.
- *Relying on information other than content.* We have further observed that in estimating the readability levels of books for emergent (i.e., K-2) readers, relying solely on content can generate readability levels that do not correlate with the ones recommended by publishers of the corresponding books. For example, the average error rate generated by using content-based predictors for 1^{st} grade books in $BookRL\text{-}Test$ with available sample text is 2.10, which

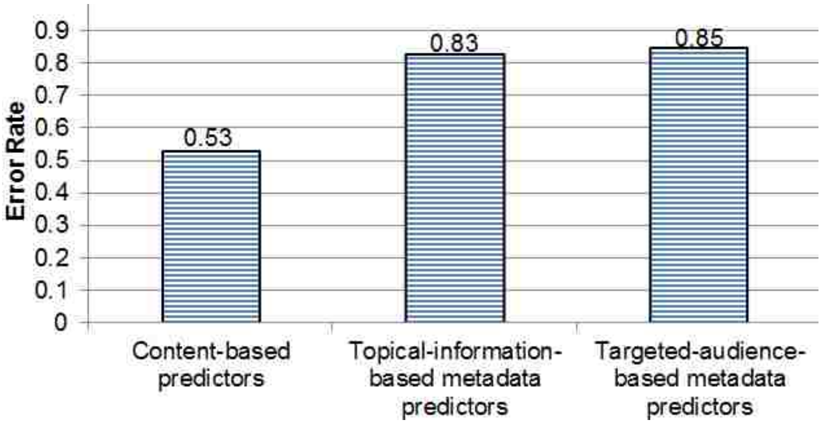


Figure 4.10: Performance evaluation of TRoLL using distinct sets of content/metadata predictors on books in $BookRL$-$Test$

is three times higher than the average error rate (i.e., 0.70) generated using up to fifty-four predictors of TRoLL on books with sample content as shown in Figure 4.11. Realizing that considering only content information can lead to imprecisely-predicted readability levels of books for emergent readers, a fact that correlates with the study discussed in [85],[12] we have designed TRoLL so that it analyzes metadata on books with or without excerpts available on-line. In doing so, the error rate obtained using content- and metadata-based predictors on 1^{st} grade books in $BookRL$-$Test$ with available sample text decreases from the 2.1 (obtained solely based on content predictors) to 0.73.

- *Using metadata.* As discussed in Sections 4.3.3 and 4.3.4, TRoLL considers two types of metadata predictors: topical information, i.e., subject headings, and targeted audience. The error rate obtained by using topical information predictors, which is 0.83 (as shown in Figure 4.10), is *higher* than the 0.75 overall error rate of TRoLL (as shown in Figure 4.11) but slightly *lower* than the error rate achieved by using only audience level predictors, which is 0.85 (as shown in Figure 4.10). This is expected, since subject headings are often available for books and is a consistent contributing factor in predicting the readability level of books, whereas audience levels are limited as opposed to other metadata/content predictors.

[12] The study verifies that using contents of books for young readers to predict their readability levels tends to yield overstated readability levels for the books.

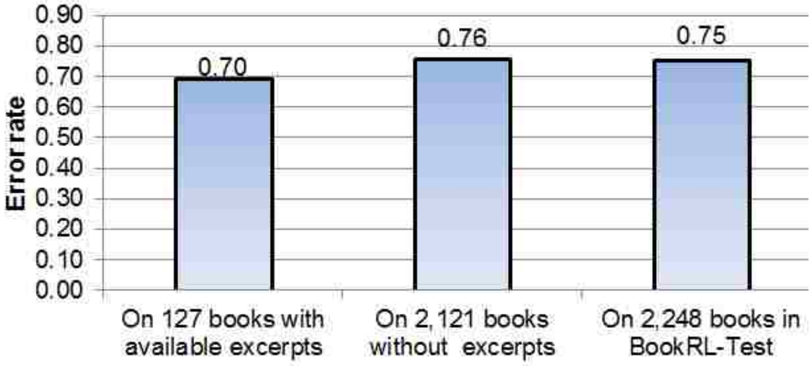


Figure 4.11: Overall performance evaluation of TRoLL using up to 45 predictors applicable to each book in $BookRL$-$Test$

- *The overall performance of TRoLL.* Based on the results of our conducted empirical study, we conclude that the readability prediction accuracy of TRoLL is consistent, regardless of the presence or absence of sample text of books. On the 127 books with available sample content in $BookRL$-$Test$, TRoLL achieves a 0.70 error rate (as shown in Figure 4.11), whereas among the 2,121 (= 2,248 - 127) books in $BookRL$-$Test$ without sample text, the 0.76 error rate generated by TRoLL is *within one* grade level off the ranges specified by the publishers of the examined books. Moreover, the overall error rate of TRoLL on $BookRL$-$Test$, in which 94% of the (2,248) books are without text, is 0.75, which is only $\frac{3}{4}$ of a grade level from the targeted grade level. This low error rate is not only an accomplishment of TRoLL, but also it cannot be achieved by *any* of the existing readability formulas/analysis tools, since *none* of them can predict the grade level of books without excerpts.

Comparing TRoLL with Others

Using the 127 (out of 2,248) books in $BookRL$-$Test$ with excerpts, we compared the grade-level prediction accuracy of TRoLL with a number of well-known readability formulas based on text content: Coleman-Liau [37], Flesch-Kincaid [76], Rix (Index) [11], and Spache [147], which we have implemented based on their formulas that are shown in Table 4.4. (See discussion on these readability formulas/tools in Section 4.2 and [19].)

Table 4.4: Popular readability formulas employed in our empirical study

Measure	**Formula**
Coleman-Liau	(0.0588 × Average number of letters per 100 words) - (0.296 × Average number of sentences per 100 words) - 15.8
Flesch-Kincaid	$(0.39 \times \frac{Number\ of\ words}{Number\ of\ Sentences}) + (11.8 \times \frac{Number\ of\ syllables}{Number\ of\ words}) - 15.59$
Rix	$\frac{Number\ of\ words\ with\ more\ than\ 6\ characters}{Number\ of\ Sentences}$
Spache	(0.121 × Average sentence length) + (0.082 × Number of unique unfamiliar words) + 0.659 (unfamiliar words can be found at http://goo.gl/nJhGMU)

Figure 4.12 shows that (i) the average error rate of the grade level predicted by TRoLL for a book with text, which is 0.7 and is the same error rate shown in Figure 4.11, is slightly more than *half* of a grade from the grade (range) determined by its publisher and (ii) the error rate of TRoLL is *at least* 26% lower than the error rate created by its counterparts. The difference in error rate achieved by TRoLL over each of its counterparts is *statistically significant*, as determined using a Wilcoxon signed-ranked test with $p < 0.001$.

We have further compared the performance of TRoLL with two other popular readability analysis tools widely-used by grade schools and reading programs in the USA, the Accelerated Reader (AR) and Lexile. Even though the algorithms of AR and Lexile are not publicly accessible, we were able to find 897 books with AR scores and 314 books with Lexile scores among the books in $BookRL\text{-}Test$ from ARbookfind.com and Lexile.com, respectively. As shown in Figure 4.13, TRoLL outperforms AR and is more accurate than Lexile in predicting the grade level of the analyzed books. The improvement in error rate achieved by TRoLL over either Lexile or AR is *statistically significant* as determined using a Wilcoxon signed-ranked test with $p < 0.001$.

Human Assessment on TRoLL

We further evaluated TRoLL to determine whether its predicted readability levels are perceived as accurate by ordinary users, which offers another perspective on the performance of TRoLL. The additional evaluation is based on real users' assessments of TRoLL which goes beyond the performance analysis conducted and presented in previous subsections. To accomplish this task,

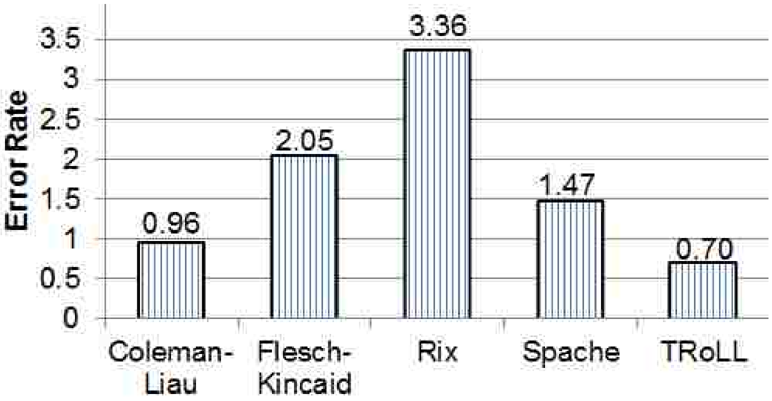


Figure 4.12: Performance evaluation on TRoLL and other readability formulas based on the 127 books with excerpts in $BookRL\text{-}Test$

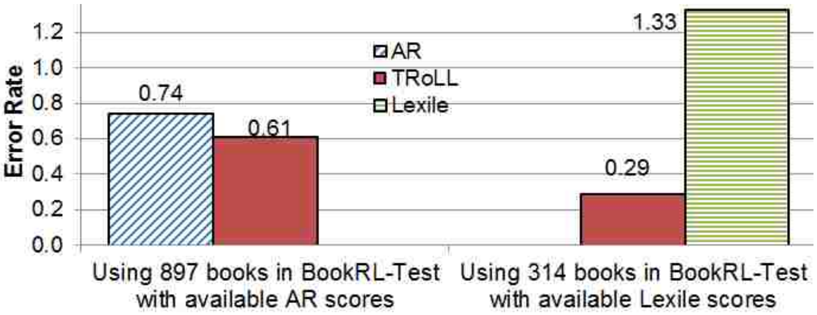


Figure 4.13: Performance evaluation on TRoLL, AR, and Lexile based on books in $BookRL\text{-}Test$

we conducted a user study using Amazon's Mechanical Turk,[13], a "marketplace for work that requires human intelligence", which allows individuals or businesses to programmatically access thousands of diverse, on-demand workers and has been used in the past to collect user feedback for multiple information retrieval tasks [80].

In the user study, we considered a set of 10 sample books with diverse readability levels. (The list of books used in the study, along with their corresponding readability levels predicted by TRoLL, is shown in Table 4.5.) We created a HIT (Human Intelligent Task) on Mechanical Turk so that for each sample book SB, each appraiser was presented six different readability levels for SB and asked to select the one that "best" captures the readability level of SB. The six readability levels were generated by AR, Coleman-Liau, Flesch-Kincaid, Rix, Spache, and TRoLL, respectively, which were presented in Section 4.4.3.

[13] https://www.mturk.com/mturk/welcome

Table 4.5: List of books and their TRoLL's readability levels employed in the user study conducted using Mechanical Turk

Book	Level	Book	Level
Arthur and the Cootie Catcher	1.5	Macbeth	10.7
Ender's Game	4.1	Mansfield Park	10
Five Little Kittens	0.9	Matilda	3.9
Good Night Moon	0.0	Pride and Prejudice	6.2
Love You Forever	1.7	The Scarlet Letter	9.3

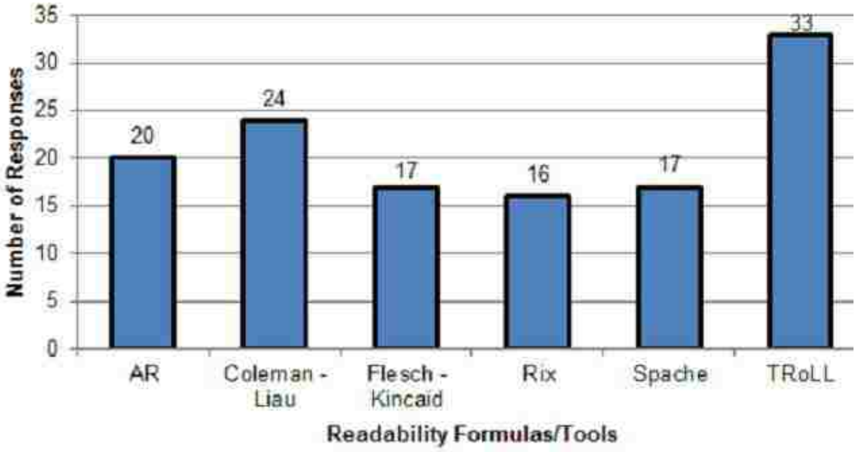
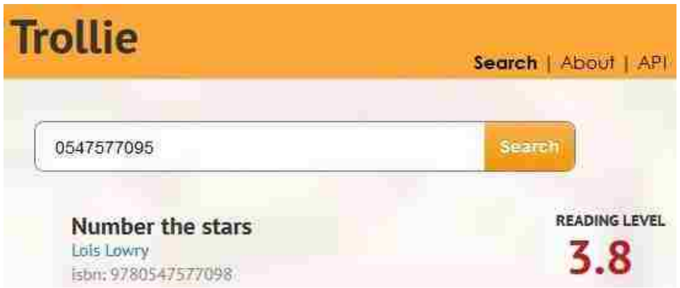


Figure 4.14: Distribution of Mechanical Turk appraisers' responses in choosing the reading levels of 10 books computed by various readability-level prediction formulas/tools

The user study was conducted between October 25 and October 30, 2013 on Mechanical Turk. Altogether, there were 127 responses among the HITs used in the study. Based on the corresponding set of responses provided by Mechanical Turk appraisers, we have verified that users tend to favor TRoLL's predicted readability level for a given book. (The distribution of the 127 collected responses among the different readability-level prediction formulas/tools is shown in Figure 4.14.) Note that the larger number of users who favor TRoLL over the remaining readability formulas/tools is statistically significant, as determined using the Wilcoxon signed-ranked test with $p < 0.05$ for Flesch-Kincaid, Coleman-Liau, Rix, and Spache, and $p < 0.1$ for AR.



Number the stars
Lois Lowry
isbn: 9780547577098
READING LEVEL
3.8

Figure 4.15: A screenshot of the online version of our readability prediction tool, TRoLL, which shows the readability level of a book, given its isbn number

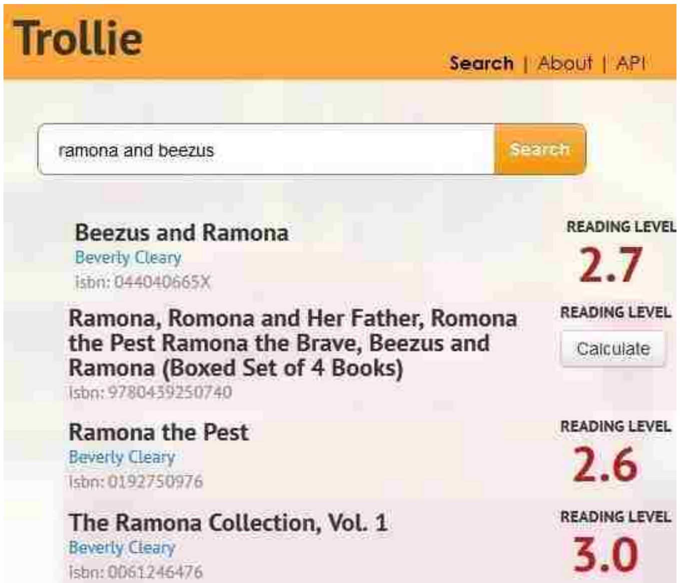
4.4.4 Trollie, an Online Prototype of TRoLL

We have implemented TRoLL and made it available as an online application, called *Trollie*.Through its user-interface, a user can either enter (a portion of) the *title* and/or *author* or *isbn* of a book, which is a unique identifier of the book. In the latter case, Trollie computes and presents the readability level of the corresponding book (through TRoLL, the back-end readability analysis tool) to the user. (See Figure 4.15 for an example.) In the former case, Trollie first conducts a search[14] of books that match the keywords captured in the (portion of the) title and/or author provided by the user. Thereafter, if the title and/or author is not unique, i.e., if multiple books partially match the user-provided keywords, the user is required to select,[15] among the retrieved books, the desired one so that Trollie can generate its corresponding readability level. (See the screenshot of Trollie shown in Figure 4.16 for an example.)

By developing Trollie, we facilitate the task of automatically determining the readability levels of books, which assists children and teenagers (parents and teachers, respectively) in locating books that they (their K-12 readers, respectively) can comprehend.

[14] The search is currently powered by OpenLibrary.org.

[15] To speed up its processing time, Trollie archives the readability levels of books that have been computed over time through its online interface. Thus, the previously-computed readability level of a book is instantly displayed; otherwise, Trollie computes the readability level of a book on-the-fly, whenever the *calculate* button is hit by the user. (See Figure 4.16 for an example.)



Beezus and Ramona
Beverly Cleary
isbn: 044040665X
READING LEVEL 2.7

Ramona, Romona and Her Father, Romona the Pest Ramona the Brave, Beezus and Ramona (Boxed Set of 4 Books)
isbn: 9780439250740
READING LEVEL Calculate

Ramona the Pest
Beverly Cleary
isbn: 0192750976
READING LEVEL 2.6

The Ramona Collection, Vol. 1
Beverly Cleary
isbn: 0061246476
READING LEVEL 3.0

Figure 4.16: A screenshot of Trollie, which shows the readability levels of books, given (a portion of) a title provided by a user

4.5 Conclusions

Statistical data compiled over the last few years has shown that the reading ability of school-age children in America is falling in comparing with most of the developed countries in the world. It is essential to encourage children/teenagers to develop good reading habits, which is crucial for them to succeed at school and in the living of a good life, the mission statement of TRoLL, a tool for regression analysis of literacy levels developed by us.

TRoLL is unique compared with existing readability formulas/ analysis tools, since it can predict the grade level of a book even without a sample text of the book by simply analyzing metadata on the book that is publicly accessible from popular online sources. TRoLL is reliable, since it applies regression analysis on a number of predictors established by using textual features on books (if they are available), Library of Congress Subject Headings of books, US Curriculum subject areas identified in books, and information about book authors to predict the grade level of K-12 books.

The development of TRoLL is a significant contribution to the educational community, since grade levels predicted by TRoLL can be used by (i) teachers, parents, and school librarians to identify reading materials suitable to their K-12 readers and (ii) K-12 students as a guide in making their own reading selections, which, in turn, can enrich their reading for learning experiences. A conducted empirical study on TRoLL has verified not only its prediction accuracy, but also its superiority over existing readability formulas/analysis tools.

For future work, we plan to extend TRoLL so that it can be used for predicting the grade levels of reading materials other than books, such as articles posted on various websites, which should facilitate the process of locating different (educational) materials, besides books, that are suitable for K-12 readers.

Chapter 5

What to Read Next?: Making Personalized Book Recommendations for K-12 Users

Abstract:

Finding books that children/teenagers are interested in these days is a non-trivial task due to the diversity of topics covered in huge volumes of books with varied readability levels. Even though K-12 readers can turn to book recommenders to look for books, the recommended books may not satisfy their personal needs, since they could be beyond/below their readability levels or fail to match their topics of interest. To address these problems, we introduce BReK12, a book recommender that makes personalized suggestions tailored to each K-12 user U based on books available on a social bookmarking site that (i) are *similar in content* to the ones that are known to be of interest to U, (ii) have been bookmarked by users with *reading patterns* similar to U's, and (iii) can be *comprehended* by U. BReK12 is an asset to its users, since it suggests books that are appealing to its users and at grade levels that they can cope with, which can increase their reading selection choices and motivate them to read. We have also developed ReLAT, the readability analysis tool employed by BReK12 to determine the grade level of books. ReLAT is novel, compared with existing readability formulas, since it can predict the grade level of a book even if an excerpt of the book is not available. We have conducted empirical studies which have verified the accuracy of ReLAT in predicting the grade level of a book and the effectiveness of BReK12 over existing baseline recommendation systems.

5.1 Introduction

Reading, an essential skill required for acquiring knowledge, is an integral part of the educational system. It is imperative to encourage K-12 students to read, since research studies have confirmed the enormous influence of reading on students' development as learners and members of society, especially at an early age. Finding books that K-12 readers are interested in, however, is a difficult task due to the diversity of topics covered in the huge volume of books that target readers of different backgrounds and age groups. Recommender systems, which have been developed to suggest items of interest to their users, are supposed to alleviate the book-finding problem. However, existing recommenders employed at well-known book-affiliated websites, such as Novelist (www.ebscohost.com/novelist) and Amazon.com, adopt a "one-size-fits-all" strategy, which makes the same suggestions to different users on a given book without considering their individual preferences [89]. On the contrary, recommenders that offer personalized book suggestions overlook the readability levels of their users, since they are conceived with a general audience in mind. As a result, even if a book recommended to a user U matches U's interests, the book might include complex (simple, respectively) content that is beyond (below, respectively) the readability level of U, which fails to sustain the mission of matching users' reading abilities with the suggested literature [6]. Moreover, these recommenders rely heavily on the existence of *personal ratings* [156] assigned to books by users, which are rarely provided by K-12 users of the existing social bookmarking sites established for them.

To address the aforementioned design problems of book recommendation systems, we have developed BReK12, a book recommender that makes personalized suggestions for K-12 users. To locate books for a user U based on U's reading ability and interests, BReK12, which is designed to be integrated into a social bookmarking site on books, analyzes U's profile, i.e., books that have been bookmarked by U on the site. If U is a new user, BReK12 treats a book provided by U as U's profile. In doing so, BReK12 bypasses the *cold-start* problem often encountered by recommenders [133]. BReK12 first infers U's readability level by analyzing the grade levels of books in his/her profile, which are determined using ReLAT, a robust readability level analysis tool that we have developed. Hereafter, BReK12 identifies a set of candidate books, among the ones archived at the site, with grade levels compatible to the inferred readability level of U. For each candidate book, BReK12 determines its *content similarity* and *readership similarity* with books in the profile of U based on the brief *descriptions* of books that are publicly available online and users' *bookmarking patterns* on the site, respectively. An aggregation strategy [12] is adopted so that the top-10 candidate books, with grade levels appropriate for U and with the highest combined content- and readership-similarity scores, are recommended to U.

BReK12 is unique, since its design goal is to suggest books to K-12 users that simultaneously match their *interests* and *reading abilities*, which in turn can motivate them to read. Unlike state-of-the-art recommenders [133], BReK12 simply employs *readily available* data, i.e., user bookmarks and brief descriptions of books, accessible from the social bookmarking sites where BReK12 is installed and book-affiliated websites, respectively to make recommendations. Moreover, BReK12 applies similarity, besides exact, matching on words to recommend books that are *similar* in content to users' bookmarks, which otherwise could be ignored.

BReK12 relies on ReLAT to determine the grade level of a book B based on the *subject areas* addressed in B, the readability level of the intended audience of books written by the *author* of B, *subject headings* assigned to B, and the *grammatical/sentence structures* in (an excerpt of) B, if any is available. Unlike existing readability formulas/tools, such as Lexile Analyzer and Flesch-Kincaid, ReLAT can predict the readability level of a book even if (a sample of) the text of the book is unavailable, which is its novelty.

Besides serving social bookmarking site users, BReK12 can also recommend books for each K-12 patron of a school/ public library, assuming that the list of books of interest provided by the library patron and the book catalog used by the library are given. In addition, BReK12 can be adapted to make recommendations for users of any book-affiliated website based on books searched by the users on the site.

The remaining of this paper is organized as follows. In Section 5.2, we discuss existing readability formulas and book recommenders. In Sections 5.3 and 5.4, we introduce ReLAT and

BReK12, respectively. In Section 5.5, we present the results of the empirical studies conducted to (i) assess the performance of ReLAT and BReK12 and (ii) compare their performances with existing approaches. In Section 5.6, we give a concluding remark and directions for future work.

5.2 Related Work

In this section, we present a number of widely-used readability formulas/analysis tools and book recommenders and compare them with ReLAT and BReK12, respectively.

5.2.1 Readability Formulas/Analysis Tools

For almost a century, hundreds of readability formulas have been developed for predicting the readability level of a text [19, 38]. Traditional readability formulas, such as Flesch-Kincaid and Coleman-Liau, rely on shallow features, which include word frequency and sentence length, to compute the grade level of a text. These formulas, however, often provide a rough estimation of text difficulty, which "(may) judge a nonsense passage as *quite readable* if the text's jumbled words are frequent, short, and organized into brief sentences" [19]. Recently-developed formulas are based on (i) *linguistic* features, such as the ones introduced by Feng et al. [51] and Collins-Thompson and Callan [38], (ii) *pre-defined word lists*, such as Lexile and Revised Dale-Chall formula, (iii) *enhanced shallow features*, such as the *Advantage-TASA Open Standard for Readability* (ATOS) formula, and (iv) *non-textual features*, such as SVM-Ranker [94], which examines images on books to predict their grade levels, and ReadAid [129], which considers information about the authors of a book and US Curriculum topics addressed in the book in addition to exploring the lexicographical and syntactical structures of the book. While most of these formulas are widely-used and popular, none of them can predict the readability level of a book if (a sample of) its text is not available, which can be achieved by ReLAT. (See [19] for an in-depth discussion of existing readability formulas.)

5.2.2 Book Recommenders

A number of book recommenders [89, 117, 156] have been proposed in the past. Amazon's recommender [89] suggests books based on the purchase patterns of its users. Yang et al. [156] analyze users' access logs to infer their preferences and apply the traditional collaborative-filtering (CF) strategy, along with a ranking method, to make book suggestions. Givon and Lavrenko [57] combine the CF strategy and social tags to capture the content of books. Similar to the recommenders in [57, 156], the one in [144] adopts the standard user-based CF framework and uses a domain ontology to determine the users' topics of interest. The recommenders in [57, 144, 156] overcome the problem that arises due to the lack of initial information to perform the recommendation task, i.e., the cold-start problem. However, unlike BReK12, they are constrained by requiring historical data in the form of *ratings* to make recommendations, which may not always be available. Moreover, the recommender in [144] relies on the existence of a book ontology, which can be labor-intensive and time-consuming to construct. In making recommendations, Park and Chang [117] analyze individual/group behaviors, such as clicks and shopping habits, and features describing books, such as their library classification, whereas $PReF$ [120] suggests books bookmarked by connections of a LibraryThing user. Similar to BReK12, $PReF$ adopts a similarity-matching strategy, which differs from the exact-matching constraint imposed in [117] and a number of content-based recommenders [61, 109]. However, neither $PReF$ nor any of the aforementioned recommenders considers the readability level of their users as part of their recommendation strategies. While BReK12 is not a recommender system for learning [133] per se, its design goal is to aid children/teenagers in developing good reading habits so that they can succeed at school and in the living of a good life. With that in mind, BReK12 is designed as an educational enrichment tool. (An in-depth discussion of existing recommender systems in the educational domain can be found in [96].)

5.3 A Grade Level Prediction Tool

As previously stated, existing readability formulas/analysis tools rely on at least a sample of a text to compute its readability level, which is a severe constraint, since text in a book is not always freely

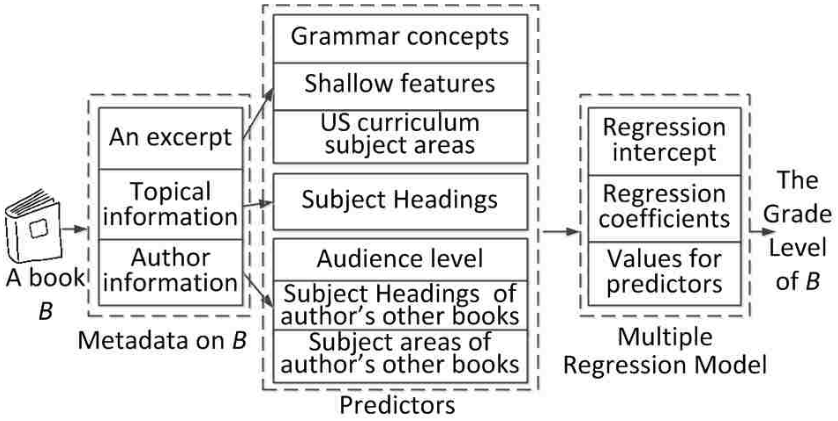


Figure 5.1: The grade-level prediction process of ReLAT, our proposed readability level analysis tool

accessible due to the copyright laws. To alleviate the text constraint, we have developed *ReLAT*[1], which determines the grade level of any book using *metadata* on books publicly accessible from reputable online sources, in addition to sample texts of books only if they are available. Analyzing a book using multiple perspectives, ReLAT can predict its grade level even if an excerpt on the book is missing. Furthermore, ReLAT *instantly* produces the grade level of a book, which is not always possible using commercial readability analysis tools. For example, Lexile offers scores for only approximately 150,000 out of the millions of books published worldwide in English, and requires direct involvement of its developers to generate the readability level of any book which has not yet been analyzed [19].

Figure 5.1 depicts the grade level prediction process of ReLAT. To determine the grade level of a book B, ReLAT takes as an input a unique identifier of B, which can be its ISBN number or its title and author. Thereafter, ReLAT extracts from reputable online sources, such as WorldCat.org and OpenLibrary.org, (i) an excerpt of B, (ii) topical information of B, and (iii) information about the author[2] of B, whatever is available. Based on the extracted data on B, ReLAT examines up to *fifty-nine predictors* to determine the grade level of B. The predictors are

[1] ReLAT is an earlier version of the readability-level prediction tool presented in Chapter 4.

[2] We have empirically verified that by considering only the *first* author of a book B, the processing time of ReLAT is minimized without affecting its accuracy in predicting the grade level of B. This is expected, since less than 10% of the hundreds of thousands of K-12 books we examined at ARbookfind.com and Scholastic.com are written by co-authors.

treated as *contributing factors*, i.e., evidences, which are used in analyzing the grade level of B. Due to the space constraint, we do not define each predictor.[3] Instead, we present the nature of predictors in each category (as shown in Figure 5.1) below.

(I) Predictors based on an *excerpt* of B examine

- *Grammar Concepts*. ReLAT analyzes the complexity of the grammar usage in B by counting the occurrences of various grammatical concepts in its sentences, which are present perfect, modal verbs, past progressive tense, parts of speech, phrases, suffixes, prefixes, and key vocabulary words.
- *Shallow Features*. ReLAT considers a number of well-established textual features commonly used by traditional readability formulas: the average number of syllables per word, average sentence length, percentage of words with at least three syllables, average number of characters per word, and absolute number of words.
- *Subject Areas*. The US Curriculum dictates which subject areas should be taught at each K-12 grade level. For example, $multiplication$ is taught at the 3^{rd} grade while $trigonometry$ at the 12^{th} grade. ReLAT relies on a Latent Dirichlet Allocation (LDA) model [22] to identify a set of *representative keywords* that best describes the content of B. Thereafter, ReLAT calculates the resemblance (using word-correlation factors [78]) between these keywords and each of the pre-defined subject areas established by the US Curriculum. The subject area SA with the highest degree of resemblance is treated as the subject area of B and the grade level associated with SA is used for predicting the grade level of B.

(II) Predictors based on *topical information* of B analyze its Subject Headings, e.g., “Biography”, which are short phrases that capture the topics covered in books and are used by the US Library of Congress to categorize books. Based on a mapping between Subject Headings and grade levels (that we have already determined using Subject Headings assigned to books with a known grade level), ReLAT identifies the grade levels that correspond to each of the Subject Headings of B. These grade levels are taken into account by ReLAT to determine the grade level of B.

[3] The list of all the predictors and the categories to which they belong can be found in tinyurl.com/dyp6ysl.

(III) Predictors that consider *author information* of B are based on the fact that, in general, K-12 authors write for a particular group of readers at a certain grade level. For this reason, ReLAT treats the *audience level* metric of an author defined by WorldCat, which captures the "intellectual level of the audience for which the item is intended," in addition to the topical information and subject areas of the author's other books (as introduced in I and II, respectively) as additional predictors that determine the overall grade level of B.

Since the information required to compute the value of a predictor can be missing, it may not be possible to use all the predictors for predicting the grade level of B. ReLAT considers various combinations of the 59 predictor coefficients and applies multiple linear regression analysis [107] (given below) on predictors applicable to B to predict its grade level.

$$c_0 + c_1x_1 + c_2x_2 + ... + c_nx_n = y \tag{5.1}$$

where c_0 is the $intercept$ term, c_i ($1 \leq i \leq n$) is the *regression coefficient* of predictor x_i, and y is the predicted grade level.

Prior to applying Equation 5.1 to predict the grade level of a book, the *intercept* and *coefficients* associated with each applicable predictor are computed through a one-time training process using the *ordinary least squares* estimation method [107] on a training set of 8,737 K-12 books written by different authors that cover diverse topics and were extracted from various publishers' websites and the Children's Literature Comprehensive Database (CLCD.com). Each training instance consists of the (values of) predictors that apply to a book B and the grade level of B determined by its publisher. Since publishers usually suggest a *range* of readability levels for each of their published books, ReLAT considers the *average* grade level of the range defined for B as its grade level during the training process. In doing so, ReLAT avoids any bias that might occur by assigning books their lower/upper grade bound during the regression training.

Example 15 Consider the book "Five Little Kittens" by Nancy Geller Jewell, which is a 32-page picture book. As stated in [85], unlike existing readability formulas that often overstate the difficulty of books for emergent readers, Accelerated Reader (AR) is a decent measure on the readability levels of books. Even though “Five Little Kittens” is appropriate for grades K-3 (i.e., readers age 5 to 8), as suggested by its publisher, its Flesch-Kincaid grade level is 4.6 and its Lexile score is 970 (which corresponds to grades 6-7). The AR grade level for the book is 2.6, which matches the target audience for the book. The AR score, however, suggests that children should be at least in the 2^{nd} grade to read “Five Little Kittens,” whereas the grade level predicted by ReLAT, which is 0.9, indicates that the book is suitable for Kindergartners, providing a grade level more compatible (than AR’s) with the book publisher’s. (See Section 5.5.3 for the performance analysis of ReLAT.) □

5.4 The Book Recommender

In this section, we present the design of BReK12. Given the profile[4] P of a user U, BReK12 selects a set of candidate books, which are compatible with the readability level of U (determined in Section 5.4.1). Hereafter, BReK12 assigns a *ranking* score to each candidate book B, which is computed using an aggregation strategy (introduced in Section 5.4.4) on the *content* and *readership similarity* of B with respect to the books in P (defined in Sections 5.4.2 and 5.4.3, respectively).

5.4.1 Identifying Candidate Books

BReK12 recognizes that “reading for understanding cannot take place unless the words in the text are accurately and efficiently decoded” [112] and only recommends books with readability levels appropriate to its users.

BReK12 applies Equation 5.2 to estimate the readability level of a user U, denoted $RL(U)$, based on the grade level of each book P_B in U’s profile predicted by ReLAT, denoted ReLAT(P_B). Note that only books bookmarked in a user’s profile during the most recent academic year are considered, since it is anticipated that the grade levels of books bookmarked by users gradually

[4] BReK12 requires the existance of at least one book in the profile of a user to make the corresponding suggestions.

Table 5.1: A number of BiblioNasium books

ID	Book Title	Grade Level
Bk_1	Mummies in the Morning	2.9
Bk_2	Captain Underpants and the Big, Bad Battle of the Bionic Booger Boy	4.7
Bk_3	The Hidden Boy	5.6
Bk_4	Dragon's Halloween	3.1
Bk_5	Junie B. Jones Smells Something Fishy	3.0
...	...	...

increase as the users enhance their reading comprehension skills over time.

$$RL(U) = \frac{\sum_{P_B \in P} ReLAT(P_B)}{|P|} \tag{5.2}$$

where $|P|$ denotes the number of books in U's profile and *average* is employed to capture the *central tendency* on the grade levels of books bookmarked by U.

Having established U's readability level, BReK12 creates $CandBks$, the subset of books archived at the bookmarking site that are within-one-grade-level range from U's. By considering books within *one* grade level above/below U's mean readability level, BReK12 recommends books with an appropriate level of complexity for U and grade levels approximate to the grade levels of books that have been read by U (as of the most recent academic year).

Example 16 Consider a BiblioNasium.com user U who has bookmarked a number of books from Dav Pilkey's "Captain Underpants" series. Based on the grade levels predicted by ReLAT for the books archived at BiblioNasium (see a sample of BiblioNasium books in Table 5.1) and U's readability level, which is 4, BReK12 does not include Bk_1 or Bk_3 in $CandBks$, since their grade levels are below/beyond the range deemed appropriate for U. □

5.4.2 Content Similarity Measure

BReK12 depends on the profile P of U to infer U's interests/preferences. To determine the degree to which the content of a book B in $CandBks$ appeals to U, BReK12 computes the *content* *similarity* between B and each book P_B in P, denoted $CSim(B,\ P)$ as defined in Equation 5.3, using a "bag-of-words" representation on the *brief descriptions* of B and P_B obtained from book-affiliated websites, such as Amazon. To compute $CSim(B,\ P)$, BReK12 employs an enhanced version of the *cosine similarity measure* based on word-correlation factors [78], which relaxes the exact-matching constraint imposed by the cosine measure and explores words in the description of B that are *analogous* to, besides the *same* as, words in the description of P_B.

Word-correlation factors in the correlation matrix introduced in [78] reflect the degree of similarity between any two words according to their (i) frequencies of co-occurrence and (ii) relative distances in Wikipedia(.org) documents. Approximately 880,000 documents covering a wide range of topics and written by more than 89,000 authors with varied writing styles were used to construct the matrix. Compared with synonyms/related words compiled by WordNet (wordnet.princeton.edu) in which pairs of words are not assigned similarity weights, word-correlation factors offer a more sophisticated measure of word similarity. In addition, word-correlation factors have been successfully employed to solve a number of content-similarity problems [120, 129].

$$CSim(B,P) = \max_{P_B \in P} \frac{\sum_{i=1}^{n} VB_i \times VP_{B_i}}{\sqrt{\sum_{i=1}^{n} VB_i^2} \times \sqrt{\sum_{i=1}^{n} VP_{B_i}^2}} \tag{5.3}$$

where B and P_B are represented as n-dimensional vectors VB = <VB_1, ..., VB_n> and VP_B = <VP_{B_1}, ..., VP_{B_n}>, respectively, n is the number of distinct words in the descriptions of B and P_B, and VB_i (VP_{B_i}, respectively), which is the $weight$ assigned to word B_i (P_{B_i}, respectively), is calculated as shown in the equations in Table 5.2.

HS_w in Table 5.2 is the set of words that are *highly similar*[5] to, but not the $same$ as, a given word w in the description of a book Bk, which is either B or P_B, $|HS_w|$ is the size of HS_w, $tf_{w,Bk}$ = $\frac{f_{w,Bk}}{\sum_{w \in Bk} f_{w,Bk}}$ is the normalized *term frequency* of w in Bk, and idf_w = $log\frac{N}{n_w}$ is the *inverse document frequency* for w in the collection of books N archived at a social bookmarking site, where n_w is the number of books in N that include w in their descriptions. Relying on the

[5] Two words are *highly similar* if their correlation factor is included in a reduced version of the aforementioned word-correlation matrix which contains 13% of the most frequently-occurring words in the Wikipedia documents.

Table 5.2: TF-IDF weighting scheme used in the enhanced cosine similarity measure in Equation 5.3

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

tf-idf weighting scheme, BReK12 prioritizes discriminating words that capture the content of its respective book.

The *max* function in Equation 5.3 emulates the "most pleasure" strategy (commonly applied in game theory and group profiling [133]). Applying this strategy, BReK12 selects the $highest$ possible score among the ones computed for each P_B in P and B. The $larger$ the number of exact-matched or highly-similar words in the descriptions of both B and P_B is, the *more likely* B is a relevant recommendation for U. Moreover, BReK12 adopts the widely-used cosine measure, which has been effectively applied to determine the degree of resemblance between any two items in content-based recommenders [109]. While content-similarity is computed by BReK12 using book descriptions, other textual information on books, such as their tag representations, can be used. Tags, however, are not always publicly available.

Example 17 To illustrate the merit of using the enhanced cosine similarity measure in Equation 5.3 to compute $CSim(B, P)$, consider the profile P of user U in Example 16 and two of the books, Bk_2 and Bk_4 as shown in Table 5.1. Using the *traditional* cosine measure, Bk_2 and Bk_4 yield the same content similarity score with respect to P. However, employing the *enhanced* cosine similarity measure, BReK12 obtains a more accurate content-similarity score for each book, since $CSim(Bk_2, P)$ = 0.57 and $CSim(Bk_4, P)$ = 0.39. These scores reflect that U is likely more interested in books similar to the ones in the "Captain Underpants" series (by Dav Pilkey) than books about "dragons," which we have verified by manually examining the profile of U. □

5.4.3 Readership Similarity Measure

$CSim$ locates books that are similar in content, to a certain degree, to the ones users have read in the past. This measure, however, does not consider books that are dissimilar in content but match users' specific preferences/interests. Hence, BReK12 explores another dimension of resemblance between each book B in $CandBks$ and books in U's profile P by using the Lennon similarity measure [87] to perform co-readership analysis on users' bookmarks on a social bookmarking site. This *readership similarity* measure (as defined in Equation 5.4) is based on the popular item-item similarity approach employed by collaborative-filtering recommenders to examine *patterns of co-occurrence* of items bookmarked by users to make recommendations [24].

$$RSim(B, P) = \max_{P_B \in P} \left(1 - \frac{min(|S_B - S_\cap|, |S_{P_B} - S_\cap|)}{min(|S_B - S_\cap|, |S_{P_B} - S_\cap|) + |S_\cap|}\right) \tag{5.4}$$

where S_B (S_{P_B}, respectively) is the set of users who bookmarked B (P_B, respectively), $S_\cap$ = $S_B \cap S_{P_B}$, $|S_\cap|$ is the number of users who bookmarked both B and P_B, $|S_B - S_\cap|$ ($|S_{P_B} - S_\cap|$, respectively) is the number of users who bookmarked B, but not P_B (P_B, but not B, respectively) at a social bookmarking site, and the use of the max function was discussed in Section 5.4.2.

In Equation 5.4, the min(imum) of the two differences between $|S_B - S_\cap|$ and $|S_{P_B} - S_\cap|$ is chosen, since by using the *smaller* of the two differences, we can *more accurately* capture the similarity between B and P_B. As a *difference* reflects the number of users who bookmark B, but not P_B (or vice versa), a *smaller* difference signifies that proportionally a *larger* number of users who bookmark one book also prefer the other book, which is a *better* indication of the degree of readership similarity between the two books.

Example 18 To illustrate the usefulness of readership similarity measure for making recommendations, consider Bk_5 in Table 5.1 and the book "The Adventures of Captain Underpants" by Dav Pilkey, denoted P_B, which is a book in the profile of user U introduced in Example 16. The contents of the books differ, since P_B is about the adventures of two fourth graders and a superhero, whereas Bk_5 details the events that occur when Junie, the main character in the book, takes her pet to school. The books, however, share some *common thread* of interest to a group of BiblioNasium users who have bookmarked both, since the books include characters of *similar age*, written in *similar literary styles*, and share the *same genre*. Relying partially on the readership similarity between Bk_5 and P, which is 0.67, BReK12 retains Bk_5 as one of the top-10 recommendations for U, which otherwise would have been ignored due to the lack of related content between the two books. □

5.4.4 Rank Aggregation

Using the computed *content-* and *readership*-similarity scores of each book B (with a readability level appropriate for U) in *CandBks*, BReK12 applies the *Borda Count voting scheme* [12] to determine the *ranking* score for B. The Borda Count voting scheme is a positional-scoring procedure such that given k ($\geq$ 1) candidates, each voter casts a vote for each candidate according to his/her preference. A candidate that is given a first-place vote receives k-1 points, a second-ranked candidate k-2 points, and so on up till the last candidate, who is awarded no points. Hereafter, the points assigned to each candidate across all the voters are added up and the candidate with the $most$ points $wins$.

The Borda Count strategy, which has been successfully applied to different information retrieval tasks [12], is employed by BReK12 to generate a single ranking score for B, denoted $Borda(B)$ as defined in Equation 5.5, that regards the content- and readership-similarity scores as equally important in determining the degree to which a user is interested in B. Using Equation 5.5, BReK12 assigns (i) k = $|CandBks|$, which is the number of candidate books selected for a user U, and (ii) C = 2, which is the number of voters, i.e., the two ranked lists of similarity scores on books computed in Sections 5.4.2 and 5.4.3, respectively. Candidate books with the top-10 Borda scores are recommended to U.

$$Borda(B) = \sum_{c=1}^{C}(k - S_c^B) \tag{5.5}$$

where S_c^B is the position on the ranking of B based on the c^{th} ranked list to be fused.

BReK12 adopts Borda as an *aggregation strategy*, since (i) its combination algorithm is $simple$ and *efficient*, which requires neither training nor compatible relevance scores that may not be available and (ii) its performance is competitive with other existing combination strategies [12].

5.5 Experimental Results

In this section, we first introduce the datasets, metrics, and evaluation protocol used for assessing the performance of ReLAT and BReK12 (in Sections 5.5.1 and 5.5.2, respectively). Hereafter, we present the results of the empirical studies conducted for evaluating the effectiveness of ReLAT and BReK12 (in Sections 5.5.3 and 5.5.4, respectively).

5.5.1 The Datasets

To the best of our knowledge, there is no benchmark dataset that can be used for evaluating readability-level prediction tools on books. Thus, to evaluate ReLAT we constructed BookGL,[6] using data extracted from CLCD.com, a website established to assist teachers, parents, and librarians in choosing appropriate books for K-12 readers, the Young Adults Book Central (YAbookscentral.com), and reputable publishers' websites. BookGL consists of 2,248 books distributed among the K-12 grade levels with the grade level (range) of each book assigned by its publisher. (See Table 5.3 for the source websites and their number of books in BookGL.) Due to the lack of consensus among researchers on the accuracy of existing readability prediction tools [19], we consider publisher-provided grade levels as the "gold-standard," since they are defined by human experts.

Even though the BookCrossing dataset (informatik.uni-freiburg.de/∼cziegler/BX) has been employed to evaluate book recommenders tailored to a general audience, it is not specifically designed for assessing the performance of book recommenders for K-12 users. Hence, we used data provided by BiblioNasium, which is a safe and secure social networking site on books developed

[6] BookGL and the set of books used to train ReLAT's multiple regression predictor (discussed in Section 5.3) are disjoint.

Table 5.3: The BookGL dataset

Online Sources	**Number of Books**
CLCD.com	663
Penguin Group	498
Simon & Schuster	388
YABC	699

exclusively for children and teenagers, to evaluate BReK12. The dataset consists of the profile of, i.e., books that have been bookmarked by, each of the 297 BiblioNasium users who joined the site within its first month of being launched. As the design methodology of BReK12 relies on brief descriptions and predicted grade levels of books, we extracted the former from Amazon.com and predicted the latter using ReLAT.

5.5.2 Metrics and Evaluation Protocol

To assess the performance of ReLAT, we apply the *(absolute) error rate* (ER) [42], which is the absolute difference between an *expected* and a *predicted* grade level for a book B.

$$ER = \frac{1}{|BookGL|} \sum_{B \in BookGL} |PR(B) - GL(B)| \tag{5.6}$$

where $|BookGL|$ is the total number of books in BookGL, $GL(B)$ is the predicted grade level of B by a readability formula/analysis tool, and $PR(B)$ is either the lower or upper bound of the grade level range of B determined by its publisher, whichever is closest to $GL(B)$. (Recall that publishers often assign a grade range, not a level, to a book.)

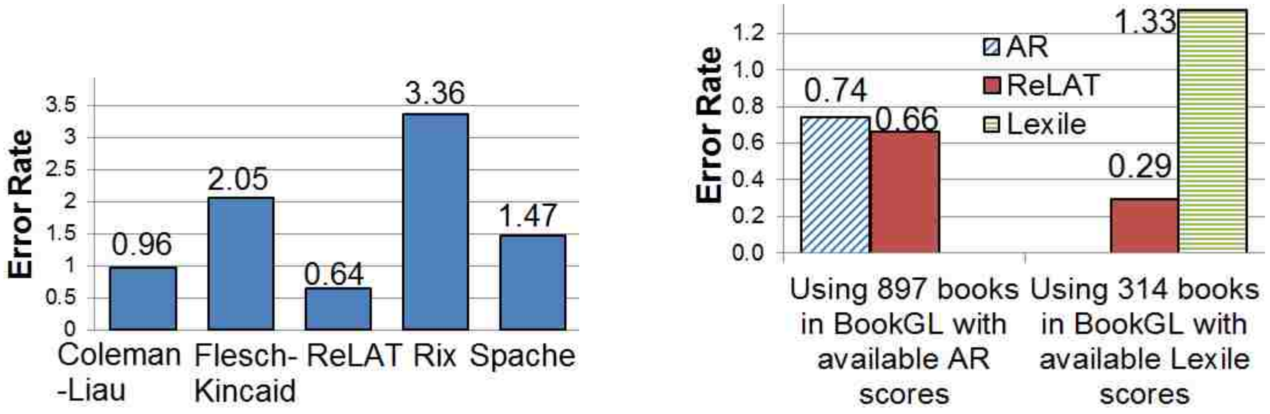
We evaluate BReK12 using *Precision@10*, *Mean Reciprocal Rank* (MRR), and *Normalized Discounted Cumulative Gain* (nDCG) [42]. Precision@10 measures the fraction of the top-10 ranked recommendations that are *relevant*, whereas MRR computes the average ranking position of the *first* relevant recommended book. nDCG determines the $overall$ ranking performance of BReK12 and *penalizes* relevant books ranked *lower* in the recommendation list. The penalization is based on a reduction, which is logarithmically proportional to the position of each relevant book in a ranked list.

We adopt the popular five-fold cross validation strategy to evaluate BReK12 (and recommender systems considered for the comparison purpose). In each of the five repetitions, 80% of the books bookmarked by a user U in the BiblioNasium dataset are used to create U's profile and the remaining 20% are reserved for the testing purpose. A recommended book B is treated as *relevant* to U if it is included in the 20% of the books withheld for the testing purpose, and is *non-relevant* otherwise, which is a commonly-employed protocol for assessing the performance of recommendation systems [61]. Since only withheld books are considered *relevant*, it is not possible to account for potentially relevant books a user has not bookmarked, which is a well-known limitation of this evaluation protocol. As the limitation affects all the recommenders evaluated in the conducted empirical studies, the results are consistent for the comparative purpose.

5.5.3 Performance Evaluation of ReLAT

Using the 127 books in BookGL with excerpts, we compared the grade-level prediction accuracy of ReLAT with a number of well-known readability formulas: Coleman-Liau, Flesch-Kincaid, Rix, and Spache, which we have implemented. (See detailed discussion on these readability formulas in [19].) Figure 5.2(a) shows that (i) on average the grade level predicted by ReLAT for a book with text (in BookGL) is about *half* a grade from the grade (range) determined by its publisher and (ii) ReLAT's error rate is *at least* 33% lower than the error rate created by its counterparts.

We also evaluated the performance of ReLAT in predicting the grade level of books for which their excerpts cannot be obtained online. Among the 2,121 books in BookGL without sample text, the 0.82 error rate generated by ReLAT shows that ReLAT's predictions are *less than one* grade level above/below the ranges specified by the publishers of the books. This low error rate is not only an accomplishment of ReLAT, but also it cannot be achieved by *any* of the existing readability formulas/analysis tools, since *none* of them can predict the grade level of books without



(a) Readability Formulas

(b) Analysis Tools

Figure 5.2: Performance evaluation of ReLAT

excerpts. The overall error rate of ReLAT on BookGL, in which 94% of the books are without text, is 0.81, which is within one grade level of the targeted grade level.

We further compare the performance of ReLAT with two popular readability analysis tools widely-accepted by grade schools and reading programs in the USA: Accelerated Reader (AR) and Lexile. Recall that the algorithms developed to compute AR and Lexile scores are not publicly accessible, but we were able to find 897 books with AR scores and 314 books with Lexile scores among the books in BookGL at ARbookfind.com and Lexile.com, respectively. As shown in Figure 5.2(b), ReLAT outperforms AR and is significantly more accurate than Lexile in predicting the grade level of the analyzed books (in BookGL).

5.5.4 Performance Evaluation of BReK12

In this section, we verify the correctness of the design methodology of BReK12 and compare its performance with a number of existing recommendation strategies.

Effectiveness of BReK12

The results of the study conducted to validate the methodologies applied by BReK12 for *selecting* and *ranking* books to be recommended for K-12 users are presented as follows:

- The Enhanced Cosine (EC) measure employed by BReK12 to perform content-similarity matching outperforms the Traditional Cosine measure (as shown in Figure 5.3), which has been verified based on the improvements in Precision@10, MRR, and nDCG that are statistically significant as determined by using the Wilcoxon signed-rank test [42] (with $p < 0.05$).

- We have observed statistically significant improvement ($p < 0.05$) when books with unsuitable readability levels for the respective BiblioNasium users are excluded prior to applying the content-similarity matching on potential book recommendations. (See EC versus EC + ReLAT Filtering in Figure 5.3.)

- The statistically significant improvements ($p < 0.01$) on various performance metrics achieved by EC + Readership over EC indicate that examining both the content and readership similarity of candidate books with respect to books in the profile of each BiblioNasium user increases the accuracy of the recommendations.

- We have empirically verified that recommending books that match users' reading abilities without considering their individual preferences is not beneficial. This is anticipated, since users might not find particular topics addressed in books appealing even if the books are suitable to their readability levels.

- When books beyond/below users' readability levels are not chosen as candidate books by BReK12, its overall effectiveness increases, according to the statistically significant improvements ($p < 0.01$) achieved by BReK12 over EC + Readership. This comparison validates the correctness of BReK12's design methodology, i.e., to recommend books of interest to individual users that they can read and understand.

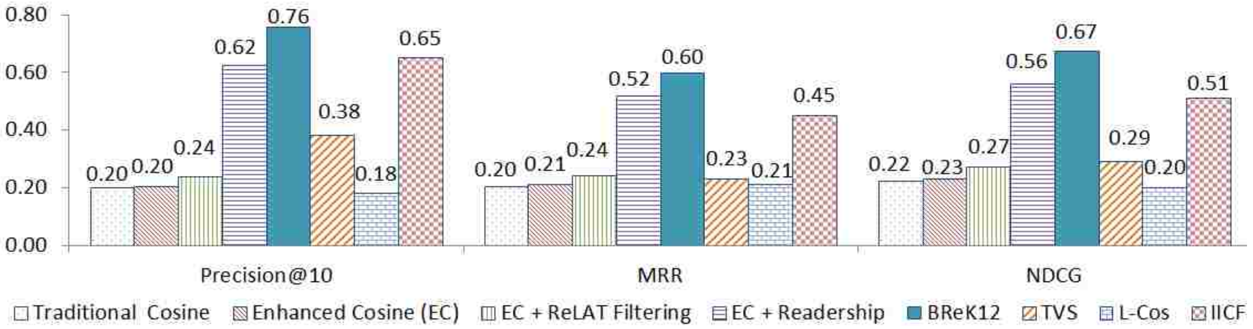


Figure 5.3: Performance evaluation of TVS, L-Cos, IICF, and BReK12 using the BiblioNasium dataset

Comparing BReK12 with Other Recommenders

As previously stated, no other personalized book recommender explicitly considers the reading abilities of its users. Thus, we have compared the performance of BReK12 with a number of recommenders developed for a general audience.

Tag Vector Similarity (TVS) [61], *L-Cosine* (L-Cos) [109], and *Item-Based Collaborative Filtering* (IICF) [24] were employed for comparison purposes, as opposed to other state-of-the-art book recommenders introduced in Section 5.2, since the latter require *personal ratings* on (K-12) books provided by individual users or *social connections* established by social bookmarking site (K-12) users, neither of which are archived by BiblioNasium. To determine which books should be recommended to a user, TVS applies the *cosine similarity measure* on *TF-IDF* tag vector representations of books,[7] whereas L-Cos considers the *weighted frequency* of each keyword in the description or title of a book. IICF, on the other hand, calculates the *degree of similarity* between any two books based on the number of users who have bookmarked both books on a social bookmarking site, which is a variation of the popular *collaborative filtering* strategy commonly adopted for making recommendations.

As shown in Figure 5.3, BReK12 outperforms its counterparts based on the evaluation metrics introduced in Section 5.5.2. The improvements achieved by BReK12 over the others are statistically significant (with $p < 0.01$). According to the computed MRR, on the average BReK12

[7] Tag descriptions on books can be extracted from the INEX 2012 Social Book Search Track dataset (inex.mmci.uni-saarland.de/tracks/books/).

users are required to browse through at least one ($\cong \frac{1}{0.60}$ = 1.67) recommended book before locating a relevant one, whereas users of TVS, L-Cos, and IICF are required to scan through at least 4 ($\cong \frac{1}{0.23}$ = 4.3 and $\cong \frac{1}{0.21}$ = 4.8) or 2 ($\cong \frac{1}{0.45}$ = 2.2) recommended books, respectively. The Precision@10 values reflect that, in general, close to 8 (out of 10) books suggested by BReK12 are relevant, as opposed to close to 4, 2, and 6 relevant books recommended by TVS, L-Cos, and IICF, respectively. The nDCG scores indicate the superiority of BReK12 over TVS, L-Cos, and IICF in ranking relevant books to be recommended *higher* in the list of suggested books.

While TVS, L-Cos, and IICF consider only textual descriptions of books or bookmaking patterns of users on a social site, BReK12 examines *multiple* contributing factors to identify potential recommendations, which increases the number of relevant reading selections for the users.

5.6 Conclusions and Future Work

We have introduced BReK12, a unique recommender tailored to K-12 readers, which makes *personalized* suggestions on books that satisfy both the *preferences* and *reading abilities* of its users. Unlike current state-of-the-art recommenders that rely on the existence of users' historical data in the form of *ratings*, which are missing among the K-12 users, BReK12 simply considers readily available *brief descriptions* on books, *patterns* of *co-occurrence* among books bookmarked on a social bookmarking site on which BReK12 is installed, and *grade levels* of books computed using our newly-developed ReLAT. ReLAT is novel, since it can determine the grade level of any book (even if a sample of the text of a book is unavailable) by analyzing the Subject Headings of books, US Curriculum subject areas identified in books, and information about the authors of books. As children continue to read more books if they can *choose* what to read [7], a significant contribution of BReK12 is to provide K-12 readers a selection of suitable books to choose from that are not only appealing to them, but can be comprehended by them. The conducted experiments demonstrate (i) the accuracy of ReLAT and its superiority over existing readability formulas/analysis tools, and (ii) the effectiveness of BReK12, which outperforms baseline recommenders in suggesting books for K-12 users.

As part of our future work, we plan to extend BReK12 so that it can suggest reading materials for struggling readers, i.e., readers with learning disabilities and those who learn English as a second language, for whom the grade level of a recommended book is an important factor to be considered.

Chapter 6

Automating Readers' Advisory to Make Book Recommendations for K-12 Readers

Abstract:

The academic performance of students is affected by their reading ability, which explains why reading is one of the most important aspects of English curriculums. Promoting good reading habits among K-12 students is essential, since research studies have confirmed the enormous influence of reading on students' development as learners and members of society. In accomplishing this task, it is indispensable to provide readers with engaging reading selections that can motivate them to read. Unfortunately, existing book recommenders have failed to offer adequate reading choices for K-12 readers, since they either ignore the reading abilities of their users or cannot acquire the much-needed information to make recommendations due to privacy issues. To address these problems, we have developed Rabbit, a book recommender that emulates the readers' advisory service offered at school/public libraries. Rabbit considers the readability levels of its readers and determines the facets, i.e., appeal factors, of books that evoke subconscious, emotional reactions on a reader. The design of Rabbit is unique, since it adopts a multi-dimensional approach to capture the reading abilities, preferences, and interests of its readers, which goes beyond the traditional book content/topical analysis. Conducted empirical studies have verified that Rabbit outperforms recommenders used at GoodReads, NoveList, and other readability-based book recommendation systems.

6.1 Introduction

Besides watching TV, text messaging, and playing computer games, children and teenagers these days often spend spare time browsing through the Internet, looking for something fun to do. YouTube, Last.fm, and Facebook are examples of popular social media sites among young audiences which offer free entertainment on demand anytime and anywhere throughout the day. Statistics reported by the New York Times[1] have shown that children/teenagers spend an average of seven hours a day on (smart) devices. This is alarming, since a significant number of children/teenagers are underachieving at school, especially in *reading*. For example, according to the 2013 National Assessment of Educational Progress, only 32% of American 4^{th} graders are proficient in reading [110]. Kids should allocate some of their free time on reading to enhance their educational experience. In order to turn the tide, educators, parents, government agents, and private organizations must join force to encourage kids to read. Unfortunately, very few existing (social) websites/(non-)government agents are equipped with the resources/technologies to cope with the problem.

To motivate K-12 students to read, it is imperative to avoid presenting these readers with books that are either too easy/difficult to read or involve topics unappealing to them which could diminish their interest in reading [7]. In fact, finding the right books for the right audience is not easy [152]. Even though existing recommenders can assist readers in finding books, they rely on either large historical data in the form of personal tags/ratings (which might not be available) or readers' connections/interactions on a social site (which may not be accessible for K-12 readers due to privacy issues). Furthermore, these systems *ignore* the reading abilities of the respective readers in recommending books. To address the shortcomings of these design methodologies, we have developed Rabbit (Readers' advisory based book recommendation tool) that makes personalized book suggestions for K-12 readers.

Rabbit is unique, since it simulates readers' advisory (RA), a service offered at school/public libraries which are established to champion and encourage reading. RA involves knowledgeable professionals in finding reading materials of interest for their patrons [138]. During the search

[1] nytimes.com/2010/01/20/education/20wired.html?_r=0

process, librarians identify the *topics*, *contents*, and *appeal factors*, i.e., literary elements, appealing to individual readers and suggest books accordingly. By offering the RA service, which is in high demand even in the era of the Web 2.0 [159], libraries provide “a vital link between library materials and readers” [138]. Unfortunately, as stated in [67], (young) readers may not approach readers’ advisors to ask for suggestions, feel their interest are *obscure* or *low-brow*, or not even visit libraries in person. By automating the RA process using a multi-dimensional recommendation strategy, we replace RA in finding books appealing to K-12 readers, which eliminates the interaction with professionals and at the same time handles any number of readers anywhere and anytime simultaneously that cannot be achieved by traditional RA.

Rabbit is novel, since besides analyzing the reading ability of a reader R, Rabbit examines appeal factors to capture the reasons why a book Bk is appealing to R. Rabbit determines the facets of Bk that instigate a subconscious, emotional reaction to Bk, which in turn impacts R’s perception on Bk. Rabbit explores the literary elements of books that identify the *rate* in which the stories unfold (pace), their *overall structure* (storyline), the *feelings* that these stories evoke on a reader (tone), subject *matters* that some readers might find unpleasant or offensive (special topics), in addition to the *qualities* of the *characters* (characterization) and the *language* and *level of details* (writing style) of the stories. Rabbit is neither affected by the cold-start problem nor requires feedback from its users, and its design surpasses the content or reading patterns explored by current state-of-the-art content/collaborative-filtering/hybrid recommenders.

While topics and content descriptions of books are freely and publicly available from reputable online sources, such as the Library of Congress,[2] appeal-factor/-term descriptions, which are fundamental for our recommendation strategy, are either determined by professionals on-the-fly or accessed through a paid subscription to RA databases. To automate the process of extracting appeal-term descriptions of books, we have developed ABET (Appeal-based extraction tool), a component of Rabbit, which relies on book reviews retrieved from well-known book-related websites, such as Amazon(.com) and Powells.com. ABET is based on effective rules that simply

[2] catalog.loc.gov/

examine typed dependencies and part-of-speech tags of words in book reviews to identify appeal factors and their terms. ABET relies on book reviews for extracting appeal-term descriptions on books, since reviews are readily available online and capture readers' varied opinions on describing literary elements of a book.

The remaining of this paper is organized as follows. In Section 6.2, we discuss existing approaches on information extraction and book recommendation. In Section 6.3, we provide a brief overview of RA. In Sections 6.4 and 6.5, we detail the design methodology of ABET and Rabbit, respectively. In Section 6.6, we present the results of the empirical studies conducted to verify the performance of ABET and Rabbit. In Section 6.7, we give a concluding remark and address directions for future work.

6.2 Related Work

Given that Rabbit is based on an extraction module, i.e., ABET, to create appeal-term descriptions of books, we discuss existing approaches that extract information from product reviews and recommend books, respectively.

6.2.1 Extracting Information from Reviews

Numerous approaches have been developed to identify and extract either features, (the polarity of) opinions, or feature-opinion pairs from reviews based on bootstrapping, natural language processing, machine learning, extraction rules, latent semantic analysis, statistical analysis, and information retrieval. (An in-depth review of state-of-the-art approaches adopted for opinion mining and extraction can be found in [115].) A product review describes products' actual features, such as the "zoom" of a camera, which is unlike a book review that evaluates "organization and writing style, possible market appeal, and cultural, political, or literary significance" of a book [141]. A book review is a form of literary criticism in which the work is analyzed based on its content, style, and merit. We have observed that existing information extraction approaches on product reviews are ill-equipped for book reviews, since sentences expressed in book reviews tend to be more elaborated than the ones used to describe products. For this reason, ABET is not designed using any particular extraction approach. Instead, it relies on simple rules to perform linguistic and semantic analysis of the content and writing style of book reviews.

6.2.2 Book Recommenders

A number of book recommendation systems have been developed in the past. Amazon's recommender suggests books based on the purchase patterns of its users [89], whereas Yang et al. [156] analyze users' access logs to infer their preferences and apply the traditional collaborative-filtering (CF) strategy to make book recommendations. Givon and Lavrenko [57] combine CF and social tags to capture the content of books for recommendation. Sieg et al. [144] rely on the standard user-based CF framework and incorporate semantic knowledge in the form of a domain ontology to capture the topics of interest to a user. The hybrid-based recommenders in [57, 144, 156], in addition to Rabbit, overcome the cold-start problem. Unlike Rabbit, however, the others require (i) historical data on the users in the form of ratings, which may not always be available, or (ii) an ontology, which can be labor-intensive and time-consuming to construct.

Unlike Rabbit, PReF [120] examines users' connections as part of the recommendation process, which may not be accessible for K-12 readers due to privacy imposed on children. BReK12 [122], which is the closest book recommender compared with Rabbit, is based on content and readability analysis. The former, however, analyzes reading patterns of users which depends on the availability of bookmarking information offered by social bookmarking site users. Furthermore, with the exception of BReK12, neither of the aforementioned recommenders considers the readability level of their users as part of their recommendation strategies. Even though Rabbit is not a recommendation system for learning, its design goal is to enhance reading selections for K-12 users. (An in-depth description of existing recommenders in the educational domain can be found in [96].)

6.3 Readers' Advisory (RA)

Rabbit emulates the readers' advisory (RA) service, which has been available at public libraries since the late 1800's [41, 138]. RA offers fiction and non-fiction readers materials of potential interest with "the help of knowledgeable and non-judgmental library staff" [138]. While the traditional RA model involves face-to-face discussions between patrons and librarians, a number of public libraries, such as William Regional Library,[3] take advantage of existing technologies and replace human interactions with online forms filled out by patrons to capture the users' interests in books [67].

Either through face-to-face conversations or filled-out online forms, a RA librarian's task is to identify the type of books preferred by readers based on the reasons behind their preferences. Besides analyzing the topical areas and content descriptions of books favored by a reader R, during the RA process, librarians examine the *appeal factors* of books that are appealing to R [138]. Appeal factors, such as the pacing or description of characters in books, are "the elements of a book—whether definable or just understood—that make readers enjoy the book" [138]. These factors capture general traits of a book that attract the attention of a reader and are considered in answering one of the most important RA questions, "Why is the reader interested in a given book?" [2]. For example, some readers might enjoy the Harry Potter books (by J. K. Rowling) because of the established friendships among students and the boarding school setting, whereas others like the fantasy aspect of the story.

To the best of our knowledge, there is no consensus on the set of appeal factors that must be considered during the RA process. The most prominent appeal factors, as articulated in RA-related literature [41, 67, 138] include: (i) characterization, (ii) frame, (iii) pacing, (iv) storyline, (v) language and writing style, (vi) tone, and (vii) special topics. The first six appeal factors are well-known literary elements of fiction/non-fiction books [40], whereas the latter identifies subjects addressed in a book that can cause emotional stress to some readers but tolerated/enjoyed by others [41]. Each appeal factor is associated with a vocabulary, which is a set of keywords,

[3] wrl.org/books-and-reading/adults/looking-good-book

Table 6.1: Sample appeal terms for each of the appeal factors considered by Rabbit

Appeal Factors	Appeal Terms
Characterization	Believable, distant, dramatic
Frame	Bittersweet, contemporary, descriptive
Language and Writing Style	Candid, complex, conversational, extravagant, poetic, prosaic
Pacing	Easy, fast, slow
Special Topics	Addiction, bullying, violence
Storyline	Action-oriented, character-centered
Tone	Dark, happy, surreal

called *appeal terms*, employed to describe the factor, which we defined based on well-known RA literature [41, 138]. The appeal factors considered by Rabbit and a sample of their respective appeal terms are shown in Table 6.1.

Based on the contents, topics, and appeal terms that describe the appeal factors of books $preferred$ by a reader R, librarians suggest other books matching (to a certain degree) the interests/preferences of R. However, due to the amount of books being published on a regular basis these days, it is an impossible task for a librarian to be familiar with every existing book to determine if it could be a potential relevant recommendation for R. For this reason, librarians turn to RA databases, which are available at NoveList Plus, Fiction Connection, Which Book, and Readers' Advisory Online, to conduct fact-based, appeal factor-oriented, and read-alike searches in locating books to suggest to a reader [41].

6.4 Appeal-Term Descriptions

While Rabbit conducts topical and content analysis of books during the RA process using data retrieved from book-related websites, appeal-term descriptions are only available through RA databases or determined by professionals. Unfortunately, accessing reputable RA databases, such as NoveList Plus, comes with a price tag, i.e., paid subscription, whereas professionals might not have read a particular book and thus it would not be possible for them to infer the corresponding appeal-term description for the book on-the-fly. To address these constraints, we have developed

ABET, a tool that automatically extracts appeal-term descriptions of books from book reviews available at well-known book-related websites, such as Amazon, Bertrams, Bookfinder4u, Bookmooch, Dogobooks, Fishpond, Powells, and Thriftbooks. As reading is a personal experience, it is anticipated that a book is (not) appealing to a reader, who is familiar with the content of the book, for various reasons. By analyzing reviews, we extract diverse readers' opinions on a book based on appeal terms that describe the corresponding appeal factors of the book, which in turn facilitates the task of identifying why books are appealing based on their literary elements.

To generate appeal-term descriptions for a given book, ABET relies on the taxonomy defined in Section 6.3, which despite being comprehensive, cannot account for a given appeal factor/term being specified differently in readers' reviews. For example, a reviewer may refer to the "Storyline" appeal factor of a book as "story" or "narrative", and (s)he may also use either "quick" or "fast" as the appeal term to describe the appeal factor "Pace". For this reason, we extend the appeal factors/terms by including the (stemmed) synonyms of each term/factor, which are identified using WordNet[4], a popular lexical database for the English language. (The complete list of appeal terms for each appeal factor can be found at goo.gl/BSwuPw.)

While the taxonomy can serve as an aid to identify potential appeal factors/terms in reviews, it is imperative to properly associate these appeal terms and appeal factors in the reviews so that appeal factor-appeal term pairs can be correctly extracted to generate an accurate appeal-term description for a given book. To accomplish this task, we have defined a number of extraction rules[5] (as given in Table 6.2) for ABET based on typed dependency relations between word pairs in sentences extracted from reviews. It is natural for ABET to turn to typed dependencies, since they capture the *semantic connection*, i.e., association, between words in sentences. (A detailed discussion on typed dependencies is available in Stanford typed dependencies manual.[6])

Rules used by ABET are simple and yet effective, which are based on written patterns identified in book reviews and capture the semantic link between appeal terms and their corresponding

[4] wordnet.princeton.edu

[5] ABET performs linguistic and semantic analysis on sentences in reviews using Stanford Part-of-Speech Tagger and Dependency Parser (nlp.stanford.edu/software/lex-parser.shtml).

[6] nlp.stanford.edu/downloads/dependencies_manual.pdf

Table 6.2: Rules considered by ABET to identify appeal factor-appeal term pairs in book reviews

Notations
$rel(A, B)$ is a *grammatical relation* between a *dominant*, i.e., *governor* or *head*, word (A) and a *subordinate*, i.e., *dependent* or *modifier*, word (B)
L_F, L_T, EL_F, and EL_T are the list of appeal factors, list of appeal terms, extended list of appeal factors, and extended list of appeal terms, respectively.
w_f is an appeal factor in L_F, and w_t is an appeal term in L_T
$w \rightsquigarrow w_f$ ($w \rightsquigarrow w_t$, respectively) denotes that w is a synonym of w_f (w_t, respectively)
$POS(w)$ is the part-of-speech tag of w which is a verb (adverb, respectively) if $POS(w)$ = "VB" ("RB", respectively)
Abbreviation: adv(erbial)mod(ifier), a(djectival)mod(ifier), c(lausal)comp(lement), d(irect)obj(ect), neg(ation modifier), nn (noun compound modifier), n(nominal)subj(ect), nsubjpass (passive nominal subject), prep(_*) (Prepositional modifier)
ABET only extracts a pair $< w_f, w_t >$ if w_t is in the corresponding vocabulary defined for w_f

Rule	Objective	Conditions	Identified Factors/Terms
1	To capture the written patterns based on a keyword, i.e., appeal term, that immediately precedes/ follows the subject or object of a sentence S, i.e., appeal factor	$A \in EL_T, B \in EL_F, rel \in$ {nn, nsubj}, (If A is a synonym of a term that applies to "Characterization" or "Storyline", then $POS(A) \notin$ {"VB, "RB"})	$B \rightsquigarrow w_f$ $A \rightsquigarrow w_t$
2		$A \in EL_F, B \in EL_T, rel \in$ {advmod, amod, prep_in, prep_about}	$A \rightsquigarrow w_f$ $B \rightsquigarrow w_t$
3	To identify an appeal term that qualifies its indirectly related appeal factor in S	$rel \in$ {nn, nsubj}, $B \in EL_F$, and $\exists rel_2(C, D) \in$ {amod, dep, ccomp}, $A = C, D \in EL_T$	$B \rightsquigarrow w_f$ $D \rightsquigarrow w_t$
4	To explicitly consider *negated* appeal terms in S	$B \in EL_F, rel \in$ {nn, nsubj}, $\exists neg(C, D)$, A (= C) is an antonym of $\bar{A} \in EL_T, D$ is a negation term	$B \rightsquigarrow w_f$ $\bar{A} \rightsquigarrow w_t$
5	To account for the multiple ways in which a reviewer can describe the setting of books or peeves/favored subject matters in books to handle special cases of "Special Topic" and "Frame" factors in S	$A \in EL_T, rel \in$ {prep_about}, A is a synonym of a term that describes "Special Topics"	w_f = "Special Topics" $A \rightsquigarrow w_t$
6		$B \in EL_T, rel \in$ {dobj, nsubj, nsubjpass}, $POS(A)$ = "VB", B is a synonym of an appeal term that describes "Special Topics"	w_f = "Special Topics" $B \rightsquigarrow w_t$
7		$A \rightsquigarrow$ "Frame" $\in EL_F, B$ (a synonym of an appeal term that describes "Frame") $\in EL_T, rel \in$ {prep_in}	w_f = "Frame" $B \rightsquigarrow w_t$

appeal factors. Consider S_A, "The narrative of the book is dramatic", and S_B, "She creates vivid, believable characters". In S_A the *subject* of the sentence, i.e., "narrative," is characterized as being "dramatic", whereas in S_B its *object*, i.e., "characters", is described as "believable". Based on the aforementioned examples it is clear that if the subject/object of a sentence is an appeal factor, then a word in the sentence that semantically describes, i.e., is directly linked to, the mentioned object/subject is often an appeal term. ABET captures these connection patterns using Rules 1 and 2 as defined in Table 6.2.

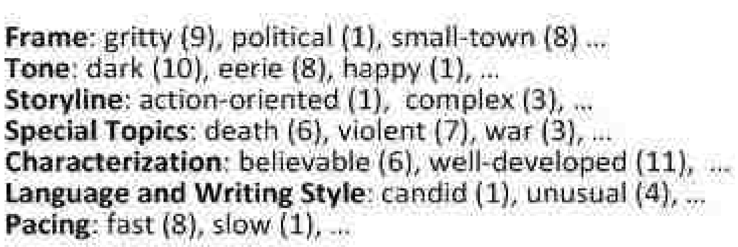
Appeal terms can also be (semantically) indirectly connected with an appeal factor in a sentence. Consider sentence S_C, "The descriptions included are extravagant". "Extravagant" is *indirectly* related to the subject of S_C, i.e., "descriptions", through the word "included". Using Rule 3, ABET examines pairs of grammatical relations that involve indirect connections among words.

Now consider S_D, “The characters are not simple”. Based on Rule 1, ABET would mistakenly describe the factor “Characterization” with the appeal term “simple”. This example reveals the need to examine pairs of grammatical relations in the presence of negated terms. ABET applies Rule 4, which identifies a negated term as a modifier of an appeal term t and then extracts as the appeal term for the corresponding factor the antonym of t (if it is included in the vocabulary defined in ABET’s taxonomy for the factor).

Together, Rules 1 to 4 account for the most common written patterns for appeal terms/factors often observed in reviews. These rules simply look for words in sentences that (directly or indirectly) describe the appeal factors of a book (considered by ABET), which are often the subjects or objects of sentences. In fact, Rules 3 and 4 take precedence over Rules 1 and 2, since once a typed dependency in a sentence is used by either of the former rules, it cannot be considered by the latter ones.

While majority of other relations (beyond the ones captured by Rules 1 to 4) seldom appeared in reviews, we observed three *special cases* that facilitate the extraction of appeal terms for “Special Topics” and “Frame”, respectively, which we defined in Rules 5 to 7. Consider S_E, “It is about violence at schools”, S_F, “Bullying is depicted in the book”, and S_G, “The action is set in a school”, which include special written patterns pertaining to the “Frame” and “Special Topics” appeal factors that are based on prepositions, subjects, and objects identified in sentences in reviews. The preposition “about” in S_E captures an appeal term employed to describe the factor “Special Topics”, i.e., “Violence”, whereas “in” in S_G is connected with an appeal term, i.e., “School”, that describes the factor “Frame”. Moreover, “bullying” in S_F, which is assigned a “VB” part-of-speech tag, is an appeal term describing the factor “Special Topics”.

ABET creates the appeal-term description for a book Bk by applying rules defined in Table 6.2 on up to 500 distinct reviews of Bk, if they are available. In generating the appeal-term description of Bk, ABET considers not only the appeal terms extracted from reviews on Bk, but also their *frequency of occurrence*. The latter captures the relative *degree of significance* of an appeal term in describing its corresponding factor based on reviewers’ varied opinions on appeal



Frame: gritty (9), political (1), small-town (8) ...
Tone: dark (10), eerie (8), happy (1), ...
Storyline: action-oriented (1), complex (3), ...
Special Topics: death (6), violent (7), war (3), ...
Characterization: believable (6), well-developed (11), ...
Language and Writing Style: candid (1), unusual (4), ...
Pacing: fast (8), slow (1), ...

Figure 6.1: ABET-generated appeal-term description for "The Hunger Games" where the number indicates the *frequency* in which a term was used to describe its corresponding appeal factor in the reviews

factors that apply to Bk. (A sample of the appeal terms and their frequencies of occurrence for each of the corresponding appeal factors identified using ABET on the book reviews for "The Hunger Games" by Suzanne Collins are shown in Figure 6.1.)

6.5 Our Proposed Recommender

In this section, we present the recommendation strategy of Rabbit for K-12 readers. Rabbit first analyzes the *profile* of a reader R, which consists of a set of N ($\geq$ 1) books either provided by R or bookmarked[7] by R on a social bookmarking site of which R is a member. Based on the profile, Rabbit identifies books (as detailed in Section 6.5.1) that are compatible with the readability level of R, which are treated as *candidate books* to be considered for recommendation. Candidate books are selected among the books available at one of the (online) book repositories, which include, but are not limited to, (i) *reputable websites*, such as OpenLibrary.org or WorldCat.org, which are two of the largest online library catalogs, (ii) *school/public libraries*, and (iii) book-related *bookmarking sites*, such as Biblionasium.com, which is a website that encourages reading among children/teenagers. Rabbit computes a ranking score, which quantifies the *degree of relevance*, of each candidate book with respect to the profile of R using a regression model (introduced in Section 6.5.3) applied to the analytical results of the book using diverse publicly available information (as discussed in Section 6.5.2).

[7] Only books bookmarked by a reader during the most recent academic year are considered, since it is anticipated that the grade levels of books bookmarked by readers gradually increase as the readers enhance their reading comprehension skills over time.

6.5.1 Candidate Books

One of the design goals of Rabbit is to suggest books that its users/readers can comprehend. It is imperative for Rabbit to locate books with grade levels adequate for each individual reader, since we realize that "reading for understanding cannot take place unless the words in the text are accurately and efficiently decoded" [112]. In order to accomplish this task, Rabbit first determines the readability level of a reader R by analyzing the grade levels of books in R's profile. The readability level of R is determined by averaging the grade level of each book (denoted P_B) in R's profile, computed using TRoLL [48], a tool for regression analysis of literacy levels, which captures the *central tendency* of the grade levels of books that have been read by R accurately. Unlike existing popular prediction formulas/tools,[8] such as Flesch-Kincaid, Lexile Framework, and ATOS (discussed in detail in [19]), TRoLL computes the grade level of any book using metadata on books publicly accessible from reputable online sources, even in the absence of book excerpts.

Having determined the readability level of R using TRoLL, Rabbit applies Equation 6.1 to determine the set of candidate books considered for recommendation.

$$SCB(R) = \{CB \mid CB \in Rep \wedge \quad TRoLL(CB) \in [\frac{\sum_{P_B \in P} TRoLL(P_B)}{|P|} \pm 0.5]\} \tag{6.1}$$

where CB is a candidate book available at a book repository Rep, $|P|$ denotes the number of books in R's profile, and $TRoLL(CB)$ ($TRoLL(P_B)$, respectively) is the grade level of CB (P_B, respectively) determined by TRoLL. By selecting books within *half a grade level* above/below the *mean* readability level of R, Rabbit considers books for recommendation within an appropriate level of (text) complexity for R based on the grade levels of books in R's profile.

[8] These formulas/tools rely on sample text of a book to compute its readability level, which is a severe constraint, since sample text is not always freely accessible due to copyright laws.

6.5.2 Analysis of Multiple Perspectives

Rabbit suggests books that not only readers can comprehend, but also they are interested in, by simultaneously examining books in the profile of R (and each candidate book CB) based on diverse publicly accessible metadata to determine (i) the *topics of interest* for R (in Section 6.5.2), (ii) *book contents* appealing to R (in Section 6.5.2), and (iii) the general *traits* that describe books favored by R (in Section 6.5.2).

Exploring Topical Information

Rabbit examines the *topical description* (i.e., topic) of a book, which is based on Library of Congress Subject Headings (LCSH) assigned to the book by professional cataloguers. LCSH, which is a de facto universal controlled vocabulary, constitutes the largest general indexing vocabulary in the English language [157]. Subject headings, which are *terms* or *phrases* that denote concepts, events, or names [134], are used by librarians to categorize and index books according to their themes. Examples of LCSH include "Computers and college students," and "Archaeology–History–18th century".

Rabbit, which explores the topical resemblance between a candidate book CB and books in R's profile P, examines the degree to which the distribution of topics in CB matches the distribution of topics of books in P. This topical similarity measure, which is defined using the well-known vector space model (VSM) and computed in Equation 6.2, prioritizes candidate books that have been assigned LCSH which match the LCSH favored by R, i.e., LCSH most frequently assigned to books in P.

$$TSim(CB, P) = \frac{\vec{CB} \cdot \vec{P}}{||\vec{CB}|| \times ||\vec{P}||} \tag{6.2}$$

where CB and P_B are represented as n-dimensional vectors $\vec{CB}$ = $< W_{CB_1}, \ldots, W_{CB_n} >$ and $\vec{P}$ = $< W_{P_1}, \ldots, W_{P_n} >$, respectively, n is the number of distinct subject headings assigned to CB and books in P, W_{CB_i}, the weight of CB_i ($1 \leq i \leq n$), is "1" if CB_i is a subject headings of CB, and is "0" otherwise, and W_{P_i}, which is the weight of P_i ($1 \leq i \leq n$), is computed as a proportion between the number of books in P that have been assigned P_i and the total number of books in P, i.e., $|P|$.

In computing the *topical similarity* measure, in addition to other similarity measures presented below, we rely on VSM, since it handles frequency distributions, which is essential in comparing candidate books based on multiple perspectives with the profile of a reader. We have empirically verified that given the short length of the descriptions for each book based on its topics, content, and appeal factors (which include very few LCSHs, words, and terms, respectively) VSM is a more reliable distance measure compared with its probabilistic counterparts, such as KL-divergence.

The Book Content Analysis

Besides determining the topics that are of interest to a reader R, Rabbit also identifies written matters covered in books that are generally appealing to R by analyzing the content description of each book in P (and each candidate book CB), which is extracted from reputable book-related websites, such as Amazon and the Library of Congress.

Rabbit computes the *content similarity* of each CB with respect to P based on the "bag-of-words" representation of the brief descriptions of CB and (books in) P. Rabbit favors candidate books with contents compatible with the contents that are most commonly addressed in books included in R's profile. To compute $CSim(CB,\ P)$, the *content similarity* score defined in Equation 6.3, Rabbit employs an $enhanced$ version of the cosine similarity measure based on word-correlation factors (WCF) [48], which relaxes the exact matching constraint imposed by the cosine measure by exploring words in the content description of CB that are analogous to, besides the same as, words in the content description of P.

WCF in the word correlation matrix reflect the *degree of similarity* between any two words according to their (i) frequencies of co-occurrence and (ii) relative distances in a collection of Wikipedia(.org) documents. Rabbit relies on WCF, as opposed to other popular similarity metrics applied to WordNet, since we have empirically verified that word-similarity scores predicted by using WCF correlate with human assessments on word similarity. Using WordSim353,[9] which is a test collection for measuring word relatedness, and STS,[10] which provides human assessments on sentence similarity for 750 pairs of sentences, we compared the performance of WCF with FaITH[11] [127], which is a feature and information theoretic-based similarity measure. (Only FaITH was considered, since as reported in [127], it outperforms other well-known information-theoretic, ontology, and hybrid-based approaches that exploit word-related information available at WordNet to estimate the degree of similarity between pairs of words.) The results of the experiments verified that the performance of WCF is comparable to that of FaITH, which is based on a Wilcoxon signed-ranked test of significance (with $p <$ 0.05) [42].

$$CSim(CB, P) = \frac{\vec{CB} \cdot \vec{P}}{||\vec{CB}|| \times ||\vec{P}||} \tag{6.3}$$

where the content of CB and P are represented as vectors $\vec{CB}$ = $<W_{CB_1}, \ldots, W_{CB_n}>$ and $\vec{P}$ = $<W_{P_1}, \ldots, W_{P_n}>$, respectively, n is the number of distinct non-stop, stemmed keywords in the brief descriptions of CB and each book in P, and W_{P_i} and W_{CB_i} are the $weights$ of keywords P_i and CB_i, which are calculated using Equations 6.4 and 6.5 such that the frequency distribution of words in $\vec{CB}$ and $\vec{P}$ is determined based on the frequency distribution of non-stop, stemmed words among the brief descriptions of CB and all the books in P, respectively.

$$W_{CB_i} = \begin{cases} \frac{f_{CB_i,CB}}{max_{CB_i \in CB}(f_{CB_i,CB})} & \text{if } CB_i \in CB \\ \frac{\sum_{c \in HS_{P_{B_i}}} f_{c,CB}}{|HS_{P_{B_i}}|} & \text{otherwise} \end{cases} \tag{6.4}$$

$$W_{P_i} = \begin{cases} \frac{f_{P_i,P}}{max_{P_i \in P}(f_{P_i,P})} & \text{if } P_i \in P \\ \frac{\sum_{c \in HS_{CB_i}} f_{c,P}}{|HS_{CB_i}|} & \text{otherwise} \end{cases} \tag{6.5}$$

where

[9] cs.technion.ac.il/~gabr/resources/data/wordsim353/

[10] goo.gl/KzMHgn

[11] Available at grid.deid.unical.it/similarity

- HS_w is the set of words that are *highly similar* to, but not the same as, a given word w in the brief description D of either CB or P. (A word is considered highly similar if it is included in a reduced version of the WCF matrix which contains 13% of the most frequently-occurring words in the Wikipedia collection.)
- $|HS_w|$ is the size of HS_w
- $f_{w,D}$ is the *frequency of occurrence* of w in D

Equation 6.4 (6.5, respectively) explicitly accounts for words in CB (P, respectively) that might be similar to, but do not exactly match, words in P (CB, respectively).

Examining Appeal-Term Descriptions

Besides analyzing the topics and content of interest to a reader R, Rabbit examines the appeal elements of books preferred by R and determines the *appeal similarity* of each candidate book CB based on the appeal-term description of CB and (each book in) the profile P of R, which are generated using ABET.

In calculating $ATSim(CB, P)$, the *appeal-term similarity* score of CB with respect to P, Rabbit adopts the cosine measure as shown in Equation 6.6. $ATSim(CB, P)$ captures the *overall degree of resemblance* between CB and the profile P of R based on the respective appeal-term distributions associated with each appeal factor, i.e., respective appeal-term descriptions generated using ABET on reviews of CB and each book in P.

$$ATSim(CB, P) = \frac{\sum_{af \in AF} \frac{\vec{CB}_{af} \cdot \vec{P_{af}}}{||\vec{CB}_{af}|| \times ||\vec{P_{af}}||}}{|AF|} \tag{6.6}$$

where AF is the set of appeal factors in the appeal-term descriptions for CB and P, $|AF|$ is the size of AF, $\vec{CB}_{af}$ and $\vec{P_{af}}$ are the n-dimensional vector representations of the appeal-term distribution of an appeal factor af for CB and P, respectively, n is the number of distinct appeal terms in the distributions of the corresponding appeal factor for CB and P, $W_{CB_{af_i}} = \frac{freq_{i,CB_{af}}}{\max_{i \in CB_{af}} freq_{i,CB_{af}}}$ is the *weight* of the i^{th} term in $C\vec{B}_{af}$, and $W_{P_{af_i}} = \frac{\sum_{P_B \in P} freq_{i,P_{B_{af}}}}{\max_{i \in P_{af}} \sum_{P_B \in P} freq_{i,P_{B_{af}}}}$ is the *weight* of the i^{th} term in P_{af}.

6.5.3 Ranking Candidate Books

Rabbit examines multiple information sources, which include *topics*, *contents*, and *general traits* of books preferred by a reader R to identify books to be recommended for R.

To predict the ranking score of each candidate book CB, Rabbit employs the multiple linear regression analysis [155], which is a classical statistical technique for building estimation models [150], as defined in Equation 6.7. The analysis accounts for the influence of multiple contributing factors, which are derived from the ranking scores of topical, content, and appeal-term descriptions of CB. The top-10 ranked books with the highest combined *degrees of similarity* are recommended to R.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n \tag{6.7}$$

where Y is the dependent variable, i.e., ranking score of CB, β_0 is the intercept parameter, $\beta_1, \ldots, \beta_n$ are the coefficients of regression, $X_1, \ldots X_n$ are the independent variables (predictors), i.e., the scores defined in Section 6.5.2 for CB, and n is the number of predictors in the regression analysis [155].

In Equation 6.7, each unknown parameter, i.e., the intercept and coefficients of regression, which is required to determine the ranking score of CB, is estimated through a one-time training process using the Ordinary Least Squares method [155] and the $Tset$ training dataset, which is introduced in Section 6.6.2. $Tset$ consists of 1,663 instances, each of which is a book b that is either a relevant or non-relevant recommendation for a given reader R. Each instance is represented as a vector of the form $< b_1, b_2, b_3, rel_R >$, where b_i is the (value of the) i^{th} predictor ($1 \leq i \leq 3$) computed for b, and rel_R is the target, which for practical reasons is “1” if b is a relevant recommendation for R, and is “0” otherwise.

The Ordinary Least Squares method calculates the residual of each training instance in $Tset$, which is the difference between the binary relevance value of b for R and the ranking score of b estimated using the (values of the) predictors in the vector representation of the corresponding training instance and Equation 6.7. Unknown parameters are estimated by minimizing the sum of squared distances between residuals of training instances in $Tset$.

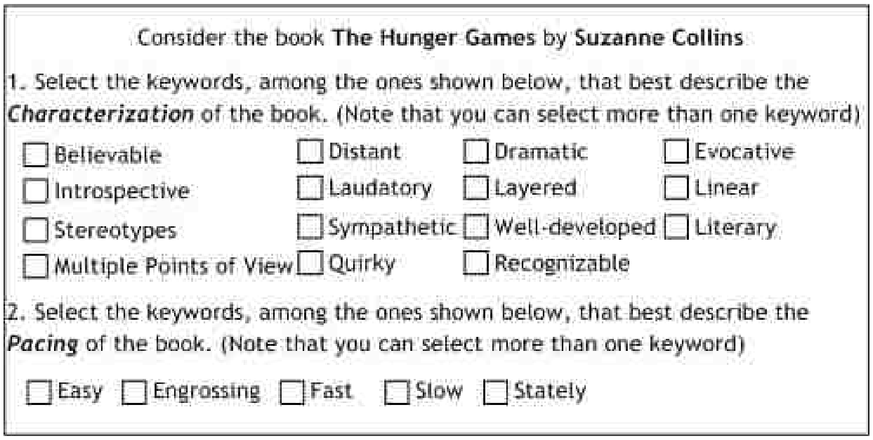
6.6 Experimental Results

In this section, we present the results of the empirical studies conducted to assess the performance of ABET and Rabbit. To conduct these studies, we relied on a number of sample sets of books, i.e., SB_1 and SB_2, which due to space constraints are available in http://goo.gl/PWE9u2. Furthermore, the statistical significance of the results presented in this section were determined using the Wilcoxon signed-ranked test.

6.6.1 Assessing the Performance of ABET

Due to the lack of existing benchmark datasets for validating the performance of tools that automatically extract appeal factor-appeal term pairs, we have assessed the performance of ABET by (i) computing the precision and recall of appeal factor-appeal term pairs extracted from book reviews, (ii) analyzing the correctness of appeal-term descriptions created by ABET, and (iii) comparing appeal-term descriptions generated by ABET with respect to the ones extracted from NoveList (Plus) on the same set of books.

We randomly selected a set of 100 books written for K-12 readers, and for each book we randomly examined a review. We *manually annotated* the appeal factor-appeal term pairs in each of the 100 examined reviews and compared the annotated pairs in each review with the ones extracted by ABET. The precision, recall, and F-measure achieved by ABET, which are 0.85, 0.82, and 0.83, respectively, verify the high accuracy of the rules defined for ABET in identifying appeal factors and their corresponding appeal terms in reviews. We have observed that majority of the pairs excluded by ABET were due to keywords used by reviewers to describe a given factor which



Consider the book **The Hunger Games** by **Suzanne Collins**

1. Select the keywords, among the ones shown below, that best describe the ***Characterization*** of the book. (Note that you can select more than one keyword)

□ Believable □ Distant □ Dramatic □ Evocative
□ Introspective □ Laudatory □ Layered □ Linear
□ Stereotypes □ Sympathetic □ Well-developed □ Literary
□ Multiple Points of View □ Quirky □ Recognizable

2. Select the keywords, among the ones shown below, that best describe the ***Pacing*** of the book. (Note that you can select more than one keyword)

□ Easy □ Engrossing □ Fast □ Slow □ Stately

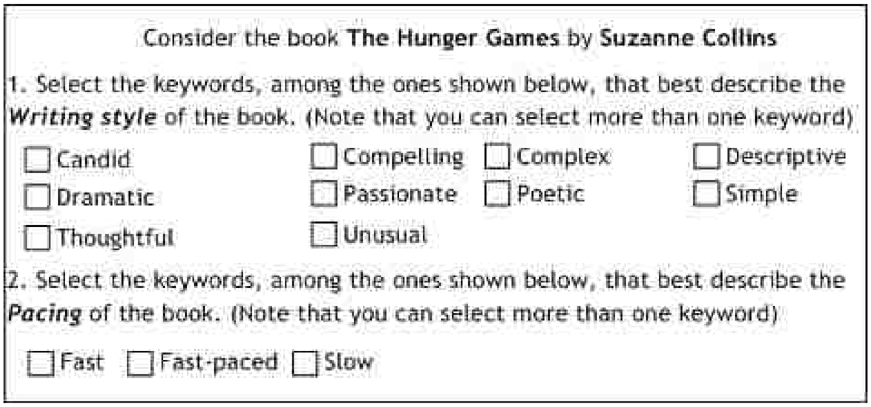
Figure 6.2: A screenshot of the Mechanical Turk survey conducted to evaluate the performance of ABET

are not included in the pre-defined vocabulary of the corresponding factor defined for ABET (as described in Section 6.4). We have also observed that poor phrasing in reviews, which in turn yields nonsensical grammatical relations between pairs of words, and lack of proper anaphora resolution have caused the majority of the extraction errors.

To further evaluate the appeal-term descriptions created by ABET, we relied on SB_1, a sample set of *eight books*, and conducted two surveys on Amazon Mechanical Turk[12]. (See a sample of the surveys in Figures 6.2 and 6.3, respectively.) For each one of the Mechanical Turk surveys, we collected 25 responses per book during the month of September 2013. Based on the average of the 200 responses collected for each survey, we compiled the results on assessing ABET and comparing its performance with NoveList, which are shown in Figures 6.4 and 6.5, respectively.

In both surveys, we asked Mechanical Turk appraisers to select the keywords, i.e., appeal terms, that best describe each appeal factor for one of the books in SB_1. While the first survey includes the vocabulary created by ABET for each appeal factor as the corresponding possible keyword choices, the second survey contains appeal terms defined by either ABET or NoveList. In the two surveys, we treated the appeal terms selected for each factor by Mechanical Turk appraisers as the "gold standard" for the factor and computed the accuracy of ABET (NoveList, respectively)

[12] Note that we asked Mechanical Turk appraisers to only participate of a HIT, i.e., survey, if they have read the books included in the corresponding survey.



Consider the book **The Hunger Games** by **Suzanne Collins**

1. Select the keywords, among the ones shown below, that best describe the ***Writing style*** of the book. (Note that you can select more than one keyword)

□ Candid □ Compelling □ Complex □ Descriptive
□ Dramatic □ Passionate □ Poetic □ Simple
□ Thoughtful □ Unusual

2. Select the keywords, among the ones shown below, that best describe the ***Pacing*** of the book. (Note that you can select more than one keyword)

□ Fast □ Fast-paced □ Slow

Figure 6.3: A screenshot of the survey conducted to evaluate the performance of ABET against NoveList

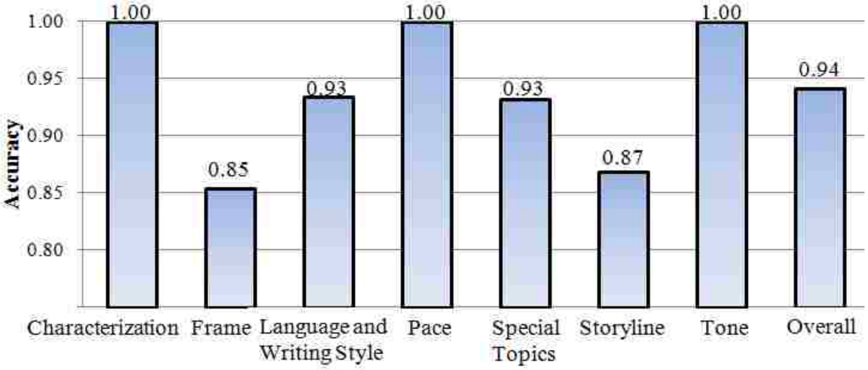


Figure 6.4: Performance evaluation of ABET conducted using Amazon Mechanical Turk

based on the proportion of terms in the gold standard of a given factor defined by Mechanical Turk appraisers which match its counterpart identified by ABET (NoveList Plus, respectively) for the factor. We relied on Mechanical Turk appraisers to perform the evaluation, since Mechanical Turk is a "marketplace for work that requires human intelligence" and allows individuals or businesses to programmatically access thousands of diverse, on-demand workers and has been used in the past to collect user feedback for multiple information retrieval tasks [80]. Furthermore, we considered NoveList for the comparison purpose, since NoveList is a premier database for readers' advisory [41] and, to the best of our knowledge, is the only RA database that includes appeal-term descriptions for books.

As shown in Figure 6.4, ABET achieves an overall 94% accuracy in identifying appeal terms (in reviews) that describe a book. More importantly, the accuracy on the identified appeal

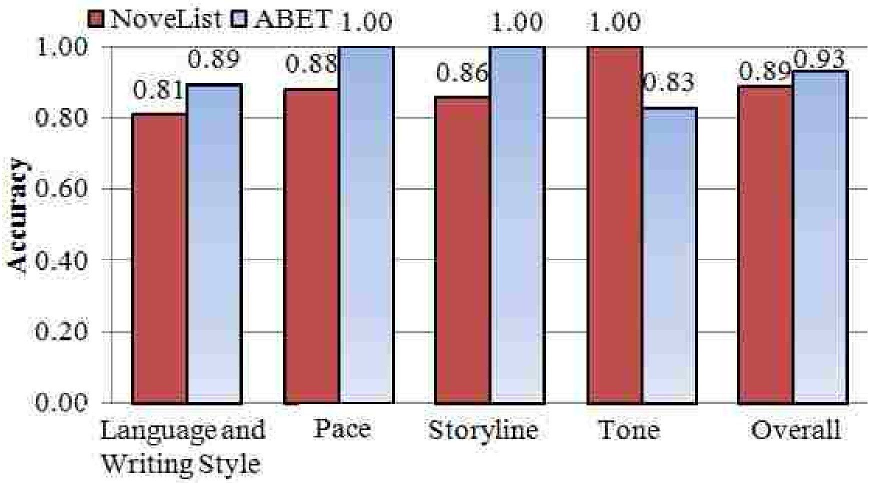


Figure 6.5: Performance evaluation of ABET and NoveList Plus using Amazon Mechanical Turk

terms for each appeal factor considered by ABET is in the upper eighty percentile or higher. Figure 6.5, on the other hand, shows the accuracy ratios of ABET and NoveList calculated using the 200 responses prepared by Mechanical Turk appraisers for the second survey. NoveList only describes books using four appeal factors, as opposed to the seven considered by ABET. For this reason, we have compared the performance of ABET and NoveList based on their common appeal factors (as shown in Figure 6.5). Based on the appraisers' assessments, we claim that appeal-term descriptions provided by ABET are favoured over the ones defined by professionals included in the RA database at NoveList. Note that the improvement in overall accuracy ratio achieved by ABET over NoveList, in addition to the improvement on "Language and Writing Style", "Pace" and "Storyline" factors, are statistically significant ($p < 0.01$).

6.6.2 Assessing the Performance of Rabbit

In this section, we discuss the empirical studies conducted to validate the design methodology and performance of Rabbit. The results of the experiments conducted to evaluate Rabbit (and compare its performance with BReK12) are based on the dataset and methodology presented in Section "Dataset and Evaluation Strategy", whereas the results of the experiments conducted to compare the performance of Rabbit with well-known recommendation modules are based on human assessments.

Dataset and Evaluation Strategy

Even though the BookCrossing dataset[13] has been employed to evaluate book recommenders tailored to a general audience, it is not specifically designed for assessing the performance of book recommenders for K-12 readers. We used data provided by BiblioNasium, which is a safe and secure social networking site on books that targets children and teenagers, to evaluate Rabbit instead. The dataset consists of the profile of books that have been bookmarked by each one of the 5,580 BiblioNasium users who joined the site within the first month of its establishment. A portion of the dataset, called $Tset$, which consists of 10% of the 5,580 BiblioNasium users and their profiles, was employed for training Rabbit's regression model, whereas the remaining users and their profiles, called $Eset$, were used for evaluation purposes. As the design methodology of Rabbit relies on *topical*, *brief content*, and *appeal-term* descriptions, in addition to the predicted grade levels of books, we retrieved the brief book descriptions and LCSH from reputable book-related websites, the appeal-term descriptions from book reviews using ABET, and the book readability levels using TRoLL.

We adopt the popular five-fold cross validation strategy to evaluate recommender systems. In each of the five repetitions, 80% of the books bookmarked by a reader R in $Eset$ yield R's profile and the remaining 20% are reserved for the testing purpose. A recommended book is treated as *relevant* to R if it is included in the 20% of the books withheld for the testing purpose, and is *non-relevant* otherwise, which is a commonly-employed evaluation protocol. Since only withheld books are considered relevant, it is not possible to account for potentially relevant books a user has not bookmarked, which is a well-known limitation of this evaluation protocol. As the limitation applies to all the recommenders evaluated in the conducted empirical studies, the results are consistent for the comparison purpose.

[13] Informatik.unifreiburg.de/$\sim$cziegler/BX

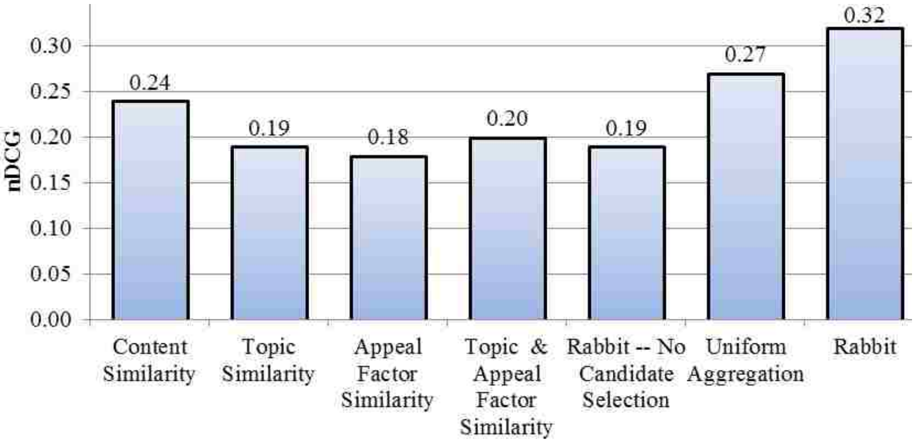


Figure 6.6: Performance evaluation of Rabbit using the $Eset$ dataset

The Evaluation of Rabbit

Conducting various empirical studies using $Eset$, we assessed the performance of Rabbit, in terms of *Normalized Discounted Cumulative Gain* (nDCG), which determines the overall (ranking) performance of a recommender and penalizes relevant recommendation positioned lower in the recommendation list. We observed that ranking candidate books solely based on *appeal factors* or *topical information* yields the lowest nDCG scores (see Figure 6.6). This is anticipated, since Library of Congress Subject Headings and appeal-term descriptions mainly identify the types of books preferred by a reader R from a general perspective, as opposed to the brief descriptions of books that explicitly capture the subjects of interest to R. Even though the content-based approach yields a relatively high nDCG score, the experimental results show that the multi-dimensional strategy of Rabbit, of which content-based analysis is a component, locates more relevant books, which is justified by the statistically significant ($p < 0.001$) difference between "Content Similarity" and "Rabbit" in Figure 6.6.

As depicted in Figure 6.6, applying the linear combination (i.e., "Uniform Aggregation") on different similarity scores considered by Rabbit yields a lower nDCG value than the one obtained by using the regression model (i.e., "Rabbit"), which is statistically significant ($p < 0.001$). The results validate the necessity of accounting for the impact, i.e., weight, of each individual similarity score.

In comparing the nDCG scores of "Rabbit" and "Rabbit_No Candidate Selection" (which are statistically significant with $p < 0.001$), we have verified that bypassing the candidate selection step would have a negative impact on the overall performance of Rabbit, since recommending books that only match the interests or general traits of books preferred by a reader R, without considering R's readability level, increases the number of non-relevant suggestions.

Rabbit versus BReK12

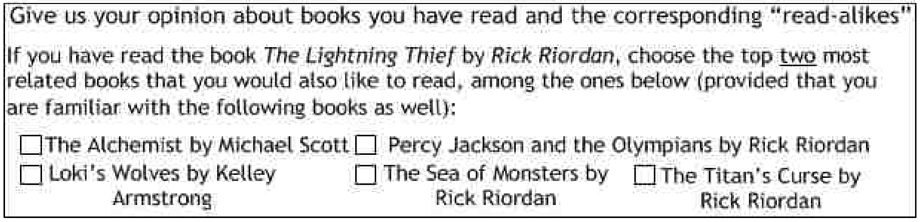
Besides validating the correctness of the recommendation strategy of Rabbit, we compare its performance with BReK12 [122] (as introduced in Section 6.2). We compare Rabbit with BReK12, since to the best of our knowledge BReK12 is the only available recommender that explicitly considers the readability level of its users in making personalized book recommendations. Furthermore, other state-of-the-art approaches for (book) recommendations are excluded for comparison purpose using $Eset$ since (as stated in Section 6.2) they require either *personal ratings* on books provided by individual users or *social connections* established by social bookmarking site users, neither are available on social websites for K-12 readers nor in the $Eset$ dataset.

We compared the performances of Rabbit and BReK12 using nDCG.Rabbit achieves a statistically significant improvement ($p < 0.001$) over BReK12 in terms of nDCG, which are 0.32 and 0.18, respectively.

Rabbit versus Other Recommendation Modules

To further validate the performance of Rabbit, we conducted a survey using Mechanical Turk appraisers on 10 sample books in SB_2, which evaluated the degree to which books recommended by Rabbit are preferred over those suggested by recommendation modules at well-known book-related websites. We have selected recommenders that adopt diverse strategies in making book suggestions: (i) Amazon, which considers purchasing patterns of its users [89], GoodReads,[14] which "combines multiple proprietary algorithms which analyze 20 billion data points to better

[14] goo.gl/99me5f



Give us your opinion about books you have read and the corresponding "read-alikes"

If you have read the book *The Lightning Thief* by *Rick Riordan*, choose the top two most related books that you would also like to read, among the ones below (provided that you are familiar with the following books as well):

□ The Alchemist by Michael Scott □ Percy Jackson and the Olympians by Rick Riordan
□ Loki's Wolves by Kelley Armstrong □ The Sea of Monsters by Rick Riordan □ The Titan's Curse by Rick Riordan

Figure 6.7: A snapshot of the survey conducted on Mechanical Turk for the book "The Lightning Thief"

predict which books people want to read next", and (iii) NoveList,[15] which examines a number of book-related information, including title, publication date, and appeal factors for recommending books.

Each survey (see Figure 6.7 for a sample) included the top-2 recommendations (such that some of them can be identical) made by Rabbit, Amazon, GoodReads, and NoveList for a given sample book Bk, respectively. Appraisers were asked to select, to the best of their knowledge, the top-two books most closely related to Bk, which were treated as the *gold standard* for Bk.

Based on the 500 responses collected during November 2013, we computed the accuracy of the top-2 recommendations made by Rabbit and each of the recommenders considered for comparison purpose. As shown in Figure 6.8, recommendations made by Rabbit and Amazon are preferred over the suggestions made by GoodReads and NoveList. Furthermore, the improvement, in terms of accuracy ratios, achieved by Rabbit over GoodReads and NoveList is statistically significant ($p <$ 0.01). In terms of the overall accuracy, Amazon outperforms Rabbit. However, their differences in nDCG are not statistically significant ($p < 0.01$).

Note that for the comparison purpose, we have evaluated the performances of Rabbit, Amazon, NoveList, and GoodReads in making recommendations in response to a book provided by a user/reader. Rabbit, however, can make suggestions regardless of the number of given books that are of interest to a reader, which Rabbit can apply to identify *patterns* of the reader's preferences. Rabbit's recommendation strategy differs from the strategies employed at Amazon and NoveList, since the latter can only examine books of interest to a user one at the time. The recommendation

[15] support.epnet.com/knowledge_base/detail.php?id=4772

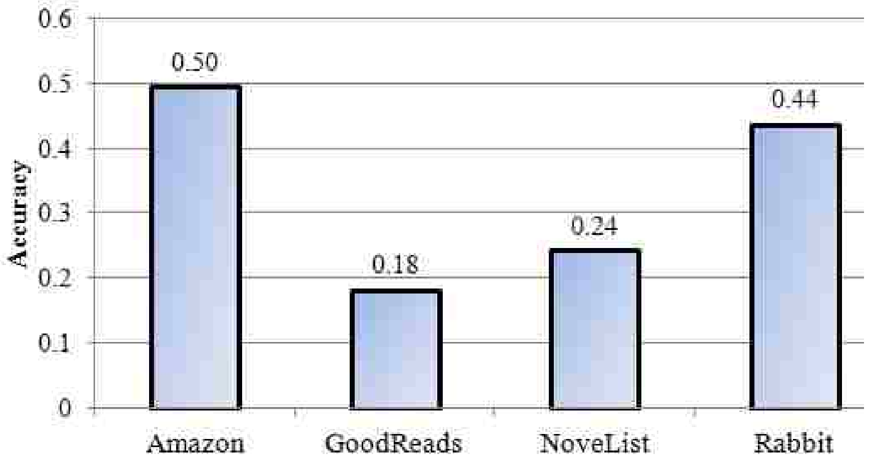


Figure 6.8: Accuracy achieved by Amazon, GoodReads, NoveList, and Rabbit based on the survey conducted using Mechanical Turk

strategies of Rabbit and GoodReads are also different, since GoodReads processes either a given book or the entire profile of a user. Furthermore, Rabbit can treat a book as a candidate suggestion immediately after the book is published, whereas Amazon requires the existence of a number of purchasing transactions involving the new book in order to suggest it to a user. In addition, in making recommendations for children and teenagers, Rabbit considers books provided directly by K-12 readers to generate personalized suggestions. Recommendations generated by Amazon that target children and teenagers, on the other hand, are the result of extensive analysis of the purchasing patterns of adults.

6.7 Conclusions

We have introduced Rabbit, a recommender which makes personalized suggestions on books that match the *interests* and *reading abilities* of its K-12 users. Rabbit emulates the readers' advisory process offered at public/school libraries to recommend books that are similar in *contents*, *topics*, and *literary elements* of other books appealing to a reader, with the latter based on extracted appeal-term descriptions. The generated appeal-term descriptions can be accessed by librarians as well in lieu of subscription-based RA databases. Rabbit can be a stand-alone tool used by readers (educators/parents, respectively) or it can be adopted by (K-12) social bookmarking sites for providing suitable reading selections.

We have developed Rabbit with K-12 readers in mind; however, the readers' advisory-based methodology of Rabbit can also be used to suggest books for adults (with reading levels below/above the 12th grade level) as well.

The results of the conducted experiments have (i) validated the design methodology of Rabbit, (ii) demonstrated the superiority of Rabbit over other recommenders that either explicitly consider or ignore the reading ability of its users using data from BiblioNasium, and (iii) verified that Rabbit performs as well as Amazon in suggesting relevant books, which is an achievement on its own, since Amazon analyzes millions of purchasing patterns made available through its website, as opposed to Rabbit which simply examines publicly accessible data on books.

As part of our future work, we plan to extend Rabbit so that it can make suggestions for K-12 audiences on items other than books, which include movies, music, websites, and other learning materials. Furthermore, Rabbit, which considers the readability levels of its readers, instead of their grade levels or ages, can be enhanced so that it can locate books that can be enjoyed and comprehended by readers with learning disabilities or readers who learn English as a second language.

Chapter 7

Conclusions and Directions for Future Work

In this dissertation, we have designed and assessed the correctness of a number of recommendation strategies which suggest reading materials of interest to readers for pleasure as well as for knowledge acquisition. With the development of the recommenders, we have made the following contributions:

- We have created new algorithms to identify reading materials that match the *specific* needs of a reader whose reading ability is at either *advanced* or *K-12* levels.
- We have taken advantage of connections explicitly established among users of a social bookmarking site to incorporate the concept of "*social trust*" as part of our recommendation strategies (whenever possible), which is based on the premise that readers often turn to people they know when seeking recommendations on products, services, and activities.
- We have developed Top-N recommendation strategies that do not rely on the availability of *large historical data* in the form of ratings, which may not be archived by bookmarking sites (for K-12 audiences) or may refer to various scales that differ in their interpretations from site to site. These developed strategies, which are neither affected by the *cold-start* nor the *long-tail*[1] problem, can make suggestions even in the absence of user-defined data, since they rely on book-related metadata that can be either retrieved or inferred from publicly and freely accessible online data sources.

[1] The long tail problem on recommender systems is the problem of large quantities of items with very few ratings on them.

- We have determined the readability level of a book, even when an excerpt of the book is not available. The readability-analysis tool is a component of our book recommenders for K-12 readers which suggest books of interest to readers by simultaneously matching their *reading abilities* and *preferences*, a task that cannot be accomplished by existing, popular commercial book recommenders.
- We have designed a fully-automated approach which extracts from book reviews appeal-term descriptions of books without relying on either professionals or subscription-based Readers' Advisory (RA) databases.
- We have created an unsupervised recommender based on RA that performs a multi-dimensional analysis on each K-12 reader to determine *what* types of book contents are favored by the reader, *which* topics the reader prefers, and *why* a book appeals to the reader. The RA-based recommender can be used as a stand-alone tool that besides guiding youth/adults in their quest for reading materials, can assist *educators/parents/librarians* in finding appealing books for children/teenagers to read.

The work in this dissertation establishes a solid foundation on making personalized reading-material recommendations for advanced as well as K-12 readers. We would like to consider future extensions of our work as follows. At present, TRoLL considers as many predictors as applicable to a book in estimating its grade level. We plan to examine diverse feature-selection models to identify, among the number of possible predictors applicable to a book, the optimal subset that yields the most accurate grade-level prediction for the book. Furthermore, TRoLL is currently designed solely for books. However, given the importance of identifying suitable reading materials in general for K-12 readers, we would like to further explore other predictors, such as HTML tags of web pages, that would allow us to adequately predict the grade levels of texts in general.

We have also recognized a number of limitations on our recommendation strategies used on a social site. First, a number of the aforementioned strategies examine the entire profile of a reader in making the corresponding suggestions. However, the topical preferences of readers can change over time. For this reason, we plan to further extend our recommendation strategies so that they can explicitly account for changes in the preferences of the readers, i.e., “concept drift” [133], and make recommendations accordingly. This can be accomplished by analysing the timestamps of bookmarks in the respective user profiles and identifying either an ideal window of time that capture the “current” interests of the individual users or changes in topical distributions over time based on items that have been bookmarked by the users, to name a few.

Besides concept drift, our recommendation strategies currently require at least an item in a user’s profile to make suggestions for the user. Compared to existing collaborative-filtering and hybrid-based recommenders which are restricted by the availability of a number of items in a user’s profile to make recommendations, our one-item requirement seems minimal. However, none of our recommendation strategies can make any suggestions to a user who has not bookmarked any items to date. To bypass this limitation, we plan to extend our proposed recommendation strategies so that without a user’s profile, the user U can still receive recommendations likely of interest as long as themes, contents, or literary elements are explicitly provided by U.

Regarding our recommendation strategies for advanced readers, we have shown that our proposed trust-based approaches for making suggestions are effective, which are based on the premise that users tend to favor recommendations made by people they know. These strategies, however, cannot account for the novelty/serendipity and diversity of recommended items [133]. With that in mind, we plan to extend the proposed strategies to include *unexpected* and *dissimilar* items among the recommendations. This can be achieved by considering reading materials that are excluded from the profiles of users’ connections, but are pertinent to the information needs of the user, as possible suggestions. The concept of providing unexpected, but relevant, suggestions also applies to K-12 readers. Consequently, we propose to further examine the performance of our K-12 recommenders using novelty-related metrics published in the literature [133].

As previously stated, the goal of our K-12 recommenders is to locate the “right material” for the “right audience”. In doing so, we aim not only to promote good the reading habits of K-12 readers, but also facilitate and encourage their learning. While we have empirically verified the correctness of our K-12 recommendation strategies, we would like to further extend their evaluations by conducting in-depth user-studies on K-12 readers. Such studies, which would require K-12 readers to directly interact with our K-12 recommenders, would allow us to quantify the degree of influence of BReK12 and Rabbit towards encouraging reading/learning among K-12 readers. Furthermore, the results of the aforementioned studies should highlight aspects of our current design methodologies, if any, that need to be enhanced/incorporated so that they can achieve their ultimate goal, i.e., aid parents/teachers/young readers in locating suitable and appealing reading materials.

Different from a number of book recommenders, such as the one used by Scholastic (scholastic.com/parents/book-search), the recommendation strategies presented in this dissertation and developed with K-12 readers in mind, explicitly consider the reading ability of a reader regardless of the school grade level or age of the reader. These strategies can be further enhanced to suggest reading materials to readers with learning disabilities or people who speak English as a second language by taking into consideration the various challenges and lack of skills to be developed by these special groups of readers as determined by psychologists and educators, respectively.

Regardless of their targeted audience, i.e., advanced or K-12 readers, our current recommendation strategies are applied to individual readers. We would like to extend our current recommendation strategies so that they can serve groups of readers. Such recommenders could assist (i) bookclub members finding reading materials of interests for the members of the club, (ii) research groups in locating academic publications relevant to the research interests of the groups, and (iii) teachers in retrieving reading materials suitable for (various projects to be pursued by) students in their classes. Moreover, we would like to further explore the enhancements necessary to create a single multimedia recommender that suggests learning materials, online videos, movies, (educational) games, web pages, and songs, in addition to books and scholarly articles.

Appendix A

A Group Recommender for Movies Based on Content Similarity and Popularity

Abstract:

People are gregarious by nature, which explains why group activities, from colleagues sharing a meal to friends attending a book club event together, are the social norm. Online group recommenders identify items of interest, such as restaurants, movies, and books, that satisfy the collective needs of a group (rather than the interests of individual group members). With a number of new movies being released every week, online recommenders play a significant role in suggesting movies for family members or groups of friends/people to watch, either at home or at movie theaters. Making group recommendations relevant to the joint interests of a group, however, is not a trivial task due to the diversity in preferences among group members. To address this issue, we introduce $GroupReM$ which makes movie recommendations appealing (to a certain degree) to members of a group by (i) employing a *merging strategy* to explore individual group members' interests in movies and create a profile that reflects the preferences of the group on movies, (ii) using *word-correlation factors* to find movies similar in content, and (iii) considering the *popularity* of movies at a movie website. Unlike existing group recommenders based on collaborative filtering (CF) which consider ratings of movies to perform the recommendation task, GroupReM primarily employs (personal) *tags* for capturing the contents of movies considered for recommendation and group members' interests. The design of GroupReM, which is simple and domain-independent, can easily be extended to make group recommendations on items other than movies. Empirical studies conducted using more than 3,000 groups of different users in the MovieLens dataset, which are various in terms of numbers and preferences in movies, show that GroupReM is highly *effective*

and *efficient* in recommending movies appealing to a group. Experimental results also verify that GroupReM outperforms popular CF-based recommenders in making group recommendations.

A.1 Introduction

During the past decades, a number of recommender systems have been developed to aid individual users in finding items of interest among the millions available, which include songs [142], books [120], and websites [29], to name a few. These recommenders, however, are tailored only to the needs of individual users. As people are gregarious by nature, a variety of activities involve groups of people who participate either online or in an old-fashioned manner, i.e., in person. To meet the demands of groups of users, group recommenders [8, 56, 131] have been proposed to identify items, such as vacation packages [101], restaurants [116], TV shows [98], songs [34, 100], or movies [114, 136], that appeal to a group as a whole (rather than individual users). As claimed by Gartrell et al. [56], effective group recommendations can have a positive impact on people's social activities. Suggesting items that satisfy (to a certain degree) the needs of members of a group, however, is a challenging task due to the diverse interests of group members, even more so when dealing with groups consisting of dissimilar members in terms of their preferences [8, 14].

One of the in-demand recommendation tasks is to suggest movies to a group. Movies offer a popular group activity for friends, families, and colleagues who gather to either see a movie at the cinema or watch a DVD at home. These people often turn to experts' reviews to find movies that match their interests and/or reach a consensus on their own regarding the movies to watch. Group recommenders on movies can streamline this process by directly suggesting movies appealing to a group. While the majority of the recommenders that have recently been introduced to make group recommendations on movies are based on the popular collaborative filtering (CF) strategy [56, 136], to the best of our knowledge, none of them adopts the content-based strategy to exploit descriptive information on movies to perform the recommendation task. In this paper, we introduce GroupReM, a group recommender system on movies. Our proposed top-N recommender, which is based on a content-based strategy to identify a ranked set of N movies that best match the (content of movies of) interest to a group, differs from existing CF-based group recommenders that adopt various strategies for predicting (individual/group) ratings on movies and suggest the movies with the highest overall rating to a group [158]. These recommenders are restricted, since "similar-minded" individuals at a movie website and the existence of large historical data to guarantee rating overlap among users [113] are required to make recommendations. GroupReM, on the other hand, simply relies on data readily available on social websites, which are tags and their frequencies of occurrence, along with bookmarked movies, to suggest movies to a group.

GroupReM considers semantic information of movies, i.e., (personal) tags at a movie website, to capture both the (i) content of movies and (ii) the preferences of members of a given group G on movies archived at the website. GroupReM applies a *rank aggregation* model on two different measures, *group appealing* and *global popularity*, computed for each candidate movie M to be considered for recommendation. The former captures the content similarity between M and the group profile of G, whereas the latter reflects the popularity of M at the movie website. GroupReM anticipates that popular movies, which are frequently bookmarked, that are similar in content (based on tags) to the group profile, which characterizes G, are of interest to the members of G. In matching (the tags in) M and the profile of G, GroupReM does not impose an exact-match constraint. Instead, GroupReM relies on pre-computed word-correlation factors [78] to determine inexact, but analogous, tags in M and G, in addition to exact-matched tags, to more accurately capture the degree of appealing of M to G.

GroupReM is (i) *simple*, since it solely employs a standard measure to combine the aforementioned content-similarity and popularity scores, (ii) *fast*, since it takes on the average less than a second to make recommendations for a group (of up to eight members), and (iii) *scalable*, since GroupReM can identify movies that capture the common interests of a group regardless of its *size* and the *degree of cohesiveness* among group members. Moreover, GroupReM requires neither training nor domain-specific knowledge to select movies to be recommended and thus can directly be adopted to make recommendations on items other than movies. We have conducted an empirical study using more than 3,000 groups of users from the MovieLens dataset [13] and verified that

GroupReM (i) generates relevant recommendations on movies tailored to the needs of a group and (ii) significantly outperforms CF-based group recommenders.

The remaining of this paper is organized as follows. In Section A.2, we discuss existing group recommenders and compare their recommendation strategies with GroupReM. In Section A.3, we detail the design of GroupReM. In Section A.4, we present the empirical study conducted to assess (compare, respectively) the performance of GroupReM (GroupReM with existing CF-based group recommenders, respectively) and illustrate the effectiveness and efficiency of GroupReM. In Section A.5, we give a concluding remark and directions for future work.

A.2 Related Work

In this section, we present a number of existing group recommenders that suggest different types of items, including vacations [101], recipes [20], TV programming [26], and music [43], and compare their recommendation approaches with GroupReM. Thereafter, we introduce representative work on recently-developed group recommenders on movies [56, 136] which differ from GroupReM in their design methodologies. An in-depth discussion on group recommenders can be found in [25, 70].

A.2.1 Non-Movie Group Recommenders

As defined in [20], there are two strategies commonly-adopted for generating group recommendations: the *aggregated models* and *aggregated predictions*. The former combines individual user models, i.e., individual user profiles that capture the preferences of a group member, into a group model from where items to be recommended for the group are identified, whereas the latter generates predictions for individual group members and then aggregates the predictions to suggest items for the group. Empirical studies conducted and presented in [20] suggest that the aggregated models strategy (which is employed by GroupReM) generally outperforms the aggregated predictions strategy.

Flytrap [43], which identifies musical tracks for a group, learns the music preferences of a user U based on the songs U has listened to and the numerical votes casted by U for the songs. Flytrap considers (i) relationships among musical genres, (ii) the influence artists have on one another, and (iii) the transitions in between songs people tend to make to perform the recommendation task. The recommender relies on domain-specific information and thus cannot be extended to suggest items other than songs, contrary to GroupReM which can directly be employed for recommending non-movie items.

CATS (Collaborative Advisory Travel System) [101] assists groups of friends in planning skiing vacations. CATS relies on an incremental method which analyzes individual user's critiques on the proposed recommendations to refine the recommendations generated for the group. Unlike GroupReM, CATS depends on user feedbacks to narrow the search space, i.e., identify items that satisfy the need of an individual, as well as a group, which is a burden on the users. Moreover, CATS has been designed to recommend items to a group of at most four users, which is a limitation, as opposed to GroupReM which does not impose a constraint on the number of group members in performing its recommendation task.

Berkovsky and Freyne [20] recommend recipes to families through an eHealth portal. The proposed CF-based group recommender considers the (i) ratings assigned to recipes on the eHealth portal and (ii) weight, i.e., influence, of each individual group member computed according to his/her activities on the portal in making recommendations. Unlike the recommender introduced in [20], GroupReM relies on the semantic content and popularity of movies to accurately perform the recommendation task.

Cantador and Castells [28] introduce an ontology-based group recommendation strategy. The proposed approach identifies users that share similar tastes/preferences, i.e., "communities of interest", according to individuals' ontology-based profiles. These clusters of related users are then exploited to generate group profiles and perform the recommendation task. Similar to GroupReM, the strategy in [28] is primarily content-based. However, the approach presented in [28] is employed to suggests photos (instead of movies), relies on ontology concepts to determine the similarity among users/items (unlike GroupReM that depends on user-defined keywords, i.e., tags, to capture users' preferences and items' descriptions), and according to the authors "'more sophisticated and statistically significative experiments need to be performed in order to properly evaluate" the correctness of applying the clustering techniques presented in [28] for group modeling and content-based collaborative filtering recommendation.

Masthoff [98] describes a number of recommendations strategies that merge individual user models in order to suggest TV shows that appeal to a group of users. Unlike GroupReM, (some of) the group recommendation strategies discussed in [98] are inspired by Social Choice Theory. Boratto et al. [26] recommend TV programming to a group by first employing a hierarchical clustering algorithm using the cosine similarity metric, which determines the similarity among pairs of users, to identify a natural community G, i.e., a group. Thereafter, a profile for G is created, which reflects the overall preference of the members in G based on the average ratings given by members of G to programs. Based on the group profile, recommendations are generated. Unlike GroupReM which generates recommendations for groups regardless of the cohesiveness among group members, the group recommender in [26] makes recommendations for groups of similar-minded individuals only, which is a restriction, since in real life groups tend to include members that may not share similar interests in various TV programming.

A.2.2 Group Recommenders on Movies

A number of group recommenders that identify movies of interest to a group have been developed in the last few years, which include the systems introduced in [14, 56, 114, 136]. Gartrell et al. [56] claim that some members of a group are more capable than others to influence the remaining group members in making decisions (i.e., relevant or non-relevant) on items suggested to the group. The authors consider several group factors, which include social interactions among group members, degrees of expertise of the members in the group, and dissimilarity among group members, to identify movies of interest to the group. While empirical studies conducted using ten groups have verified the effectiveness of the proposed recommender, it relies heavily on the interaction activities among group members that may not always exist or become available.

Baltrunas et al. [14] conduct an empirical study to assess the effectiveness of alternative rank aggregation strategies, such as Spearman Footrule, Borda Count, Least Misery, and Average, for combining individual ranking predictions using a CF-based algorithm to make group recommendations. Similar to the approach in [14], the group recommender developed by O'Connor et al. [114] adopts a Least Misery strategy to combine individual ratings predicted by a CF-based algorithm. Basu Roy et al. [136] prune and merge rating lists predicted for individual members of a group G using the popular Average and Least Misery aggregation strategies, in addition to considering pairwise disagreement lists of movies, to recommend movies of interest to G. Unlike GroupReM, the group recommenders in [14, 114, 136] are based on the aggregated prediction strategy. According to the research work conducted in [20], this strategy has been empirically determined to be less effective than the aggregated model, which GroupReM adopts.

A.3 Our Proposed Group Recommender

In this section, we present our proposed recommender, GroupReM, which suggests movies appealing (to a certain degree) to members of a group who are users of a movie website, such as Netflix (netflix.com) and MovieLens (movielens.umn.edu). GroupReM relies on *tags* assigned to (represent the content of) movies and the *popularity* of each movie to make recommendations.

As group members of a movie website often have diverse preferences in movies, GroupReM first assesses the interest of each individual member U of a given group G based on the tags assigned by U to movies bookmarked in his/her profile. Tags and their frequencies of occurrence in group members' profiles are combined to create the *group profile* of G which reflects the common interests of the group members (as detailed in Section A.3.1). Thereafter, using word-correlation factors (introduced in Section A.3.2), GroupReM determines the movies, among the ones available at the website which are not included in the profile of any member of G, that are similar (based on tags) to the ones bookmarked by the members of G to a certain degree to gener-

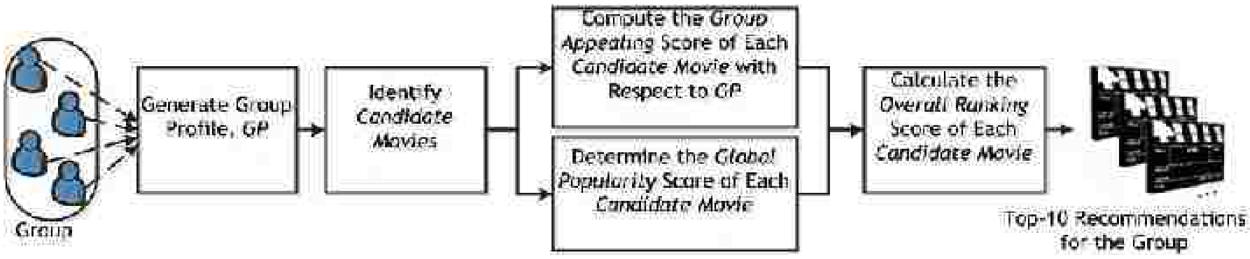

Group
Generate Group Profile, GP
Identify Candidate Movies
Compute the Group Appealing Score of Each Candidate Movie with Respect to GP
Determine the Global Popularity Score of Each Candidate Movie
Calculate the Overall Ranking Score of Each Candidate Movie
Top-10 Recommendations for the Group

Figure A.1: Processing steps of the proposed group recommender on movies, GroupReM

ate the set of *candidate movies* that the group is likely interested in (as described in Section A.3.3). Using a *rank aggregation function* (as presented in Section A.3.4), GroupReM computes the overall ranking score of each candidate movie M. The *ranking score* of M is based on (i) the *group appealing* score of M for G (as defined in Section A.3.4) and (ii) the *popularity* score of M (as computed in Section A.3.4). The former is calculated according to the number of tags assigned to (represent the content of) M that exactly-match or are analogous to the ones which characterize the profile of G, whereas the latter reflects the overall interest of the website users on M. The top-10 ranked candidate movies are recommended to G. The overall process of GroupReM is illustrated in Figure A.1.

A.3.1 Creating a Group Profile

As the goal of group recommenders is to suggest movies of interest to a group, GroupReM analyzes the preference of each group member in movies and creates a *group profile* which reflects the types of movies preferred (to a certain degree) by the group as a whole. To construct the profile, GroupReM employs an *aggregated model* [20] that merges *individual user models*, i.e., the movies each group member is interested in, into a *group model*, which indicates the collective interest of the group members in movies.

GroupReM identifies the preference in movies of each individual member U of a group G by considering the movies bookmarked by U and tags assigned by U to the movies. Personal tags, i.e., tags defined by an individual user, are employed to represent (the content of) a movie M of interest to a user, as opposed to tags in the tag cloud[1] of M at a movie website, since GroupReM aims to capture U's description of M. Hereafter, GroupReM proceeds to create the *group profile* for G which includes all the personal tags (and their combined frequencies) assigned by the members of G to movies in their individual profiles. The *higher* the frequency of a tag T in G is, the more *adequately* T is in reflecting the *joint interest* of the group members on movies, since the high frequency of T reflects that T is more often used by members of G to describe movies they are interested in than other tags with lower frequencies.

Example 19 Consider a group G with three different MovieLens members. Figure A.2 shows (a portion of) the profile of each member U in G, which includes the personal tags assigned by U to movies bookmarked in his/her profile. By combining the tags (and cumulating the corresponding frequencies) in the individual group member profiles, GroupReM creates a group profile for G. As shown in Figure A.2, tags such as "drama" and "animation" reflect the types of movies that are of interest to each member of G, since the tags are included in the personal profile of each group member, as opposed to tags such as "mermaid" and "war", which are preferred by one out of three group members. □

A.3.2 Word-Correlation Factors

GroupReM relies on the pre-computed word-correlation factors in the word-correlation matrix [78] to determine the *similarity* between any two tags, which facilitates the task of identifying *candidate movies* to be considered for recommendation (as detailed in Section A.3.3). Moreover, GroupReM takes advantage of the word-correlation factors in calculating the *group appealing* score of a candidate movie with respect to a group profile (as discussed in Section A.3.4).

Word-correlation factors were calculated using a set of approximately 880,000 Wikipedia documents (wikipedia.org). Each correlation factor indicates the degree of similarity of the two cor-

[1] The tag cloud of a movie M, which provides the collective description on the content of M bookmarked by users at a movie website W, can be inferred by collecting each tag assigned to M by users at W, in addition to their $frequencies$.

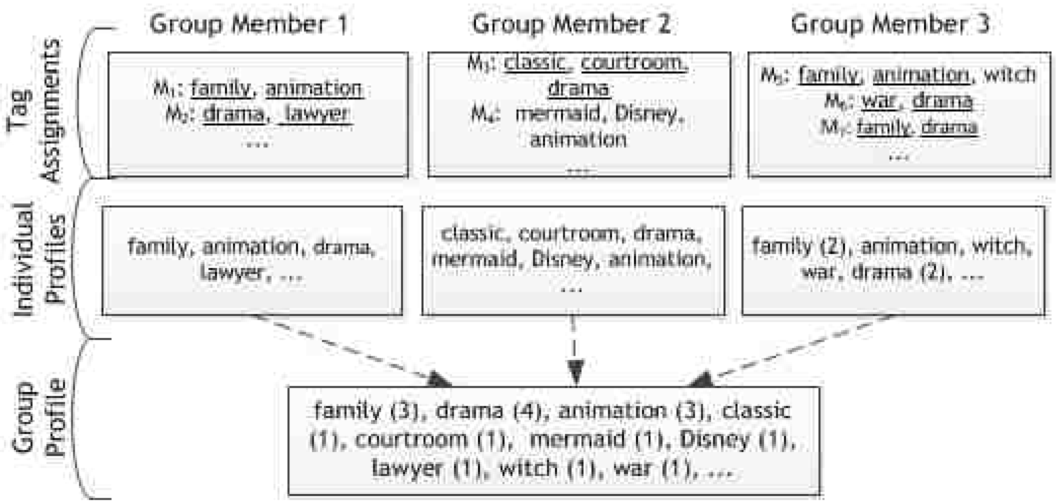

Group Member 1
Group Member 2
Group Member 3
Tag Assignments
M1: family, animation
M2: drama, lawyer
...
M3: classic, courtroom, drama
M4: mermaid, Disney, animation
...
M5: family, animation, witch
M6: war, drama
M7: family, drama
...
Individual Profiles
family, animation, drama, lawyer, ...
classic, courtroom, drama, mermaid, Disney, animation, ...
family (2), animation, witch, war, drama (2), ...
Group Profile
family (3), drama (4), animation (3), classic (1), courtroom (1), mermaid (1), Disney (1), lawyer (1), witch (1), war (1), ...

Figure A.2: The group profile created by GroupReM for a given group G based on the tags in the profiles of three MovieLens users, who are members of G

responding words[2] 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 with different writing styles, and the documents cover a wide range of topics with diverse word usage and contents. Compared with synonyms/related words compiled by the well-known 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. Despite the existence of a number of measures that rely on WordNet to determine the semantic similarity between pairs of words, such as Lesk [15] and LCH [83], GroupReM depends on word-correlations, which have been successfully adopted to determine the similarity between words in a number of applications, such as document classification [119] and text retrieval [130].

[2] Words in the Wikipedia documents were *stemmed* (i.e., reduced to their grammatical roots) after all the *stop-words*, 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 non-stop, stemmed (key)words/tags.

A.3.3 Identifying Candidate Movies to be Recommended

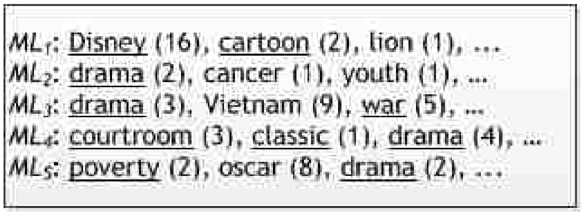
As the number of movies available at a movie website W can be large, i.e., in the hundreds of thousands, it is inefficient to analyze each movie of W to identify those of interest to the members of a group G at W, since the comparisons would significantly prolong the processing time of GroupReM to make recommendations. To minimize the number of comparisons and thus reduce the processing time required in generating recommendations for G, GroupReM applies a *blocking strategy*[3] on movies archived at W to obtain the subset of movies that are potentially of interest to the members of G (to various degrees), denoted *Candidate_Movies*, to be considered for recommendation.

The blocking strategy adopted by GroupReM first considers the personal tags assigned by a group member U of G for each of his/her bookmarked movies, uM. A movie M archived at W [4] is included in *Candidate_Movies* if *each* of the personal tags assigned by U to uM exactly matches or is highly similar to at least a tag in the tag cloud of M. As tags are concise and valid content descriptors of an item [64], it is anticipated that movies in *Candidate_Movies* are of interest to (at least one of the members of) G, since each movie in *Candidate_Movies* shares a number of same (or analogous) tags (to a certain degree) with the ones in the group profile for G.

To select movies to be included in *Candidate_Movies*, GroupReM relies on a reduced version of the word-correlation matrix (introduced in Section A.3.2) which contains 13% of the most frequently-occurred 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 [63]. By adopting a reduced version of the word-correlation matrix to determine potentially similar movies, the overall processing time of GroupReM is significantly reduced without affecting its accuracy [126].

[3] A blocking strategy is a filtering technique that reduces the potentially very large number of comparisons to be made among records [33], i.e., movies available at a movie website in our case.

[4] A movie archived at a movie website is included in the set of candidate movies if it has not been bookmarked by any member of a given group.



ML_1: Disney (16), cartoon (2), lion (1), ...
ML_2: drama (2), cancer (1), youth (1), ...
ML_3: drama (3), Vietnam (9), war (5), ...
ML_4: courtroom (3), classic (1), drama (4), ...
ML_5: poverty (2), oscar (8), drama (2), ...

Figure A.3: (Portions of the) Tag clouds of potential candidate movies in which tags exactly-matched or highly similar to the personal tags assigned to a movie shown in Figure A.2 are underlined

Example 20 Consider the five movies archived at MovieLens, i.e., ML_1, ML_2, ML_3, ML_4, and ML_5, as shown in Figure A.3, which are not bookmarked by any member of the group shown in Figure A.2. To determine which one of the five movies should be treated as candidate movies for the group G introduced in Example 19, GroupReM compares personal tags assigned to each movie shown in Figure A.2 with the tags (in the tag cloud) of each movie shown in Figure A.3. Given that each of the personal tags assigned to M_1 is highly similar to at least a tag in the tag cloud of ML_1, i.e., the word-correlation factors of “family” and “Disney” (“animation” and “cartoon”, respectively) can be found in the reduced version of the word-correlation matrix, ML_1 is selected as a candidate movie. Furthermore, each of the personal tags assigned to describe M_3 (M_6, respectively) exactly matches its counterpart in ML_4 (ML_3, respectively). Therefore, ML_4 (ML_3, respectively) is a candidate movie. In addition, since the “drama” tag in M_7 exactly matches its counterpart in (the tag cloud of) ML_5 and the remaining personal tag of M_7, i.e., “family”, is highly similar to another tag in (the tag cloud of) ML_5, i.e., “poverty”, ML_5 is also selected as a candidate movie. Although M_2, M_3, M_6, and M_7 include a tag, i.e., “drama”, which is also a tag in the tag cloud of ML_2, none of the remaining personal tags assigned to describe the content of either M_2, M_3, M_6, or M_7 exactly matches or is similar to a tag in ML_2, and thus ML_2 is not treated as a candidate movie. □

A.3.4 Generate Group Recommendations

Having identified the set of candidate movies to be considered for recommendation to a group, GroupReM proceeds to rank each of the candidate movies by relying on two different scores, the *group appealing* and *popularity* scores, presented in Sections A.3.4 and A.3.4, respectively. The two scores are combined using an *aggregation function*, as defined in Section A.3.4, and the top-10 candidate movies with the highest combined scores are recommended to the group.

Appealing Scores of Movies

To determine the degrees of interests of members in a group G on a candidate movie M, GroupReM computes the *group appealing* score of M for G, denoted *GrpApp*(M, G), by accumulating the word correlation factors among the tags that capture the types of movies members of G are interested in, i.e., tags in the group profile of G, and tags in the tag cloud of M. In computing the *GrpApp* score of M for G, GroupReM relies on the word-correlation matrix introduced in Section A.3.2, instead of the reduced word-correlation matrix employed in Section A.3.3, since the former provides a more accurate similarity measure between (tags representing) M and G than the reduced matrix. The *GrpApp* score of M for G is defined as

$$GrpApp(M, G) = \sum_{g \in GP} \sum_{m \in M} wcf(g, m) \times \frac{freq_g}{Max(freq_{GP})} \times \frac{freq_m}{Max(freq_M)} \tag{A.1}$$

where g (m, respectively) is a tag in the group profile GP of G (the tag cloud of M, respectively), $wcf(g, m)$ is the word-correlation factor of g and m in the word-correlation matrix , $freq_g$ ($freq_m$, respectively) is the frequency of occurrence of tag g (m, respectively) in GP (the tag cloud of M, respectively), $Max(freq_{GP})$ ($Max(freq_M)$, respectively) is the highest frequency of any tag in GP (the tag cloud of M, respectively), and $\frac{freq_g}{Max(freq_{GP})}$ ($\frac{freq_m}{Max(freq_M)}$, respectively) denotes the *weight* of g (m, respectively).

$Freq_g$ ($freq_m$, respectively) in Equation A.1 is an indicator of the relative *degree of significance* of tag g (m, respectively) in representing (the content of) GP (M, respectively), since it reflects the frequency in which group members (number of users at a movie website, respectively) have chosen g (m, respectively) to represent movies of interest for G (the content of M, respectively). The *larger* $freq_g$ ($freq_m$, respectively) is, the more *significant* g (m, respectively) is in characterizing GP (describing M, respectively). In addition, by relying on the *weight* of each tag in GP (the tag cloud of M, respectively) in Equation A.1, GroupReM ensures that exactly-matched (or highly-similar) tags between GP and M do not inflate the group appealing score of M if they are not significant/representative tags to G (M, respectively).

Example 21 To illustrate the merit of using word-correlation factors in computing the *GrpApp* score of a candidate movie, consider the group profile shown in Figure A.2 and the candidate movies ML_1 and ML_5 as shown in Figure A.3. Both movies include a tag, i.e., "Disney" and "drama", respectively in their corresponding tag clouds that exactly matches its counterpart in the group profile of G (as shown in Figure A.2), which implies that the *GrpApp* score of ML_1 and ML_5 should be similar. Taking into account the remaining, i.e., non-exact-matched but analogous, tags in the tag clouds of the aforementioned movies in calculating their respective *GrpApp* scores, GroupReM computes a more accurate group appealing score for each candidate movie. *GrpApp*(ML_1, G), computed using Equation A.1, is 3.5, whereas *GrpApp*(ML_5, G) is 1.0, which correctly reflects that G, as a whole, is more interested in *family*, *animated* movies than *dramatic* movies, as captured in the group profile of G. □

Popularity Scores of Movies

In addition to computing the *GrpApp* score of a candidate movie M for G, GroupReM also considers the *global popularity* score of M, denoted *GlbPop*(M), which exploits the "wisdom of the crowd" [21], i.e., the collective interest in M expressed by users at the movie website of which members of G are users, and provides a higher ranking on M if it is *more frequently* bookmarked at the website than other candidate movies.

Popular movies which attract the attention of users at a movie website are more likely to be bookmarked by the users. GroupReM weights the fact that frequently-bookmarked movies may also be of interest to members of G. While solely relying on the popularity of an item in performing the recommendations task (which does not apply to GroupReM) can lead to less diverse and useless recommendations [160], Adomavicius and Kwon [3] claim that the accuracy of the recommendations can be enhanced by considering the popularity of an item during the recommendation process.

GlbPop, which is considered by GroupReM as an additional decision factor besides *GrpApp* to rank M to make recommendations, is computed as the *total number* of users at W who have *bookmarked* M.

Rank Aggregation

Having determined the group appealing and global popularity scores of each movie M in *Candidate_Movies*, GroupReM computes the *ranking score* of M by applying a popular linear combination measure, called $CombMNZ$ [86], which is frequently used in fusion experiments [39]. CombMNZ considers multiple existing lists of rankings on an item I to determine a joint ranking of I, a task known as rank aggregation or data fusion.

$$CombMNZ_I = \sum_{c=1}^{N} I^c \times |I^c > 0| \tag{A.2}$$

where N is the number of ranked lists to be fused, i.e., the number of input ranked lists, I^c is the normalized score of I in the ranked list c, and $|I^c > 0|$ is the number of non-zero, normalized scores of I in the lists to be fused.

Prior to computing the ranking score of M, it is necessary to transform the original scores in each individual ranked list into a *common range*, which can be accomplished by applying Equation A.3 to each score in each ranked list so that it is within the range [0, 1], a common range [86].

$$I^c = \frac{S^I - I^c_{min}}{I^c_{max} - I^c_{min}} \tag{A.3}$$

where S^I is the score of item I in the ranked list c prior to be normalized, I^c_{max} (I^c_{min}, respectively) is the maximum (minimum, respectively) score available in c, and I^c is the normalized score for I in c.

GroupReM normalizes the group appealing and global popularity scores of M computed in Sections A.3.4 and A.3.4, respectively using Equation A.3. Thereafter, using CombMNZ, GroupReM (i) sets N = 2 (in Equation A.2), which is the number of (input) ranked lists of normalized scores with the original ones computed in Sections A.3.4 and A.3.4, respectively, (ii) determines the $overall\ ranking$ score of each movie M in *Candidate_Movies* using Equation A.2, and (iii) recommends the top-10 ranked movies to (the members of) G.

By adopting this fusion strategy, GroupReM considers the strength of each evidence, i.e., the *GrpApp* and *GlbPop* scores, as opposed to simply positioning higher in the ranking movies with a high *GrpApp* or *GlbPop* score.

Example 22 Consider the candidate movies ML_1, ML_3, ML_4, and ML_5 as shown in Figure A.3, along with their respective (normalized) group appealing and global popularity scores as shown in Table A.1. Using CombMNZ as a rank aggregation measure, GroupReM identifies the most relevant movies, i.e., movies of interest, for G. Even though the (normalized) global popularity score of ML_1 is slightly lower than the global popularity score of ML_3, GroupReM positions ML_1 higher than ML_3 in the ranking of movies to be recommended. This is because ML_1 is more appealing for (members of) G based on the tags in the tag cloud of ML_1 and the tags in the group profile of G that depict the movie preferences of (the members of) G.

As shown in Table A.1, the global popularity score of ML_5 is relatively high; however, its group appealing score is significantly lower in comparison with the group appealing scores of the remaining candidate movies. As a result, GroupReM positions ML_5 lower in the ranking of movies to be recommended than the remaining candidate movies in Figure A.3. □

Candidate Movie	Group Appealing Score	Global Popularity Score	Ranking
ML_1	0.92	0.65	**3.14**
ML_3	0.71	0.80	3.02
ML_4	0.53	0.56	2.18
ML_5	0.27	0.75	2.04

Table A.1: Normalized scores for the candidate movies shown in Figure A.3 with respect to the group profile of G shown in Figure A.2 as computed by GroupReM

A.4 Experimental Results

In this section, we first introduce the dataset (in Section A.4.1) employed for assessing the performance of GroupReM. Thereafter, we present the evaluation protocol and group formation strategy adopted for creating the groups used for the evaluation purpose (in Sections A.4.2 and A.4.3, respectively). We define the metric which quantifies the accuracy and ranking approach of GroupReM (in Section A.4.4). We detail the empirical study conducted for verifying the effectiveness and efficiency of GroupReM and compare its performance with existing group recommenders on movies (in Section A.4.5).

A.4.1 Dataset

To evaluate GroupReM in recommending movies appealing (to a certain degree) to the members of a group, we consider the MovieLens dataset [13], a dataset released by the ACM HetRec Conference in 2011. Statistical information on MovieLens is shown in Table A.2. (See detailed information on the dataset at grouplens.org/system/files/hetrec2011-movielens-readme.txt.) Note that the MovieLens dataset was not developed for assessing the performance of group recommenders, since pre-defined groups of users are not provided in the dataset. For this reason, we create our own groups of users for the evaluation purpose (see details in Section A.4.3).

MovieLens Dataset	
# of Distinct Users	2,113
# of Distinct Movies	10,197
# of Distinct Tags	13,222
# of Distinct Tag-Movie Assignments	47,957
Average # of Movies Bookmarked per User	13
Average # of Distinct Tags Assigned to Movies per User	23
Average # of Distinct Tags Assigned to a Movie	8
Average # of Ratings Assigned to Movies per User	405
Average # of Ratings Assigned to a Movie	85

Table A.2: Statistical information of the MovieLens dataset

A.4.2 Evaluation Protocol

To assess the relevancy of group recommendations suggested by GroupReM, we have adapted a standard approach to partition the movies bookmarked by each user in the MovieLens dataset into two subsets and employed the five-fold cross validation approach [95]. In evaluating the recommendations made by GroupReM for a given group G, in each of the five repetitions, 80% of the movies bookmarked in MovieLens by each member U of G were treated by GroupReM as included in the individual profile of U and the remaining 20% were reserved for the testing purpose, i.e., to assess the relevance of the recommendations generated for (U in) G. A recommendation made by GroupReM is treated as *relevant* for (U in) G, if the recommended movie is included in the 20% of the movies (bookmarked by U) withheld for the testing purpose, a commonly-employed protocol for assessing recommendation systems [18, 61].

A.4.3 Group Formation

To the best of our knowledge, there are no benchmark datasets available for assessing the performance of group recommenders, needless to say group recommenders on movies. For this reason, we employ a popular strategy for generating groups (of users in the MovieLens dataset introduced in Section A.4.1) for evaluation purpose.

In creating groups for evaluating the recommendations generated by GroupReM, we consider two important factors: the $size$ and $cohesiveness$ of a group [8, 14]. By varying the *group sizes*, we can assess the difficulty in reaching *consensus* among members of small versus large groups. We consider groups with 2 to 8 members, which are comparable to the group sizes defined in [8, 14], to demonstrate the effectiveness of GroupReM in recommending movies for small, as well as large, groups.

Besides group size, *group cohesiveness* is another important criterion [8] in evaluating group recommenders. By using groups that include members with various degrees of cohesiveness, i.e., different degrees of user-to-user similarity, we can verify the correctness of GroupReM in generating recommendations for groups of users that may or may not share common preferences in movies, since the latter is more challenging than the former in terms of satisfying their mutual interest. Altogether, three different types of groups, i.e., *highly similar*, *dissimilar*, and *random*, are considered. *Random groups* are formed by randomly selecting users from MovieLens, regardless of their preferences on movies. *Highly-similar* groups include members with common interests in the same types of movies, whereas *dissimilar groups* reflect groups of people that are different in terms of their preferences in movies. To determine the users who should be included in highly-similar and dissimilar groups, we adapted the strategy employed in [14], which calculates the user-to-user similarity, denoted *User_Sim*, on each pair of users in MovieLens. The *User_Sim* metric is introduced in [8] and computed as

$$User\_Sim(u, u') = \frac{|\{i \mid i \in I_u \bigwedge i \in I_{u'} \bigwedge |rating(u,i) - rating(u',i)| \geq 2\}|}{|\{i \mid i \in I_u \bigvee i \in I_{u'}\}|} \tag{A.4}$$

where I_u ($I_{u'}$, respectively) denotes the set of items, i.e., movies in our case, rated by user u (u', respectively), i is an item, $rating(u, i)$ ($rating(u', i)$, respectively) denotes the rating assigned to i by u (u', respectively), and $|rating(u, i) - rating(u', i)| \geq 2$ constraints a movie M to be treated as “shared” between u and u’ if they both rated M within 2 units of each other on the scale of 0 to 5, which indicates that u and u' assigned a similarly high (low, respectively) rating to M. The high (low, respectively) similar ratings provided by u and u' on i indicates that u and u' share the same preference on i.

In computing the $User\_Sim$ score between any two users (as defined in Equation A.4), only pairs of users who have rated at least 5 common items are considered, a common practice among CF-based recommenders which ensures that the correlation between two users, i.e., $User\_Sim$ score, is not high (low, respectively) solely based on the same ratings assigned to a small set of items, i.e., less than 5 movies in our case, by the two users [14].

We follow the strategy proposed by the authors in [14], who consider the distribution of user pairs in a given dataset (based on their user-to-user similarity) and treat the 33% of user-pairs with the highest user-to-user similarity score as *highly-similar* users. Based on the distribution of user-to-user similarity scores for each pair of users in MovieLens (as shown in Figure A.4), we observe that pairs of users with a 0.11 $User\_Sim$ score or higher fall within the range of 33% user-pairs who achieve the highest user-to-user similarity. Hence, a group of MovieLens users whose user-to-user similarity among each other is higher or equal to 0.11 is treated as a *highly-similar* group. Applying the same strategy to determine highly-similar users, we treat the 33% of users-pairs with the lowest $User\_Sim$ scores as dissimilar users. As it turns out, user-pairs with a $User\_Sim$ score less than or equal to 0.06 constitute the 33% of user-pairs in MovieLens with the lowest user-to-user similarity (as shown in Figure A.4), and these users are treated as members of *dissimilar* groups.

Based on the group formation protocol defined above, we created 3,150 distinct groups, which are uniformly distributed among highly-similar, dissimilar, and random groups. In addition, each set of the 1,050 groups that share the same degree of cohesiveness is uniformly distributed based on the pre-defined group sizes, i.e., 2 to 8 members. Thus, for each distinct group size there are 150 groups in which group members share the same (pre-determined) degree of cohesiveness.

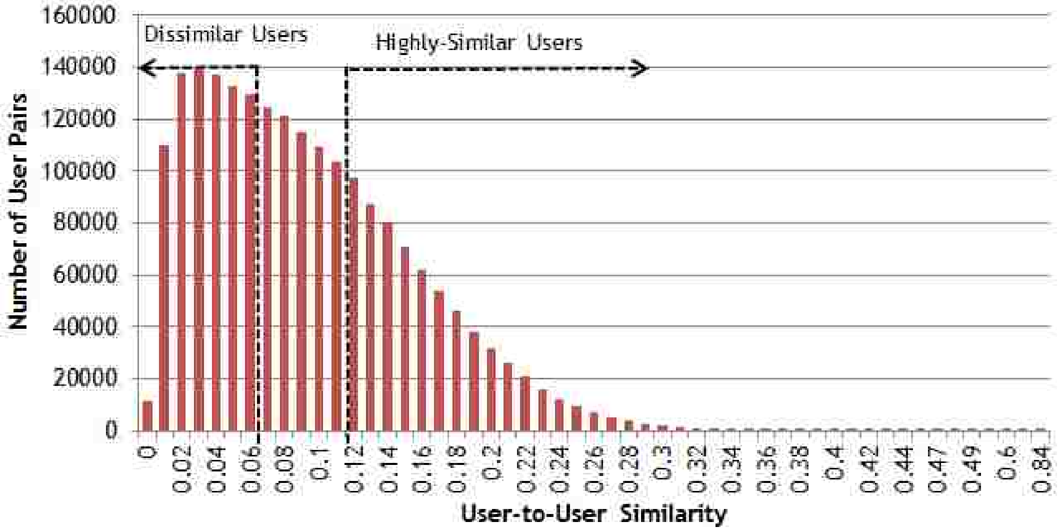


Figure A.4: User-to-user similarity distribution in the MovieLens dataset

A.4.4 Metrics

To assess the overall performance and ranking strategy of GroupReM, we employ the Normalized Discounted Cumulative Gain ($nDCG$) [42] measure, which is a standard IR metric often used for evaluating group recommenders [8, 14]. Due to the lack of "ground truth" required to assess the recommendations generated by GroupReM for a given group G of a particular size that includes members (without) sharing the same degree of cohesiveness, we calculate the $nDCG$ for G as the *average* of the $nDCG$ value computed for each of the group members in G, following the experimental setting adopted by Amer-Yahia et al. [8].

$nDCG_{10}$, as defined in Equation A.5 for evaluating the relevance of each batch of *top-10* recommendations generated by GroupReM, *penalizes* relevant movies ranked *lower*. The penalization is based on a relevance reduction, which is logarithmically proportional to the relative position of each relevant movie in a ranked list of recommended movies (as shown in Equation A.6). The *higher* the $nDCG_{10}$ score is, the *better* the ranking strategy adopted by the corresponding recommender system RS is, since a high $nDCG_{10}$ score on a list of recommendations L indicates that relevant recommendations generated by RS are positioned high in L.

$$nDCG_{10} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{M} \sum_{k=1}^{M} \frac{DCG_{10,k}}{IDCG_{10,k}} \tag{A.5}$$

where N (which is 150 in our case) is the number of groups with a pre-defined number of group members such that the members share the same pre-determined degree of cohesiveness (as detailed in Section A.4.3), i is the i^{th} group for which GroupReM generates movie recommendations, M is the number of group members in i, k is the k^{th} group member in i, $IDCG_{10,k}$ (in Equation A.5) is the best possible $DCG_{10,k}$ value for the recommendations generated by GroupReM for k,[5] and

$$DCG_{10,k} = \sum_{j=1}^{10} \frac{(2^{rel_j} - 1)}{log_2(1 + j)} \tag{A.6}$$

where rel_j is the binary relevant judgment of the recommended movie at the j^{th} ranking position and is assigned a value of "1" if the movie is a *relevant* recommendation for k (as defined in Section A.4.2) and is assigned a "0", otherwise.

A.4.5 The Effectiveness and Efficiency of GroupReM

In this section, we first verify the correctness of relying on *word-correlation factors* and the *popularity* of movies to generate group recommendations (as presented in Section A.4.5). Thereafter, we compare the *performance* of GroupReM with existing CF-based group recommenders (in Section A.4.5) and assess the *efficiency* of GroupReM and CF-based group recommenders in performing the recommendation task (in Section A.4.5).

The Correctness of GroupReM

As stated in Section A.3.4, GroupReM depends on the *group appealing* (based on word-correlation factors) and *global popularity* scores to generate recommendations of interest to a group. To verify

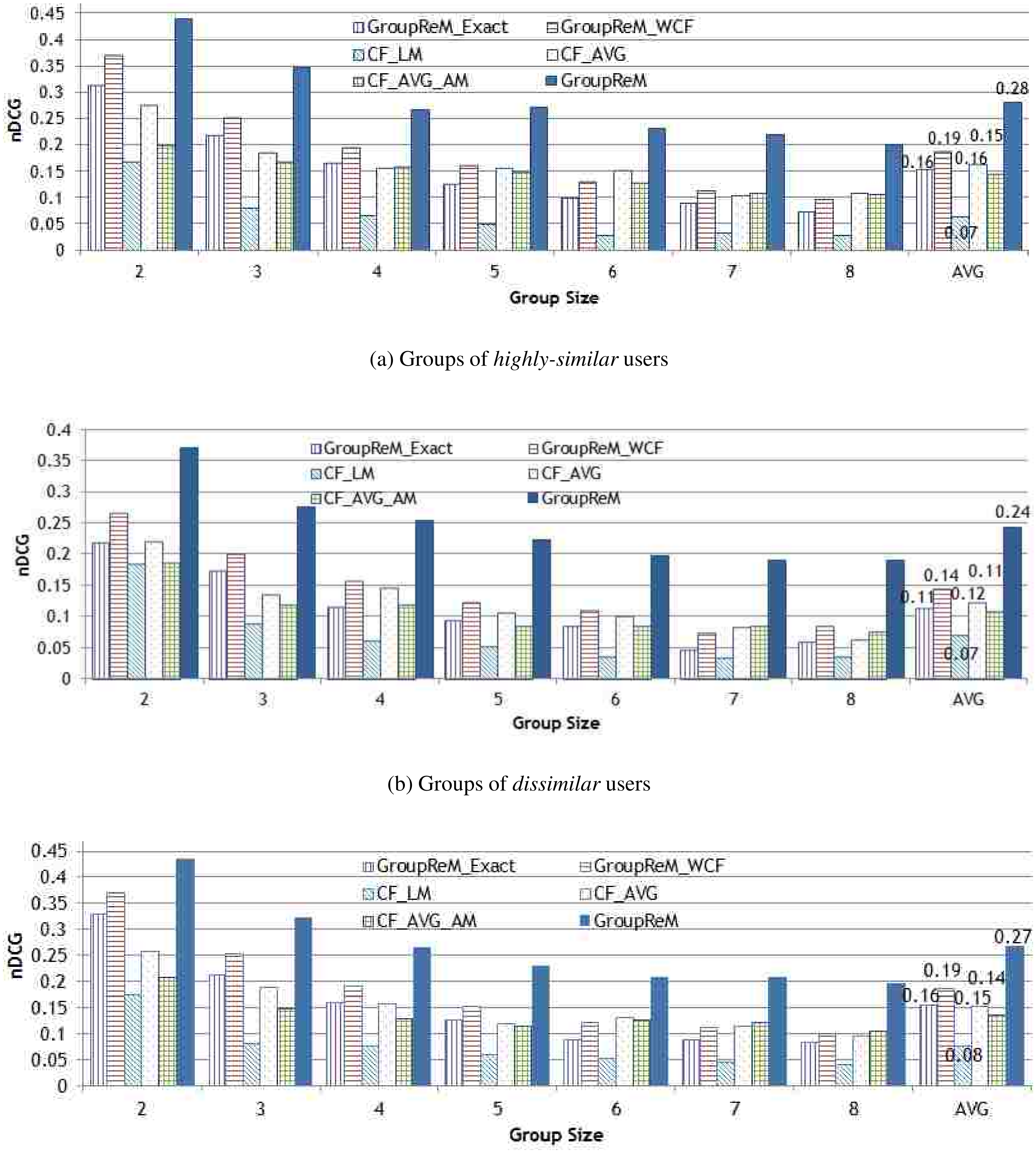
[5] $IDCG_{10,k}$ is computed as $DCG_{10,k}$ using an *ideal* ranking such that the ten recommendations are arranged in descending order based on their relevant judgment scores in the ranked list.

the effectiveness of GroupReM in making group recommendations on movies, we conducted an empirical study in which we compared two alternative implementations of GroupReM. The first alternative, denoted *GroupReM_Exact*, relies solely on the group appealing score computed on exactly-matched tags for generating movie recommendations for a group G. In this case, the group appealing score of a candidate movie M for G is calculated using the $Dice$ coefficient [42] on the tags in (the tag cloud of) M and the tags characterizing (the group profile of) G. The second alternative, denoted *GroupReM_WCF*, relies on the word-correlation factors and considers analogous, besides exactly-matching, tags. GroupReM_WCF computes the group appealing score of each candidate movie using Equation A.1.

As illustrated in Figure A.5, regardless of the degree of cohesiveness among group members in groups of any size, GroupReM_WCF consistently improves the accuracy of the recommendations generated by GroupReM_Exact. The 3% overall improvement on the (average) $nDCG$ achieved by GroupReM_WCF over GroupReM_Exact, using the MovieLens dataset and the groups introduced in Section A.4.3, indicates that relaxing the exact-matching constraint by adopting word-correlation factors enhances the accuracy of movies recommended to a group by GroupReM_WCF. In addition, at least 8% overall improvement on the (average) $nDCG$ scores achieved by GroupReM over GroupReM_WCF, using the aforementioned dataset, validates the fact that the global popularity score (as defined in Section A.3.4) further increases the accuracy of group recommendations than simply using the group appealing scores of movies to perform the group recommendation task (as illustrated in Figures A.5(a)-A.5(c)). Note that the differences between GroupReM_WCF and GroupReM_Exact with respect to GroupReM, in terms of $nDCG$, are statistically significant, as determined using a Wilcoxon Rank Sum Test ($p <$ 0.05).

Comparing the Performance of GroupReM with Existing Group Recommenders

To further verify and demonstrate the *effectiveness* of GroupReM, we compare its performance with two well-known CF recommenders on movies, which are based on $Average$ (CF_AVG) and $Least\ Misery$ (CF_LM) aggregation strategies [8, 14], respectively. Given that GroupReM adopts



(a) Groups of *highly-similar* users

(b) Groups of *dissimilar* users

(c) Groups of *random* users

Figure A.5: $nDCG$ scores computed for (the alternative implementations of) GroupReM and alternative implementations of the collaborative filtering approach based on $average$ and $least\ misery\ aggregation$ strategies on 3,150 groups of various sizes. All the differences in $nDCG$ are statistically significant with respect to GroupReM (Wilcoxon, $p < 0.05$).

an aggregated model approach to make recommendations, we also compare its performance with a CF recommender that employs an $average$ aggregated model strategy (CF_AVG_AM). We have chosen CF-based recommenders for comparisons, since to the best of our knowledge there is no group recommender *on movies* that depends primarily on content descriptions to make recommendations.

Given a group G, both CF_AVG and CF_LM first generate movie recommendations for individual members of G by employing the well-known CF strategy. Thereafter, the recommenders proceed to merge the recommendations generated for individual group members to create the list of movies to be recommended to G. While CF_AVG computes the score of a movie M for G by *averaging* the ratings of M predicted for each individual group member in G, CF_LM defines the score of M for G as the smallest predicted rating of M among all the rating predictions of M determined for each of the individual members of G. The top-10 movies with the highest ratings are recommended to G. (A more in-depth discussion on CF_AVG and CF_LM can be found in [8, 14].) The CF_AVG_AM approach, on the other hand, generates a single group profile by averaging the ratings of each movie bookmarked by each individual member of G. Thereafter, the well-known CF approach is employed to generate a list of the top-10 highest ranked movies for (the profile of) G.

Prior to comparing the performance of the aforementioned recommenders with GroupReM, we have determined the relevance of each movie recommended by CF_AVG, CF_LM, and CF_AVG_AM for each of the groups constructed in Section A.4.3 using the MovieLens dataset, evaluation protocol, and metric detailed in Sections A.4.1, A.4.2, and A.4.4, respectively.

Figures A.5(a), A.5(b), and A.5(c) show the $nDCG$ scores achieved by GroupReM, CF_LM, CF_AVG, and CF_AVG_AM for highly-similar, dissimilar, and random groups of different sizes, respectively. The average $nDCG$ score of GroupReM computed for groups with *highly-similar* users is 0.28, which is at least 12% higher than the average $nDCG$ scores achieved by either CF_LM, CF_AVG, or CF_AVG_AM, which are 0.07, 0.16, and 0.14, respectively. The average $nDCG$ score achieved by GroupReM for groups with *dissimilar* (*random*, respectively) users is 0.24 (0.27, respectively), which also outperforms the average $nDCG$ scores achieved by CF_LM, CF_AVG, and CF_AVG_AM on the same groups, which are 0.07, 0.12, and 0.11 (0.08, 0.15, and 0.14, respectively). All of these $nDCG$ values achieved by GroupReM are statistically significant over CF_LM, CF_AVG, and CF_AVG_AM (as verified using a Wilcoxon Rank Sum Test for $p < 0.05$).

A higher $nDCG$ value indicates that GroupReM is more effective than CF_LM, CF_AVG, and CF_AVG_AM in detecting and ranking higher in the list of recommended movies the ones that are relevant, i.e., of interest, to a group, regardless of the number of members in the group or the similarity among group members in terms of their preferences in movies.

Observations

Since only movies reserved for the testing purpose (as detailed in Section A.4.2) are considered relevant, it is not possible to account for the potentially relevant movies that the users have not bookmarked. As a result, the $nDCG$ scores in our empirical study are underestimated, which is a well-known limitation of the evaluation protocol (introduced in Section A.4.2) applied to recommender systems [66]. As this limitation affects all the evaluated recommenders, i.e., (alternative implementations of) GroupReM, CF_AVG, CF_LM, and CF_AVG_AM, the $nDCG$ values are consistent for the comparative evaluations [18].

Regardless of the degrees of cohesiveness among group members, the $nDCG$ scores computed for GroupReM (CF_AVG, CF_LM, and CF_AVG_AM, respectively) consistently *decrease* when the group *size increases*. This decrease in $nDCG$ score is expected as *more* users are involved in a group, the *harder* it is to reach consensus among members in terms of choosing movies that represent the collective interests of the group. Moreover, regardless of the size of the groups under evaluation, the $nDCG$ scores computed for GroupReM (CF_AVG, CF_LM, and CF_AVG_AM, respectively) are slightly *higher* when considering groups with *highly-similar* users. This is anticipated, since the *more similar* the group members are with one other in terms of their preferences in movies, the *more likely* they will treat each recommendation the same, i.e., as (non-)relevant.

The results of the analysis on the performance of GroupReM (and other recommenders used for comparison purposes), in terms of the degree of cohesiveness among group members, correlates with the empirical study conducted in [8, 14].

Note that the fact that CF_AVG_AM and CF_AVG outperform CF_LM is anticipated, since the latter adopts a least misery strategy which favors the "least happy" group member in making recommendations. Furthermore, CF_AVG, CF_LM, and CF_AVG_AM rely on identifying "similar-minded" users within a movie community, i.e., a movie website, to generate movie recommendations. The search is applied to each member of a given group G. In doing so, CF_AVG, CF_LM, and CF_AVG_AM solely consider users of a movie website who rate the same movies as the ones that have been bookmarked and rated by members of G. Hence, the *less* "similar-minded" the users are (with respect to a member U of G), the *less* reliable are the ratings predicted for movies to be recommended to U (and G). GroupReM, on the other hand, does *not* require locating "similar-minded" users to perform the recommendation task. Instead, GroupReM, relies on content-similarity on tags and the popularity scores of the candidate movies.

Efficiency of GroupReM

Besides assessing the effectiveness of GroupReM, CF_AVG, CF_LM, and CF_AVG_AM on making movie recommendations to a group (in Section A.4.5), we have also validated the overall *efficiency* of (the variations of) GroupReM, CF_AVG, CF_LM, and CF_AVG_AM in suggesting movies of interest to a group.

Figure A.6 shows the average time (in seconds) required for (the alternative implementations of) GroupReM, CF_AVG, CF_LM, and CF_AVG_AM to generate recommendations for the 1,050 groups of various sizes, such that group members share the same degree of cohesiveness among one another, using the 5-fold evaluation strategy detailed in Section A.4.2. While GroupReM_Exact achieves the shortest processing time, which is 68 seconds, the additional processing time required by GroupReM, which is 66 (= 134-68) seconds, is relatively insignificant,

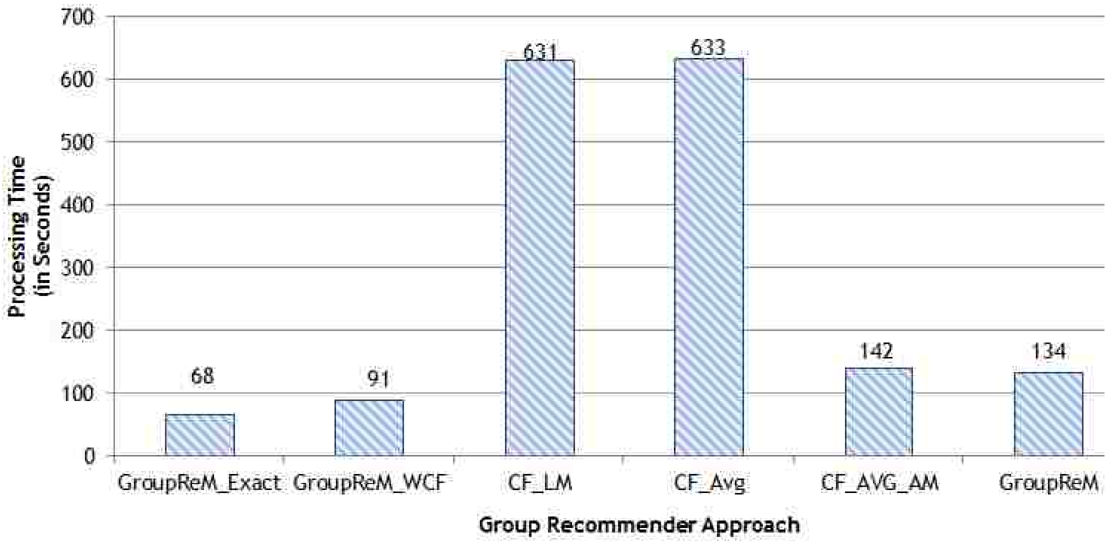


Figure A.6: (Average) Time for (alternative implementations of) GroupReM, CF_AVG, CF_LM, and CF_AVG_AM to generate recommendations for 1,050 groups (regardless of their sizes), such that group members share the same degree of cohesiveness among each other, using the 5-fold evaluation strategy detailed in Section A.4.2

compared with the degree of accuracy achieved by GroupReM in generating recommendations of interest to a group, as shown in Section A.4.5.

GroupReM and CF_AVG_AM require similar processing time to generate recommendations. When compared with CF_AVG and CF_LM, however, GroupReM requires significantly less time, i.e., as illustrated in Figure A.6, the processing time of CF_AVG and CF_LM increases by at least *8 minutes* in comparison with the processing time of GroupReM.

To further assess the efficiency of GroupReM, we consider 450 (= 3 × 15) groups (regardless of the degree of cohesiveness among the members of the group) of MovieLens users of each pre-defined size, i.e., 2 to 8, for evaluation purpose (as detailed in Section A.4.3). We computed the average processing time of GroupReM in generating recommendations for each one of the 450 groups of pre-defined size. As illustrated in Figure A.7, the (average) time (in milliseconds) required by GroupReM to generate group recommendations does not exponentially increase when the number of group members increases. Instead, as determined by the curve created using the Microsoft Excel Trend/Regression tool (also shown in Figure A.7), the increase in processing

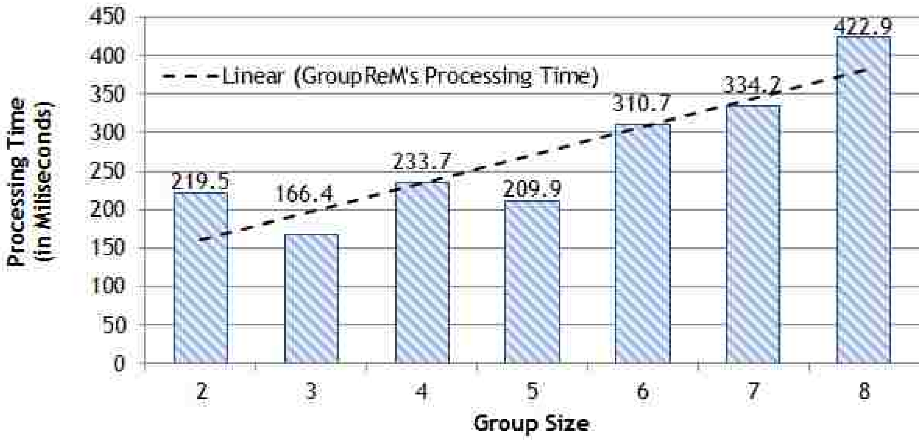


Figure A.7: (Average) Processing time of GroupReM for generating movie recommendations for groups including a certain number of group members, which is computed using groups of a predefined size, i.e., 2 to 8, as defined in Section A.4.3

time of GroupReM when the number of group members increases follows a *linear* trend, which demonstrates the scalability of GroupReM.

We have also evaluated whether the total number of movies bookmarked by the members of a group can significantly affect the group recommendation processing time of GroupReM. To draw a conclusion, we considered the 3,150 groups defined Section A.4.3 and calculated the processing time of GroupReM in generating recommendations for each of the groups, regardless of the size of the groups or the degree of cohesiveness among group members. As anticipated, the processing time (in milliseconds) required for GroupReM to generate recommendations *increases* as the total number of movies bookmarked by group members *increases*, as illustrated in Figure A.8. However, even though the total number of movies bookmarked by group members is in the thousands, the processing time of GroupReM in suggesting movies of interest to a group is at most 2.5 seconds, which is a relatively short period of time. Furthermore, the increase in processing time follows a *polynomial* trend, as determined by the curve created using Microsoft Excel Trend/Regression tool and as shown in Figure A.8.

Note that independently of the 3,150 groups introduced in Section A.4.3, we have empirically evaluated GroupReM on generating recommendations for groups of up till 100 members. Based on the conducted experiments, we have observed that (i) the total number of movies book-

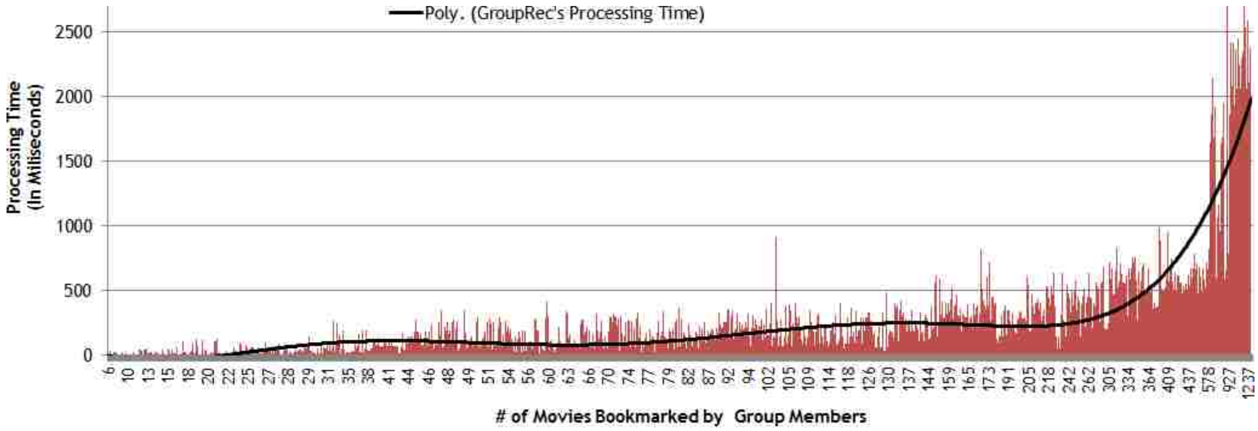


Figure A.8: Average time for GroupReM to generate group recommendations for groups with different number of bookmarked movies among group members, which is computed using 3,150 groups created in Section A.4.3

marked by group members remains in the thousands and (ii) the processing time of GroupReM is at most 5 seconds, even when considering groups of approximately 100 members with thousands of movies bookmarked among them.

A.4.6 Limitations of the Current Implementation of GroupReM

GroupReM, as currently developed, adopts a Top-N strategy and suggests a list of N movies to a group of users at a given time [49]. The current design of GroupReM does not consider the dynamic preferences of group members that may evolve over time. Moreover, the satisfaction of a group member U on the recommended items, i.e., movies in our case, may depend on other group members. As stated in [14, 99], U can be influenced by other group members through emotional contagion and conformity. The former claims that U's satisfaction may be increased if other group members are satisfied with the recommendations, whereas the latter states that the opinions of other users may influence U's opinions. The recommendation strategy adopted by GroupReM, however, does not consider that some members of a group are more capable than others to influence the remaining group members in making decisions on the (non-)relevance of movies suggested to the group, an issue to be addressed as future work.

A.5 Conclusions and Future Work

With the popularity of social activities in which groups of people are involved, either online or in person, group recommenders that are designed for identifying items of interest to a group play a significant role in social networking. One of the item domains that predominates on group recommenders is movies. Groups of friends, family members, and acquaintances, who gather to watch a movie at home or at the cinema, can use the service of a group recommender to find movies pertaining to their interests. Identifying movies to be recommended that appeal a group, however, is a non-trivial task due to the personal (and often diverse) preferences of group members in movies. We have introduced GroupReM, a group recommender on movies, which advances the current technology in solving the problem.

To suggest movies for members of a given group G at a movie website W, GroupReM first constructs a *group profile* for G, which captures the collective interests of members of G in movies. Hereafter, GroupReM relies on a simple *aggregation model* to determine the ranking score of each candidate movie M archived at W, which has not been bookmarked by members of G and is potentially of interest to G, based on the (i) *content similarity* between M and the group profile of G and (ii) *popularity* of M at W so that the top-10 ranked movies are recommended to G.

Unlike existing group recommenders on movies, which are based on the collaborative-filtering (CF) strategy and rely solely on the ratings assigned to movies to perform the recommendation task, GroupReM takes the advantage of the richness of semantic information, i.e., (personal) tags, which are available at any movie website. Considering the content-similarity of movies and a group profile, GroupReM is not constrained to find users at a movie website who are "similar-minded" based on ratings assigned to the same movies to suggest movies to a group, as CF-based group recommenders do. In addition, GroupReM employs word-correlation factors and considers non-exact-matched, but analogous, tags to more adequately determine the degree of appeal of a movie to a group, which in turn enhances the accuracy of the recommendations.

We have conducted an empirical study using more than 3,000 groups of various sizes and degrees of cohesiveness among group members, who are users in the MovieLens dataset, to verify the *effectiveness* and *efficiency* of GroupReM. The experimental results indicate that GroupReM is highly accurate in suggesting movies appealing (to a certain degree) to the members of a group. We have compared the performance of GroupReM with three well-known CF-based recommenders and verified that GroupReM outperforms the aforementioned recommenders by a large margin, and the *average* processing time of GroupReM is significantly shortened in comparison to its counterparts.

GroupReM relies on personal tags assigned to movies that have been bookmarked by group members to create group profiles and identify movies to be recommended. Occasionally, personal tags may not be available or they may be too broad in describing (the content of) a movie. We plan to investigate strategies that can be applied to *infer* tags that adequately represent the content of movies, if personal tags are missing or too general, which can further enhance the accuracy of the recommendations made by GroupReM. We also intent to enhance the recommendation strategy of GroupReM by considering the fact that some members of a group may influence the remaining group members in making decisions on (non-)relevant items suggested to the group.

References

[1] The National Illiteracy Action Project: 2007-2012. Technical report, 2012. http://www.talkingpage.org/NIAP2007.pdf.

[2] H. Acerro. Unpeeling Appeal Factors: The Gooey Center of Readers' Advisory. Technical report, The Official Blog of the Association for Library Service to Children, 2012. alsc.ala.org/blog/2012/05/unpeeling- appeal-factors-the-gooey-center-of-readers-advisory/.

[3] G. Adomavicius and Y. Kwon. Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 24:896– 911, 2012.

[4] 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.

[5] D. Alexander. Director's Opening Statement on the FY 2003 President's Budget Request for the House Subcommittee on Labor-HHS-Education Appropriations. Technical report, National Institute of Child Health and Human Development, 2002. http://www.nichd.nih.gov/about/budget/testimony/Pages/dir_FY2003.aspx.

[6] R. Allington. *What Really Matters for Struggling Readers: Designing Research-Based Programs,* 3^{rd} *Edition*. Pearsons, 2011.

[7] R. Allington and E. Gabriel. Every Child, Every Day. *Reading: The Core Skill*, 69(6): 10–15, 2012.

[8] S. Amer-Yahia, S. Basu Roy, A. Chawlat, G. Das, and C. Yu. Group Recommendation: Semantics and Efficiency. *VLDB Endowment*, 2(1):754–765, 2009.

[9] E. Andersen. If You Want to Succeed in Business, Read More Novels. Technical report, Forbes, 2002. http://www.forbes.com/sites/erikaandersen/2012/05/31/if-you-want-to-succeed-in-business-read-more-novels/.

[10] 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 *International Conference on World Wide Web (WWW)*, pages 199–208, 2008.

[11] J. Anderson. Lix and Rix: Variations on a Little-known Readability Index. *Journal of Reading*, 26:993–1022, 1983.

[12] J. Aslam and M. Montague. Models for Metasearch. In *International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 267–276, 1997.

[13] GroupLens Research at University of Minnesota. Available at http://www.grouplens.org/system/files/hetrec2011-movielens-2k.zip.

[14] L. Baltrunas, T. Makcinskas, and F. Ricci. Group Recommendations with Rank Aggregation and Collaborative Filtering. In *ACM Conference on Recommender Systems (RecSys)*, pages 119–126, 2010.

[15] S. Banerjee and T.Pedersen. Extended Gloss Overlaps as a Measure of Semantic Relatedness. In *International Joint Conference on Artificial Intelligence(IJCAI)*, pages 205–810, 2003.

[16] C. Basu, H. Hirsh, W. Cohen, and C. Nevill-Manning. Technical Paper Recommendation: A Study in Combining Multiple Information Sources. *Journal of Artificial Intelligence Research (JAIR)*, 1:231–252, 2001.

[17] F. Belem, E. Martings, T. Pontes, J. Almeida, and M. Goncalves. Associative Tag Recommendation Exploiting Multiple Textual Features. In *International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 1033–1042, 2011.

[18] A. Bellogin, I. Cantador, and P. Castells. A Study of Heterogeneity in Recommendations for a Social Music Service. In *International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, pages 1–8, 2010.

[19] R. Benjamin. Reconstructing Readability: Recent Developments and Recommendations in the Analysis of Text Difficulty. *Educational Psychology Review*, 24:63–88, 2012.

[20] S. Berkovsky and J. Freyne. Group-Based Recipe Recommendations: Analysis of Data Aggregation Strategies. In *ACM Conference on Recommender Systems (RecSys)*, pages 111–118, 2010.

[21] S. Berkovsky, J. Freyne, and M. Coombe. Aggregation Trade Offs in Family Based Recommendations. In *Australasian Joint Conference on Advances in Artificial Intelligence (AI)*, pages 646–655, 2009.

[22] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research (JMLR)*, 3:993–1022, 2003.

[23] T. Bogers and A. van den Bosch. Recommending Scientific Articles Using CiteULike. In *ACM Conference on Recommender systems (RecSys)*, pages 287–290, 2008.

[24] T. Bogers and A. van den Bosch. Fusing Recommendations for Social Bookmarking Websites. *International Journal of Electronic Commerce (IJEC)*, 15(3):33–75, 2011.

[25] L. Boratto and S. Carta. State-of-the-Art in Group Recommendation and New Approaches for Automatic Identification of Groups. *Studies in Computational Intelligence*, 324:1–20, 2011.

[26] L. Boratto, S. Carta, A. Chessa, M. Agelli, and M. Clemente. Group Recommendation with Automatic Identification of Users Communities. In *IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, pages 547–550, 2009.

[27] S. Bourke, K. McCarthy, and B. Smyth. Power to the People: Exploring Neighbourhood Formations in Social Recommender System. In *ACM Conference on Recommender Systems (RecSys)*, pages 337–340, 2011.

[28] I. Cantador and P. Castells. Extracting Multilayered Communities of Interest from Semantic User Profiles: Application to Group Modeling and Hybrid Recommendations. *Computers in Human Behavior*, 27(4):1321–1336, 2011.

[29] I. Cantador, A. Bellogin, and D. Vallet. Content-based Recommendation in Social Tagging Systems. In *ACM Conference on Recommender Systems (RecSys)*, pages 237–240, 2010.

[30] 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 *ACM Conference on Information and Knowledge Management (CIKM)*, pages 1227–1236, 2009.

[31] H. Chen and A. Chen. A Music Recommendation System Based on Music Data Grouping and User Interests. In *ACM Conference on Information and Knowledge Management (CIKM)*, pages 231–238, 2001.

[32] X. Chen. Google Books and WorldCat: A Comparison of Their Content. *Online Information Review*, 36(4):507–516, 2012.

[33] P. Christen. Automatic Record Linkage Using Seeded Nearest Neighbour and Support Vector Machine Classification. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 151–159, 2008.

[34] I. Christensen and S. Schiaffino. Entertainment Recommender Systems for Group of Users. *Expert Systems with Applications*, 38:14127–14135, 2011.

[35] S. Chu, S. Tse, E. Loh, and K. Chow. Collaborative Inquiry Project-based Learning: Effects on Reading Ability and Interests. *Library and Information Science Research* , 33:236–243, 2011.

[36] C. Clark and K. Rumbold. Reading for Pleasure: a Research Overview. Technical report, National Literacy Trust, 2006.

[37] M. Coleman. A Computer Readability Formula Designed for Machine Scoring. *Applied Psychology*, 60(2):283–284, 1975.

[38] K. Collins-Thompson and J. Callan. A Language Modeling Approach to Predicting Reading Difficulty. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*, pages 193–200, 2004.

[39] G. Cormack, C. Clarke, and S. Buettcher. Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods. In *International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 758–759, 2009.

[40] C. Coulter and M. Smith. The Construction Zone: Literary elements in Narrative Research. *Educational Researcher*, 38(8):577–590, 2009.

[41] A. Cox and K. Horne. Fast-Paced, Romantic, Set in Savannah: A Comparison of Results from Readers' Advisory Databases in the Public Library. *Public Library Quarterly*, 31(4): 285–302, 2012.

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

[43] A. Crossen, J. Budzik, and K. Hammond. Flytrap: Intelligent Group Music Recommendation. In *ACM International Conference on Intelligent User Interfaces (IUI)*, pages 184–185, 2002.

[44] M. Crowhurst and G. Piche. Audience and Mode of Discourse Effects on Syntactic Complexity in Writing at Two Grade Levels. *Research in the Teaching of English*, 13(2):101–109, 1979.

[45] A. Davison and R. Kantor. On the Failure of Readability Formulas to Define Readable Texts: A Case Study from Adaptations. *Reading Research Quarterly*, 17(2):187–209, 1982.

[46] 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.

[47] M. De Marneffe. Generating Typed Dependency Parses from Phrase Structure Parses. In *Language Resources and Evaluation Conference (LREC)*, pages 449–454, 2006.

[48] J. Denning, M.S. Pera, and Y.-K. Ng. A Readability Level Prediction Tool for K-12 Books. In Submission, January 2014.

[49] M. Deshpande and G. Karypis. Item-based Top-N Recommendation Algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[50] M. Ekstrand, P. Kannan, J. Stemper, J. Butler, J. Konstan, and J. Riedl. Automatically Building Research Reading Lists. In *ACM Conference on Recommender Systems (RecSys)*, pages 159–166, 2010.

[51] L. Feng, N. Elhadad, and M. Huenerfauth. Cognitively Motivated Features for Readability Assessment. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 229–237, 2009.

[52] L. Feng, M. Jansche, M. Huenerfauth, and N. Elhadad. A Comparison of Features for Automatic Readability Assessment. In *International Conference on Computational Linguistics (COLING)*, pages 276–284, 2010.

[53] R. Fisher. Philosophy in Primary Schools: Fostering Thinking Skills and Literacy. *Reading*, 35:67–73, 2003.

[54] UNESCO Institute for Statistics. Adult and Youth Literacy. Technical report, 2013. http://www.uis.unesco.org/literacy/Documents/fs26-2013-literacy-en.pdf.

[55] D. Friedman and L. Hoffman-Goetz. A Systematic Review of Readability and Comprehension Instruments Used for Print and Web-Based Cancer Information. *Health Education & Behavior*, 33(3):352–373, 2006.

[56] M. Gartrell, X. Xing, Q. Lv, A. Beach, R. Han, S. Mishra, and K. Seada. Enhancing Group Recommendation by Incorporating Social Relationship Interactions. In *ACM International Conference on Supporting Group Work (GROUP)*, pages 97–106, 2010.

[57] S. Givon and V. Lavrenko. Predicting Social-Tags for Cold Start Book Recommendations. In *ACM Conference on Recommender Systems (RecSys)*, pages 333–336, 2009.

[58] GoodRead's Official Blog. Announcing GoodReads Personalized Recommendation. goodreads.com/blog/show/303-announcing-goodreads-personalized-recommendations, 2011.

[59] A. Graesser, D. McNamara, M. Louwerse, and Z. Cai. Coh-Metrix: Analysis of Text on Cohesion and Language. Behavior Research Methods. *Instruments and Computers*, 36(2): 193–202, 2004.

[60] T. Griffiths and M. Steyvers. Finding Scientific Topics. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 101:5228–5235, 2004.

[61] Z. Guan, C. Wang, J. Bu, C. Chen, K. Yang, D. Cai, and X. He. Document Recommendation in Social Tagging Services. In *International Conference on World Wide Web (WWW)*, pages 391–400, 2010.

[62] R. Gunning. *The Technique of Clear Writing*. McGraw-Hill, 1952.

[63] N. Gustafson and Y.-K. Ng. Augmenting Data Retrieval with Information Retrieval Techniques by Using Word Similarity. In *Natural Language and Information Systems (NLDB)*, pages 163–174, 2008.

[64] I. Guy, N. Zwerdling, I. Ronen, D. Carmel, and E. Uziel. Social Media Recommendation Based on People and Tags. In *International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 194–201, 2010.

[65] M. Heilman, K. Collins-Thompson, and M. Eskenazi. An Analysis of Statistical Models and Features for Reading Difficulty Prediction. In *Workshop on Innovative Use of NLP for Building Educational Applications (NAACL-HLT BEA)*, pages 71–79, 2008.

[66] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An Algorithmic Framework for Performing Collabor- ative Filtering. In *International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 230–237, 1999.

[67] N. Hollands. Improving the Model for Interactive Readers' Advisory Service. *Reference & User Services Quarterly*, 45(3):205–212, 2006.

[68] C. Hoscher and G. Strube. Web Search Behavior of Internet Experts and Newbies. *Computer Networks*, 33:337–346, 2000.

[69] L. Hunt. The Effect of Self-Selection, Interest, and Motivation upon Independent Instructional, and Frustrational Levels. *The Reading Teacher*, 50:278–282, 1997.

[70] A. Jameson and B. Smyth. Recommendation to Groups. In *P. Brusilovsky, A. Kobsa, and W. Nejdl (Eds.), The Adaptive Web: Methods and Strategies of Web Personalization*, pages 596–627, 2007.

[71] Q. Jiang, J. Zhu, M. Sun, and E. Xing. Monte Carlo Methods for Maximum Margin Supervised Topic Models. In *Neural Information Processing Systems Foundation (NIPS)*, pages 1601–1609, 2012.

[72] Y. Jiang, A. Jia, Y. Feng, and D. Zhao. Recommending Academic Papers via Users' Reading Purposes. In *ACM Conference on Recommender Systems (RecSys)*, pages 241–244, 2012.

[73] T. Joachims. Optimizing Search Engines Using Click- through Data. In *ACM SIGKDD Conference on Knowledge, Discovery, and Data Mining*, pages 133–142, 2002.

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

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

[76] J. Kincaid, R. Fishburne, R. Rogers, and B. Chissom. Derivation of New Readability Formulas (Automated Readability Index, Fog Count, and Flesch Reading Ease formula) for Navy Enlisted Personnel. Technical Report 8-75, Chief of Naval Technical Training, 1975.

[77] B. Knijnenburg, S. Bostandjiev, J. O'Donovan, and A. Kobsa. Inspectability and Control in Social Recommenders. In *ACM Conference on Recommender Systems (RecSys)*, pages 43–50, 2012.

[78] J. Koberstein and Y.-K. Ng. Using Word Clusters to Detect Similar Web Documents. In *Knowledge Science, Engineering and Management (KSEM)*, pages 215–228, 2006.

[79] I. Konstas, V. Stathopoulos, and J. Jose. On Social Networks and Collaborative Recommendation. In *International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 195–202, 2009.

[80] M. Koolen, J. Kamps, and G. Kazai. Social Book Search: Comparing Topical Relevance Judgments and Book Suggestions or Evaluation. In *ACM Conference on Information and Knowledge Management (CIKM)*, pages 185–194, 2012.

[81] Y. Koren. Factorization Meets the Neighborhood: a Multifaceted Collaborative Filtering Model. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 426–434, 2008.

[82] Y. Koren. Factor in the Neighbors: Scalable and Accurate Collaborative Filtering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 4(1):Article 1, 24 pages, 2010.

[83] C. Leacock and M. Chodorow. *Combining Local Context and WordNet Similarity for Word Sense Identification*. The MIT Press, 1998.

[84] A. Leahy. The Importance of Reading for All of Us. Technical report, 2012. http://www.huffingtonpost.com/anna-leahy/the-importance-of-reading_b_1623078.html.

[85] Renaissance Learning. ATOS vs. Lexile Which Readability Formula is Best? Available at kmnet. renlearn.com/Library/R003520002GE7114.pdf, 2006.

[86] J. Lee. Analyses of Multiple Evidence Combination. In *International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 267–276, 1997.

[87] J. Lennon, P. Koleff, J. Greenwood, and K. Gaston. The Geographical Structure of British Bird Distributions: Diversity, Spatial Turnover and Scale. *Journal of Animal Ecology*, 70: 966–979, 2001.

[88] H. Li, Y. Gu, and S. Koul. Review of Digital Library Book Recommendation Models. SSRN (dx.doi.org/10.2139/ssrn.1513415), 2009.

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

[90] L. Liu, B. Fang, and W. Zhang. Speak the Same Language with Your Friends: Augmenting Tag Recommenders with Social Relations. In *ACM Conference on Hypertext and Hypermedia (HT)*, pages 45–50, 2010.

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

[92] M. Milone. The Development of ATOS: The Renaissance Readability Formula. http://doc.renlearn.com/KMNet/ R004250827GJ11C4.pdf, 2012.

[93] H. Ma, D. Zhou, C. Liu, M. Lyu, and I. King. Recommender Systems with Social Regularization. In *ACM International Conference on Web Search and Data Mining (WSDM)*, pages 287–296, 2011.

[94] Y. Ma, E. Fosler-Lussier, and R. Lofthus. Ranking-based Readability Assessment for Early Primary Children's Literature. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 548–552, 2012.

[95] C. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 2003.

[96] N. Manouselis, H. Drachsler, K. Verbert, and E. Duval. *Recommender Systems for Learning*. Springer Briefs in Electrical and Computer Engineering, 2013.

[97] P. Massa and P. Avesani. Trust-aware Recommender Systems. In *ACM Conference on Recommender Systems (RecSys)*, pages 17–24, 2007.

[98] J. Masthoff. Group Modeling: Selecting a Sequence of Television Items to Suit a Group of Viewers. *User Modeling and User-Adapted Interaction (UMUAI)*, 14:37–85, 2004.

[99] J. Masthoff and A. Gatt. In Pursuit of Satisfaction and the Prevention of Embarrassment: Affective State in Group Recommender Systems. *User Modeling User-Adapted Interaction (UMUAI)*, 16(3-4):281–319, 2006.

[100] J. McCarthy and T. Anagnost. MusicFX: An Arbiter of Group Preferences for Computed Supported Collaborative Workouts. In *ACM Conference on Computer Supported Cooperative Work (CSCW)*, pages 363–372, 1998.

[101] K. McCarthy, M. Salamo, L. Coyle, L. McGinty, B. Smyth, and P. Nixon. Group Recommender Systems: A Critiquing Based Approach. In *ACM International Conference on Intelligent User Interfaces (IUI)*, pages 267–269, 2006.

[102] G. McLaughlin. SMOG Grading–A New Readability Formula. *Reading*, 12(8):639–646, 1969.

[103] S. McNee, N. Kapoor, and J. Konstan. Don't Look Stupid: Avoiding Pitfalls when Recommending Research Papers. In *ACM Conference on Computer Supported Cooperative Work (CSCW)*, pages 171–180, 2006.

[104] Inc. Metametrics. Lexile: A System for Measuring Reader Ability and Text Difficulty: A Guide for Educators. Technical report, Scholastic, 2008. http://teacher.scholastic.com/products/sri_reading_assessment/pdfs/SRI_ProfPaper_Lexiles.pdf.

[105] J. Miller. *Cataloging Correctly for Kids: An Introduction to the Tools, 4th Ed.*, chapter Sears list of subject Headings, pages 75–79. American Library Association, 2006.

[106] G. De Francisci Morales, A. Gionis, and C. Lucchese. From Chatter to Headlines: Harnessing the Real-Time Web for Personalized News Recommendation. In *ACM International Conference on Web Search and Data Mining (WSDM)*, pages 153–162, 2012.

[107] R. Myers. *Classical and Modern Regression with Applications*. Duxbury Press Belmont, 1990.

[108] A. Naak, H. Hage, and E. Ameur. A Multi-criteria Collaborative Filtering Approach for Research Paper Recommendation in Papyres. In *E-Technologies: Innovation in an Open World*, pages 25–39. Springer Berlin Heidelberg, 2009.

[109] C. Nascimento, A. Laender, A. da Silva, and M. Goncalves. A Source Independent Framework for Research Paper Recommendation. In *ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 297–306, 2011.

[110] National Assessment of Educational Progress. The Nation's Report Card. http://nationsreportcard.gov/ reading_math_2013//student-groups. 2013.

[111] A. Neumann. A survey of recommender systems at major sti providers. In *Recommender Systems for Information Providers*, Contributions to Management Science, pages 1–12. Physica-Verlag HD, 2009. ISBN 978-3-7908-2133-8.

[112] J. Oakhill and K. Cain. The Precursors of Reading Ability in Young Readers: Evidence from a Four-Year Longitudinal Study. *School Science Review*, 16(2):91–121, 2012.

[113] P. Ochi, S. Rao, L. Takayama, and C. Nass. Predictors of user perceptions of web recommender systems: How the basis for generating experience and search product recommendations affects user responses. *International Journal of Human-Computer Studies*, 68: 472–482, 2010.

[114] M. O'Connor, D. Cosley, J. Konstan, and J. Riedl. PolyLens: A Recommender System for Groups of Users. In *European Conference on Computer-Supported Cooperative Work (ECSCW)*, pages 199–218, 2001.

[115] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

[116] M. Park, H. Park, and S. Cho. Restaurant Recommendation for Group of People in Mobile Environments Using Probabilistic Multi-criteria Decision Making. In *Asia-Pacific Conference on Computer-Human Interaction (APCHI)*, pages 114–122, 2008.

[117] 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.

[118] D. Parra and P. Brusilovsky. Collaborative Filtering for Social Tagging Systems: An Experiment with CiteULike. In *ACM Conference on Recommender Systems (RecSys)*, pages 237–240, 2009.

[119] M.S. Pera and Y.-K. Ng. A Naive Bayes Classifier for Web Document Summaries Created by Using Word Similarity and Significant Factors. *International Journal on Artificial Intelligence Tools (IJAIT)*, 19(4):465–486, 2010.

[120] M.S. Pera and Y.-K. Ng. With a Little Help From My Friends: Generating Personalized Book Recommendations Using Data Extracted from a Social Website. In *IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, pages 96–99, 2011.

[121] M.S. Pera and Y.-K. Ng. A Personalized Recommendation System on Scholarly Publications. In *ACM Conference on Information and Knowledge Management (CIKM)*, pages 2133–2136, 2011.

[122] M.S. Pera and Y.-K. Ng. What to Read Next?: Making Personalized Book Recommendations for K-12 Users. In *ACM Conference on Recommender Systems (RecSys)*, pages 113–120, 2013.

[123] M.S. Pera and Y.-K. Ng. A group recommender for movies based on content similarity and popularity. *Journal of Information Processing & Management (IPM)*, 49(3):673–687, 2013.

[124] M.S. Pera and Y.-K. Ng. Exploiting the Wisdom of Social Connections to Make Personalized Recommendations on Scholarly Articles. To Appear in Journal of Intelligent Information Systems (JIIS), Springer, 2014.

[125] M.S. Pera and Y.-K. Ng. Automating Readers' Advisory to Make Book Recommendations for K-12 Readers. In Submission, January 2014.

[126] M.S. Pera, W. Lund, and Y.-K. Ng. A Sophisticated Library Search Strategy Using Folksonomies and Similarity Matches. *Journal of the American Society for Information Science and Technology (JASIST)*, 60(7):1392–1406, 2009.

[127] G. Pirro and J. Euzenat. A Feature and Information Theoretic Framework for Semantic Similarity and Relatedness. In *International Semantic Web Conference (ISWC)*, pages 615–630, 2010.

[128] A. Pudhiyaveetil, S. Gauch, H. Luong, and J. Eno. Conceptual Recommender System for CiteSeerX. In *ACM Conference on Recommender Systems (RecSys)*, pages 241–244, 2009.

[129] R. Qumsiyeh and Y.-K. Ng. ReadAid: A Robust and Fully-Automated Readability Assessment Tool. In *IEEE International Conference on Tools with Artificial Intelligence(ICTAI)*, pages 539–546, 2011.

[130] R. Qumsiyeh, M.S. Pera, and Y.-K. Ng. Generating Exact- and Ranked Partially-Matched Answers to Questions in Advertisements. In *International Conference on Very Large Databases (VLDB)*, pages 217–228, 2012.

[131] J. Recio-Garcia, G. Jimenez-Diaz, A. Sanchez-Ruiz, and B. Diaz-Agudo. Personality Aware Recommendations to Groups. In *ACM Conference on Recommender Systems (RecSys)*, pages 325–328, 2009.

[132] Renaissance Learning. Matching Books to Students: How to Use Readability Formulas and Continuous Monitoring to Ensure Reading Success. http://doc.renlearn.com/KMNet/R003544312GE0BA6.pdf, 2011.

[133] F. Ricci, L. Rokach, B. Shapira, and P. Kantor. *Recommender Systems Handbook*. Springer, 2011.

[134] L. Robare. Basic Subject Cataloging Using LCSH. Washington, DC: Library of Congress, Cataloging Distribution Service. Available at http://loc.gov/catworkshop/courses/basicsubject/pdf/LCSH_Instructor_2011.pdf, 2007.

[135] R. Robinson, M. McKenna, and K. Conradi. *Issues & Trends in Literacy Education*. Pearson, 2011.

[136] S. Basu Roy, S. Amer-Yahia, A. Chawla, G. Das, and C. Yu. Space Efficiency in Group Recommendation. *VLDB Journal*, 19(6):877–900, 2010.

[137] R. Salmon, C. Ribeiro, and S. Amarala. Fact based search engine: News fact finder utilizing nave bayes classification. *Quality Issues in the Management of Web Information*, 50:121–143, 2013.

[138] J. Saricks. *Readers' Advisory Service in the Public Library, 3^{rd} Ed.* ALA Store, 2005.

[139] School Renaissance Inst. Inc. The ATOS Readability Formula for Books and How it Compares to Other Formulas. Technical Report ED449468, ERIC Document Reproduction Service, 2000.

[140] S. Schwarm and M. Ostendorf. Reading Level Assessment Using Support Vector Machines and Statistical Language Models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 523–530, 2005.

[141] R. Shaban. A guide to writing book reviews. jephc.com/full_article.cfm?content_id=388, 2006. Journal of Emergency Primary Health Care.

[142] Y. Shavitt, E. Weinsberg, and U. Weinsberg. Building Recommendation Systems Using Peer-to-peer Shared Content. In *ACM Conference on Information and Knowledge Management (CIKM)*, pages 1457–1460, 2010.

[143] A. Shepitsen, J. Gemmel, B. Mobasher, and R. Burke. Personalized Recommendations in Social Tagging Systems Using Hierarchical Clustering. In *ACM Conference on Recommender Systems (RecSys)*, pages 259–266, 2008.

[144] A. Sieg, B. Mobasher, and R. Burke. Improving the Effectiveness of Collaborative Recommendation with Ontology-based User Profiles. In *International Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec)*, pages 39–46, 2010.

[145] D. Smith, A. Stenner, I. Horabin, and M. Smith. The Lexile Scale in Theory and Practice: Final Report. Technical Report ED307577, ERIC Document Reproduction Service, 1989.

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

[147] G. Spache. A New Readability Formula for Primary-Grade Reading Materials. *Elementary School*, 53(7):410–413, 1953.

[148] K. Sugiyama and M. Kan. Scholarly Paper Recommendation via User's Recent Research Interests. In *ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 29–38, 2010.

[149] M. Sun, F. Li, J. Lee, K. Zhou, G. Lebanon, and H. Zha. Learning Multiple-question Decision Trees for Cold-start Recommendation. In *ACM International Conference on Web Search and Data Mining (WSDM)*, pages 445–454, 2013.

[150] H. Tan, Y. Zhao, and H. Zhang. Conceptual Data Model-based Software Size Estimation for Information Systems. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 19(2):Article 4, 2009.

[151] K. Tanaka-Ishii, S. Tezuka, and H. Terada. Sorting Texts by Readability. *Computational Linguistics*, 36(2):203–227, 2010.

[152] S. Vanneman. Keep them reading. *School Library Monthly*, 27(3):21–22, 2010.

[153] W. DuBay. The Principles of Readability. www.nald.ca/library/ research/readab/readab.pdf, 2004.

[154] J. Wang, Q. Li, and Y. Chen. User Comments for News Recommendation in Social Media. In *International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 881–882, 2010.

[155] J. Wooldridge. *Introductory Econometrics: A Modern Approach*. South-Western Pub, 2009.

[156] C. Yang, B. Wei, J. Wu, Y. Zhang, and L. Zhang. CARES: A Ranking-oriented CADAL Recommender system. In *ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 203–212, 2009.

[157] K. Yi and L. Chan. Revisiting the syntactical and structural analysis of library of congress subject headings for the digital environment. *Journal of the American Society for Information Science and Technology (JASIST)*, 61(4):1532–2890, 2010.

[158] C. Yu, L. Lakshmanan, and S. Amer-Yahia. It Takes Variety to Make a World: Diversification in Recommender Systems. In *EDBT*, pages 368–378, 2009.

[159] D. Zabel. *Reference Reborn: Breathing New Life into Public Services Librarianship*. Libraries Unlimited, 2010.

[160] H. Zheng, D. Wang, Q. Zhang, H. Li, and T. Yang. Do Clicks Measure Recommendation Relevancy?: An Empirical User Study. In *ACM Conference on Recommender Systems (RecSys)*, pages 249–252, 2010.

[161] J. Zobel. *Writing for Computer Science*. Springer, 2004.