Scientific Information Understanding via Open Educational Resources (OER)

Xiaozhong Liu
School of Informatics and Computing
Indiana University
Bloomington, Bloomington, IN, USA, 47405
liu237@indiana.edu

Zhuoren Jiang
College of Transportation Management
Dalian Maritime University
Dalian, China, 116026
jzr_dimu@hotmail.com

Liangcai Gao
Institute of Computer Science and Technology
Peking University
Beijing, China, 100871
glc@pku.edu.cn

ABSTRACT
Scientific publication retrieval/recommendation has been investigated in the past decade. However, to the best of our knowledge, few efforts have been made to help junior scholars and graduate students to understand and consume the essence of those scientific readings. This paper proposes a novel learning/reading environment, OER-based Collaborative PDF Reader (OCPR), that incorporates innovative scaffolding methods that can: 1. automatically characterize student emerging information need while reading a paper; and 2. enable students to readily access open educational resources (OER) based on their information need. By using meta-search methods, we pre-indexed 1,112,718 OERs, including presentation videos, slides, algorithm source code, or Wikipedia pages, for 41,378 STEM publications. Based on the computational information need, we use text mining and heterogeneous graph mining algorithms to recommend high quality OERs to help students better understand the scientific content in the paper. Evaluation results and exit surveys for an information retrieval course show that the OCPR system alone with the recommended OERs can effectively assist graduate students better understand the complex STEM publications. For instance, 78.42% of participants believe the OCPR system and recommended OERs can provide precise and useful information they need, while 78.43% of them believe the recommended OERs are close to exactly what they need when reading the paper. From OER ranking viewpoint, MRR, MAP and NDCG results prove that learning to rank and cold start solutions can efficiently integrate different text and graph ranking features.

Categories and Subject Descriptors
H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval

General Terms
System, Algorithms, Experimentation, Measurement

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires specific permission and/or a fee. Request permissions from Permissions@acm.org.

SIGIR ’15, August 09 - 13, 2015, Santiago, Chile.
DOI: http://dx.doi.org/10.1145/2766462.2767750.

Keywords
Scaffolding; Information Need Characterization; Education; Heterogeneous Graph Mining; Evaluation; Information Understanding

1. INTRODUCTION AND MOTIVATION

"I don’t think I need those books... I cannot understand them anyway...” This was a student’s response when I suggested that she should find some statistics methods books in the university library. Actually, her response is quite representative when students face STEM (Science, Technology, Engineering, and Mathematics) readings. In the last decade, while the volume of STEM publications has increased dramatically in university and digital libraries, few efforts have been made to help students/junior scholars understand them. From an educational viewpoint, understanding the content of scientific publications in STEM course environments remains daunting [10], i.e., Information Access ≠ Information Understanding. For instance, a number of search/recommendation algorithms ensure Google Scholar and Microsoft Academic give users access to the high quality scientific readings, but junior scholars/graduate students may not necessarily understand them. In a recent study of STEM [8], PhD students (in computer science domain) claimed that the complex models, formulas, and methods in the readings were deemed too difficult to understand because of the students’ limited knowledge in computer science and mathematics. Students need innovative scaffolding methods/systems to better assist in understanding scientific publications.

While the content of scientific publications is challenging and difficult for students, a large number of open educational resources (OERs) are increasingly available and have great potential to assist students better understand the essence of these publications. For example, scholars, organizations and institutions commonly share conference slides, video lectures, tutorials, Wikipedia pages, datasets and source codes generated from research. As these OERs are often seamlessly blended or integrated with other scholarly publications or research topics, students can, we surmise, better understand the papers by leveraging those OERs. In this paper, we define it as OER-based Scaffolding. In a study by Liu [8], for example, students expect that a blend of OER-based learning objects could significantly improve learning. Unfortunately, not all OERs exist in a publication as cited references; some may be scattered across different social media (e.g., TED, SlideShare, authorStream, GitHub, SourceForge, YouTube, Google Code, VideoLecture, and

1In education, scaffolding refers to a variety of instructional techniques used to move students progressively toward stronger understanding and, ultimately, greater independence in the learning process. http://edglossary.org/scaffolding/
Wikipedia). While the cost of manually locating those OERs for large-scale STEM topics or publications is prohibitive, as another challenge, the existing system cannot use those innovative OERs to address students’ emerging information needs, i.e., using or recommending OERs to address students’ questions on a particular section/paragraph/sentence/formula of a scientific publication.

Motivated by these observations, we propose a new solution and an innovative system framework to help students and scholars better understand and learn from scientific publications in a course environment by leveraging OERs, i.e., OER-based scaffolding. A new learning system, called the **OER-based Collaborative PDF Reader (OCPR)**, is to be implemented as a first step towards this effort. Theoretically speaking, the system is also designed to accommodate the needs of students in MOOCs (massive open online courses) in addition to students from varied backgrounds. As Figure 1 shows, the system can auto-characterize student’s emerging information need while reading a paper, while recommending high quality OERs to help them understand the publication.

With the exception of using text mining for OER recommendation (i.e., using highlighted text to find relevant OERs), OCPR also employs heterogeneous graph mining to enhance scaffolding performance by investigating the weighted relations between OER pair; OER and paper; OER-topic association; paper-topic association; and the topic-topic relationships, which help to prioritize important OERs and filter out the noisy resources. The meta-path and supervised random-walk-based graph algorithms [11] will be used to explore the important OERs (vertices) given student’s implicit and explicit information needs (topic and paper seed vertices). For example, we can assume that when a student is reading a paper, she can be interested in finding OERs relevant to the paper, specific sentences in the paper, the paper’s topics, or weekly teaching topics. From an information need perspective, for link-based recommendations, student information need is characterized by some seed vertices on the heterogeneous networks, i.e. the paper or topics that the student is currently reading, and supervised random walk algorithms can be used to recommend candidate OERs.

Evaluation result shows that, first, auto-recommended OERs can provide important and useful information to help students better understand the content of the scientific publications, i.e., MRR (mean reciprocal rank) score is 0.8609 (students find the useful OERs in the 1st or 2nd position in the result ranking list). This assumption is also confirmed in the exit survey, where 78.42% of participants believe the OCPR system along with recommended OERs can provide precise and useful information they need, while 78.43% of them believe the recommended OERs are close to exactly what they need. From system perspective, 80.39% of participates believe the OCPR system and real-time OER recommendation function is easy to use. Second, based on the students’ judgments, we found all the OER recommendation can be potentially useful, and the learning to rank method can be quite useful to enhance the recommendation performance. As Table 5 shows, learning to rank and cold start methods significantly ($p < 0.0001$) outperform other methods.

The contribution of this paper is fourfold. First, we propose a novel auto-scaffolding method to help students better understand the scientific readings in a course environment by leveraging various kinds of OERs. Second, a new system, OCPR, is employed to capture student reading behavior while enabling them to ask questions or highlight the confusing part of the readings. Third, novel algorithms are proposed to characterize students’ information need and recommend OERs to them while reading a paper. Last but not least, an experiment (with 51 participates) is employed to validate the usefulness of the system and to evaluate the OER recommendation algorithms.

## 2. RELATED WORK

Scaffolding is defined as a “process that enables a child or novice to solve a problem, carry out a task, or achieve a goal which would be beyond his unassisted efforts” [25]. Since 1976, the term scaffolding has been widely used in educational research [14], and the scaffolding construct is being applied more broadly [16]. In particular, the concept of scaffolding is applied to the studies of computer-assisted learning environments, i.e. computer-mediated scaffolding [16]. Naturally, the focus of scaffolding research is on system design and usage. One of the most recent efforts is to utilize existing social tagging and annotation tools. Social tagging and annotation have produced positive results in a number of tasks, including promoting student learning [6, 18, 24].

However, prior studies also found those scaffolding approaches, by leveraging social tagging or annotation, can be quite limited [13], and students cannot essentially benefit from such systems when reading a challenging text. Not until recently did researchers begin to focus on the usefulness of OER. For instance, Dennis, et al. [4] found that an additional link to video presentation had significantly positive impacts on students’ learning based on quiz scores. More recently, Liu [8, 9] found OERs, like YouTube video, dataset, and Wikipedia page, can assist students better understand the scientific readings. Meanwhile, he found information retrieval methods, i.e., language model and BM25, can help to build OER index for a large number of publications automatically. However, this research cannot address students’ emerging information need while reading a paper. For instance, existing method cannot help students to understand a specific sentence or paragraph. Meanwhile, existing methods generate a large number of noisy OERs [8]. For instance, Liu found a large number of YouTube videos do not have high quality text descriptions, and methods based on text similarity may be not accurate.

For this study, we use text retrieval and heterogeneous graph mining plus random walk [21] to recommend OERs while filtering out the noisy data. Similar approaches have been used to recommend YouTube video. For instance, Baluja et al., [2] used user-video graph plus random walk to provide personalized video suggestions for users. A heterogeneous network is defined as a directed graph $G = (\mathcal{V}, \mathcal{E})$ with an object type mapping function $\tau : \mathcal{V} \to \mathcal{A}$ and a link type mapping function $\phi : \mathcal{E} \to \mathcal{R}$, where each object $v \in \mathcal{V}$ belongs to one particular object type $\tau(v) \in \mathcal{A}$, each link $e \in \mathcal{E}$ belongs to a particular relation $\phi(e) \in \mathcal{R}$, and if two links belong to the same relation type, the two links share the same starting object type as well as the ending object type. The network scheme, denoted as $T_G = (\mathcal{A}, \mathcal{R})$, is a meta template for the heterogeneous network $G = (\mathcal{V}, \mathcal{E})$ with the object type mapping $\tau : \mathcal{V} \to \mathcal{A}$ and the link mapping $\phi : \mathcal{E} \to \mathcal{R}$, which is a directed graph defined over object types $\mathcal{A}$, with edges as relations from $\mathcal{R}$ [11].

The concept of meta-path was first proposed in [21], which can systematically capture the semantic relationships between objects in a heterogeneous information network scenario. A meta-path $\mathcal{P}$ is a path defined on the graph of network schema $T_G = (\mathcal{A}, \mathcal{R})$, and
is denoted in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \ldots \xrightarrow{R_l} A_{l+1}$, which defines a composite relation $R = R_1 \circ R_2 \circ \ldots \circ R_l$ between types $A_1$ and $A_{l+1}$, where $\circ$ denotes the composition operator on relations. Different meta-path-based mining tasks are studied, including similarity search [21], relationship prediction [19,20], user-guided clustering [22], and recommendation [11, 26, 27]. It turns out that meta-path serves as a very critical feature extraction tool for most of the mining tasks in a heterogeneous information network. In this paper, we also employed the supervised random walk methods which have been investigated in [7, 11] and prior studies show that the supervised random walk can significantly enhance the classical unsupervised graph mining methods.

3. RESEARCH METHODS
In this section we discuss the research method in detail which includes: to collect different kinds of candidate OERs for scientific publications (3.1), to design OCPR system (3.2), and to design the text- and graph-based OER recommendation algorithms (3.3).

3.1 OER Collection Indexation via Meta-Search
In order to recommend high-quality OERs to help students better understand scientific readings, we need to index different kinds of OERs. In this study, we collect four kinds of OERs, presentation video, slides, source code, and Wikipedia pages. For OER collection, the most straightforward method is to search for the paper title (exact match) in different search engines. However, this method has two major limitations. First, given a relatively long publication title, if we use exact string match, search engines can hardly find any results. On average, based on our experiment with some random sampled publications, only 0.35 resources were retrieved (for each testing publication). We also found that the quality of resources was not good. A large portion consisted of publisher or digital library access pages for the target publication, which are not helpful for scholars and students to understand the paper. Second, the computational cost of this method is quite high. Given the very large potential publication collection, we needed to send multiple queries for each publication. However, most search engines, i.e., Google and YouTube, restrict the number of automated visits, which makes this method inefficient.

<table>
<thead>
<tr>
<th>Resource type</th>
<th>Search engines</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slides</td>
<td>Google, Slideshare</td>
<td>Google: [Keyword] AND (filetype:ppt OR slides); Slideshare: [Keyword]</td>
</tr>
<tr>
<td>Video</td>
<td>YouTube, TED, Videolecture</td>
<td>[Keyword]</td>
</tr>
<tr>
<td>Source code</td>
<td>GitHub, SourceForge</td>
<td>[Keyword]</td>
</tr>
</tbody>
</table>

In this research, we used a more economical method to cope with this problem. We assumed each publication was composed of a list of topics with each topic represented by an author-assigned keyword. For each resource type, we first aggregated all resources retrieved using the publication keywords (topics) and then used a ranking algorithm to identify the most important resources based on publication content. By using this method, we can significantly reduce the amount of queries sent to different search engines. For instance, take ACM DL corpus as an example, for 41,378 publications, we only need to collect OERs for 9,263 popular author assigned keywords (appear at least 5 times in the corpus). Such resources are highly likely to be relevant to the publication content and could help students or scholars understand the essence of the publication. For this method, we assumed that helping scholars or students understand the topics (keywords) of the paper will eventually help them to understand the paper itself.

In order to effectively generate OERs for each topic, we use meta-search approaches where a Boolean query was sent to one or more search engines for each scientific keyword and OER type. The detailed query list for each resource type is listed in the Table 1. The first column is the OER type. The target query for each type was sent to one or more search engines or web services. For exam-
ple, in order to get the slides OERs for a topic (keyword), the query "[Keyword] AND (filetype:ppt OR slides)" (column 3) was sent to Google (column 2), where [Keyword] was replaced with the target keyword content. Similarly, for video OERs, the query "[Keyword]" was sent to YouTube, TED, and Videolecture. For slides and source code, different queries were sent to different search engines. For performance reasons, we indexed the Wikipedia page dump (2014 May) locally.

In this experiment, we used the top 15 retrieved results from each search engine to aggregate the final result collection for each resource category. Each retrieved result is a triple of title, URL, and snippet, and we use the result URL to download the HTML page content. We then used a list of predefined rules (regular expressions) to identify the key content on the HTML page. For example, for each YouTube or Videolecture video, we extract the video description and tags to represent the content of the video. However, if the predefined rules don’t work for a specific page, we just use the HTML (after removing HTML tags and script code) as the content of the target resource.

In most cases, the result collection was a combination of informative resources and noisy results. Experience in information retrieval reveals that since different search engines return very diverse and erratic results for the same or similar query, irrelevant data may pollute the search results and mislead users. To address this problem, we used an information retrieval ranking algorithm, language model 15 [15, 28], to prioritize those informative resources for the target scientific keywords while filtering out the irrelevant information. In other words, we want to use ranking fusion to find out the most important resources from multiple ranking lists. Because the OERs will be used to help scholars and students better understand the essence of the readings, most retrieval resources (low ranked), at this stage, will be filtered out. We also use the paper abstract as query to search the OER content to find the candidate OERs for a publication. So, for each keyword $K_i$, we will index top $m$ OERs with similarity score $S(K_i, R_j) = P_{LM}(K_i|R_j)$, where $P_{LM}(K_i|R_j)$ is the language model ranking score for keyword probability given a OER, $R_j$, content. Similarly, we also index the publication-OER similarity score, $P_{LM}(P_i|R_j)$ ($P_i$ is a paper, represented by title and abstract content). So, for topic, $K_i$, we index triples $\{K_i, R_j, P_{LM}(K_i|R_j)\}$, and for paper, $P_i$, we index triples $\{P_i, R_j, P_{LM}(P_i|R_j)\}$.

The performance of this method is highly dependent on the quality of the OER text description. However, different kinds of OERs have varying text quality. Wikipedia pages (extracted from Wikipedia dump) have very high text quality, but a large number of YouTube/Videolecture videos and Google/Slideshare slides do not associate with high quality text description. Consequently, a text search may generate some noisy OER data which may pollute the recommendation results. In order to solve this problem, we will use heterogeneous graph mining method to recommend OERs to students (introduced in Section 3.3). The connections between OERs, then, can be important. For instance, Wikipedia pages are interlinked via incoming/outgoing links, and each YouTube, Slideshare, Videolecture, TED, GitHub, and SourceForge OER has "related" OERs. The related resources are provided by the target online service. While the algorithm behind that is a black box, most related resources come from 1. resource content similarity, 2. the ownership of these resources (i.e., if two resources uploaded by the same user), and 3. user co-visitation (or co-watched) probability [3].

We index the relationships between OERs in a graph database, i.e., $R_1 \rightarrow R_2$.

Note that, unlike other meta-search problems, the ranking lists of some OER types in this research were totally disjoint. For example, for videos, we sent queries to YouTube, TED, and Videolecture. Their indexes are almost disjoint (i.e., TED videos may not exist in YouTube video libraries), and some sophisticated metasearch ranking fusion algorithms are not appreciate, e.g., Borda’s method [1] and Markov chain methods [5].

### 3.2 OER-based Collaborative PDF Reader

To validate the new scaffolding hypothesis and help students better understand the scientific publications in a course environment, we design and implement the novel learning and reading system, OER-based Collaborative PDF Reader (OCPR). The new system has three main functionalities:

- **Capture evidence and characterize students’ emerging implicit/explicit information needs when reading a scientific paper.** For instance, students can ask a specific question given a piece of text, which serves as evidence of an explicit information need, or highlight part of a text in the paper, as evidence of an implicit information need. In either case, the OCPR is able to capture the student reading behavior and the reading context from the target PDF document.

- **Automatically recommend high quality OERs, such as videos, slides, source code, or Wikipedia pages, to resolve students’ information needs, while helping them understand a publication more effectively.**

- **Other students (classmates) can find their colleagues’ interaction in the PDF system and actively contribute to this auto-learning process by recommending, changing, annotating, or removing certain resources given an information need.** They can also provide additional comments and suggestions to help others.

The first two functions focus on OER-based scaffolding, and the third one mainly addresses user incentive. For instance, students, when reading the paper, can find her colleagues’ interactions with the system (by clicking the question marks on the side of PDF reading), i.e., the questions or highlights with the target paper. Meanwhile, students could discuss for an existing interaction by using OCPR. We will investigate the third function in the future study, and in this study, we focus on the first two.

At the backend, the Collaborative PDF Reader has access to the OER index (created in last section), the list of assigned class readings, and the algorithm package, which guarantees efficient OER recommendations and different scaffolding methods.

OCPR is designed for course usage, and it can manage class readings and student information. By using the OCPR, the class instructor can easily manage the class reading list by uploading the class syllabus (in a pre-defined format). In this syllabus, the instructor needs to identify the weekly reading schedule and weekly teaching topic; he or she can then upload the readings for each week to the server. For instance, the following is an exemplar OCPR syllabus snippet, which is corresponding a one week teaching schedule. The exemplar snippet has two topics (for teaching), and the algorithm package, which guarantees efficient OER recommendations and different scaffolding methods.

OCPR is designed for course usage, and it can manage class readings and student information. By using the OCPR, the class instructor can easily manage the class reading list by uploading the class syllabus (in a pre-defined format). In this syllabus, the instructor needs to identify the weekly reading schedule and weekly teaching topic; he or she can then upload the readings for each week to the server. For instance, the following is an exemplar OCPR syllabus snippet, which is corresponding a one week teaching schedule. The exemplar snippet has two topics (for teaching), and the algorithm package, which guarantees efficient OER recommendations and different scaffolding methods.

OCPR is designed for course usage, and it can manage class readings and student information. By using the OCPR, the class instructor can easily manage the class reading list by uploading the class syllabus (in a pre-defined format). In this syllabus, the instructor needs to identify the weekly reading schedule and weekly teaching topic; he or she can then upload the readings for each week to the server. For instance, the following is an exemplar OCPR syllabus snippet, which is corresponding a one week teaching schedule. The exemplar snippet has two topics (for teaching), and the algorithm package, which guarantees efficient OER recommendations and different scaffolding methods.
Additionally, the instructor can also upload the student roster to the system, which will automatically register each student while sending student email invitation to download the OCPR reading system. By logging into the reading system, students can access the class weekly readings and use the new scaffolding functions.

The main function of OCPR is OER based scaffolding, which captures evidence of students’ emerging implicit or explicit information needs when reading a scientific paper and recommends high quality OERs to address their information needs. As Figure 1 shows, by using OCPR, students could ask a specific question given a piece of text, which serves as evidence of an explicit information need, or highlight part of a text in the paper, as evidence of an implicit information need. In either case, the OCPR is able to capture the selected or input question and its reading context from a student in the PDF document. The evidence of students’ information needs will then be sent to the backend recommendation and ranking algorithms for information need characterization. For instance, as Figure 1 shows, user highlighted text is focusing on “unsupervised topic modeling for citation analysis”, and the paper content mainly addresses “dynamic topic modeling and PageRank-based citation recommendation”. The recommended Wikipedia pages (on the right) are focusing on those topics.

The algorithms presented in the next section recommend the optimized OERs given the information need, which will be able to help students better understand the essence of the targeted reading. A student can also select the specific type of OERs in the “right click menu”, you may want use “context menu” instead, i.e., Mac is ctrl + click that he or she prefers and the system will capture and log the selected OERs. Meanwhile, students can also provide relevance and usefulness feedback for system recommended OERs. For instance, as Figure 1 shows, students can click “Good”, “OK”, “Bad” or “Not Sure” for each recommended OER given their information need. The judgments and student click information will be saved as system logs, which will be important for OER recommendation algorithms, i.e., training learning to rank model, and algorithm evaluation.

Unlike the common practice of accessing PDF documents locally on one’s personal computer, OCPR gives students and their instructor cloud access on the server, i.e., everyone works on the same document. All users’ asynchronous interactions with the PDFs will be saved in a central database along with their user IDs, and the interactions - i.e., PDF highlighting, proposing questions, recommending OERs, OER judgments to an existing information need - will be useful to characterize computational user information need and to train OER recommendation model.

### 3.3 OER Recommendation and Ranking

As aforementioned, student’s emerging information needs while reading a paper can be very complex. For instance, when a student highlights a paragraph in a paper, it’s challenging to understand the reason and motivation. We employ multiple recommendation features to characterize students’ information need, \( \Phi(f_1(\chi), f_2(\chi), \ldots, f_k(\chi)) \), where each feature, \( f_i \), is a OER ranking function given input space \( \chi \). We propose three OER ranking assumptions: 1) straightforwardly, student’ information needs can be related to the OCPR interactions while reading, i.e., explicit questions and high-light content of the paper; 2) students’ information may be related to the target paper and paper’s associated topics; 3) in a course environment, student’s information need may address the weekly teaching schedule (in the syllabus). Because, even a single paper can cover multiple topics, the weekly topics from the course syllabus can be important. Each assumption will be represented by a number of OER ranking features, and the input space \( \chi \) includes 1) user question or highlight text, 2) paper and paper’s topic, and 3) weekly topic. While we can hardly estimate the importance of each feature, in this paper, we use learning to rank to weight each of those, \( \Phi_{\alpha, \beta, \gamma} \). The training data come from the OCPR collected OER relevance judgments. However, we also need to address the cold start problem. When training data is sparse, we assume information need is a linear function, \( \Phi_{\text{cold}} \), and the online features (assumption 1) should be more important than other offline ones.

From a performance viewpoint, the first assumption is based on online text ranking features. Since it cannot host any complex algorithms, we used text information retrieval algorithm to cope with this problem. The second and third assumptions are offline, and we can pre-index a number of OERs closely related to paper, paper’s topic and syllabus topics with higher computational cost. We use heterogeneous graph mining to implement it.

Figure 2 illustrates the graphical OER recommendation problem, where the OER recommendation is conceptualized as a random walk problem on a heterogeneous graph. Four kinds of vertices, paper, topic, weekly topic and OER, are interconnected by using different kinds of edges. More detailed information is depicted in Table 2.

The links and paths between the vertices can be important to enhance the OER recommendation performance. As section 3.1 shows, we use meta-search and crawler to index the OER by using keyword search. However, this method generates noisy data which may pollute the recommendation results. For instance, given the keyword (topic), “Question Answering”, the system may find a Wikipedia page or YouTube video “Question the Answers” (content relevant), which is a pop song. Simply using content match, this noisy resource may pollute the OER recommendation result. However, from a random walk viewpoint, via meta-path \( \Phi^* 
\rightarrow R \leftarrow R' \), the OER, \( R' \), useful probability equals the random walk probability from the seed (query) topic \( K \) to the candidate OER \( R' \) (via all the related interim OERs). Given the earlier example, even though “Question the Answers” is related to the keyword with text match, but, all its related resources are about the related music and artists (not related to the topic), and, then, the random walk probability from the seed topic to this noisy OER will be very low. Comparing with that, the correct Wikipedia page’s interlinked pages are more likely to be relevant to the target topic or paper, and the random walk probability (following this meta-path) can be high. Similarly, because a large number of YouTube videos or Slideshare slides do not associate with high quality textual descriptions, we cannot totally trust the video/slide-topical or video/slide-paper similarly, and the relationship between videos/slides can be important. For instance, if two YouTube videos’ co-watched probability is high, they can be related [3], and this information may not totally depend on the video text description. For this experiment, we collected video, slide or source code relationships by using YouTube, Videolecture, GitHub or Slideshare API. We find, in most cases, useful OER’s related OERs can be also relevant to the target topic/paper, and noisy video’s related ones are not likely to be relevant to the target keyword. Random walk can be important to filter those noisy OERs.

Explained it in another way, from the random walk perspective, user information needs are represented by different kinds of seed
Similarly, we also generate the edges calculated by using language model with Dirichlet smoothing [28]. The weight of $K \rightarrow_P P$ is the normalized topic authority score, which characterize the importance (or contribution) of each paper. Note that topic $P_i$ is contributed by paper $P_j$. Every paper $P_j$ is linked to the target topic $P_i$, and all the readings from that week connected to the target topic(s), $P \rightarrow W$. 

We can also consider two or multiple parallel meta-paths leading to the same type of query nodes, for example, $A_1 \rightarrow_R A_2 \rightarrow_R A_3$.
... $R_{i} \to A_{i+1}$ and $(A_{1}) \to R_{1} \to (A_{2}) \to R_{2} \to \ldots \to R_{i} \to (A_{i+1})$, where $A_{i+1} = A'_{i+1}$. In this case, we can define similarity from different sets of objects to a result node from different meta-paths:

$$s(Q_{1} \cup Q_{2}, r) = sMP_{1}(Q_{1}, r)^{\alpha} \cdot sMP_{2}(Q_{2}, r)^{1-\alpha}$$

For instance, $K^* \rightarrow R \overset{t}{\leftarrow} R^* \overset{r}{\leftarrow} K^*$ is a combined restricted meta-path, which combines two meta-paths: $K^* \rightarrow R \overset{t}{\leftarrow} R^*$ and $K^* \overset{r}{\leftarrow} R^*$. So, based on the above formula, the ranking score of candidate OER, $R^*$, is the combination of two random walk scores for both sub-meta-paths. Theoretically, we need to tune parameter $\alpha$ for each meta-path to optimize the weight of each sub-meta-path. For this study, because of the sparseness of the training data, we set $\alpha = 0.5$. More sophisticated parameter tuning will be addressed in future work.

Based on text mining methods and meta-path based methods, we proposed 21 ranking features for OER recommendation. All the features investigated in this study are listed in Table 3. Please note that, we don’t consider the edge direction, as long as the two vertices are connected, the meta-path can be created. There are three types of features: online feature, semi-online feature, offline feature. Online features can only be calculated online, offline features are pre-calculated, while semi-online features combine online calculation and offline calculation results.

After we calculate all the ranking features, we need to integrate them as a OER recommendation mode, $\Phi(f_{1}(\chi), f_{2}(\chi), \ldots, f_{k}(\chi))$. In this study, we use two methods. First, without user judgments, we generate $\Phi_{odd}(f_{1}(\chi), f_{2}(\chi), \ldots, f_{k}(\chi))$ based on a low-cost linear function (cold start). We assume the online or semi-online features (No. 1-4), based on user OCPR interactions, are more important than offline features. So, the online features’ weights are 3 times higher than the offline features.

When OER usefulness judgments are available (collected by OCPR system), we use learning to rank to statistically combine different ranking features, $\Phi_{L2R}(f_{1}(\chi), f_{2}(\chi), \ldots, f_{k}(\chi))$, while avoiding manual parameter tuning. As this study is not focusing on learning to rank, we used a relative simple algorithm, Coordinate Ascent [12], which iteratively optimizes a multivariate objective ranking function, for OER ranking feature integration and algorithm evaluation.

4. EXPERIMENT

In this section, we describe the experimental setting, results and exit survey.

4.1 Dataset

We tested this reading system and the associated OER ranking algorithms in a real learning environment. A graduate-level information retrieval course at Indiana University is used for this experiment. 51 students (masters and PhDs) voluntarily participated this experiment (33 male and 18 female), and they were required to use the OCPR system for 8 weeks (with 8 required readings) 3. They could use OCPR functions to get access to the system recommended OERs. Meanwhile, we asked each participant to provide OER relevance judgments for the top 5 system-recommended OERs (they can rate more). There are totally valid 5,132 judgments we collected (for 1,051 student requests). We use those judgments to train the learning to rank model and to evaluate the algorithm performance. For all the requests, there are 77 explicit questions from students (each question is also associated with a piece of highlight text). On average, each paper got 9.36 questions, and the average character length for a question is 42.25 (with highlight character length 64.08). Based on students’ feedback, averagely, they use the system about 3 hours per week when reading the paper.

At the backend of OCPR, we created a heterogeneous graph for OER recommendation. As aforementioned, there are 4 kinds of vertices and 10 kinds of relations in this heterogeneous graph, for paper vertices, we used 41,370 publications from 1,553 venues (mainly from the ACM digital library). The paper vertices are connected to 9,263 keyword labeled topics. For this course, based on course syllabus, there are a total of 34 weekly topics. By using meta-search, we collected a total of 1,112,718 OERs. More detailed information can be found in Table 4. Note that even though we only used a small number of readings in this experiment, the keyword co-occurs, keyword-paper contribution, and citation relationships on the large graph will help to enhance the OER recommendation performance. Theoretical, user could use OCPR to read any paper in this collection.

<table>
<thead>
<tr>
<th>Type</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper</td>
<td>41,378</td>
</tr>
<tr>
<td>Keyword</td>
<td>9,263</td>
</tr>
<tr>
<td>Weekly Topic</td>
<td>34</td>
</tr>
<tr>
<td>Open Education Resource(wiki)</td>
<td>128,772</td>
</tr>
<tr>
<td>Open Education Resource(sourcecode)</td>
<td>45,775</td>
</tr>
<tr>
<td>Open Education Resource(slides)</td>
<td>151,616</td>
</tr>
<tr>
<td>Open Education Resource(video)</td>
<td>786,555</td>
</tr>
<tr>
<td>Open Education Resource(total)</td>
<td>1,112,718</td>
</tr>
</tbody>
</table>

4.2 Experiment results

For all the OER judgments, participants rated 21.63% of the recommended OERs as “Good”, 29.36% as “OK”, 40.10% as “Bad” and 8.9% as “Not Sure”. For this experiment, we used the cold start solution to recommend OERs to students. From an NDCG viewpoint, we score $\text{Good} = 2, \text{OK} = 1, \text{Bad} = 0$. The OER recommendation performance can be found in Table 5. As the OER recommendation is more like a QA problem, and students are more interested to find the first useful resource, we use MRR as the indicator to train the learning to rank model. For evaluation, 10-folder cross-validation was used.

The ranking feature 3, 14, and 17 are the most important features (highest feature weights in Coordinate Ascent model). Ranking feature 14 and ranking feature 17 were offline feature. Feature 14 ($K^* \rightarrow K^* \rightarrow R \overset{t}{\leftarrow} R^* \overset{r}{\leftarrow} K^*$) carried the keyword citation, keyword-OER similarity, keyword-OER related and OER-

<table>
<thead>
<tr>
<th>Type</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>paper has keywords</td>
<td>587,252</td>
</tr>
<tr>
<td>paper is assigned to weekly topic (for 8 required readings)</td>
<td>34</td>
</tr>
<tr>
<td>paper-wiki relationship based on similarity</td>
<td>8,068,471</td>
</tr>
<tr>
<td>paper-slides relationship based on similarity</td>
<td>8,039,382</td>
</tr>
<tr>
<td>paper-sourcecode relationship based on similarity</td>
<td>7,973,611</td>
</tr>
<tr>
<td>paper-video relationship based on similarity</td>
<td>8,112,065</td>
</tr>
<tr>
<td>paper-OER relationship based on similarity (total)</td>
<td>32,193,309</td>
</tr>
<tr>
<td>keyword cites keyword</td>
<td>206,322</td>
</tr>
<tr>
<td>keyword-keyword co-occurrence</td>
<td>1,450,910</td>
</tr>
<tr>
<td>keyword is contributed by paper</td>
<td>3,577,111</td>
</tr>
<tr>
<td>keyword-wiki relationship based on similarity</td>
<td>2,796,341</td>
</tr>
<tr>
<td>keyword-slides relationship based on similarity</td>
<td>2,684,119</td>
</tr>
<tr>
<td>keyword-sourcecode relationship based on similarity</td>
<td>557,436</td>
</tr>
<tr>
<td>keyword-video relationship based on similarity</td>
<td>2,094,405</td>
</tr>
<tr>
<td>keyword-OER relationship based on similarity (total)</td>
<td>7,532,291</td>
</tr>
<tr>
<td>weekly topic-weekly topic co-occurrence</td>
<td>114</td>
</tr>
<tr>
<td>weekly topic-OER relationship based on similarity</td>
<td>68,000</td>
</tr>
<tr>
<td>OER is related to OER</td>
<td>1,983,866</td>
</tr>
</tbody>
</table>

Table 4: Vertices and edges in the heterogeneous graph
Table 3: Measures of different OER ranking algorithms (significant test: 1. cold start vs. other features; 2. L2R vs. cold start; * p < 0.01, ** p < 0.001, *** p < 0.0001)

<table>
<thead>
<tr>
<th>Ranking</th>
<th>MAP@3</th>
<th>MAP@5</th>
<th>MAP@all</th>
<th>NDCG@3</th>
<th>NDCG@5</th>
<th>NDCG@all</th>
<th>P@1</th>
<th>P@3</th>
<th>P@5</th>
<th>MRR(O)</th>
<th>MRR(Good)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature1</td>
<td>0.6375</td>
<td>0.8146</td>
<td>0.8222</td>
<td>0.7904</td>
<td>0.8093</td>
<td>0.8140</td>
<td>0.7412</td>
<td>0.6000</td>
<td>0.8257</td>
<td>0.7872</td>
<td></td>
</tr>
<tr>
<td>Feature2</td>
<td>0.5317</td>
<td>0.7222</td>
<td>0.7306</td>
<td>0.6430</td>
<td>0.7321</td>
<td>0.7379</td>
<td>0.5665</td>
<td>0.4890</td>
<td>0.6992</td>
<td>0.6249</td>
<td></td>
</tr>
<tr>
<td>Feature3</td>
<td>0.3014</td>
<td>0.4894</td>
<td>0.4974</td>
<td>0.4228</td>
<td>0.5480</td>
<td>0.5597</td>
<td>0.6082</td>
<td>0.5266</td>
<td>0.5885</td>
<td>0.5362</td>
<td></td>
</tr>
<tr>
<td>Feature4</td>
<td>0.2970</td>
<td>0.4860</td>
<td>0.4949</td>
<td>0.4323</td>
<td>0.5415</td>
<td>0.5474</td>
<td>0.6262</td>
<td>0.6133</td>
<td>0.2838</td>
<td>0.2678</td>
<td></td>
</tr>
<tr>
<td>Feature5</td>
<td>0.4586</td>
<td>0.6372</td>
<td>0.6452</td>
<td>0.5756</td>
<td>0.6597</td>
<td>0.6680</td>
<td>0.4812</td>
<td>0.3871</td>
<td>0.5524</td>
<td>0.7070</td>
<td></td>
</tr>
<tr>
<td>Feature6</td>
<td>0.6436</td>
<td>0.8188</td>
<td>0.8258</td>
<td>0.7446</td>
<td>0.8045</td>
<td>0.8076</td>
<td>0.7635</td>
<td>0.6322</td>
<td>0.8393</td>
<td>0.7085</td>
<td></td>
</tr>
<tr>
<td>Feature7</td>
<td>0.6598</td>
<td>0.8123</td>
<td>0.8188</td>
<td>0.7421</td>
<td>0.8013</td>
<td>0.8042</td>
<td>0.7624</td>
<td>0.6086</td>
<td>0.8315</td>
<td>0.7320</td>
<td></td>
</tr>
<tr>
<td>Feature8</td>
<td>0.5556</td>
<td>0.7322</td>
<td>0.7382</td>
<td>0.6602</td>
<td>0.7358</td>
<td>0.7386</td>
<td>0.6329</td>
<td>0.4980</td>
<td>0.7384</td>
<td>0.5853</td>
<td></td>
</tr>
<tr>
<td>Feature9</td>
<td>0.5827</td>
<td>0.7545</td>
<td>0.7615</td>
<td>0.6913</td>
<td>0.7601</td>
<td>0.7633</td>
<td>0.6847</td>
<td>0.5196</td>
<td>0.7775</td>
<td>0.6427</td>
<td></td>
</tr>
<tr>
<td>Feature10</td>
<td>0.6196</td>
<td>0.7926</td>
<td>0.7994</td>
<td>0.7144</td>
<td>0.7814</td>
<td>0.7839</td>
<td>0.7318</td>
<td>0.5965</td>
<td>0.8227</td>
<td>0.6493</td>
<td></td>
</tr>
<tr>
<td>Feature11</td>
<td>0.376</td>
<td>0.5625</td>
<td>0.5705</td>
<td>0.4942</td>
<td>0.6041</td>
<td>0.6093</td>
<td>0.2459</td>
<td>0.2294</td>
<td>0.4307</td>
<td>0.3819</td>
<td></td>
</tr>
<tr>
<td>Feature12</td>
<td>0.3378</td>
<td>0.5651</td>
<td>0.5732</td>
<td>0.4906</td>
<td>0.6067</td>
<td>0.6120</td>
<td>0.2518</td>
<td>0.2322</td>
<td>0.4355</td>
<td>0.3811</td>
<td></td>
</tr>
<tr>
<td>Feature13</td>
<td>0.5926</td>
<td>0.7682</td>
<td>0.7739</td>
<td>0.6640</td>
<td>0.7648</td>
<td>0.7663</td>
<td>0.6871</td>
<td>0.5604</td>
<td>0.7871</td>
<td>0.6307</td>
<td></td>
</tr>
<tr>
<td>Feature14</td>
<td>0.3577</td>
<td>0.5461</td>
<td>0.5542</td>
<td>0.4801</td>
<td>0.5396</td>
<td>0.5369</td>
<td>0.2118</td>
<td>0.2035</td>
<td>0.4013</td>
<td>0.3535</td>
<td></td>
</tr>
<tr>
<td>Feature15</td>
<td>0.3591</td>
<td>0.5474</td>
<td>0.5555</td>
<td>0.4809</td>
<td>0.5392</td>
<td>0.5377</td>
<td>0.2141</td>
<td>0.2059</td>
<td>0.4027</td>
<td>0.3543</td>
<td></td>
</tr>
<tr>
<td>Feature16</td>
<td>0.3495</td>
<td>0.5376</td>
<td>0.5459</td>
<td>0.4748</td>
<td>0.5361</td>
<td>0.5391</td>
<td>0.2071</td>
<td>0.1933</td>
<td>0.3986</td>
<td>0.3541</td>
<td></td>
</tr>
<tr>
<td>Feature17</td>
<td>0.3536</td>
<td>0.5454</td>
<td>0.5531</td>
<td>0.4768</td>
<td>0.5403</td>
<td>0.5396</td>
<td>0.2071</td>
<td>0.2024</td>
<td>0.3976</td>
<td>0.3479</td>
<td></td>
</tr>
<tr>
<td>Feature18</td>
<td>0.6089</td>
<td>0.7787</td>
<td>0.7848</td>
<td>0.7069</td>
<td>0.7708</td>
<td>0.7733</td>
<td>0.7047</td>
<td>0.5729</td>
<td>0.8010</td>
<td>0.6414</td>
<td></td>
</tr>
<tr>
<td>Feature19</td>
<td>0.6080</td>
<td>0.7772</td>
<td>0.7834</td>
<td>0.7153</td>
<td>0.7764</td>
<td>0.7791</td>
<td>0.7212</td>
<td>0.5533</td>
<td>0.8038</td>
<td>0.6729</td>
<td></td>
</tr>
<tr>
<td>Feature20</td>
<td>0.5937</td>
<td>0.7657</td>
<td>0.7714</td>
<td>0.6996</td>
<td>0.7628</td>
<td>0.7653</td>
<td>0.6894</td>
<td>0.5475</td>
<td>0.8666</td>
<td>0.6379</td>
<td></td>
</tr>
<tr>
<td>Feature21</td>
<td>0.5997</td>
<td>0.7694</td>
<td>0.7754</td>
<td>0.7045</td>
<td>0.7694</td>
<td>0.7777</td>
<td>0.7047</td>
<td>0.5388</td>
<td>0.7907</td>
<td>0.6825</td>
<td></td>
</tr>
<tr>
<td>Cold Start</td>
<td>0.7114***</td>
<td>0.8585***</td>
<td>0.8946***</td>
<td>0.8079***</td>
<td>0.8559***</td>
<td>0.8511***</td>
<td>0.8600***</td>
<td>0.7208***</td>
<td>0.9140***</td>
<td>0.8179***</td>
<td></td>
</tr>
<tr>
<td>L2R</td>
<td>0.7409***</td>
<td>0.9090***</td>
<td>0.9161**</td>
<td>0.8426***</td>
<td>0.8789***</td>
<td>0.8811***</td>
<td>0.8871*</td>
<td>0.7373***</td>
<td>0.5329*</td>
<td>0.8609***</td>
<td></td>
</tr>
</tbody>
</table>

OER related information. Similarly, feature 17 ($K^{cont} \rightarrow P \rightarrow R \rightarrow R^* \rightarrow K^*$) carries keyword-paper contribution, paper-OER similarity, keyword-OER similarity and OER-OER related information, in which, the keyword-paper contribution information is also calculated from citation information and key-paper related information (via LLDA). Question-OER (feature 3) is an online feature, and this indicates the question content from the student can improve the recommendation performance. From a performance viewpoint, evaluation results show that, first, auto-recommended OERs can provide important and useful information to help stu-
students better understand the content of the scientific publications, i.e., both learning to rank and cold start methods’ MRR (Good) score is higher than 0.8 (which means students are finding the useful OERs in the 1st or 2nd position in the result ranking list). In this table, MRR (OK) investigated the multiplicative inverse of the rank of the first OER user rated “OK”, and MRR (Good) used first user rated “Good” OER in the ranking list. This finding is also confirmed in the exit survey where 78.42% of participants believe the OCPR system along with recommended OERs can provide precise and useful information.

Second, based on the student judgments, we found that a number of OER recommendation features, including online and offline, can be potentially useful, and the learning to rank method can be useful to enhance the recommendation performance comparing with cold start model ($p < 0.01$). When training instances are sparse or not available, manual parameter setup (cold start) can be an effective solution. Cold start significantly outperforms other ranking features ($p < 0.0001$), except for MRR(Good) ($p = 0.014$). From a ranking perspective, both MAP and NDCG proved that the cold start solution and learning to rank can be efficient for OER recommendation. Third, we find that different kinds of seed vertices can be useful. Weekly topic, $W^*$, is especially useful. For instance, features 18-21 performed well in this experiment, which means class teaching schedule can be important in helping the system better understand student information needs. The evaluation results prove that cold start solution and learning to rank significantly outperform the classic text-based or graph-based approaches.

From performance viewpoint, as the online or semi-online features are implemented via language model and greedy match and the offline features are pre-computed, the system performance is acceptable. In most cases, after students highlight text or ask a question, system can give students OER results within 5 seconds.

4.3 Exit Survey

From a usability perspective, we used an exit survey to investigate the usefulness of the new reading environment and the effectiveness of the new OER-based scaffolding method. In the survey, we asked 13 questions for each participant. More detailed information can be found in the project website. Based on students’ feedback, 78.42% of participants believe the OCPR system along with recommended OERs can provide precise and useful information, while 78.43% believe the recommended OERs are close to what they need when reading the target paper. From a system perspective, 80.39% of participants believe the OCPR (prototype) system, i.e., highlight and ask question function, is easy to use.

Meanwhile, based on students’ feedback, we found Wikipedia page and video presentation (or video lecture) are more helpful for them to understand the publication, compared with slides and source code. In this experiment, very few students actually read the source code to understand the paper content. Overall, 49.02% of students found the system can effectively help them better understand the target publication, while 33.33% found the system and OERs somewhat helpful. Only 1.96% of the participants described the recommended OERs as useless.

5. ANALYSIS AND CONCLUSION

In this study, we propose a novel task to assist students to better understand scientific publications by leveraging high-quality OERs. By using the OCPR system, a number of reading behaviors are captured to characterize students’ information needs. Text and graph mining algorithms are used as ranking features to recommend high-quality OERs. Experiment results show that both text mining and graph mining algorithms can potentially contribute to the OER recommendation. Meanwhile, different seed vertices on the heterogeneous graph, i.e., paper, topic, weekly topic, highlights, and questions, are important to characterize student information need. Learning to rank is efficient to integrate different OER ranking features, when training data is available. However, when training data is sparse or unavailable, human generated ranking rules can be useful to cope with the cold start problem. We also find heterogeneous graph mining can be useful to solve this proposed problem. When OER text quality is not high (e.g. some YouTube videos have low quality descriptions), relations between the OERs become very important to filter out the noisy resources. For instance, we find the relations between OERs and keyword relations are useful to enhance the OER recommendation performance which is difficult for classical text mining methods.

From statistical analysis of student survey responses, we found only a handful of them like to propose an explicit question when using the OCPR system: only 77 questions out of the 1,051 requests are found to be explicit, although proposing explicit questions can help them locate the useful OERs. Instead, most students prefer to highlight some text and ask for system help. Evaluation shows that highlighted text provides important information to find the correct OER (features 1 and 2). However, noisy OER can pollute the quality of the recommendation result. In this case, graph mining methods help to enhance the OER ranking accuracy as well as filtering out the noisy OERs. For instance, features 13, 18, 20, and 21 based on paper topic or weekly topic achieve good recommendation and ranking performance.

One major limitation of this study is the lack of personalization. Intuitively, different students may have different kinds of information needs even though they pose the same question or select the same part of a text in a paper. For example, given the same piece of text in a target paper that students want to understand, some students may prefer to watch a video, while others may want to read a Wikipedia page or access the source code of the scaffolding. In order to tailor the recommendation results for each student, we need to generate a computational profile for each of them. However, we will also need to address the challenge of data sparseness problem for personalizing model training.

We also hope to have a chance to use the user comment function to enhance the OER ranking and scaffolding performance in the future. Supported by OCPR, other students (classmates) and instructors can actively contribute to the auto-learning process by recommending, changing, annotating, or removing certain resources given an information need. Users can easily click the question marks on the side of PDF reading for viewing or commenting on their colleagues’ interactions with the system, which provide important incentives for using the OER-scaffolding functions. In the future, we will enhance the OER recommendation by using this function and motivate students participation.

Finally, this system can serve other purposes in addition to student learning. For instance, by using OCPR, a system can help journal readers better understand the scientific content within the articles. Different experiments and ranking features need to be designed to achieve this goal.

6. REFERENCES


[22] Y. Sun, B. Norick, J. Han, X. Yan, P. S. Yu, and X. Yu. Integrating meta-path selection with user guided object clustering in heterogeneous information networks. In Proc. of 2012 ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD’12), Beijing, China, 2012.


[27] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. Personalized entity recommendation: A heterogeneous information network approach. In Proc. 2014 ACM Int. Conf. on Web Search and Data Mining (WSDM’14), New York, 2014.