Pre-print of
Joeran Beel, Stefan Langer, Bela Gipp, and Andreas Nürnberger. 2014. The Architecture and Datasets of Docear's Research Paper Recommender System. In *Proceedings of the 3rd International Workshop on Mining Scientific Publications (WOSP 2014) at the ACM/IEEE Joint Conference on Digital Libraries (JCDL 2014)*. Downloaded from http://www.docear.org.

The Architecture and Datasets of Docear's Research Paper Recommender System

Joeran Beel
Otto-von-Guericke University
Dept. of Computer Science
Magdeburg
Germany
beel@ovgu.org

Stefan Langer
Docear
Magdeburg
Germany
langer@docear.org

Bela Gipp
University of California
Berkeley, USA
& National Institute of
Informatics, Tokyo, Japan
gipp@nii.ac.jp

Andreas Nürnberger
Otto-von-Guericke University
Dept. of Computer Science
Magdeburg
Germany
andreas.nuernberger@ovgu.de

ABSTRACT
In the past few years, we have developed a research paper recommender system for our reference management software *Docear*. In this paper, we introduce the architecture of the recommender system and four datasets. The architecture comprises of multiple components, e.g. for crawling PDFs, generating user models, and calculating content-based recommendations. It supports researchers and developers in building their own research paper recommender systems, and is, to the best of our knowledge, the most comprehensive architecture that has been released in this field. The four datasets contain metadata of 9.4 million academic articles, including 1.8 million articles publicly available on the Web; the articles' citation network; anonymized information on 8,059 Docear users; information about the users' 52,202 mind-maps and personal libraries; and details on the 308,146 recommendations that the recommender system delivered. The datasets are a unique source of information to enable, for instance, research on collaborative filtering, content-based filtering, and the use of reference management and mind-mapping software.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *information filtering.*

General Terms
Algorithms, Design, Experimentation

Keywords
Dataset, recommender system, mind-map, reference manager, framework, architecture

1. INTRODUCTION
Researchers and developers in the field of recommender systems can benefit from publicly available architectures and datasets[1]. *Architectures* help with the understanding and building of recommender systems, and are available in various recommendation domains such as e-commerce [1], marketing [2], and engineering [3]. *Datasets* empower the evaluation of recommender systems by enabling that researchers evaluate their systems with the same data. Datasets are available in several recommendation domains, including movies[2], music[3], and baby names[4].

In this paper, we present the architecture of *Docear*'s research paper recommender system. In addition, we present four datasets containing information about a large corpus of research articles, and Docear's users, their mind-maps, and the recommendations they received. By publishing the recommender system's architecture and datasets, we pursue three goals.

First, we want researchers to be able to understand, validate, and reproduce our research on Docear's recommender system. In our previous papers, we could often not go into detail of the recommender system due to spacial restrictions. This paper gives the information on Docear's recommender system that is necessary to allow the re-implementation of our approaches and to reproduce our findings.

Second, we want to support researchers when building their own research paper recommender systems. Docear's architecture and datasets ease the process of designing one's own system, estimating the required development times, determining the required hardware resources to run the system, and crawling full-text papers to use as recommendation candidates.

Third, we want to provide real-world data to researchers who have no access to such data. This is of particular importance, since the majority of researchers in the field of research paper recommender systems have no access to real-life recommender systems [4]. Our datasets allow analyses beyond the analyses we have already published [5–8]. For instance, our datasets might be used to evaluate collaborative filtering algorithms, perform citation analysis, or explore the use of reference managers.

This paper will present related work, provide a general overview of Docear and its recommender system, introduce the architecture, and present the datasets.

2. RELATED WORK
Several academic services published datasets, and hence have eased the process of researching and developing research paper recommender systems. *CiteULike*[5] and *Bibsonomy*[6] published

[1] Recommendation frameworks such as *LensKit or Mahout* may also be helpful for researchers and developers, but such frameworks are not the subject of this paper.

[2] http://grouplens.org/datasets/movielens/
[3] http://labrosa.ee.columbia.edu/millionsong/
[4] http://www.kde.cs.uni-kassel.de/ws/dc13/
[5] http://www.citeulike.org/faq/data.adp
[6] https://www.kde.cs.uni-kassel.de/bibsonomy/dumps/

datasets containing the social tags that their users added to research articles. The datasets were not originally intended for recommender system research but are frequently used for this purpose [9–11]. *CiteSeer* made its corpus of research papers public[7], as well as the citation graph of the articles, data for author name disambiguation, and the co-author network [12]. CiteSeer's dataset has been frequently used by researchers for evaluating research paper recommender systems [4]. Kris Jack et al. compiled a dataset based on the reference management software *Mendeley* [13]. The dataset includes 50,000 randomly selected personal libraries from 1.5 million users. These 50,000 libraries contain 4.4 million articles with 3.6 million of them being unique. For privacy reasons, Jack et al. only publish unique IDs of the articles and no title or author names. In addition, only those libraries having at least 20 articles were included in the dataset. Sugiyama and Kan released two small datasets[8], which they created for their academic recommender system [14]. The datasets include some research papers, and the interests of 50 researchers. The CORE project released a dataset[9] with enriched metadata and full-texts of academic articles, and that could be helpful in building a recommendation candidate corpus.

Architectures of research paper recommender systems have only been published by a few authors. The developers of the academic search engine *CiteSeer(x)* published an architecture that focused on crawling and searching academic PDFs [15], [16]. This architecture has some relevance for recommender systems since many task in academic search are related to recommender systems (e.g. crawling and indexing PDFs, and matching user models or search-queries with research papers). Bollen and van de Sompel published an architecture that later served as the foundation for the research paper recommender system *bX* [17]. This architecture focuses on recording, processing, and exchanging scholarly usage data. The developers of *BibTiP* [18] also published an architecture that is similar to the architecture of bX (both bX and BibTip utilize usage data to generate recommendations).

3. DOCEAR AND ITS RECOMMENDER SYSTEM

Docear is an open source literature suite for organizing references and PDFs, including the PDFs annotations. Docear is available for Windows, Mac OS, and Linux and offers a recommender system for publicly available research papers on the Web. In contrast to most other reference managers, Docear uses mind-maps. Figure 1 shows a screenshot depicting the management of PDFs in Docear, including annotations. A user can create several categories (e.g. "Academic Search Engines") and sub-categories (e.g. "Google Scholar"). Each category contains a number of PDFs, and for each PDF, its annotations that are made by the user – e.g. highlighted text, comments, and bookmarks – are displayed. If the cursor is moved over a PDF or annotation, bibliographic data such as the title and authors, is shown.

For the remainder of this paper, it is important to note that each element in the mind-map – i.e. each category, PDF, or annotation – is called a "node". Each node has some descriptive text (e.g. the category label or PDF's file name), an option to a link to a file or web page (a click on the node opens the linked file or web page), and some further attributes such as the bibliographic data. For each node, the dates when the node was created, modified, and moved are stored. If a node links to a PDF, the PDF's title is extracted and stored as an invisible attribute to the node. A "circle" on a node indicates that the node has child nodes but that they are currently hidden in the mind-map.

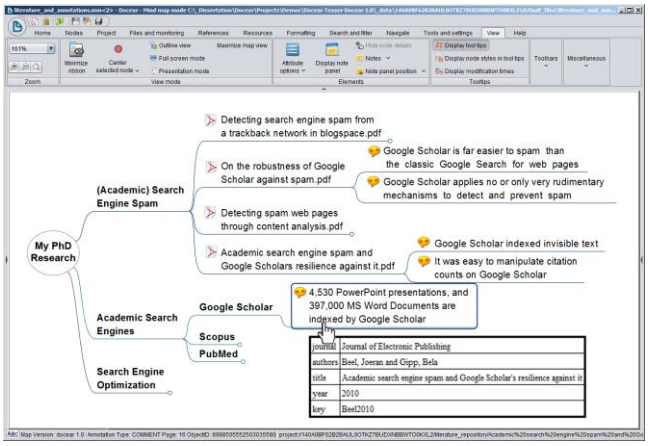

Figure 1: Screenshot of Docear

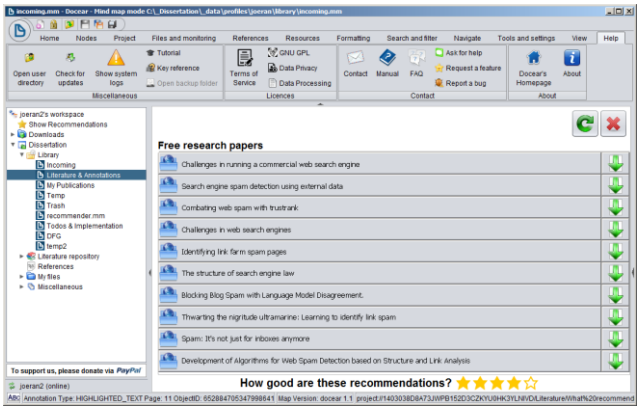

Figure 2: User-interface of Docear's recommender system

Every five days, recommendations are displayed to the users at the start-up of Docear. In addition, users may explicitly request recommendations at any time. Docear displays recommendations as a set of ten research papers, as long as ten papers are available to recommend, otherwise less recommendations are shown (Figure 2). A click on a recommendation opens the PDF in the user's web browser. Users can rate each recommendation set on a scale of one to five. The rating is then used to evaluate the effectiveness of different recommendation algorithms.

4. ARCHITECTURE

Docear itself is a JAVA desktop software with its source code hosted on *GitHub*[10]. The recommender system is also primarily written in JAVA and runs on our web servers. The source code is not (yet) publicly available. To enable communication between the Desktop software and the servers, we implemented a RESTful Web Service. Figure 3 illustrates the architecture and the particular components, which are explained in detail in the following sections, along with technical details.

[7] http://csxstatic.ist.psu.edu/about/data
[8] http://www.comp.nus.edu.sg/~sugiyama/SchPaperRecData.html
[9] http://core.kmi.open.ac.uk/intro/data_dumps

[10] https://github.com/Docear/

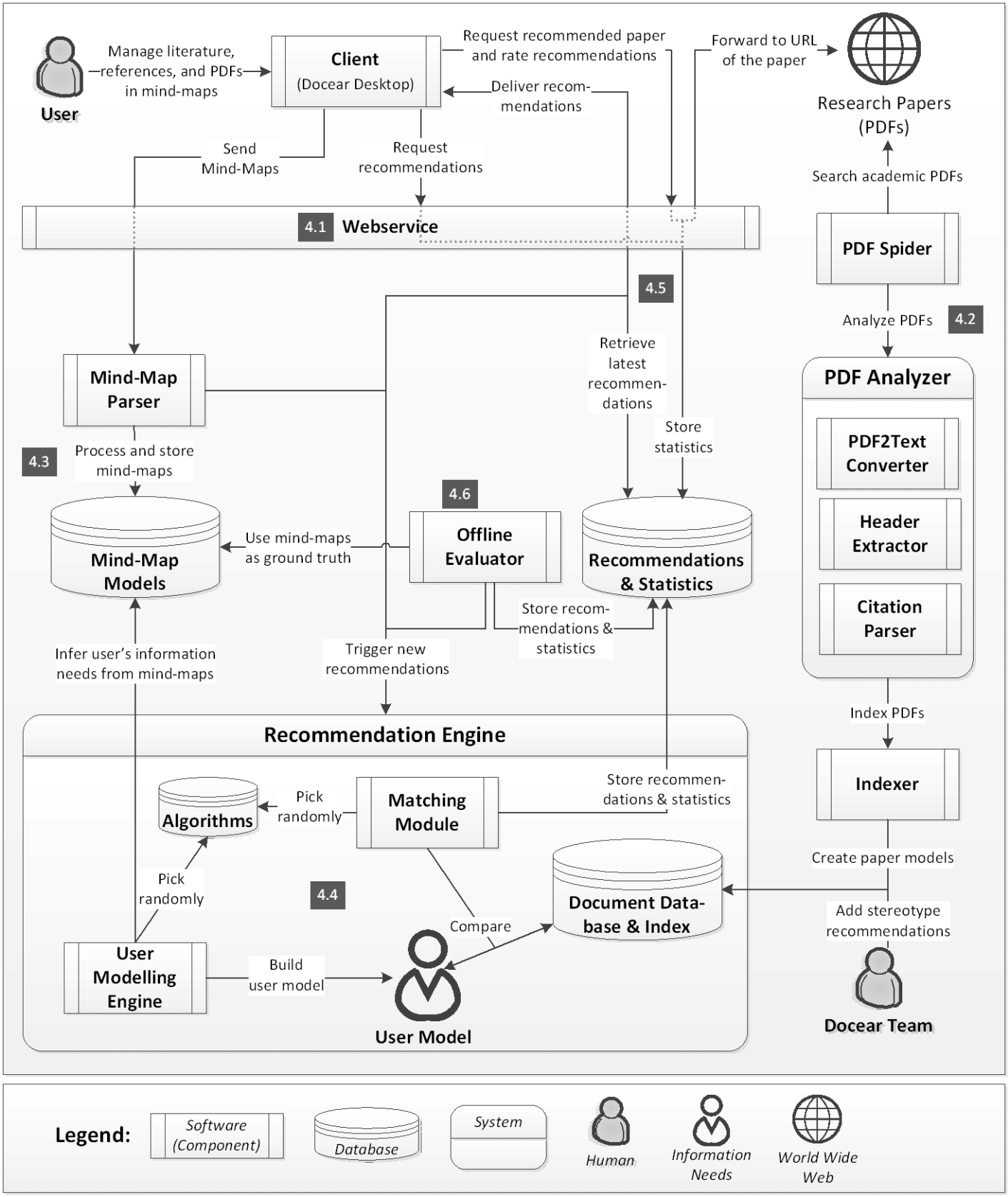

Figure 3: Architecture of Docear's research paper recommender system

4.1 Web Service / API

Docear's RESTful Web Service (*Jersey*)[11] is responsible for several tasks, including user registration and delivering recommendations. In Table 1, the most important methods relating to recommendations are listed. Third parties could use the Web Service, for instance, to request recommendations for a particular Docear user and to use the recommendations in their own application (if the third party knew the user's username and password). However, it should be noted that, for now, we developed the Web Service only for internal use, that there is no documentation available, and that the URLs might change without prior notification.

Table 1: POST and GET requests for Docear's web service

Task	URL	Type
Upload a mind-map	*https://api.docear.org/user/{username}/mindmaps/*	POST
Request recommendations	*https://api.docear.org/user/{username}/recommendations/*	GET
Confirm the receipt of recommendations	*https://api.docear.org/user/{username}/recommendations/{recommendationsSetId}*	POST
Download a recommended paper	*https://api.docear.org/user/{username}/recommendations/fulltext/{hash}*	GET
Send rating	*https://api.docear.org/user/{username}/recommendations/{recommendationsSetId}*	POST

4.2 Building the corpus

The *Spider* crawls the Web for academic PDF files, which serve as recommendation candidates. Each PDF is converted into text, and the header information and citations are extracted. The text conversion is done with *jPod*[12], a PDF library we found to be more effective than the commonly used *PDFBox*. The header extraction is done with *ParsCit*[13] and *Docear's PDF Inspector* [19]. The citation extraction is also conducted with ParsCit, which we modified to identify the citation position within a text[14]. Once all information is extracted, it is indexed with *Apache Lucene*[15] and stored in Lucene's file-based data storage.

Instead of indexing the original citation placeholder with *[1]*, *[2]*, etc. the unique Docear ID of the cited document is indexed (e.g. *dcr_doc_id_54421*) (Figure 4). This allows to apply weighting schemes, such as TF-IDF to citations, i.e. CC-IDF [15], and searching with Lucene for documents that cite a certain paper. It also allows for the matching of user models and recommendation candidates based on terms *and* citations at the same time. For instance, a user model could consist of the terms and citation "*cancer, sun, dcr_doc_id_54421, skin"* and those papers would be recommended that contain the terms *cancer*, *sun* and *skin* and that cite the document *dcr_doc_id_54421*.

In addition to the papers that were found by the Spider, we selected a few papers manually and added them to the corpus of recommendation candidates. These papers are about academic writing and search, i.e. topics we assume to be relevant for the majority of our users. These papers are recommended with the stereotype approach, which is later explained in detail.

4.3 Collecting information about users

Docear's recommender system needs access to the users' data, i.e. their mind-maps, to be able to infer the users' information needs. To get access to the users' mind-maps, Docear stores a copy of the mind-maps in a temporary folder on the users' hard drive, whenever a mind-map was modified and saved by the user. Every five minutes – or when Docear starts – Docear sends all mind-maps located in the temporary folder to the Web Service. The Web Service forwards these mind-maps, i.e. XML files, to the Mind-Map Parser (JAVA), which is based on *nanoXML*[16]. All nodes of the mind-maps, including attributes (text, links to files, titles of linked PDFs, and bibliographic data) are extracted from the XML file and stored in a graph database (*neo4j*[17]). Citations in the mind-maps are replaced with the corresponding Docear-IDs, similarly to the replace-process of citations in the research articles (see section 4.2 and Figure 4). A 'citation' in a mind-map can either be a link to a PDF file, or the bibliographic data that is attached to a node. This means, if a node in a mind-map links a PDF on the user's hard drive, the PDF is identified (via its title) and the link in the mind-map is replaced with the Docear-ID of the cited article, linked PDF respectively. If the cited article is not already in Docear's database, the article is added and a new Docear-ID is created.

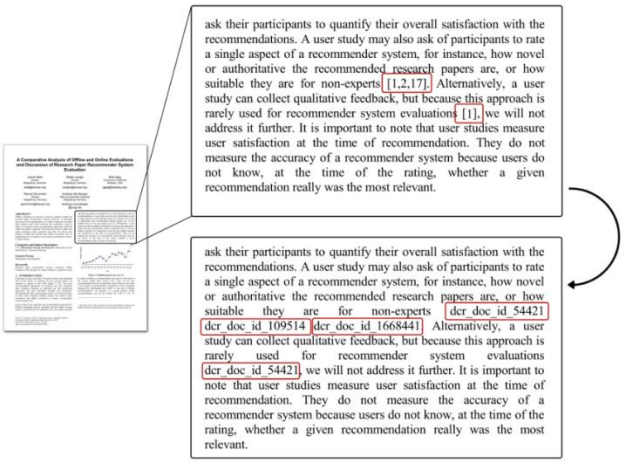

Figure 4: Converting in-text citations to Docear-IDs

4.4 Generating user models & recommendations

The recommendation engine (JAVA) is the central part of Docear's recommender system. It creates new user models and recommendations whenever new mind-maps are uploaded to the server or after recommendations have been delivered to a user. Generating recommendations in advance has the disadvantage that a significant amount of computing time is wasted. Of 1.4 million

[11] http://jersey.java.net
[12] http://sourceforge.net/projects/jpodlib/
[13] http://aye.comp.nus.edu.sg/parsCit/
[14] Meanwhile, our modifications were integrated into ParsCit.
[15] http://lucene.apache.org
[16] http://nanoxml.sourceforge.net/orig/
[17] http://www.neo4j.org/

generated recommendations, only 308,146 recommendations (21.3%) were delivered to the users. In other words, 79.7% of the computing power could have been saved if recommendations were only created when they actually were needed. However, on average, it took 52 seconds to calculate one set of recommendations with a standard deviation of 118 seconds, and users would probably not want to wait so long for receiving recommendations. This rather long computing time is primarily caused by the many statistics that we calculate for each set of recommendations, along with a few algorithms that require intensive computing power. We also run several additional services on the recommendation servers that require a lot of computing power (e.g. PDF processing), and this slows down the recommendation process. If we would disable statistics, concentrate on a few algorithms, and use a dedicated server for the recommender system, it should be possible to generate recommendations in real-time. However, since we need the statistics, and want to evaluate different variations of the recommendation approaches, pre-generating recommendation seems the most feasible solution to us.

Docear's recommender system applies two recommendation approaches, namely stereotype recommendations and content-based filtering (CBF). Every time the recommendation process is triggered, one of these approaches is randomly chosen.

The stereotype approach [20] is chosen with a probability of 1%. It generalizes over all users, and assumes that they are all researchers (which is not exactly true, because some users only use Docear for its mind-mapping functionality). The recommender system then recommends papers that are potentially interesting for researchers, i.e. the books and research articles about academic writing that we manually added to the corpus (see 4.2). Compiling the stereotype recommendations requires a lot of labor and we primarily did this to use the results as a baseline to compare the CBF performance against it [21].

The content-based filtering approach analyzes the users' mind-maps and recommends research papers whose content is similar to the content of the mind-maps. 'Similarity' is based on the number of terms or citations that user-models and research papers have in common. The user modeling process varies by a number of variables that are stored in the algorithms database (*MySQL & Hibernate*[18]). These variables are randomly arranged to assemble the final user-modeling algorithm, each time recommendations are generated. In the first step, the *feature type* to use from the mind-maps is randomly chosen. The feature type may be terms, citations, or both. Then, a number of other variables are chosen such as the *number of mind-maps* to analyze, the *number of features* the user model should contain, and the *weighting scheme* for the features. For instance, one randomly arranged algorithm might utilize the one hundred most recently created citations in the user's mind-maps, weight the citations with CC-IDF, and store the five highest weighted citations as a user model. Another algorithm might utilize all the terms from the two most recently created mind-maps, weight terms based on term frequency and store the 50 highest weighted terms as user model.

The *user model* is represented by a list of terms and/or citations that are supposed to describe the user's information needs. The user-modeling engine randomly chooses whether to store the user model as a weighted or un-weighted list in the database. An un-weighted list is a plain list of terms or citations such as *sun*, *skin*, *dcr_doc_id_54421*, *cancer* ordered by the terms' and citations' weight (the features are always sorted by weight, but the weight is discarded when storing the user model). The weighted-list is a vector in which the weights of the individual features are stored, in addition to the features themselves. Docear uses both weighted and un-weighted lists to research the differences in their effectiveness.

The *matching module* is responsible for finding the appropriate recommendations for a given user model. To match user models and recommendation candidates, Apache Lucene is used, i.e. the user model is sent as a search query to Lucene. From Lucene's top 50 search results, a set of ten papers is randomly selected as recommendations. Choosing papers randomly from the top 50 results decreases the overall relevance of the delivered recommendations, yet increases the variety of recommendations, and allows for the analyzing of how relevant the search results of Lucene are at different ranks. The exact matching algorithm is randomly arranged. Among others, the field at which the candidates Lucene should search for is randomly chosen, for instance in the title only, or in the candidates' full-text.

Once the recommendations are created, they are stored in the recommendation database (MySQL & Hibernate). The system stores for which user the recommendations were generated, by which algorithm, as well as some statistical information such as the time required to generate recommendations and the original Lucene ranking. The recommendations are not yet delivered to the user but only stored in the database.

4.5 Delivering recommendations

To display recommendations to a user, the Docear desktop software sends a request to the Web Service. The Web Service retrieves the latest created recommendations and returns them to Docear, which displays the recommendations to the user. The Web Service stores some statistics, such as when the recommendations where requested and from which Docear version. After recommendations are displayed to the user, a new set of recommendations is generated.

Each recommendation set receives a label that is displayed in Docear above the recommendations (Figure 2). Some labels such as "Free research papers" indicate that the recommendations are free and organic. Other labels such as "Research papers (Sponsored)" indicate that the recommendations are given for commercial reasons. For each user, the label is randomly chosen, when the user registers. The label has no effect on how the recommendations are actually generated. We randomly assign labels only to research the effect of different labels on user satisfaction. For more information refer to [5].

When users click on a recommendation, a download request is sent to Docear's Web Service. The Web Service again stores some statistics, such as the time when the user clicked the recommendation. Then the user is forwarded to the original URL of the recommended paper. Forwarding has the disadvantage that papers occasionally are not available any more at the time of the recommendation since they were removed from the original web server. However, caching PDFs and offering them directly from Docear's servers might have led to problems with the papers' copyright holders.

4.6 Offline evaluation

The *Offline Evaluator* (JAVA) runs occasionally to evaluate the effectiveness of the different algorithms. The offline evaluator creates a copy of the users' mind-maps and removes that citation

[18] http://hibernate.org/

that was most recently added to the mind-maps. In addition, all nodes from the copy are removed that were created after the most recent citation was added. The offline evaluator then selects a random algorithm and creates recommendations for the users. The offline evaluator checks if the removed citation is contained in the list of recommendations and stores this information in the database. It is assumed that if an algorithm could recommend the removed citation, the algorithm was effective. The more often an algorithm could recommend a removed citation, the more effective it is. For more details on the offline evaluator, and potential shortcomings of offline evaluations, refer to [22].

4.7 Technical details

The recommender system runs on two servers. The first server is an Intel Core i7 PC with two 120GB SSDs, one 3 TB HDD, and 16 GB RAM. It runs the PDF Spider, PDF Analyzer, and the mind-map database (*neo4j*), and its load is usually high, because web crawling and PDF processing require many resources. The second server is an Intel Core i7 with two 750 GB HDDs, and 8 GB RAM. It runs all other services including the Web Service, mind-map parser, MySQL database, Lucene, and the offline evaluator. The server load is rather low on average, which is important, because the Web Service is not only needed for recommendations but also for other tasks such as user registration. While long response times, or even down times, for e.g. the PDF spider are acceptable, user registration should always be available. Overall, we invested around 14 person-months development time in the past three years to implement the recommender system[19]. This estimate does not include the development time for the Docear Desktop software.

5. DATASETS

We publish four datasets relating to the research papers that Docear's spider found on the web (see 5.1), the mind-maps of Docear's users (see 5.2), the users themselves (5.3), and the recommendations delivered to the users (see 5.4). Due to spacial restrictions, the following sections provide only an overview of the most important data, particularly with regard to the randomly chosen variables. Please note that all variables are explained in detail in the *readme* files of the datasets. For an empirical evaluation of the different variables please refer to [21], or analyze the datasets yourself. All datasets are available at http://labs.docear.org.

5.1 Research papers

The *research papers* dataset contains information about the research papers that Docear's PDF Spider crawled, and their citations.

The file *papers.csv* contains information about 9.4 million research articles. Each article has a unique *document id*, a *title*, a *cleantitle*, and for 1.8 million articles, a *URL* to the full-text is provided. The 1.8 million documents were found by Docears PDF Spider, and for each of these documents, titles were extracted with Docear's PDF Inspector or parsed from the web page that linked the PDF. The remaining 7.6 million documents in the dataset were extracted from the 1.8 million documents' bibliographies. In this case, no full-text URL is available and the document's title was extracted from the bibliography with ParsCit. Based on a small random sample of 100 documents, we estimate that 71% of the articles are written in English. Other languages include German, Italian, Russian, and Chinese. It also appears that the papers cover various disciplines, for instance, social sciences, computer science, and biomedical sciences. However, several of the indexed documents are of non-academic nature, and sometimes, entire proceedings were indexed but only the first paper was recognized.

Document disambiguation is only based on the documents' "cleantitle". To generate a cleantitle, all characters are transformed to lowercase, and only ASCII letters from a to z are kept. If the resulting cleantitle is less than half the size of the original title, the original title is used as cleantitle – this prevents e.g. Chinese titles to be shortened to a string of length zero. If two documents have the same cleantitle, the documents are assumed identical. Comparing documents only based on such a simplified title is certainly not very sophisticated but it proved to be sufficiently effective for our needs.

The file *citations.csv* contains a list of 572,895 papers with 7.95 million citations. These numbers mean that of the 1.8 million PDFs, 572,895 PDFs could be downloaded and citations could be extracted, and on average, each of the PDFs contained around 14 references. The dataset also contains information where citations occur in the full-texts. For each *citing->cited* document pair, the position of a citation is provided in terms of character count, starting from the beginning of the document. This leads to 19.3 million entries in *citations.csv*, indicating that, on average, each cited paper is cited around three times in a citing document. The dataset allows building citation networks and hence calculating document similarities, or the document impact. Since the position of the citations is provided, document similarity based on citation proximity analysis could be calculated [23], which is an extension of co-citation analysis.

Due to copyright reasons, full-texts of the articles are not included in the dataset. Downloading the full-text is easily possible, since the URLs to the PDFs are included (as long as the PDFs are still available on the Web).

5.2 Mind-maps / user libraries

Every month, 3,000 to 4,000 newly created and modified mind-maps are uploaded to Docear's server. Some mind-maps are uploaded for backup purposes, but most mind-maps are uploaded as part of the recommendation process. Information on the latter ones is provided in the mind-map dataset.

The file *mindmaps.csv* contains information on 52,202 mind-maps created by 12,038 users who agreed that we could publish their information. Docear does not only store the latest version of a mind-map but keeps each revision. Information about 390,613 revisions of the 52,202 mind-maps is also included in *mindmaps.csv*. This means, on average there are around seven to eight revisions per mind-map. All mind-maps and revisions in the dataset were created between March 2012 and March 2013. There are three different types of mind-maps. First, there are mind-maps in which users manage academic PDFs, annotations, and references (Figure 1). These mind-maps represent data similar to the data included in the Mendeley dataset (see section 2). While Mendeley uses the term "personal libraries" to describe a collection of PDFs and references, Docear's "mind-maps" represent also collections of PDFs and references but with a different structure than the ones of Mendeley. Second, there are mind-maps to draft assignments, research papers, theses, or books (Figure 2). These mind-maps differ from the first type as they typically contain only few PDFs and references, but they include additional data such as images, LaTeX formulas, and more text. The third type of mind-maps, are "normal" mind-maps

[19] This is a very rough estimate, as we did not keep track of the exact working hours for the recommender system.

that users create to brainstorm, manage tasks, or organize other information. Due to privacy concerns, this dataset does not contain the mind-maps themselves but only metadata. This includes a list of all the mind-maps and revisions, their file sizes, the date they were created, and to which user they belong. The data may help to analyze how often researchers are using reference management software, for how long they are using it, and how many papers they manage in their mind-maps, personal collections respectively.

The file *mindmaps-papers.csv* contains a list 473,538 papers that are linked eight million times in 12,994 mind-maps. This means, of the 52,202 mind-maps, 24.8% contain at least one link to a PDF, and PDFs are linked 17 times in a mind-map on average. The paper IDs in *mindmaps-papers.csv* are anonymized and do not correlate with paper IDs from the research paper dataset, nor does *mindmaps-papers.csv* contain titles of the linked papers. It should also be noted that the 473,538 papers are not necessarily contained in *papers.csv* as *papers.csv* contains only information of the publicly available PDFs and their citations. These limitations were made to ensure the privacy of our users.

5.3 Users

There are three types of users in Docear, namely registered users, local users, and anonymous users. *Local users* chose not to register when they install Docear. Consequently, they cannot use Docear's online services such as recommendations or online backup, and we do not have any information about these users, nor do we know how many local users there are. *Registered users* sign-up with a username, a password, and an email address and they can use Docear's online services. During the registration process, these users may provide information about their age and gender. Between March 2012 and March 2014, around 1,000 users registered every month, resulting in 21,439 registered users. *Anonymous users* decline to register but still want to use some of Docear's online services. In this case, Docear automatically creates a user account with a randomly selected user name that is tied to a users' computer. Anonymous users cannot login on Docear's website, but they can receive recommendations as their mind-maps are transferred to Docear's servers, if they wish to receive recommendations. Due to spam issues, no new anonymous users were allows since late 2013. Until then, around 9,500 anonymous user accounts were created by non-spammers.

The file *users.csv* contains anonymized information about 8,059 of the 21,439 registered users, namely about those who activated recommendations and agreed to have their data analyzed and published. Among others, the file includes information about the users' date of registration, gender, age (if provided during registration), usage intensity of Docear, when Docear was last started, when recommendations were last received, the number of created mind-maps, number of papers in the user's mind-maps, how recommendations were labeled, the number of received recommendations, and click-through rates (CTR) on recommendations. The CTR expresses the ratio of received and clicked recommendations. If a user receives 100 recommendations, and clicked eight of them, CTR is 8%. CTR is a common performance measure in online advertisement and recommender systems evaluation and allows for the analyzing of the effectiveness of recommendation algorithms.

The file *users_papers.csv* contains a list of 6,726 users and 616,651 papers that the users have in their collections, i.e. mind-maps. This means, on average, each user has linked or cited 92 documents in his or her mind-maps. The paper IDs in *users_papers.csv* do not correlate with the IDs from the research paper dataset, to ensure the users' privacy.

The users-dataset may help to identify how differences between users affect users' satisfaction with recommendations. For instance, we found that older users are more likely to click on recommendations than younger users [7], and that the labelling of recommendations has an effect on user satisfaction [5]. The dataset also allows analyses about the use of reference managers, for instance, how intensive researchers are using Docear.

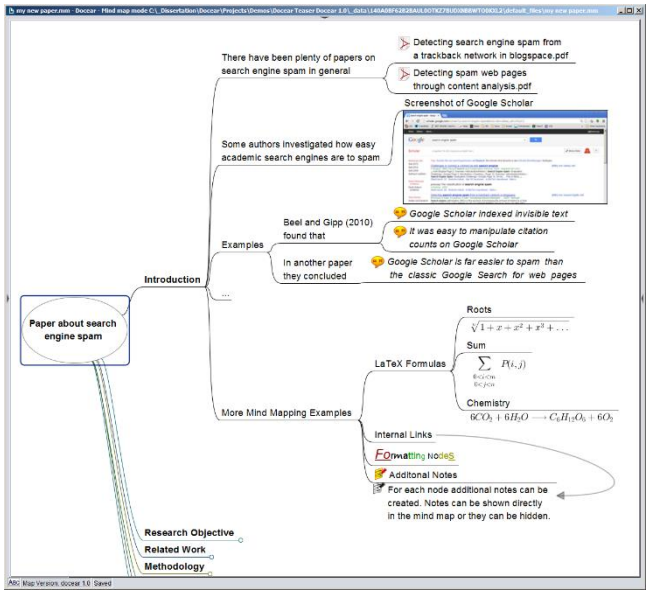

Figure 5: Screenshot of a research paper draft in Docear

5.4 Recommendations

Between March 2013 and March 2014, Docear delivered 31,935 recommendation sets with 308,146 recommendations to 3,470 users. Of the delivered sets, 38.7% were explicitly requested by the users. The remaining 62.2% were delivered automatically when the Docear Desktop software was started. Among the 308,146 recommendations, there were 147,135 unique documents. In other words, from Docear's 1.8 million documents, 9% were actually recommended. The recommendation dataset splits into two files.

The file *recommendation_sets.csv* contains information about the 31,935 delivered recommendation sets. This includes the number of recommendations per set (usually ten), how many recommendations were clicked, the date of creation and delivery, the time required to generate the set and corresponding user models, and information on the algorithm that generated the set. There is a large variety in the algorithms. We stored whether stop words were removed, which weighting scheme was applied, whether terms and/or citations were used for the user modelling process, and several other variables were applied that are described in more detail in the dataset's readme file.

The file *recommendations.csv* contains information about the 308,146 recommendations that Docear delivered. This information includes all details contained in r*ecommendation_sets.csv* and additional information, such as at which position a recommendation was shown, and which document was recommended (again, we anonymized the paper IDs).

6. SUMMARY & FINAL REMARKS

In this paper, we presented the architecture of Docear's research paper recommender system, and introduced four datasets containing metadata about research articles, and information about Docear's users, their mind-maps, and the recommendations they received.

Docear's architecture is unique in the domain of research paper recommendations. Most of the previously published architectures are rather brief, and architectures such as those of bX and BibTip focus on co-occurrence based recommendations. These approaches are primarily relevant for recommender systems with many users. Docear's architecture is comprehensive, explaining the individual components, the required hardware, and the integrated software libraries. Hence, the architecture should provide a good introduction for new researchers and developers on how to build a research paper recommender system. Due to the focus on content-based filtering, the architecture is also relevant for building recommender systems for rather few users.

The datasets are also unique. While the research paper dataset is rather small, and the metadata is probably of a rather low quality, the dataset contains 1.8 million URLs to freely accessible full-text articles that are from various research fields and languages, and the dataset contains information where citations in a paper occur. The mind-map dataset is smaller than the dataset e.g. of Mendeley, but it was not pruned, and hence allows for analyses for users with less than 20 papers in their collections. The dataset also contains the information of how often a paper occurs in a mind-map. This information could be used to infer implicit ratings that are not only binary (linked/not linked) but to weight the implicit rating. The datasets about Docear's users and recommendations contain extensive information, among others, about users' demographics, the number of received and clicked recommendations, and specifics about the algorithms that recommendations were created with. This data allows for analyses that go beyond those that we already performed, and should provide a rich source of information for researchers, who are interested in recommender systems or the use of reference managers.

For the future, we plan to release updated datasets annually or bi-annually, and we invite interested researchers to contact us for cooperation.

7. REFERENCES

[1] L. Palopoli, D. Rosaci, and G. M. Sarné, "A Multi-tiered Recommender System Architecture for Supporting E-Commerce," in *Intelligent Distributed Computing VI*, Springer, 2013, pp. 71–81.

[2] Y.-L. Lee and F.-H. Huang, "Recommender system architecture for adaptive green marketing," *Expert Systems with Applications*, vol. 38, no. 8, pp. 9696–9703, 2011.

[3] M. E. Prieto, V. H. Menéndez, A. A. Segura, and C. L. Vidal, "A recommender system architecture for instructional engineering," in *Emerging Technologies and Information Systems for the Knowledge Society*, Springer, 2008, pp. 314–321.

[4] J. Beel, S. Langer, M. Genzmehr, B. Gipp, C. Breitinger, and A. Nürnberger, "Research Paper Recommender System Evaluation: A Quantitative Literature Survey," in *Proceedings of the Workshop on Reproducibility and Replication in Recommender Systems Evaluation (RepSys) at the ACM Recommender System Conference (RecSys)*, 2013, pp. 15–22.

[5] J. Beel, S. Langer, and M. Genzmehr, "Sponsored vs. Organic (Research Paper) Recommendations and the Impact of Labeling," in *Proceedings of the 17th International Conference on Theory and Practice of Digital Libraries (TPDL 2013)*, 2013, pp. 395–399.

[6] J. Beel, S. Langer, M. Genzmehr, and A. Nürnberger, "Persistence in Recommender Systems: Giving the Same Recommendations to the Same Users Multiple Times," in *Proceedings of the 17th International Conference on Theory and Practice of Digital Libraries (TPDL 2013)*, 2013, vol. 8092, pp. 390–394.

[7] J. Beel, S. Langer, A. Nürnberger, and M. Genzmehr, "The Impact of Demographics (Age and Gender) and Other User Characteristics on Evaluating Recommender Systems," in *Proceedings of the 17th International Conference on Theory and Practice of Digital Libraries (TPDL 2013)*, 2013, pp. 400–404.

[8] J. Beel, S. Langer, M. Genzmehr, and B. Gip, "Utilizing Mind-Maps for Information Retrieval and User Modelling," in *Proceedings of the 22nd Conference on User Modelling, Adaption, and Personalization (UMAP)*, 2014, vol. 8538, pp. 301–313.

[9] W. Huang, S. Kataria, C. Caragea, P. Mitra, C. L. Giles, and L. Rokach, "Recommending citations: translating papers into references," in *Proceedings of the 21st ACM international conference on Information and knowledge management*, 2012, pp. 1910–1914.

[10] P. Jomsri, S. Sanguansintukul, and W. Choochaiwattana, "A framework for tag-based research paper recommender system: an IR approach," in *Proceedings of the 24th International Conference on Advanced Information Networking and Applications (WAINA)*,, 2010, pp. 103–108.

[11] L. Rokach, P. Mitra, S. Kataria, W. Huang, and L. Giles, "A Supervised Learning Method for Context-Aware Citation Recommendation in a Large Corpus," in *Proceedings of the Large-Scale and Distributed Systems for Information Retrieval Workshop (LSDS-IR)*, 2013, pp. 17–22.

[12] S. Bhatia, C. Caragea, H.-H. Chen, J. Wu, P. Treeratpituk, Z. Wu, M. Khabsa, P. Mitra, and C. L. Giles, "Specialized Research Datasets in the CiteSeerx Digital Library," *D-Lib Magazine*, vol. 18, 2012.

[13] K. Jack, M. Hristakeva, R. G. de Zuniga, and M. Granitzer, "Mendeley's Open Data for Science and Learning: A Reply to the DataTEL Challenge," *International Journal of Technology Enhanced Learning*, vol. 4, no. 1/2, pp. 31–46, 2012.

[14] K. Sugiyama and M.-Y. Kan, "Scholarly paper recommendation via user's recent research interests," in *Proceedings of the 10th ACM/IEEE Annual Joint Conference on Digital Libraries (JCDL)*, 2010, pp. 29–38.

[15] K. D. Bollacker, S. Lawrence, and C. L. Giles, "CiteSeer: An autonomous web agent for automatic retrieval and identification of interesting publications," in *Proceedings of the 2nd international conference on Autonomous agents*, 1998, pp. 116–123.

[16] Y. Petinot, C. L. Giles, V. Bhatnagar, P. B. Teregowda, H. Han, and I. Councill, "A service-oriented architecture for digital libraries," in *Proceedings of the 2nd international conference on Service oriented computing*, 2004, pp. 263–268.

[17] J. Bollen and H. Van de Sompel, "An architecture for the aggregation and analysis of scholarly usage data," in *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, 2006, pp. 298–307.

[18] A. Geyer-Schulz, M. Hahsler, and M. Jahn, "Recommendations for virtual universities from observed user behavior," in *Proceedings of the 24th Annual Conference of the Gesellschaft für Klassifikation e.V.*, 2002, pp. 273–280.

[19] J. Beel, S. Langer, M. Genzmehr, and C. Müller, "Docears PDF Inspector: Title Extraction from PDF files," in *Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'13)*, 2013, pp. 443–444.

[20] E. Rich, "User modeling via stereotypes," *Cognitive science*, vol. 3, no. 4, pp. 329–354, 1979.

[21] J. Beel, S. Langer, and G. M. Kapitsaki, "Mind-Map based User Modelling and Research Paper Recommendations," in *Under Review. Pre-print available at http://www.docear.org/publications/.*

[22] J. Beel, S. Langer, M. Genzmehr, B. Gipp, and A. Nürnberger, "A Comparative Analysis of Offline and Online Evaluations and Discussion of Research Paper Recommender System Evaluation," in *Proceedings of the Workshop on Reproducibility and Replication in Recommender Systems Evaluation (RepSys) at the ACM Recommender System Conference (RecSys)*, 2013, pp. 7–14.

[23] B. Gipp and J. Beel, "Citation Proximity Analysis (CPA) - A new approach for identifying related work based on Co-Citation Analysis," in *Proceedings of the 12th International Conference on Scientometrics and Informetrics (ISSI'09)*, 2009, vol. 2, pp. 571–575.