Noname manuscript No.
(will be inserted by the editor)

A Qualitative Study of Large-Scale Recommendation Algorithms for Biomedical Knowledge Bases

Ehsan Noei · Tsahi Hayat · Jessica Perrie · Recep Çolak · Yanqi Hao · Shankar Vembu · Kelly Lyons · Sam Molyneux

Received: date / Accepted: date

Abstract The frequency at which new research documents are being published causes challenges for researchers who increasingly need access to relevant documents in order to conduct their research. Searching across a variety of databases and browsing millions of documents to find semantically relevant material is a time-consuming task. Recently, there has been a focus on recommendation algorithms that suggest relevant documents based on the current interests of the researchers. In this paper, we describe the implementation of seven commonly used algorithms and three aggregation algorithms. We evaluate the recommendation algorithms in a large-scale biomedical knowledge base with the goal of identifying relative weaknesses and strengths of each algorithm. We analyze the recommendations from each algorithm based on assessments of output as evaluated by 14 biomedical researchers. The results of our research provide unique insights into the performance of recommendation algorithms against the needs of modern-day biomedical researchers.

Keywords biomedical science · knowledge base · co-citation similarity · bibliographic coupling · semantic similarity · co-author similarity · large scale

Ehsan Noei, Tsahi Hayat, Jessica Perrie, Kelly Lyons
University of Toronto
E-mail: e.noei@utoronto.ca, {tsahi.hayat, jsscperrie}@gmail.com, kelly.lyons@utoronto.ca

Recep Çolak, Yanqi Hao, Shankar Vembu
Meta
E-mail: {rcpcolak, y3hao6}@gmail.com, shankar@argmix.com

Sam Molyneux
sam@chanzuckerberg.com
E-mail: The Chan Zuckerberg Initiative

1 Introduction

A frequent challenge for science researchers is to keep up to date with relevant research. This task is especially challenging because of the many disparate sources of literature, a growing number of publications being produced, and the relative lack of powerful tools to support context-specific methods for exploration [32]. Specifically, in biomedical research areas, which account for 30% of journals, the number of publications has doubled in the past 20 years [81].

One way to improve the research discovery process is by providing researchers with recommendations for relevant documents or related bodies of work. In this paper, we analyze and compare seven recommendation algorithms and three aggregation algorithms that were implemented in a large-scale biomedical-focused discovery and distribution platform called Meta [75]. Meta's underlying semantic network contains over 90 biomedical controlled vocabularies and ontologies, five core entities (papers, researchers, institutions, journals, and concepts), and relations among the entities (e.g., researchers write documents, documents mention concepts, and journals publish documents). At the time of this research, Meta indexed over 27 million documents with 1.7 million full-text documents. Designing a recommendation system for such a large-scale platform is challenging and the choice of which algorithm(s) to deploy must take into account: runtime performance (how fast the algorithms execute on a large-scale knowledge base); *coverage* (the percentage of documents in the knowledge base for which recommendations can be generated); and, quality of recommendations (the *usefulness* and relevance of the recommended documents to researchers).

We investigate the quality of recommended documents and the performance of recommendation algorithms through a set of experiments. Specifically, we use a qualitative experiment to assess and compare the recommendations returned by ten different algorithms (seven base algorithms and three rank aggregation algorithms) using metrics of *coverage*, *diversity*, *serendipity*, *novelty*, *usefulness*, and *accuracy* [14, 52, 57, 61]. The results of our qualitative experiments provide unique insights into the strengths and weaknesses of different kinds of recommendation algorithms in biomedical research. Our experiments compare recommendations based on meta-data, full-text, semantic relationships, citation networks, co-authorship, and combinations of these features. To the best of our knowledge, our experiments are the first to compare such a diverse set of recommendation algorithms in a working system that is in production.

The remainder of this paper is organized as follows. In Section 2, we survey relevant work on scientific databases and recommender systems. The implementation of the approach is explained in Section 3. The recommendation algorithms are described in Section 4. The evaluation method and findings are presented in Section 5. Finally, we conclude and provide future directions in Section 6.

2 Related Work

In this section, we introduce existing knowledge bases first. Then, we present related research in this area.

2.1 Knowledge Bases

Scientific databases have emerged as one of the milestones in the modern scientific enterprise. One of the main goals of these resources is to refine the methods of information retrieval and augment citation analysis [9, 28, 33]. Major online scientific databases that are currently in use by biomedical researchers are PubMed [22], Google Scholar (GS) [37], Web of Science (WoS) [24], Scopus [31], Semantic Scholar (S2) [91], and Meta [72]. Most of these online scientific databases make use of recommendation systems and algorithms to some degree and identify relevant documents based on certain criteria or data.

PubMed [82] is a free online resource that contains references from the MEDLINE database [38] as well as other life science journals and books [22]. It mostly focuses on medicine and biomedical literature (as Meta), whereas the other resources described below include journals from various scientific fields [33]. Related documents are identified by the number of terms in common. Approximately 2 million terms are used and weighted based on the number of different documents in the database that contain the term, the number of times the term occurs in the first and second documents and the location of the term.

GS [37] is another free service that crawls the web and finds scholarly documents and documents. Documents are indexed by their meta-tags or through automatic format inspection. Compared to PubMed, GS provides very limited search fields (title, author, publication year, all text, and publisher); however, GS supports full-text search, which distinguishes it from PubMed and WoS. For authors with a GS profile, GS uses full-text analysis and algorithms to recommend documents related to one's own publications, and, recently, authors can follow research related to that of another author [1].

WoS [23] is developed and maintained by Clarivate Analytics (formerly the Institute of Scientific Information of Thomson Reuters), and, in comparison to other resources, covers the oldest publications with archived records dating back to 1900 [23, 33]. The WoS indexing procedure is manual and editors update the journal *coverage* by identifying and evaluating promising new journals or deleting journals that have become less useful [99]. WoS finds relevant documents using keywords [45] in the search query and citation-based methods. Recently, WoS and GS started a collaborative effort to interlink their data sources, allowing researchers to search in GS and move to WoS for deeper citation analyses, such as in-depth citation history research [24, 58, 97].

Scopus [31] was launched at nearly the same time as GS, and it is developed and maintained by Elsevier. Scopus is the largest abstract and citation database of peer-reviewed literature. Similar to WoS, the indexing procedure

is manual and the journals are evaluated based on a number of criteria [31]. In comparison to other generic resources, such as WoS and GS, Scopus offers a wider range of search fields called proximities. Related documents are suggested based on shared references, authors, or keywords.

S2 [91] was launched in late 2015 and is developed and maintained at the Allen Institute for Artificial Intelligence [4]. It is a free AI-based scholarly search engine. S2 uses natural language processing, data mining, and machine learning techniques to discern the content of research documents. S2 covers over 40 million research documents [51].

Meta makes newly published findings available to researchers by allowing users to subscribe to any context or entity in the semantic network [75]. Meta's goal is to make it quicker and easier for researchers to filter through scientific documents, find the most important work, and most relevant research tools and products. Because of the variety of available data, that is in Meta's knowledge base, several recommendation algorithms have been implemented and deployed in Meta that recommend documents based on criteria such as citation networks, text content, semantic tag content, and co-authorship information.

In addition to the large-scale databases described above, there are several other systems that focus on some aspects of search and discovery of scientific documents such as recommendation, citation management, and citation analysis. When compared to the major databases described above, these tools have less extensive *coverage* of the scientific literature but do offer some models of recommendation systems.

CiteSeer [16], which covers computer and information systems literature, was the first to provide automated citation indexing and citation linking [62]. It also uses meta-data in combination with word-based measures for the document recommendation.

Mendeley [46] is a reference manager that can also recommend documents based on user profiles or based on a given document using a content-based approach, more specifically, Term Frequency-Inverse Document Frequency (TF-IDF) similarity [80] implemented on top of the Lucene open-source libraries [79]. Mendeley uses meta-data such as user-defined tags, abstracts, mesh-terms, and title, as its fields [48]. For personalized recommendations, it uses item-based collaborative filtering (IBCF) with Apache Mahout [47, 48]. Collaborative filtering is based on the notion that people who agree in their subjective evaluation of past documents (i.e., like-minded users) are likely to agree again in their future evaluation of knowledge bases [85]. Once two like-minded users are determined, items one user likes are recommended to the other user, and vice versa.

Docear [12] is an experimental tool specifically designed for managing and annotating PDFs and recommending documents publicly available on the web. Two types of recommendation algorithms are implemented, namely stereotype recommendations and content-based filtering, both of which are highly dependent on what users have added to their profiles.

TheAdvisor [60] uses a variant of the PageRank algorithm [43] on the citation graph. One interesting feature of the TheAdvisor is that users are able to personalize the recommendations by changing variables of the algorithm to find relevant documents whose relations to the query are not obvious, older documents, or more recent documents. The effectiveness of some of these techniques is limited in that recommendations are either based solely on the similarity between user preferences or on network statistics derived from a user's citation list [44].

We implemented seven base algorithms and three aggregation algorithms that aggregate results from the seven base algorithms in Meta. The algorithms are all inspired by existing work [5, 6, 29, 35, 54, 55, 69, 94] and are customized for the Meta dataset of biomedical documents. Retrieval of biomedical literature always had its unique methods due to the rich knowledge bases, such as the Unified Medical Language System (UMLS), Medical Subject Headings (MeSH), and the Systematized Nomenclature of Medicine (SNOMED) that enable the indexing of documents into concepts, for various purposes, such as retrieval [76]. The algorithms, summarized in Table 1, are described in detail in Section 4.

2.2 Related Literature

Methods and solutions for evaluating recommender systems have been extensively discussed in the earlier literature [2, 41, 49, 92]. Recommender systems can be evaluated using various techniques such as online or offline methods. We discuss related work along with two topics of (i) evaluation of recommender systems and (ii) evaluation metrics.

2.2.1 Evaluation of Recommender Systems

Online Evaluation and User Study. In user studies, evaluators are asked to interact with a recommender system and perform some evaluation tasks. Similarly, online evaluations also benefit from actual users but in a natural course [2]. User participation is essential in both types of evaluation. Box *et al.* [17] provides a general study of online evaluation design. Krishnan *et al.* [59] presents a comparison of online recommender systems concerning human decisions. User studies are popular for evaluation of recommender systems [2]. For example, Lee *et al.* [64], Ma *et al.* [67], and Middleto *et al.* [73] proposed recommender systems for specific applications, such as personalized user searches, and evaluated their solutions using user studies. Raamkumar *et al.* [83] proposed a solution (integrated discovery of similar documents) in order to helps users find similar documents. They evaluated their solution using ACM Digital Library by asking 121 researchers to answer different evaluation questions. Mogenet *et al.* [74] reported that although online evaluation is the most reliable way to evaluate the results of their experiments, it is not as fast as offline evaluation. They compared metrics such as precision and recall when recommending

jobs and found that only a few metrics from the online evaluation are highly correlated with their offline evaluation counterparts.

Offline Evaluation. Offline evaluation benefits from historical data to evaluate a recommender system. Offline evaluation is among popular techniques because frameworks and evaluation measures can be developed for evaluations. However, offline evaluation is limited in understanding the impact of metrics such as serendipity and diversity on user experience [52, 86]. Offline evaluation lacks the ability to measure the actual propensity of users interacting with the recommender system in the future. As an example, data may change over time and the current predictions might not reflect correct recommendations in the future [2]. Moreover, evaluation measures such as accuracy do not capture metrics such as serendipity and novelty [2]. Cañamares *et al.* [21] examines the steps that need to be taken in the offline evaluation of recommender systems, such as user studies and evaluation metrics. Gruson *et al.* [39] compared offline evaluation results with online evaluation as the gold standard. They reported problems from both bias and variance in offline estimators can be mitigated by identifying proper experiments to A/B test. Beel *et al.* [10] compared offline evaluation results to online evaluation in the context of research-paper recommender systems. They reported that offline evaluations do not always reflect the same result as online evaluations. However, they reported a strong correlation between online evaluation results and user studies.

In this paper, we evaluate recommendation algorithms in the context of biomedical knowledge bases with the help of 14 expert evaluators. We show that algorithms such as B-CCP (see Section 4) perform better than other algorithms in biomedical knowledge bases.

2.2.2 Evaluation Metrics

McNee *et al.* [71] argue that measuring accuracy is not enough for evaluation of recommender systems. Konstan *et al.* [56] discuss the importance of novelty in recommender systems. Ge *et al.* [34] study the coverage metrics and Smyth and McClave [95] discuss diversity metrics. Kaminskas and Bridge [52] focus on beyond-accuracy metrics for evaluating recommender systems, including serendipity, novelty, and coverage. They observe correlations between metrics such as diversity and novelty when conducting an offline evaluation. Schein *et al.* [89] discuss the metrics that should be used for evaluating cold-start recommender systems, i.e., for the items that no one has yet evaluated nor ranked them.

In this work, we incorporate all the important metrics for the evaluation of recommender systems. We evaluate six metrics of accuracy, coverage, diversity, novelty, serendipity, usefulness in the context of biomedical knowledge bases.

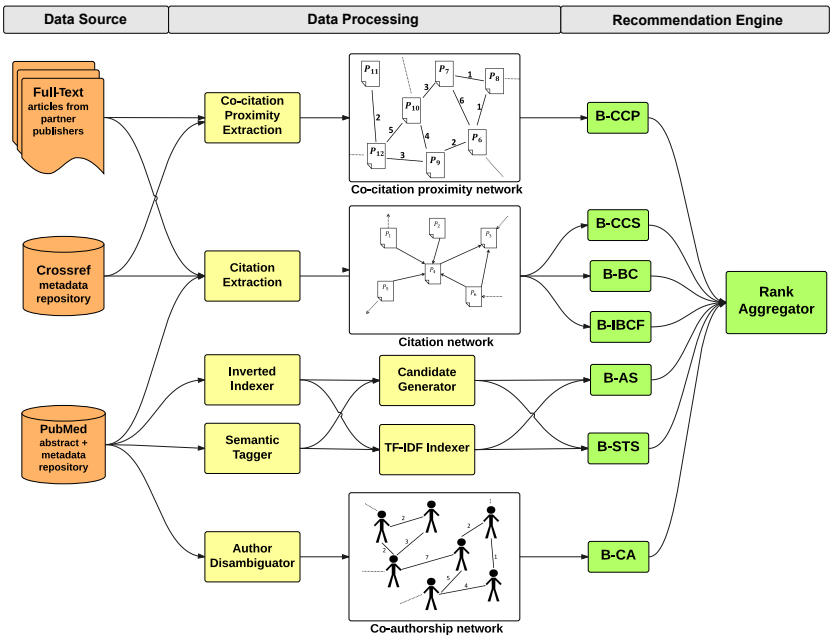

Fig. 1: System schema: data flow of the recommendation engine.

3 The Recommender System in a Nutshell

Figure 1 gives an overview of the recommender system that we evaluated. The algorithms that are discussed in this paper were integrated into Meta's document-to-document recommendation system [75] and make use of its large-scale semantic knowledge base. As shown in Figure 1, the document-to-document recommendation system has three main components:

(i) *Data Source.* Public and private data sources that feed the knowledge network, including PubMed [82], Crossref [26], and full-text documents.
(ii) *Data Processing.* An Extract, Transform, Load (ETL) pipeline that disambiguates the entities and discovers relations among them.
(iii) *Recommendation Engine.* The recommendation engine includes the seven base recommendation algorithms are described in Section 4, and the aggregation algorithms that combine recommendations from the base recommenders to generate the final set of recommendations.

3.1 Data Source

Three main data sources are used to populate the knowledge base: PubMed [22], Crossref [26], and full-text documents.

PubMed. PubMed is the central repository for all biomedical publications. PubMed provides a detailed API through which biomedical journals and conferences can be retrieved [22]. A PubMed record contains a title, abstract, and meta-data (e.g., authors, affiliations, keywords, DOI, and ISSN). Each PubMed document has a unique id (PMID) corresponding to a unique digital object identifier (DOI) registered by Crossref [26].

Crossref. Crossref is a non-profit association of scholarly publishers that develop the infrastructure to distribute and maintain DOIs [26]. From Crossref, we gathered meta-data for 50.9 million documents and citations.

Full-Text Documents. Our third data source is full-text documents from publisher partners of Meta which, at the time of our experiment, included Elsevier, Sage, DeGruyter, PLoS, BMC, among others. The Meta full-text pipeline contains various adapters for diverse publishers and extracts both meta-data and citation information from full-text content, which arrives in both XML [18] and PDF formats.

3.2 Data Processing

Each document goes through a disambiguation engine which has two main tasks. The first is disambiguating the authors of the document where the goal is to associate the document with the existing authors in the database or assign a newly discovered author. Meta's author disambiguation algorithm is modeled after the winning algorithms of the KDD Cup 2013, Author Disambiguation challenge (track-2) [65, 66]. Given a manually disambiguated document-author assignment training set, a random forest classifier [42] is trained to discriminate between correct and incorrect author-document assignments. Given an existing document-to-author assignment database and a newly published document, the algorithm compares the document against each candidate author's profile which included over 43 predictive features at the time of our experiment, using the classification model. If the author with maximum match probability achieves a threshold, the document is assigned to this candidate author, otherwise, a new author profile is generated and the document is assigned as the first document of the newly discovered author. The 43 predictive features span five major categories: author name similarity metrics (e.g. Levenstein, Jaro-Winkler, and Jaccard [36]), document content similarity (mostly based on TF-IDF [84]), affiliation similarity, co-authorship information, and author's active time compatibility. Meta's author disambiguation algorithm achieves an F1 score of 0.73, AU-ROC of 0.94, and AU-PRC of 0.60.

The second disambiguation process deals with concept mentions. Once a concept mention is recognized through an entity recognizer, such as GNAT [40], DNORM [63], and NeJI [20], it is normalized into the canonical name from UMLS [15] and becomes a semantic tag. Among the many concept types, we used only the Medical Subject Headings (MeSH) in our algorithms.

Next, documents go through a citation extraction phase, during which references listed by the documents are identified and resolved into unambiguous,

Table 1: Summary of recommendation and rank aggregation algorithms that are utilized in our system.

Name	Short Description
B-CCS: Co-Citation Similarity	Recommends documents cited by similar citing documents [69, 94].
B-BC: Bibliographic Coupling	Recommends documents with similar references [54].
B-IBCF: Item-Based Collaborative Filtering	Treats citations as user-item purchases, recommends items to users that are similar to ones user already bought.
B-CCP: Co-Citation Proximity	Recommends documents that are co-cited and close together in the text [35].
B-AS: Abstract Similarity	Recommends documents with similar text content.
B-STS: Semantic Similarity	Recommends documents with similar semantic content.
B-CA: Co-Authorship	Recommends documents with similar/shared authors [78, 96].
A-BS: Beam Search Aggregation	Aggregates based on heuristics using beam search [6].
A-BL: Borda Aggregation	Aggregates by simply averaging over the ranks [27].
A-MS: Merge Sort Aggregation	Aggregates based on merge sort based heuristic [6].

directed DOI-DOI pairs and added into the citation network of Meta which has roughly 580 million citations. For documents with full-text, if possible, we also extract pairwise proximities of the references. Finally, the text and semantic tag components of the documents are indexed into an inverted index, which is built using Hadoop MapReduce [93] based TF-IDF builder [68].

3.3 Recommendation Engine

The recommendation algorithms operate on the transformed data in Meta's semantic knowledge network. The algorithms are implemented using a diverse technology stack: Hadoop, Java, Python, and MySQL. Some of the algorithms depend heavily on the Hadoop based MapReduce framework, while others are implemented with direct SQL queries. The recommended documents produced by the base algorithms are aggregated using a number of rank aggregation algorithms. A list of the base and aggregation recommendation algorithms is provided in Table 1. In the next section, the recommendation algorithms are explained in detail.

4 Recommendation Algorithms

The document-to-document recommendation problem can be stated as: given a database of documents, P where $|P| = n$ and a document, p_i that is of interest to a researcher R, recommend a list of k documents, $RP = (p_1, p_2, \ldots, p_k)$ to R such that p_j, $j = 1, \ldots, k$ are judged to be related to p_i and/or in some way useful to R. The list may be a partially ordered list such that p_1 is considered to be more relevant than p_j, $j = 2, \ldots, k$.

In this work, we focus on implementing and evaluating well-studied recommendation algorithms and analyzing their performance in a large scale biomedical knowledge base. We implemented seven based recommendation algorithms

Fig. 2: Citation structures of sample documents. Citation-based algorithms produce the following recommendations for Document E in order: **B-CCS** → (A,C); (B,D). **B-BC** → Z; Y. **B-IBCF** → (A,B,C); D. **B-CCP** → A; D; B; C; (F,G).

(as listed in Table 1) on a database with more than 24 million biomedical documents. All the research documents do not share the same structure or, if they do, their full text may not be publicly accessible. Therefore, we considered the algorithms that would work for such cases, i.e., the ability to leverage various available data types. We also implemented three different algorithms, customized for our dataset of biomedical documents, that aggregate results from the seven base algorithms [5, 6, 29, 35, 54, 69, 94]. Table 1 summarizes the recommendation algorithms.

4.1 Base Recommendation Algorithms

The base recommendation algorithms make use of citation information, content information in abstracts, the full-text of the documents, and authorship information.

4.1.1 Citation-based Algorithms

We generated a citation network of the documents in our database by gathering citations from 50.9 million documents from across the sciences, meta-data from 24.6 million PubMed documents and the full-text of over 16 million documents using a fully automated technique. Our resulting citation network has over 17 million nodes (which is a subset of the biomedical documents in the 50.9 million documents) and over 350 million edges. The base algorithms that use the citation network are: Co-Citation Similarity (B-CCS), Bibliographic Coupling (B-BC), Item-Based Collaborative Filtering (B-IBCF), and Co-Citation Proximity (B-CCP). Figure 2 illustrates a sample data set of three documents with citations indicated.

Co-Citation Similarity (B-CCS). Intuitively, documents that are cited by the same document or co-cited [69, 94] many times are likely to be similar to each other. This notion of similarity provides us with a basis for the recommendation. Referring to the example in Figure 2, given Paper E, B-CCS recommends Documents A and C ahead of Paper B or Paper D because Paper E is co-cited

with Paper A in two documents (Documents Y Z) and Paper E is co-cited with Paper C in two documents as well (also Documents Y and Z). However, Paper E is only co-cited with Paper B in one document (Paper Z) and is only co-cited with Paper D in one document (Paper Y).

The notion of co-cited documents can be captured by using incoming citation vectors. Given a citation network that contains n documents, we define the incoming citation vector $vector^i_{in}$ of a paper p_i as an n-dimensional bit vector $vector^i_{in} = (b^i_1, b^i_2, \ldots, b^i_n)$ where $b^i_j = 1$ if p_j cites p_i, otherwise $b^i_j = 0$. Then, p_i and p_k are co-cited by paper p_j if $b^k_j = b^i_j = 1$. Two documents with many 1's in the same position in their incoming citation vectors are co-cited by many documents.

To recommend documents related to paper p_i, we can apply standard vector similarity metrics such as cosine similarity [98] on $vector^i_{in}$ and $vector^j_{in}$ for all documents p_j to find documents that are most co-cited with p_i. Cosine similarity also normalizes similarity scores by the norms of the vectors, intuitively weighting documents with many incoming citations less than documents with few incoming citations. However, cosine similarity gives an equal weight to all coordinates of $vector^i_{in}$ and $vector^j_{in}$. Suppose there is a hypothetical paper p_k that cites a lot of documents, then for many documents p_x, in the vectors $vector^x_{in}$, $b^x_k = 1$. Conversely, if a paper p_c cites few documents, then in the vectors $vector^c_{in}$, $b^x_c = 1$ for only a few documents p_x. Intuitively, coordinate c should contribute more than k because it is rarer; two documents co-cited by a document with few outgoing citations is worth more than being co-cited by a document with many outgoing citations. To account for this, we normalize the incoming citation vectors by dividing each coordinate of $vector^i_{in}$ and $vector^j_{in}$ by the number of outgoing citations of the document represented by the coordinate before applying cosine similarity.

The number of pairwise similarity computations grows quadratically with the number of documents in the database and is around 10^{14} for 25 million documents. To speed up this computation, we only consider pairs of documents with at least one common incoming citation, and this resulted in a 10^5-fold decrease in the number of pairwise similarity computations.

Bibliographic Coupling (B-BC). Documents having similar citation profiles are intuitively more similar than documents with different citation profiles [54]; this gives us yet another basis for recommendation. In this case, we compute the n-dimensional outgoing citation vector for each paper p_i as $vout_i = (b^i_1, b^i_2, \ldots, b^i_n)$ where $b^i_j = 1$ if p_i cites p_j and $b^i_j = 0$ otherwise. Then, p_i and p_k both cite paper p_j if $b^k_j = b^i_j = 1$. Two documents with many 1's in the same position in their outgoing citation vectors cite many of the same documents.

We then employ the same algorithm used for co-citation similarity (B-CCS) except with the citation edges reversed. We normalize outgoing citation vectors by penalizing coordinates that represent documents with many incoming citations (those that are cited by many documents); then, given a document, we compute the cosine similarity between it and every other document to obtain documents with highly similar citation profiles as recommendations. The

penalization step is the same as in B-CCS. The intuition behind it is: two documents citing a document with few incoming citations is worth more than citing a document with many incoming citations.

In the example in Figure 2, for Paper E, B-BC recommends Paper Z before Paper Y because Paper Z has more citations in common with Paper E (both co-cite Documents F and G). Paper Y only has one citation in common with Paper E.

Similar to our approach used for pairwise similarity computations in co-citation similarity (B-CCS) algorithm, we only consider pairs of documents with at least one common outgoing citation resulting in a 10^5-fold decrease in the number of computations.

Item-based Collaborative Filtering (B-IBCF). The item-based collaborative filtering algorithm is implemented by Apache Hadoop [7]. Using the citation network, we treat each citation edge as a user-item interaction. Paper p_i citing paper p_j represents user p_i buying item p_j. We treat all our documents as both items and users and recommend documents (items) to documents (users) based on citations. We perform the standard item-based collaborative filtering approach [87]: given a user (document) p_i, we want to recommend items (documents) to p_i that p_i does not already have (does not already cite), and are similar to items that p_i already has (already cites). Just like the co-citation similarity algorithms, the similarity is based on vector similarity. Given an item (p_j), its user vector is the binary vector of users (documents) that have purchased (cited) this item (p_j). So, for example, if the incoming citation vector for paper p_j is $vector^j_{in} = (b^j_1, b^j_2, \ldots, b^j_n)$ where $b^j_i = 1$ if p_i cites p_j and $b^j_i = 0$ otherwise, then we consider p_j as an item that is bought by those users p_i where $b^j_i = 1$. Since these vectors are binary, we use Hadoop's log-likelihood vector similarity measure [50] to compute item similarity between items that user p_i has bought, and items that p_i does not have and pick the best items by averaging similarity scores across all items that p_i has. Intuitively, given a paper p_i, we recommend documents most similar to its citations (using log-likelihood similarity, which is intuitively co-citation similarity).

As shown in the example in Figure 2, for Paper E, B-IBCF recommends Documents A, B, and C because Paper Z (which has more citations in common with Paper E than Paper Y) cites all of these documents and E does not (i.e., does not have them in its list). Next, B-IBCF recommends documents Paper D because Z does not cite it (have it in its list) but Y does. Documents F and G are not recommended because Paper E also cites (has) them.

The primary difference between B-IBCF and B-CCS is that given an input paper p, B-CCS finds documents closest to p using co-citation similarity. B-IBCF, however, does not look at the input document, it instead treats the input document as a set of documents by looking at its citations, and then recommends documents closest to its citations by averaging co-citation similarity between its citations and those of other documents.

Co-Citation Proximity (B-CCP). The co-citation proximity approach is based on citation proximity analysis [35]. The intuition behind the algorithm is that if citations occur close together in the text of a document, then the cited documents are likely to be more closely related than if the citations were further apart. We use a different weighting scheme for the proximity occurrences than Gipp and Beel [35] and we aggregate the occurrence values.

We processed each paper p to extract all possible citation pairs between the documents referenced in the citation list of p. Each citation pair is given a proximity type (group – within the same square brackets, sentence, paragraph, section, or paper) based on the minimal distance between each citation. The proximity type is calculated by parsing the structure of the document's XML format [18] or applying minor heuristics.

Relationship weights are used to quantify the different minimum proximities between citation pairs and are summed across document pairs to indicate their similarity. For example, co-citations in the same document are assigned a weight of 1, co-citations in the same section, a weight of 2. If paper p_i and paper p_j are cited once within the same sentence (a total relation weight of 4) but paper p_i and paper p_k are cited within the same section in three additional documents (a total relation weight of $2{\times}3 = 6$), then paper p_i has a stronger similarity to paper p_k than to paper p_j. We also experimented with and applied the approach to larger datasets (over 16 million documents) than what Gipp and Beel used (1.2 million) [35].

Referring back to the example in Figure 2, for Paper E, B-CCP recommends documents based on minimal citation proximity to Paper E over the multiple documents in which Paper E is cited (Documents Y and Z). The recommended documents are ordered as follows: Paper A which is cited in the same sentence as a citation to Paper E (weight of 4) in Paper Y and in the same section (weight of 2) in Paper Z (total weight is 6); Paper D which is cited in the same group as Paper E (weight of 5) in Paper Y; Paper B which is cited in the same sentence as Paper E (weight of 4) in Paper Z; Paper C which is cited in the same document (Paper Z) as Paper E (weight of 1) and in the same section as Paper E (weight of 2) in Paper Y (total weight of 3); and, Documents F and G together with each having a weight of 2 (In both Paper Y and Z, Paper F is cited in the same document as E which is $1{+}1 = 2$ and Paper G is cited in the same section as Paper E in Paper Z for a weight of 2).

One issue with this approach is the situation in which paper p_i and paper p_j are cited in the same sentence but used to contrast each other [35]. This is not a significant issue in our case because our large collection of documents means that consistently co-cited documents will have a stronger connection. Additionally, even if two documents are co-cited in the context of a disagreement and/or conflict because they propose opposing theories, the fact that they are frequently co-cited may make them strongly related (i.e., such that one would be a good recommendation for the other).

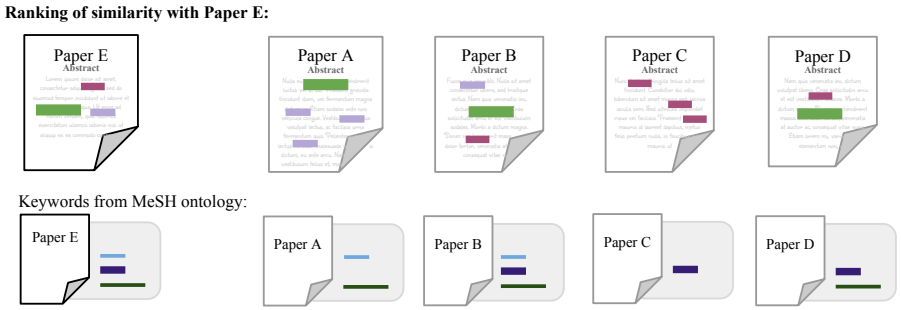

Fig. 3: Example of common words and keywords (based off MeSH ontology) represented by rectangles in the documents. Content-based algorithms produce the following recommendations for Paper E in order: **B-AS** $\rightarrow$ A, B, C, D (using words); **B-STS** $\rightarrow$ B, A, D, C (using keywords).

4.1.2 Content-Based Algorithms

We can also identify similar documents to recommend based on the content of the document or its abstract. These similarity-based algorithms make use of terms and semantic meaning of the terms in the text.

Abstract Similarity (B-AS). Almost every document includes an abstract that typically summarizes the document's focus, methods, experiments, results, and contributions in a succinct and efficient manner. Many knowledge base search engines index only the abstract (rather than the full-text of the document) because abstracts provide sufficient information about the full document. Two documents with similar abstracts are likely to be similar documents; therefore, we used the text of abstracts as a basis for recommending documents. To determine abstract similarity, we use a TF-IDF similarity measure on the words of the abstract. TF-IDF is calculated as the product of the term frequency (TF: the number of times a term t occurs in a document) and the inverse document frequency (IDF: a measure of how common or rare the term is across all documents).

Using the B-AS algorithm to recommend documents for Paper E in Figure 3, Paper A is recommended before Paper B because Paper A contains three instances of an infrequent word (highlighted in light purple). Paper B is recommended before Paper C because Paper B contains one instance of the infrequent word and two frequent words (highlighted in green and pink). Documents C and D both contain frequent words in common with Paper E, but Paper C contains more instances of words in common with Paper E (three vs. two); hence, it is recommended before Paper D.

To obtain accurate TF-IDF similarity, first, we normalize the abstracts by tokenizing them into words, eliminating external token punctuation, and stop-word tokens. TF-IDF is then calculated on a token level. We calculate the inverse document frequency of each token on our entire document abstract

dataset (size approximately 14 million). Inverse document frequency of a token t amongst all n documents $p_i \in P$ in the dataset is defined as:

$$IDF(t,P)=\begin{cases}\sqrt{log(n/DF(t,P))} & \text{if } DF(t,P)\neq 0\\ 0 & \text{if } DF(t,P)=0\end{cases}$$

where $DF(t,P)$ is the number of documents in the set P in which t occurs.

Then, given two abstracts from documents p_i and p_j, we compute their TF-IDF vectors; that is, their abstracts expanded into d-dimensional bit vectors, where d is the number of distinct words that occur in all abstracts (in our database this is approximately 9 million distinct words) such that each position in the vector for paper p_i contains $TF(t,p_i)\times IDF(t,P)$ for the corresponding token t. The term frequency TF of a token t in p_i is defined as: $TF(t,p_i)=\sqrt{\text{count}(t,p_i)}$, where $\text{count}(t,p_i)$ is the number of times t occurs in the abstract of paper p_i.

Given the two TF-IDF vectors, $TF-IDF_i$ and $TF-IDF_j$ for p_i and p_j respectively, we compute their cosine similarity to obtain the final similarity score. Intuitively, this similarity score captures abstracts that share similar terms, strengthened by the number of times the term occurs in the abstracts under consideration and penalized by the commonality of the term amongst all abstracts. Thus, we expect rare terms that occur frequently in both abstracts to indicate strong similarity between the abstracts.

Suppose for a given paper p_i in our dataset, we want to obtain the top 50 documents similar to p_i using abstract TF-IDF similarity. This computation is extremely inefficient as it requires $\approx 25000000^2=6.25\times 10^{14}$ similarity calculations. Therefore, as a fast approximation for a given document abstract, we consider only those document abstracts that share at least one rare term with it. We define a term t as rare when $DF(t,P)\leq 5000$. This step significantly cuts down the number of similarity calculations to approximately 2×10^{11} (more than $3,000$-fold decrease). For the top recommended documents, the abstracts should intuitively share at least one rare term, so this filtering step should not eliminate too many documents and, in practice, this heuristic search space reduction strategy works well.

Semantic Similarity (B-STS). The B-AS algorithm is very sensitive to ambiguity and synonymy problems. To overcome this issue, we aimed to use semantic relationships to infer indirect mentions. Traditional TF-IDF similarity based systems are not able to identify similarity among different terms for the same concept but normalized field/concept annotations provide a principled way to detect and measure similarity. Hence, we applied named entity recognition algorithms to all documents in our database to identify mentions of concepts such as gene, chemicals, diseases, and research areas, which are all included in the MeSH ontology [77].

There are about $28,000$ terms and $139,000$ supplementary concepts in MeSH. For every document, we capture a summary of the document based on the fields it contains. Intuitively, documents that share more fields are more

similar than documents that share fewer fields. As in the abstract similarity algorithm (B-AS), we use TF-IDF similarity to compute semantic similarity in exactly the same way, except instead of using normalized tokens representing words of the abstract, we use fields associated with the document. TF-IDF inherently treats documents that share many rare fields as closest to each other. The term frequency of a term t and paper p_i is either 0 or 1 because our field/term tagger only tags the existence of each field in a document. As in abstract similarity, we only compare similarities between documents which share at least one rare field (term, t), where rare is defined as occurring in at most $5,000$ documents in the set P of documents: $DF(t, P) \leq 5000$. This heuristic filtering approach reduces the number of pairs we have to compare to 72.2 billion (6.25×10^{14}) without jeopardizing the quality of the recommendations.

Going back to the example in Figure 3, having reduced the words to their semantic fields, the frequency of instances within each document no longer has an impact. Paper B is recommended first because it shares the most infrequent terms with Paper E. Paper A and then Paper D are recommended next because Paper A still contains a term more infrequent than Paper D. Finally, Paper C is recommended because it contains one infrequent term in common with Paper E.

4.1.3 Co-Authorship Similarity (B-CA)

The main idea behind co-authorship based recommendations is that documents which share authors are likely to be related to each other [78, 96]. We take a simple approach by first building the co-authorship network where the set of nodes $P = \{p_1, p_2, \ldots, p_n\}$ represents the set of n documents and a weighted edge between two documents, (p_i, p_j) represents the number of shared co-authors between documents p_i and p_j. Then, for a given paper p_i, we traverse the co-author network graph to each of its one- and two-hop neighbors p_j to calculate the shared-author scores as the sum of the weighted edges in the path from p_i to p_j. Each one- and two-hop neighbors p_j is ranked by its shared-author score with p_i and the documents with the highest scores are recommended (ties are broken randomly).

As shown in the example in Figure 4, in one and two hops from Paper E, Paper B has six co-authors (three on the path E-A-B, one on the path E-B, and two on the path E-C-B), and, hence, is the first recommendation. Paper A is next because it has four co-authors on the one- and two-hop paths (one on E-A and three on E-B-A), while Paper C is last because it only has three co-authors on the paths (one on E-C, and two on E-B-C).

4.2 Aggregation Algorithms

We implemented three rank aggregation methods [5, 6, 29] to aggregate results from the base algorithms described above. Given a set of n elements and K complete rankings or permutations of these elements $\pi_1, \pi_2, \ldots, \pi_K$, the goal

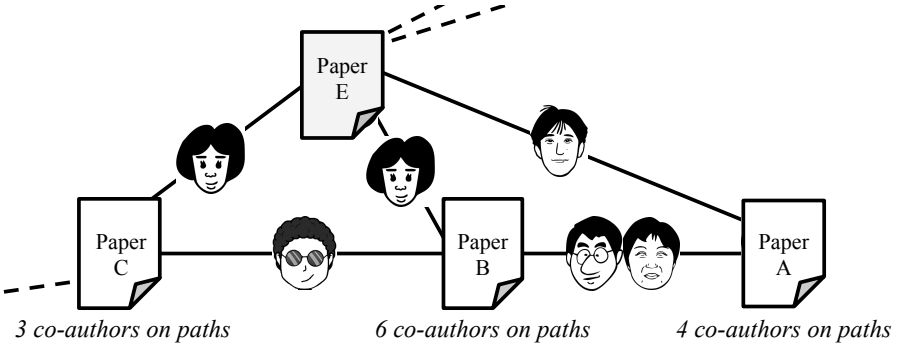

Fig. 4: Co-authorship structure where common authors are shown as icons along paths. Recommendations for Document E are as follows: **B-CA** → B, A, C.

is to find the Kemeny optimal ranking π [53], i.e., the ranking that minimizes $\sum_{i=1}^{K} d(\pi, \pi_i)$, where $d(\cdot, \cdot)$ is the number of pairwise disagreements between a pair of rankings, also known as the Kendall distance. When complete rankings are not available, we place all the unranked objects at the bottom of the list and consider all objects in this set to be tied with each other. The problem of finding the Kemeny optimal ranking is NP-hard [8]. See Ali and Meilă [6] for a comprehensive survey of algorithms to compute Kemeny ranking. We use three different algorithms to approximate the Kemeny ranking. The precedence matrix $Q \in R^{n \times n}$ has entries Q_{ij} that represent the fraction of times an element i is ranked higher than element j, i.e., $Q_{ij}=(1/K)\sum_{k=1}^{K} I(i \prec_{\pi_k} j)$, where $I(\cdot)$ is the indicator function, and $\prec_\pi$ is the precedence operator for ranking π.

4.2.1 Beam Search (A-BS)

The set of all permutations can be represented in the form of a tree, where each permutation can be traced in a path from the root to a leaf. Note that every path from the root to an internal node in the tree represents a partial ranking. We use beam search to explore the set of all permutations and output the optimal ranking. The basic idea is to consider only B candidate solutions (partial rankings) at each level of the tree, where B is a user-defined parameter known as beam width, and these candidates represent the best partial rankings found so far by the heuristic search algorithm. The tree is then explored in a breadth-first fashion from the root all the way down to the leaves. The optimal solution is then selected from the best B candidates found at the lowest level of the tree. A greedy version of the algorithm can be derived by setting $B = 1$, where at each level only one candidate solution is considered greedily. In the other extreme, when $B = \infty$, the algorithm explores all the possible exponential number of rankings/paths in the tree.

In order to select the best B candidate solutions at each level of the tree, we need to define a cost function to score partial rankings. This cost function can be defined using the precedence matrix Q as: $C(\pi_p)=\sum_{(i,j)\in\pi_p} Q_{ij}$, where π_p is a partial ranking and $\{(i,j)\}$ is the set of all pairs (i,j) such that $i \prec_{\pi_p} j$ in the partial ranking, including transitive pairs. Our implementation of the algorithm takes about $3.58s/document$ on a single machine with 8 threads.

4.2.2 Borda Counts (A-BL)

A simple algorithm to aggregate rankings is to rank objects based on their average ranking computed from all the multiple rankings [27]. This is equivalent to sorting the elements based on the column sum of the precedence matrix, i.e., $argsort_i \sum Q_{ij}$. Our implementation of the algorithm takes about $0.161s/document$ on a single machine with 8 threads.

4.2.3 Sort-Based Approximation (A-MS)

Comparison-based sorting algorithms, such as merge sort or quick sort [25], can be adapted to aggregate rankings using the precedence matrix Q [6]. Instead of comparing pairs of elements i and j in the sorting algorithm, we compare Q_{ij} and Q_{ji}. We refer the reader to [88] for more details on comparison sort methods for rank aggregation. In our experiments, we adapted merge sort to solve the rank aggregation problem. Our implementation takes about $0.159s/paper$ on a single machine with 8 threads.

5 Evaluation

In this section, first, the evaluation method is explained; then, in Section 5.2, the observations and findings are presented and discussed.

5.1 Approach

We provided 14 expert evaluators, who are active researchers in the biomedical field, with the output of each recommendation algorithm. The evaluators selected documents from their field of expertise and then rated the quality of the recommendations returned for those documents by each of the algorithms according to six evaluation metrics (see Section 5.1.4). Figure 5 summarizes the setting for the evaluation process.

5.1.1 Recruitment and Participants

We advertised for graduate researchers and postdoctoral fellows at a center for cellular and biomedical research at a major research university. We also asked them to share the recruitment invitation within their networks. Eventually, 14

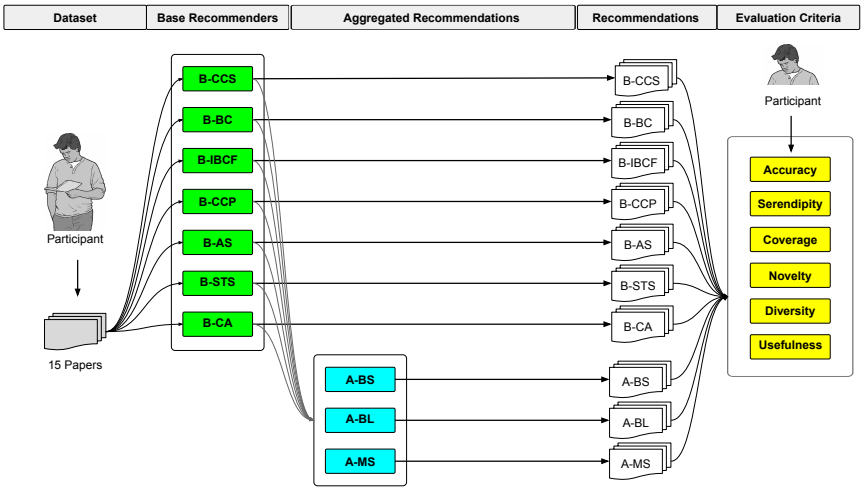

Fig. 5: High-level overview of evaluation experiments detailing how results from one algorithm were fed into another.

people agreed to participate in our study. Participants were compensated for their participation.

5.1.2 Presentation

To ensure high-quality annotations and to calibrate the participant ratings, a random document was returned in each set of recommendations. The participants were informed about the existence of random documents. There were 1,990 random documents inserted into the lists and 1,977 out of 1,990 were rated 1, resulting in greater than 99.3% correctness in *accuracy* ratings. Each participant selected 15 documents from their field of study for a total of 14×15=210 documents. There was one duplicate document; thus, a total of 209 documents were used to evaluate the algorithms. Table 2 shows the number of documents in different categories of publications. As shown in Table 2, the majority of the selected are research documents as the primary focus of this work is research documents. We also compared the keywords associated with selected documents with the keywords associated with the random documents to see how similar are the random documents and selected documents. The results show that 62%, 27%, 26%, 20%, and 5% of the selected documents are associated with humans, female, male, animals, and mice keywords, respectively. Similarly, 44%, 10%, 9%, 47%, and 19% of the random documents are associated with the aforementioned keywords, respectively.

Table 2: Distribution of participants' selected documents in different publication categories.

Category	Number of Documents
research support non-u.s. government	155
research support, extramural	60
research support, u.s. government, funding support by Public Health Service (p.h.s)	33
research support, u.s. government, non-p.h.s.	28
review	8
comparative study	8
research support n.i.h. intramural	4
comment	3
validation study	1
evaluation study	1
research support american recovery and reinvestment act	1

5.1.3 Document Selection

Participants were instructed to select research documents that they knew well and were told that they would need to be able to judge the correctness, interestingness, and relatedness of the recommended documents. In selecting the 15 documents, participants were also asked to try to select documents as diverse as possible: some old, but mostly relatively new documents (last four to five years); some from high impact journals, and some from medium or low impact journals. They were told that there was no restriction on the subject as long as the document was in the biomedical field and indexed in PubMed.

5.1.4 Scoring Criteria and Metrics

For each document, the top 9 recommended documents returned by each algorithm (plus one random document) were provided to participants. The participants were asked to rate each individual recommended document on a scale of 1 to 3 according to *accuracy*, that indicates the relevance and relatedness of the recommended documents to the selected document, (1 is *not at all* accurate, 2 is *somewhat* accurate and 3 is *very* accurate). The evaluation metrics [52] (besides *accuracy*) are:

- ***Coverage***: Checks that highly relevant documents are not missing.
- ***Diversity***: Measures *diversity* of the recommended documents according to factors such as time, author, topic, and method.
- ***Novelty*** Indicates that the user did not know about the recommended documents before.
- ***Serendipity***: Means that the documents are both novel and surprising.
- ***Usefulness***: Measures the value of the recommendations to the user's research.

5.1.5 Analysis

For each recommendation algorithm, first, we measure the distribution of evaluation scores (i.e., *very*, *somewhat*, and *not at all*) across each metric (see Figure 6). Then, in order to rank the performance of each recommendation algorithm, we apply the Scott-Knott test [90] on the ratings with a confidence level of 95% ($\alpha = 0.05$). The Scott-Knott test is a classification technique that ranks the performance of the algorithms into statistically distinct groups of ranks. It recursively divides the algorithms into two statistically significantly different groups of ranks (if possible), and repeats this process on the divided groups. The test continues until no group of algorithms could be divided. The output of the Scott-Knott test is an ordered set of recommendation algorithms (see Figure 7).

5.2 Findings

We begin by presenting the results of the evaluators' scores for the documents recommended by each algorithm. Figure 6 shows the distribution of the scores for the recommended documents. Of the 209 documents selected by the evaluators, we were able to generate recommendations for 157 documents using B-AS and B-CA algorithms, 148 documents using B-CCS, 131 documents using B-IBCF, 129 using B-STS, 128 using B-BC, and 114 using B-CCP. Figure 7 illustrates the output of the Scott-Knott test across each metric. Each document had recommendations from a minimum of two base recommenders. Based on the number of times an algorithm is listed within the top-performing set of algorithms, the overall best algorithm is B-CCP (Co-Citation Proximity) as it is ranked amongst the best algorithms across five metrics (see Table 3). In what follows, we discuss the performance of each algorithm in detail across each metric.

5.2.1 Accuracy

Accuracy is an important metric as it is an assessment of the relevance and relatedness of the recommended documents to the selected document. As in Figure 6, B-CCP, that uses co-citations to recommend documents, achieves the best *accuracy* scores indicating 62.85% *very* accurate results. In addition, B-CCP outperforms other algorithms when it comes to *accuracy* as depicted in Figure 7. In contrast, B-CA has the weakest performance amongst the evaluated algorithms with only 29.63% *very* accurate results. Moreover, B-CA is the only algorithm that scores lower than two on average which suggests that the rest of the algorithms all recommend *somewhat* or *very* accurate documents. The notion behind co-authorship-based recommendations does not necessarily optimize for *accuracy*. Indeed, in the age of increasingly multi-disciplinary studies and multi-authored documents, we find it reasonable that

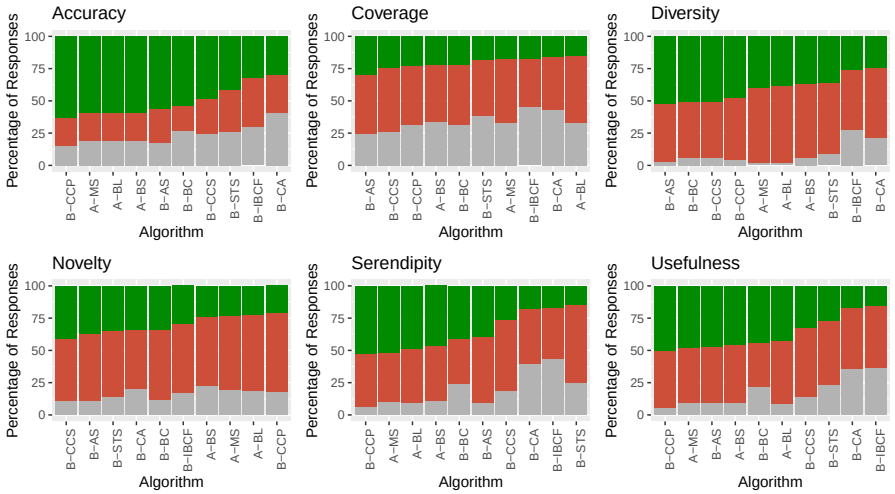

Fig. 6: Normalized distribution of the evaluators' ratings (i.e., scores) across each metric. The top (green), middle (red), and bottom (gray) colors indicate the percentages of scores that are marked as *very*, *somewhat*, and *not at all*, respectively. The algorithms are ordered by the proportion of the responses that indicate *very* scores.

co-authorship-based recommendations are not as accurate as other methods which directly optimize for content or citation similarity.

The algorithms that stand in the second place are A-MS, A-BL, A-BS, and B-AS; except for B-AS, the rest of algorithms in the second rank are aggregation algorithms. Overall, the aggregation algorithms (i.e., A-BS, A-BL, and A-MS) perform better than the base algorithms (with the exception of B-CCP) as they are all in the top-performing group with a mean score of 2.4, and they succeed in harnessing the better recommendations from the base algorithms.

Figure 8 shows the correlation structure according to Spearman rank correlation (ρ) [100] as well as the clustering of six metrics we used for evaluation. *Accuracy* and *novelty* are slightly negatively correlated ($\rho = -0.06$). Novel recommendations are the ones that the researchers did not know about before and so may be more likely to be rated as not accurate. The negative correlation disappears between *accuracy* and *serendipity* ($\rho = 0.17$), as by definition a serendipitous recommendation needs to be surprising in a good way which would be very unlikely if the recommendation is not accurate.

B-CCP (Co-Citation Proximity) outperforms all the other algorithms with the highest average accuracy (i.e., 2.48/3*), whereas B-CA (Co-Authorship) performs the worst with a mean accuracy of* 1.9/3*.*

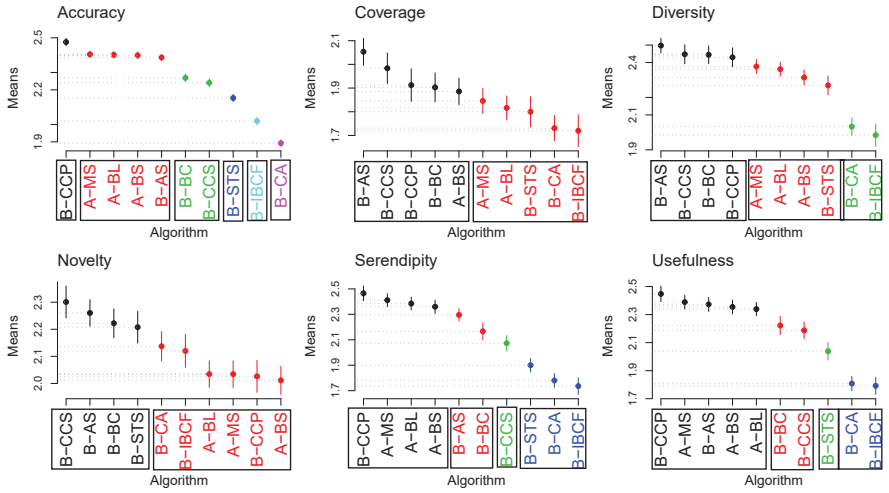

Fig. 7: Groups of ranks of the recommendation algorithms.

5.2.2 Coverage

The Scott-Knott test (Figure 7) divides the algorithms into two groups of ranks where one group outperforms the other one in terms of *coverage*. B-AS, B-CCS, B-CCP, and B-BC are the base algorithms that are ranked as the top ones. The only aggregation algorithm that is amongst the top algorithms is A-BS which aggregates the base algorithms using beam search. As shown in Figures 6 and 7, all algorithms perform poorly on the *coverage*; except for B-AS, that compares the abstracts of documents, the rest of algorithms received a mean evaluation score of less than two.

> *Recommendation algorithms score between* 1.72/3 *and* 2.10/3 *on coverage. B-AS (Abstract Similarity) recommend documents with the best coverage (i.e.,* 2.1/3*) while B-IBCF (Item-Based Collaborative Filtering) has the weakest performance in terms of coverage (i.e.,* 1.72/3*).*

5.2.3 Diversity

For the *diversity* metric, which seems to be weakly interacting with other metrics (see Figure 8), all algorithms except B-IBCF and B-CA do relatively well. In particular, all the other base algorithms perform very close to each other (with an average score of between 2.27 and 2.50). The aggregation algorithms, as shown in Figure 7, are all placed in the second rank on *diversity* while four base algorithms (B-AS, B-CCS, B-BC, and B-CCP) hold the first place. Moreover, as in Figure 6, all the algorithms except B-IBCF and B-CA achieve

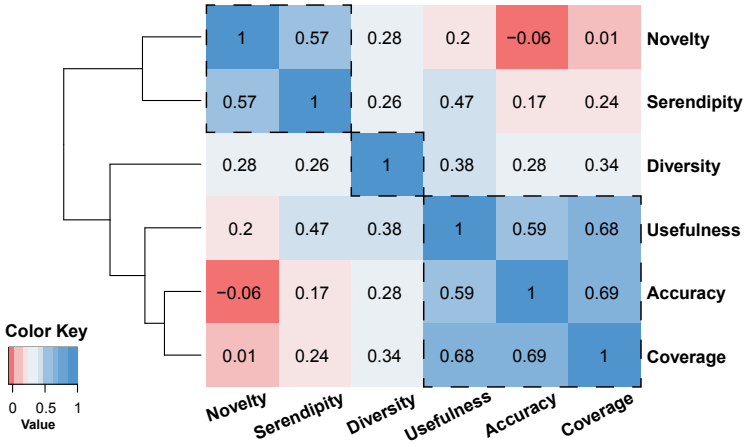

Fig. 8: Spearman rank correlation between evaluation metrics.

a proportion of less than 10% *not at all* diverse results. Interestingly, the documents that were recommended by A-MS, A-BL, and B-AS were almost all *very* or *somewhat* diverse documents with only 2.29%, 2.30%, and 2.37% *not at all* diverse recommendations.

In the analysis of *novelty*, *diversity*, and *serendipity* metrics, a different picture emerges. It is evident, and perhaps not surprising, that *usefulness*, *accuracy* and *coverage* are highly correlated with each other ($\rho > 0.69$), while *novelty* and *serendipity* are tightly correlated as well ($\rho = 0.57$). *diversity* on the other hand shows moderate correlation (between $\rho = 0.27$ and $\rho = 0.34$) with all the metrics, but strong (anti)correlation with none.

> *Except for B-IBCF (Item-Based Collaborative Filtering) and B-CA (Co-Authorship), all algorithms perform similarly well in recommending very or somewhat diverse documents with an average score of between* 2.27 *and* 2.50.

5.2.4 Novelty

Regarding the *novelty* metric, the recorded scores are between 2.0 and 2.3 on average. As depicted in Figure 7, the Scott-Knott test divides the algorithms into two groups of ranks. All algorithms perform relatively well and achieve an average score of more than 2.01, yet the average scores for the first group of ranks are more than 2.21. Four base algorithms, i.e., B-CCS, B-AS, B-BC, and B-STS, are ranked in the first place for recommending novel documents. However, none of the aggregation algorithms are placed in the first group of ranks (Figure 7), and they are all placed in the second rank. Given the interactions between metrics, an intuitive explanation of this phenomenon might be that the aggregation algorithms inherently try to increase consensus and hence

focus on the overlapping recommendations across the base algorithms, which in turn reduces the *novelty*. This is because the more a document appears in the recommended list of documents of diverse algorithms, the less it is likely to be a novel one.

> *All the recommendation algorithms perform relatively well and achieve an average novelty score of more than* 2.01/3 *on average. Four base algorithms, including B-CCS (Co-Citation Similarity), B-AS (Abstract Similarity), B-BC (Bibliographic Coupling), and B-STS (Semantic Similarity), are placed in the first group of ranks with an average score of more than* 2.21/3.

5.2.5 Serendipity

Moving to the *serendipity*, strong performers are B-CCP, A-MS, A-BL, and A-BS (Figures 6 and 7). Three aggregation algorithms perform well in terms of recommending serendipitous documents. Intuitively, in order to achieve high *serendipity* scores, the recommendation has to be good in some sense in addition to being novel. The aggregation algorithms strike a balance between keeping some *novelty* and goodness of recommendations. On the *serendipity* metric, we observe a clear dominance of the aggregation algorithms over base recommenders, with the exception of B-CCP that also performs well regardless of being a base algorithm. B-CCP is the only base algorithm that is statistically significantly better than all the other base algorithms. The top-ranked algorithms achieve an average score of between 2.36 and 2.47. The weakest three algorithms with an average score of below 2 are B-STS, B-CA, and B-IBCF. This might be because the approaches that are designed to increase *serendipity* rely on different heuristics, such as returning uncertain results, to generate surprising and unexpected recommendations [52]. However, the aforementioned algorithms use citations, authors, and semantic contents for recommendations that can reduce the "surprise" factor of the returned documents. Comparing the evaluation results of B-CCP and B-CCS, we observe that although B-CCP is lower in novelty, i.e., users did not know about the recommended documents before, B-CCP ranks higher for serendipity, i.e., recommended documents are both novel and surprising. In other words, B-CCP does not produce a lot of novel documents in comparison to B-CCS but the ones that are produced are surprising ones.

> *Aggregation algorithms, including A-MS (Merge Sort Aggregation), A-BL (Borda Aggregation), and A-BS (Beam Search Aggregation), plus B-CCP (Co-Citation Proximity) outperform all the other base algorithms on serendipity metric. On the other hand, B-STS (Semantic Similarity), B-CA (Co-Authorship), and B-IBCF (Item-Based Collaborative Filtering) are the weakest algorithms with an average serendipity score of below* 2/3.

5.2.6 Usefulness

Similar to *serendipity*, in addition to B-CCP and B-AS, all the aggregation algorithms are amongst the top-performing algorithms. Indeed, this pattern is very similar to the *serendipity* metric and can easily be explained by the fact that *usefulness* and *serendipity* are correlated with $\rho = 0.48$. As such, optimizing for *usefulness* seems to have also optimized and achieved top results for *serendipity* metric as well. B-CA and B-IBCF have the weakest performance on *usefulness* with an average score of 1.8/3. A reason for such a relatively weak performance can be their recommendation strategy where B-CA returns the documents that have similar or the same authors and B-IBCF recommends documents with similar citations.

Aggregation algorithms, including A-MS (Merge Sort Aggregation), A-BL (Borda Aggregation), and A-BS (Beam Search Aggregation), along with two base algorithms, including B-CCP (Co-Citation Proximity) and B-AS (Abstract Similarity), have the best performance in recommending useful documents.

Table 3 summarizes the total number of times each algorithm was amongst the best performers. As shown in Table 3, B-CCP hits the best performance for five evaluation metrics, while *novelty* is the only area that B-CCP is not ranked in the first place. However, despite being in the second place on *novelty*, it achieves an average score of 2/3 which is an acceptable score (having considered that the score of 2 means *somewhat* novel). The second best algorithm, in terms of the number of times it appears in the first place, is B-AS which is another base algorithm. B-AS uses abstract similarity approach and performs relatively well in terms of *coverage*, *diversity*, *serendipity*, and *usefulness*. The third place belongs to an aggregation algorithm, i.e., A-BS, that applies beam search in order to aggregate the recommendations from the base algorithms.

On the other hand, the worst overall performance belongs to B-IBCF and B-CA. B-IBCF applies an item-based collaborative filtering approach and B-CA recommends according to shared or similar co-authors. This observation suggests that co-authorship and item-based collaborative filtering are not well suited for biomedical research applications.

The best overall performance is achieved by B-CCP (Co-Citation Proximity) being ranked in the first place for five of the evaluation metrics. On the other hand, the worst performance belongs to B-IBCF (Item-Based Collaborative Filtering) and B-CA (Co-Authorship). Depending on the metric(s) of interest, an appropriate algorithm should be selected according to Table 3.

Table 3: Total number of times when each algorithm is ranked in the first place by the Scott-Knott test [90] across each metric.

Algorithm	Metric						
	Total	Accuracy	Coverage	Diversity	Novelty	Serendipity	Usefulness
B-CCP	5	✓	✓	✓		✓	✓
B-AS	4		✓	✓	✓		✓
A-BS	3		✓			✓	✓
B-CCS	3		✓	✓	✓		
B-BC	3		✓	✓	✓		
A-BL	2					✓	✓
A-MS	2					✓	✓
B-STS	1				✓		
B-IBCF	0						
B-CA	0						

5.3 Limitations

We study seven base algorithms and three aggregations that are commonly used [5, 6, 29, 35, 54, 69, 94] because the focus of our study is evaluating well-studied recommendation algorithms and analyzing their performance in large scale production knowledge bases. However, future studies should shed more light on other recommendation algorithms such as rank-based aggregation and LP approximation (A-LP). A-LP solves the problem of finding the Kemeny optimal ranking by posing it as an integer linear program (ILP) [3]. Unlike other aggregation algorithms, the amount of overlap p has a significant effect on the runtime complexity of A-LP as the reduction in the number of variables and constraints is quadratic and cubic in $1/p$, respectively. Also, adding more base algorithms will affect LP more than other aggregation methods, unless p increases by the same rate. We omitted using the LP-based algorithm as it was prohibitively slow for the majority of papers. Therefore, even though the LP can be solved using off-the-shelf LP solvers, in practice, we found this to be prohibitively expensive due to a large number of transitivity constraints. Furthermore, in our implementation of algorithms, for example, we use the TF-IDF in order to measure similarities (see Section 4). Future research can expand our findings using other solutions such as word embedding.

For the evaluations (see Section 5), our participants were graduate researchers and postdoctoral fellows at a center for cellular and biomedical research at a major research university so they were all early career researchers. Future experiments could include researchers at various career stages. Each of the 14 participant selected 15 documents with a total of 209 documents (see Section 5.1.2). We used the 209 documents to evaluate the algorithms. Future studies can expand this work by inviting more participants with more documents. We asked participants to rate the recommended papers based on their qualitative assessment of accuracy, coverage, diversity, novelty, serendipity, and usefulness. There are other measures that could be used and participants

could be asked to provide more general assessments (rather than ratings). Future work could also consider quantitative assessments [10, 13, 19, 30, 70].

6 Conclusion

In this paper, we discuss the implementation and evaluation of seven base recommendation algorithms and three aggregation algorithms, aimed at providing relevant document recommendations for biomedical knowledge bases. The base recommendation algorithms utilize diverse sets of features, such as a citation network, text content, semantic tags, and co-authorship information.

We conducted a qualitative experiment to identify the quality of recommendation algorithms for biomedical researchers. We compared the performance of the algorithms on six metrics and identified the algorithms that perform better according to each. For each metric, we rank the algorithms using the Scott-Knott test. For example, B-CCP is ranked first for *accuracy* and it outperforms all the other algorithms with the highest average *accuracy* (i.e., 2.48/3). However, we find that, on average, the majority of the algorithms (nine out of ten) recommend documents that are somewhat or very accurate. The best overall performance is achieved by B-CCP (Co-Citation Proximity) being ranked in the first place according to five of the evaluation metrics (i.e., *accuracy*, *coverage*, *diversity*, *serendipity*, and *usefulness*).

Results of the experiments demonstrate the impact of limitations in the upstream algorithms. For example, recommendation algorithms that rely on semantic similarity require semantic tags for most documents in the knowledge base and author similarity requires author disambiguation within the knowledge base. The results of the study have had a real-world impact in helping to determine which algorithms to deploy in the Meta production system.

In the future, we will focus on personalizing the user experience, which is an active research area [11, 60], by making recommendations based on the user's document library, past reading lists, social interactions, and interests (e.g., identified through subscriptions to researchers and topics) automatically without any input from the user. The findings of this study can also be used to develop similar recommendation systems for other entity types (i.e., authors, journals, institutions, and concepts) that constitute Meta's semantic knowledge network.

Acknowledgment

This research was supported in part by a Natural Sciences and Engineering Research Council (NSERC) Engage Grant and an NSERC Strategic Partnership Project Grant. The authors wish to recognize contributions of Bahar Ghadiri Bashardoost and Yuyang Liu.

References

1. A. Acharya, "Follow related research for key authors," October 13, 2017, last accessed: December 4, 2017. [Online]. Available: https://scholar.googleblog.com/2017/10/follow-related-research-for-key-authors.html
2. C. C. Aggarwal *et al.*, *Recommender systems*. Springer, 2016, vol. 1.
3. S. Agmon, "The relaxation method for linear inequalities," *Canadian Journal of Mathematics*, vol. 6, pp. 382–392, 1954.
4. AI2, "Leverage AI to combat information overload," 2017, last accessed: 11 September 2017. [Online]. Available: http://allenai.org/semantic-scholar/
5. N. Ailon, M. Charikar, and A. Newman, "Aggregating inconsistent information: Ranking and clustering," *Journal of the ACM*, vol. 55, no. 5, 2008.
6. A. Ali and M. Meilă, "Experiments with Kemeny ranking: What works when?" *Mathematical Social Sciences*, vol. 64, pp. 28–40, 2012.
7. Apache, "Introduction to item-based recommendations with hadoop," 2019, last accessed: 21 February 2019. [Online]. Available: http://mahout.apache.org/users/recommender/intro-itembased-hadoop.html/
8. J. Bartholdi III, C. Tovey, and M. Trick, "Voting schemes for which it is can be difficult to tell who won the election," *Social Choice and Welfare*, vol. 6, pp. 157–165, 1989.
9. J. Beel, B. Gipp, S. Langer, and C. Breitinger, "paper recommender systems: a literature survey," *International Journal on Digital Libraries*, vol. 17, no. 4, pp. 305–338, 2016.
10. J. Beel and S. Langer, "A comparison of offline evaluations, online evaluations, and user studies in the context of research-paper recommender systems," in *International conference on theory and practice of digital libraries*. Springer, 2015, pp. 153–168.
11. 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 International Workshop on Reproducibility and Replication in Recommender Systems Evaluation*, ser. RepSys '13. New York, NY, USA: ACM, 2013, pp. 15–22.
12. J. Beel, S. Langer, B. Gipp, and A. Nürnberger, "The architecture and datasets of Docear's research paper recommender system," *D-Lib Magazine*, vol. 20, no. 11, p. 1, 2014.
13. C. T. Bergstrom, J. D. West, and M. A. Wiseman, "The eigenfactor metrics," *International Journal of NeuroScience*, vol. 28, no. 45, pp. 11 433–11 434, 2008.
14. J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowledge-based systems*, vol. 46, pp. 109–132, 2013.
15. O. Bodenreider, S. J. Nelson, W. T. Hole, and H. F. Chang, "Beyond synonymy: Exploiting the UMLS semantics in mapping vocabularies." in *Proceedings of the AMIA symposium*. American Medical Informatics Association, 1998, p. 815.

16. 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*. ACM, 1998, pp. 116–123.
17. G. Box, W. Hunter, and J. Hunter, "Statistics for experimenters," *Wiley*, 1978.
18. T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau, "Extensible markup language (xml) 1.0," 2000.
19. J. S. Breese, D. Heckerman, and C. M. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, 1998, pp. 43–52.
20. D. Campos, S. Matos, and J. L. Oliveira, "A modular framework for biomedical concept recognition," *BMC bioinformatics*, vol. 14, no. 1, p. 281, 2013.
21. R. Cañamares, P. Castells, and A. Moffat, "Offline evaluation options for recommender systems," *Information Retrieval Journal*, pp. 1–24, 2020.
22. K. Canese and S. Weis, "PubMed: The bibliographic database," *The NCBI Handbook [Internet]*, 2013, last accessed: 15 December 2017. [Online]. Available: http://www.ncbi.nlm.nih.gov/books/NBK153385/
23. Cision, "Acquisition of the Thomson Reuters Intellectual Property and Science Business by Onex and Baring Asia completed," 2016, last accessed: 15 December 2017. [Online]. Available: http://www.prnewswire.com/
24. Clarivate, "Web of Science: Core collection help," 2017, last accessed: 15 January 2019. [Online]. Available: https://images.webofknowledge.com/images/help/WOS/hp_full_record.html
25. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2009.
26. Crossref, "Crossref," [Online] Available: http://www.crossref.org/, 2019.
27. J.-C. de Borda, "Mémoire sur les élections au scrutin," *Histoire de l'Académie Royale des Sciences, Paris*, pp. 657–664, 1781.
28. Y. Ding, G. Zhang, T. Chambers, M. Song, X. Wang, and C. Zhai, "Content-based citation analysis: The next generation of citation analysis," *Journal of the Association for Information Science and Technology*, vol. 65, no. 9, pp. 1820–1833, 2014.
29. C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, "Rank aggregation methods for the web," in *Proceedings of the 10th International Conference on World Wide Web*. ACM, 2001, pp. 613–622.
30. M. D. Ekstrand, P. Kannan, J. A. Stemper, J. T. Butler, J. A. Konstan, and J. T. Riedl, "Automatically building research reading lists," in *Proceedings of the Fourth ACM Conference on Recommender Systems*. ACM, 2010, pp. 159–166.
31. Elsevier, "The largest up-to-date collection of global, unbiased and expertly sourced research," 2017, last accessed: 2018 December 15. [Online]. Available: https://www.elsevier.com/solutions/scopus/content

32. P. Fafalios and Y. Tzitzikas, "Stochastic reranking of biomedical search results based on extracted entities," *Journal of the Association for Information Science and Technology*, vol. 68, no. 11, pp. 2572–2586, 2017.
33. M. E. Falagas, E. I. Pitsouni, G. A. Malietzis, and G. Pappas, "Comparison of PubMed, Scopus, Web of Science, and Google Scholar: Strengths and weaknesses," *The Journal of the Federation of American Societies for Experimental Biology*, vol. 22, no. 2, pp. 338–342, 2008.
34. M. Ge, C. Delgado-Battenfeld, and D. Jannach, "Beyond accuracy: evaluating recommender systems by coverage and serendipity," in *Proceedings of the fourth ACM conference on Recommender systems*, 2010, pp. 257–260.
35. B. Gipp and J. Beel, "Citation Proximity Analysis (CPA): A new approach for identifying related work based on co-citation analysis," in *ISSI'09: 12th International Conference on Scientometrics and Informetrics*, 2009, pp. 571–575.
36. W. H. Gomaa and A. A. Fahmy, "A survey of text similarity approaches," *International Journal of Computer Applications*, vol. 68, no. 13, pp. 13–18, 2013.
37. Google, "Google scholar: About," 2020. [Online]. Available: https://scholar.google.ca/intl/en/scholar/about.html
38. T. Greenhalgh, "How to read a paper: the medline database," *Bmj*, vol. 315, no. 7101, pp. 180–183, 1997.
39. A. Gruson, P. Chandar, C. Charbuillet, J. McInerney, S. Hansen, D. Tardieu, and B. Carterette, "Offline evaluation to make decisions about playlistrecommendation algorithms," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019, pp. 420–428.
40. J. Hakenberg, C. Plake, R. Leaman, M. Schroeder, and G. Gonzalez, "Inter-species normalization of gene mentions with GNAT," *Bioinformatics*, vol. 24, no. 16, pp. i126–i132, 2008.
41. J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, pp. 5–53, 2004.
42. T. K. Ho, "Random decision forests," in *Proceedings of 3rd international conference on document analysis and recognition*, vol. 1. IEEE, 1995, pp. 278–282.
43. A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme, "Information retrieval in folksonomies: Search and ranking," in *European semantic web conference*. Springer, 2006, pp. 411–426.
44. Y. Huang, N. Contractor, and Y. Yao, "CI-KNOW: Recommendation based on social networks," in *Proceedings of the International Conference on Digital Government Research*. Digital Government Society of North America, 2008, pp. 27–33.
45. Y. Ishida, T. Shimizu, and M. Yoshikawa, "An analysis and comparison of keyword recommendation methods for scientific data," *International Journal on Digital Libraries*, pp. 1–21, 2020.

46. K. Jack, "Mendeley: Crowdsourcing and recommending research on a large scale," 2011, online; accessed 2015-02-25. [Online]. Available: http://www.slideshare.net/KrisJack/mendeley-crowdsourcing-and-recommending-research-on-a-large-scale
47. ——, "Mahout becomes a researcher: Large scale recommendations at Mendeley," 2012, last accessed: 15 December 2017. [Online]. Available: http://www.slideshare.net/KrisJack/mahout-becomes-a-researcher-large-scale-recommendations-at-mendeley
48. ——, "Mendeley: Recommendation systems for academic literature," 2012, last accessed: 15 December 2017. [Online]. Available: http://www.slideshare.net/KrisJack/mendeley-recommendation-systems-for-academic-literature
49. D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, "An introduction to recommender systems," *New York: Cambridge*, 2011.
50. I. Jolliffe, *Principal component analysis*. Springer, 2011.
51. N. Jones, "AI science search engines expand their reach," November 11, 2016, last accessed: 15 December 2017. [Online]. Available: http://www.nature.com/news/ai-science-search-engines-expand-their-reach-1.20964
52. M. Kaminskas and D. Bridge, "Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 7, no. 1, p. 2, 2017.
53. J. Kemeny and J. Snell, *Mathematical models in social sciences*. Blaisdell, New York, 1962.
54. M. M. Kessler, "Bibliographic coupling between scientific papers," *American documentation*, vol. 14, no. 1, pp. 10–25, 1963.
55. R. Klavans and K. W. Boyack, "Which type of citation analysis generates the most accurate taxonomy of scientific and technical knowledge?" *Journal of the Association for Information Science and Technology*, vol. 68, no. 4, pp. 984–998, 2017.
56. J. A. Konstan, S. M. McNee, C.-N. Ziegler, R. Torres, N. Kapoor, and J. Riedl, "Lessons on applying automated recommender systems to information-seeking tasks," in *AAAI*, vol. 6, 2006, pp. 1630–1633.
57. D. Kotkov, S. Wang, and J. Veijalainen, "A survey of serendipity in recommender systems," *Knowledge-Based Systems*, vol. 111, pp. 180–192, 2016.
58. R. Kreisman, "Thomson Reuters-Google Scholar linkage offers big win for STM users and publishers," 2013.
59. V. Krishnan, P. K. Narayanashetty, M. Nathan, R. T. Davies, and J. A. Konstan, "Who predicts better? results from an online study comparing humans and an online recommender system," in *Proceedings of the 2008 ACM conference on Recommender systems*, 2008, pp. 211–218.
60. O. Küçüktunç, E. Saule, K. Kaya, and Ü. V. Çatalyürek, "Towards a personalized, scalable, and exploratory academic recommendation service," in *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, 2013, pp.

636–641.
61. M. Kunaver and T. Požrl, "Diversity in recommender systems–a survey," *Knowledge-Based Systems*, vol. 123, pp. 154–162, 2017.
62. S. Lawrence, C. L. Giles, and K. Bollacker, "Digital libraries and autonomous citation indexing," *IEEE Computer*, vol. 32, no. 6, pp. 67–71, 1999.
63. R. Leaman, R. I. Doğan, and Z. Lu, "DNorm: Disease name normalization with pairwise learning to rank," *Bioinformatics*, vol. 29, no. 22, pp. 2909–2917, 2013.
64. B.-H. Lee, H.-N. Kim, J.-G. Jung, and G.-S. Jo, "Location-based service with context data for a restaurant recommendation," in *International Conference on Database and Expert Systems Applications*. Springer, 2006, pp. 430–438.
65. C.-L. Li, Y.-C. Su, T.-W. Lin, C.-H. Tsai, W.-C. Chang, K.-H. Huang, T.-M. Kuo, S.-W. Lin, Y.-S. Lin, Y.-C. Lu *et al.*, "Combination of feature engineering and ranking models for paper-author identification in KDD cup 2013," in *Proceedings of the 2013 KDD Cup Workshop*. ACM, 2013, p. 2.
66. J. Liu, K. H. Lei, J. Y. Liu, C. Wang, and J. Han, "Ranking-based name matching for author disambiguation in bibliographic data," in *Proceedings of the 2013 KDD Cup Workshop*. ACM, 2013, p. 8.
67. Z. Ma, G. Pant, and O. R. L. Sheng, "Interest-based personalized search," *ACM Transactions on Information Systems (TOIS)*, vol. 25, no. 1, pp. 5–es, 2007.
68. C. D. Manning, P. Raghavan, and H. Schütze, "Scoring, term weighting and the vector space model," *Introduction to Information Retrieval*, vol. 100, 2008.
69. I. Marshakova-Shaikevich, "System of document connections based on references," *Scientific and Technical Information Serial of VINITI*, vol. 6, pp. 3–8, 1973.
70. S. M. McNee, Istvan, D. Cosley, P. Gopalkrishnan, S. K. Lam, A. M. Rashid, J. A. Konstan, and J. Riedl, "On the recommending of citations for research papers," in *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work*, 2002.
71. S. M. McNee, J. Riedl, and J. A. Konstan, "Being accurate is not enough: how accuracy metrics have hurt recommender systems," in *CHI'06 extended abstracts on Human factors in computing systems*, 2006, pp. 1097–1101.
72. Meta, "Meta," 2020. [Online]. Available: https://www.meta.org/
73. S. E. Middleton, N. R. Shadbolt, and D. C. De Roure, "Ontological user profiling in recommender systems," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, pp. 54–88, 2004.
74. A. Mogenet, T. A. N. Pham, M. Kazama, and J. Kong, "Predicting online performance of job recommender systems with offline evaluation," in *Proceedings of the 13th ACM Conference on Recommender Systems*, 2019, pp. 477–480.

75. S. D. Molyneux and A. C. Molyneux, "System and method for establishing a dynamic meta-knowledge network," Apr. 4 2017, uS Patent 9,613,321.
76. R. Moskovitch, F. Wang, J. Pei, and C. Friedman, "JASIST special issue on biomedical information retrieval," *Journal of the Association for Information Science and Technology*, vol. 68, no. 11, pp. 2525–2528, 2017.
77. S. J. Nelson, "Medical terminologies that work: The example of MeSH," in *Proceedings of the 2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks (ISPAN)*. IEEE, 2009, pp. 380–384.
78. M. E. Newman, "The structure of scientific collaboration networks," *Proceedings of the National Academy of Sciences*, vol. 98, no. 2, pp. 404–409, 2001.
79. E. Noei and A. Heydarnoori, "Exaf: A search engine for sample applications of object-oriented framework-provided concepts," *Information and Software Technology*, vol. 75, pp. 135–147, 2016.
80. E. Noei, F. Zhang, S. Wang, and Y. Zou, "Towards prioritizing user-related issue reports of mobile applications," *Empirical Software Engineering*, pp. 1–33, 2018.
81. A. Plume and D. van Weijen, "Publish or perish? The rise of the fractional author," *Research Trends*, vol. 38, no. 3, pp. 16–18, 2014.
82. PubMed Help, November 27, 2017, last accessed: 15 December 2017. [Online]. Available: http://www.ncbi.nlm.nih.gov/books/NBK3827/
83. A. S. Raamkumar, S. Foo, and N. Pang, "Can i have more of these please?: Assisting researchers in finding similar research papers from a seed basket of papers," *Electronic Library*, vol. 36, no. 3, pp. 568–587, 2018.
84. A. Rajaraman and J. D. Ullman, *Mining of massive datasets*. Cambridge University Press, 2011.
85. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*. New York, NY, USA: ACM, 1994, pp. 175–186.
86. A. Said, B. Fields, B. J. Jain, and S. Albayrak, "User-centric evaluation of a k-furthest neighbor collaborative filtering recommender algorithm," in *Proceedings of the 2013 conference on Computer supported cooperative work*, 2013, pp. 1399–1408.
87. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*. ACM, 2001, pp. 285–295.
88. F. Schalekamp and A. Zuylen, "Rank aggregation: Together we are strong," in *Proceedings of the 11th Workshop on Algorithm Engineering and Experiments*, 1998, pp. 38–51.
89. A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, 2002, pp. 253–260.

90. A. J. Scott and M. Knott, "A cluster analysis method for grouping means in the analysis of variance," *Biometrics*, pp. 507–512, 1974.
91. Semantic Scholar, "Semantic scholar," 2019, last accessed: 15 January 2019. [Online]. Available: https://www.semanticscholar.org/
92. G. Shani and A. Gunawardana, "Evaluating recommendation systems," in *Recommender systems handbook*. Springer, 2011, pp. 257–297.
93. K. Shvachko, H. Kuang, S. Radia, R. Chansler *et al.*, "The hadoop distributed file system." in *MSST*, vol. 10, 2010, pp. 1–10.
94. H. Small, "Co-citation in the scientific literature: A new measure of the relationship between two documents," *Journal of the American Society for information Science*, vol. 24, no. 4, pp. 265–269, 1973.
95. B. Smyth and P. McClave, "Similarity vs. diversity," in *International conference on case-based reasoning*. Springer, 2001, pp. 347–361.
96. K. Sugiyama and M.-Y. Kan, "Serendipitous recommendation for scholarly papers considering relations among researchers," in *Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries*. ACM, 2011, pp. 307–310.
97. ——, "A comprehensive evaluation of scholarly paper recommendation using potential citation papers," *International Journal on Digital Libraries*, vol. 16, no. 2, pp. 91–109, 2015.
98. P.-N. Tan, *Introduction to data mining*. Pearson Education India, 2018.
99. J. Testa, "The Thomson Reuters journal selection process," 2016, last accessed: 15 December 2017. [Online]. Available: http://scientific.thomsonreuters.com/wok/benefits/essays/journalselection/
100. J. H. Zar, "Significance testing of the spearman rank correlation coefficient," *Journal of the American Statistical Association*, vol. 67, no. 339, pp. 578–580, 1972.