CiteSight: Supporting Contextual Citation Recommendation Using Differential Search

Avishay Livne[1], Vivek Gokuladas[2], Jaime Teevan[3], Susan T. Dumais[3], Eytan Adar[1,4]
[1] Computer Science & Engineering and [4] School of Information, University of Michigan, Ann Arbor, MI 48109, USA
[2] Qualcomm, San Diego, CA 92121, USA
[3] Microsoft Research, Redmond WA 98052, USA
{avishay,eadar}@umich.edu, gvivekcbe@gmail.com, {teevan,sdumais}@microsoft.com

ABSTRACT
A person often uses a single search engine for very different tasks. For example, an author editing a manuscript may use the same academic search engine to find the latest work on a particular topic or to find the correct citation for a familiar article. The author's tolerance for latency and accuracy may vary according to task. However, search engines typically employ a consistent approach for processing all queries. In this paper we explore how a range of search needs and expectations can be supported within a single search system using *differential search*. We introduce *CiteSight*, a system that provides personalized citation recommendations to author groups that vary based on task. CiteSight presents cached recommendations instantaneously for online tasks (e.g., active paper writing), and refines these recommendations in the background for offline tasks (e.g., future literature review). We develop an active cache-warming process to enhance the system as the author works, and context-coupling, a technique for augment sparse citation networks. By evaluating the quality of the recommendations and collecting user feedback, we show that differential search can provide a high level of accuracy for different tasks on different time scales. We believe that differential search can be used in many situations where the user's tolerance for latency and desired response vary dramatically based on use.

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

Keywords
Citation recommendation, personalization, differential search.

1. INTRODUCTION
The act of citation in academic work is a critical piece of scientific production. Citations are used to assign credit, justify decisions, norm behavior, and increase awareness of one's own work. Research on these motivations has become a 'cottage industry' [3], generating a vast array of typologies and classification schemes. While the act of citing is ultimately the same regardless of motivation, the execution of citation work varies dramatically and has led to a large ecosystem of workflows, datasets, tools, and search engines that address different aspects of the problem.

SIGIR '14, July 06 - 11 2014, Gold Coast , QLD, Australia

http://dx.doi.org/10.1145/2600428.2609585

While idealized citation work may involve a set of co-authors obtaining, interpreting, synthesizing, and indexing all relevant citations prior to writing a paper, in reality the process includes many different tasks done by different individuals at different times. During paper writing, for example, an author may draw from existing knowledge of related literature to reference a specific paper that they know well. Or they may find themselves making an argument that requires finding and analyzing papers they have not read. They may also preform self-audits at the completion of writing to ensure appropriate citation of the latest work [6]. Each of these citation behaviors has a different tolerance for novelty, latency, and accuracy, and is done with different competing attentional demands. As a result, each activity is typically supported in a different way. For example, EndNote's *Cite While You Write* allows authors to reference specific citations without leaving the context of the paper. For the more ambiguous citation needs that come up during writing, authors sometimes use the text of the paper itself, leaving notes for themselves or coauthors (e.g., "[alice: citation?]") that are addressed at a later time using an academic search engine, general search engine, or other means.

We present CiteSight, a *differential search* system that provides contextualized citation recommendations using the most appropriate contexts and approaches to match authors various needs as they write a paper. Differential search reflects a class of search infrastructures where different use cases require different levels of service. For example, CiteSight caches highly relevant, personalized, content related to the paper's authors, venue, and existing citations, and searches through this content to instantaneously identify relevant citations as a person writes. Simultaneously, CiteSight performs a deeper analysis on the broader literature in the background that can be accessed when the author is ready to consider other relevant literature.

CiteSight indexes metadata for over 2.3M Computer Science manuscripts provided by Microsoft Academic. While the corpus is small by Web standards, querying over it is computationally expensive because our queries are long and complex, and many features must be dynamically calculated using text and network structure. Providing recommendations in near real time—a desirable feature to limit disruption of writing "flow" [24]—necessitates a tradeoff. Either one settles for faster features with worse performance, or restricts the size of the database. CiteSight achieves an effective compromise by pre-caching documents that are likely targets of inline citation: those the author is most likely to know. By mining our repository for past behavior, features such as who the authors are and what venue is being targeted can be used to personalize the cache and increase the hit rate. Additionally, dynamic monitoring of local behaviors, such as citations in the working manuscript can enhance the cache through the identifica-

tion of co-cited literature. While cached results are provided nearly instantaneously, CiteSight continues to search the full index by responding to additional user hints and updating results to make broader recommendations for literature to read.

An additional challenge of citation recommendation is that it often requires recommending items for which there is very little information. While we would like to use past citation behavior as a hint for the appropriateness of a particular recommendation, citations are power-law distributed (with a cut-off) [8]. This means that most papers are rarely cited and have minimal citation context. CiteSight introduces *citation-coupling* as a way to enrich the citation graph by borrowing contexts from related papers.

Citation recommendation is representative of a broad class of problems where search and recommender systems are integrated with user interfaces. Differential search captures the idea that, depending on the task, end-users have different criteria for evaluating and using the results. Differences may range from the latency required for certain tasks but may also focus on other factors such as the novelty of results. By understanding that the demands of the user, in relation to the "intelligent" component, can vary significantly given what the end-user knows, needs, and wants, it is possible to create systems that can better support this variation.

The contribution of this work includes an architecture to support the varied citations tasks ranging from sub-10ms inline recommendations to "deeper" queries. To support this mechanism we introduce a dynamic, personalized cache-warming technique that makes use of a range of signals to improve performance. We additionally describe context-coupling, a technique to enhance sparse network structures to support better recommendations. Finally, we describe ways in which the design of CiteSight can support a range of enhanced "slow search" [39] features including collaborative search, summarization, and citation audits.

2. RELATED WORK

Significant research has been devoted to identifying the papers that are the most likely to be referenced by a given manuscript. Techniques explored include collaborative filtering [24][42], topic modeling [10][19][28][37], machine learning [5][36] and machine translation [14][18][21]. CiteSight leverages these to identify relevant citations, and extends them through a novel context-coupling technique that augments paper descriptions.

A number of citation management systems have been built using these techniques. For example, TheAdvisor [20] expands the bibliography entered by an author to identify related papers in the citation graph, seeded by a keyword-based query. Scienstein [12] allows authors to submit entire papers with their associated bibliography to find related work. CiteSense [45] helps users make sense of literature via an interface that provides a screen with papers citing or cited by the reviewed paper along with the citation contexts that serve as contextual cues. CiteSight extends this work by supporting a variety of different types of citation related tasks, such as the inline identification of targeted reference.

Some citation management systems act as intermediaries between the user and academic search engines (e.g., Google Scholar, CiteseerX and Microsoft Academic Search) [34][45]. For example, ActiveCite [34] provides recommendations by forwarding searches to Google Scholar and integrating the results. Because CiteSight uses its own recommendation engine, it can offer personalized recommendations and take into account the interaction of the user with the system when ranking recommendations.

Personalization is used extensively and effectively in Web search, driven by user features such as demographics [41], session context [4], and past queries [40]. As is the case for web search [38] and browsing [2], we find that citation behavior is consistent within individuals and use this to our advantage. CiteSight leverages features unique to academic writing, such as paper structure, authors, venues, and citation behavior. We are only aware of one other research project that provides personalization for scholarly search [5], in which machine learning is used to refocus queries to an academic search engine. Additionally, reference management and social bookmarking tools like Mendeley, Zotero and CiteULike exploit users' profiles and apply collaborative filtering to enhance their search engine. Google Scholar provides a service that recommends newly published articles to users based on their publication history. While the service offers personalized recommendations, it does so based on past publications only. As such it can help users keep up to date with research that is related to their past research, but it does not support interactive search for papers relevant to current interests. While personalization tailors results to an individual, multiple authors often write academic papers and the citations must reflect all of their backgrounds. Citation search is, in effect, a collaborative search problem [30][33], and CiteSight uses this to personalize (or "groupize" [41]) its results for multiple authors.

Search engines are used for many different purposes, ranging from navigational queries targeted at individual webpages to exploratory searches [23][44]. To address this, recent information retrieval research has explored the optimization of multiple objectives. For example, Bennett et al. [4] personalize search results using short-term search session behavior and long-term search behavior differently based on user behavior. Of particular relevance to CiteSight is the notion of *slow search* [39], which explores how to optimize search for time constraints beyond the traditional tight constraints adhered to by commercial search engines. CiteSight's architecture represents one approach to the slow search problem by operating on a variety of time scales, from milliseconds (when hitting the cache), to seconds (when computing global recommendations), to longer (when collaborating authors leave notes like "Alice: please insert relevant citation here").

Almost all citation recommendation systems require the user to leave the context of a manuscript to find citations. However, systems such as the Remembrance Agent [31] are designed to facilitate associative memory by continuously displaying in situ information that might be relevant to content recently typed by a user. This approach focuses on previously-seen information. We draw inspiration from this approach for the presentation of information in-line with the task but extend it to support background processing. He et al. [15] identify sentences where adding a citation might be appropriate, along with the corresponding citation.

This paper presents a novel system for citation recommendation with a unique, integrated user experience. CiteSight builds on existing citation prediction research by introducing a context-coupling technique to augment what is known about individual papers. It also identifies the appropriate context needed to provide personalized citation recommendations to author groups on varying time scales. This allows CiteSight to support a greater variety of citation related tasks than previous systems, including the inline identification of targeted reference.

3. THE CITESIGHT USER EXPERIENCE

We begin our discussion of the CiteSight system by describing the user experience. In subsequent sections, we explain how the system is implemented, evaluate the quality and speed of the citation recommendations it produces, and discuss how the system works in practice using a small-scale qualitative user study.

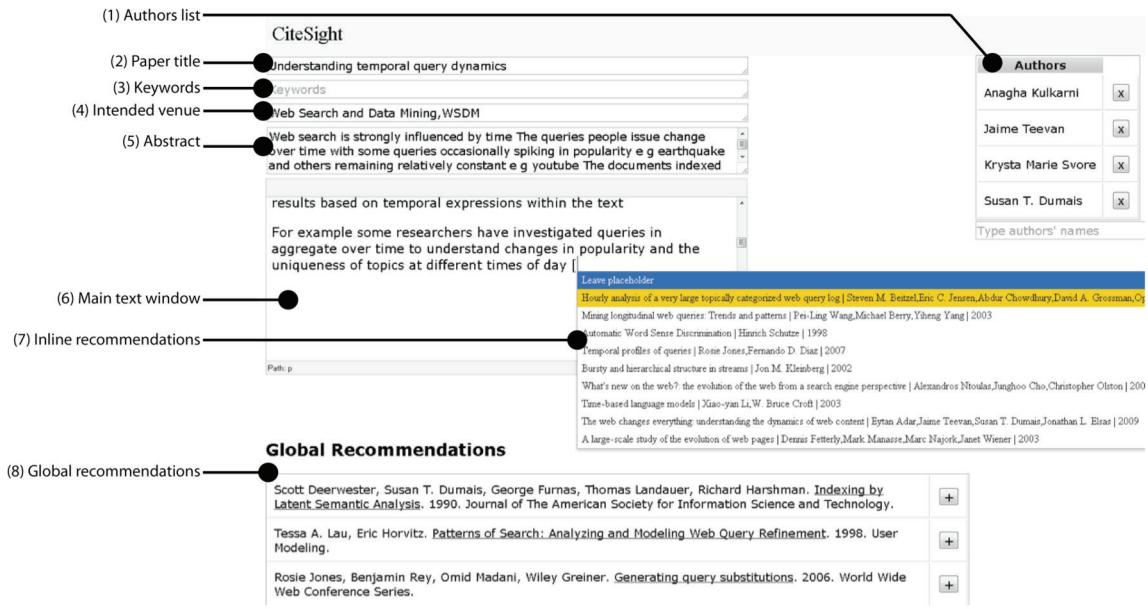

Figure 1. A screenshot of the CiteSight user interface. CiteSight allows the user to enter the (1) paper authors, (2) title, (3) keywords, (4) intended venue, and (5) abstract. When the user opens a citation bracket in the main text window (6), inline citations recommendations (7) are provided based on the paper meta-data and text surrounding the bracket. Global recommendations (8) are also provided. The sample text shown in the figure is from an existing paper, and the golden highlighted reference, which is the first suggested by CiteSight, is the actual citation used by the authors.

The CiteSight backend was designed and implemented as an API so that it could function with a variety of front-ends (e.g., Word or Emacs). For demonstration and evaluation purposes, we implemented a limited, but functional, Web-based prototype, shown in Figure 1. An author creates and edits a manuscript in the main text region (Figure 1(6)). Additional text fields allow the author to enter metadata such as their name and coauthors (1), the title (2), keywords (3), the venue for submission (4), and abstract (5). Although no metadata are required, CiteSight can make use of them to provide the best possible recommendations. Author and venue names are autocompleted from our large database, in part to make this quick for the author, and in part to eliminate ambiguity in which personalized models to load. Various objects (e.g., authors, papers, and venues) are linked to the Microsoft Academic site.

Once entered, the different fields provide context for the citation recommendation process. Whenever a field is changed via the CiteSight client, the new content is fed back to the backend system, and used to populate the cache and generate recommendations. Once the user is satisfied with their work, the system allows authors to export the text with the citations embedded in LaTeX format, along with the bibliography in BibTeX format.

While editing a manuscript, citation recommendations are provided in two forms to support different citation creation tasks: as local inline recommendations (Figure 1(7)), and as global recommendations (Figure 1(8)).

3.1 Inline Citation Recommendations

CiteSight allows authors to easily search for and include the appropriate citation references as they enter text into the main text window shown in Figure 1 (6). CiteSight "queries" consist of the citation content (the entered text) and contextual information (e.g., author, venue, other selected citations, etc.) which are evaluated by the server to generate recommendations. To reduce distractions to the author [7][9], CiteSight recommendations are only triggered when an author opens a citation bracket ('['). This produces a list of up to ten papers in a popup window (Figure 1 (7)). Papers are identified in this list with a short summary that includes title, authors and publication date. The CiteSight user can accept one of the recommendations by selecting it. When a recommendation has been accepted, a citation to the paper is added inline and the paper is added to the bibliography.

Initial recommendations for the inline process rely on the CiteSight cache, and thus can be delivered nearly instantaneously. If the author leaves the recommendation window open, new suggestions are automatically delivered from the full CiteSight index. We informally experimented with mechanisms for delivering these new results, including showing an entirely new result list or appending new results to the end of the existing list. We settled on complete replacement as we found that there was often sufficient time for the author to skim and select from the list before it was changed. Sometimes, the original cache-based recommendations will reappear in this revised list but this is not required.

If the displayed results are unsatisfactory or the author ignores them, they may type additional text to refine the recommendations. As long as the citation bracket is open, the recommendation list is *filtered* using the typed terms. For example, in Figure 1 if the author were to type "Bruce Croft" while the recommendation list is open, the list would be updated to show papers written by that author. Entered text that matches the paper summary is automatically highlighted in the filtered results. If the author accepts a recommendation, the hint text is replaced by the recommendation.

If the author chooses not to accept a recommendation, they can instead leave a placeholder (e.g., "[cite]" or "[add citation]"), and "Leave placeholder" is an explicit option provided in the inline recommendation drop-down box. In this way, the system naturally supports leaving citation-related notes to oneself or one's collaborators (e.g., "[Alice: please add reference]"). While placeholders help remind the authors to return, they further serve as a hint to the system that it should continue to search for relevant recom-

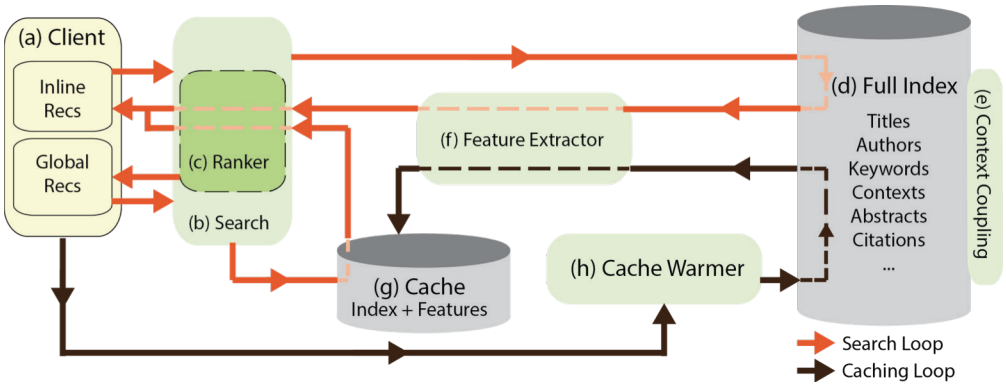

Figure 2. A diagram of the CiteSight citation recommend process. When the user enters paper content into the CiteSight client (a), a search is triggered (b) that identifies the most relevant papers using the full index (d) and cache (g).

mendations within the given context, identifying content that may take significant time to find. Author can receive new citation recommendations by returning to the placeholder and clicking on it. For placeholders that include a request for a coauthor, when that coauthor begins editing they can receive recommendations personalized to their citation history.

3.2 Global Citation Recommendations

The CiteSight interface also provides global recommendations to support the exploration of generally relevant papers (Figure 1(8)). The motivation is to offer a service similar to e-commerce's "customers who bought this item bought these items as well" to help the authors identify potentially unknown papers.

High-level metadata, including title, authors, and venue, is provided for each recommended paper. The CiteSight user can click on any of the globally recommended papers to see the corresponding entity page on the academic search site and download the paper, if available. The global recommendation list is dynamically updated as the user progresses through paper editing or the system discovers interesting citations through deeper, offline analysis.

CiteSight currently restricts the number of updates to the global recommendation list by ignoring certain changes. In particular, global recommendations are not identified based on the text used within the body paper, as the local inline recommendations are. Instead, they are based on meta-data and co-citation patterns, with papers boosted that were frequently co-cited with papers already cited in the manuscript. This process is described in greater detail in Section 4.3.2. We believe it is worth exploring different mechanisms for displaying new results to find an optimal solution that is non-disruptive but ensures high precision and recall.

4. THE CITESIGHT SYSTEM

We now discuss how the CiteSight system is implemented. An illustration of the CiteSight recommendation process is shown in Figure 2. As an author enters content in the client (a), that information is constantly sent to the server. For example, when an author edits their manuscript, or enters metadata (e.g., authors or title), the server receives the new content. The server then issues this content as a semi-structured query (b) to a set of indices (d+g) to generate a ranked list of recommended citations (c). This process consists of two procedures: candidate retrieval and ranking (as seen in the *search loops* in the figure). The ranking procedure is applied up to two times: once on cached papers and again after more candidate papers are retrieved from the full index. A second piece of CiteSight (labeled as *caching loop*) constantly warms (h) the cache (g) with papers and pre-calculated features.

4.1 The CiteSight Index

CiteSight uses Lucene [22] to index a corpus of academic papers provided by Microsoft Academic. Our prototype includes 2.3M Computer Science papers from 1970-2010 (inclusive), written by 19 million disambiguated authors and published at 20,077 venues. We include only papers that were cited 3 or more times to reduce noise and improve performance.

Microsoft Academic's database provides citation information, including the citations from one paper to another and the specific sentences in a paper in which the reference occurs (which we refer to as the citation context). For example, a paper A may be cited by another paper B. The authors of B include a sentence such as: "Smith et al., described the first use of the ABC procedure [A]." This sentence is one of the citation contexts for paper A (clearly, a paper may have many citation contexts). Previous research has found that using a paper's citation context to identify relevant papers can lead to higher recommendation accuracy than using the paper's abstract or full-text [16]. These contexts directly map to our task as it is possible that two authors—a previous one, and our CiteSight user—will refer to the same paper in similar ways. To leverage the different text elements, we index the title, abstract, and citation contexts of each paper.

In addition to these textual features, CiteSight uses additional indexed *paper metadata* for ranking including authorship, citation count, publication venue, and references (both incoming and outgoing). Additionally, we store *author metadata* including the author's name, published papers, and number of times each paper was cited (e.g., to calculate H- or G-index style statistics for an author). Finally, we record *venue metadata* including the venue-to-venue citations and co-citations and rough estimates of venue reputation (e.g., a venue-based H-index).

CiteSight was designed to also include personal paper repositories (e.g., CiteULike, Endnote, or Zotero). While these would likely improve recommendation performance, a large quantity of such personal repositories was not available to us. Thus, in this paper, we focus exclusively on the academic search engine data.

4.1.1 Citation Context Coupling

Citation context has been shown to be valuable for identifying relevant citations [16]. However, many papers have very limited citation data [8]. In order to enrich citation contexts for rarely-cited papers, we introduce a novel technique, context-coupling, to impute citation contexts. Context-coupling effectively imputes citation contexts for low-citation papers by leveraging the con-

texts of similar papers. For our purpose, the similarity of papers is determined by the structure of the citation network.

Consider for example the citation network illustrated in Figure 3. Document A is the paper "A proximity language model for information retrieval" and paper B is "An exploration of proximity measures in information retrieval." The two documents are deemed similar to each other as they are *co-cited* several times. For example, paper D, "A proximity probabilistic model for information retrieval" mentions both A and B. In our hypothetical example, A has only a handful of citation contexts which we would like to expand to better describe paper A. To do so, we wish to propagate to A some of the contexts describing similar papers. For example, paper E "Investigation of partial query proximity in web search" cites B using the following text—*"... In addition to the traditional criteria concerning individual query terms such as tf and idf, the proximity between query terms in a document is often believed to be a useful criterion for document ranking..."* This citation context could also adequately describe paper A and is a good candidate for *coupling*.

The context-coupling algorithm consists of two steps. First, for each paper we find the most similar co-cited papers. We utilize the Adamic-Adar [1] similarity metric computed over the incoming links (as we wish to identify papers that are referenced similarly). In the next step, we scan all the citation contexts to each of the most similar papers identified in step one. We compare each citation context with the citation contexts already available for the target paper and copy them, treating them as if they were attributed to the target paper. The similarity of each citation context is computed using cosine similarity over the TF-IDF representation of the citation context and the content originally available for the target paper (receiving equal weight in the prototype)

CiteSight uses context-coupling to enrich citation context when the papers are indexed. We also experimented with a variant of forward aggregation which was proposed by Metzler et al. [26]. In this approach, content is propagated transitively (e.g., G→A in Figure 3). However, unlike Metzler et al.'s approach, where all contexts are copied, we propagate only the most similar contexts.

4.2 The CiteSight Cache

The resulting index is large. In particular, it is large enough that it is impossible to pre-compute and keep updated all of the contextual meta-data features for each paper in the index that that system uses for ranking. Identify relevant papers in the index and computing the related meta-data features in response to a query can take half a second or longer. This is too slow to support real-time interaction. Existing research suggests that for a computer's response to appear instantaneous, it must respond in less than 100 milliseconds [27]. To support a faster response time, CiteSight uses a cache with a dynamic *cache warming system* (see the cache "loop" in Figure 2). By restricting the number of papers in the cache (~1000 to 2000 in a given session) and pre-calculating various features as papers are loaded into the cache, response rates can be pushed to under 10 milliseconds.

As is the case for the primary index, papers in the cache are indexed by Lucene using their title, abstract, and citation context. However, the meta-data related features (see Section 4.3) used for ranking are pre-computed for each paper. The cache contains the subset of papers that are identified as likely to be relevant to the author and paper being written. While we may like to identify those papers that an author is familiar with this is not realistic. Instead, we optimistically cache those papers that are likely relevant, erring on the side of recall.

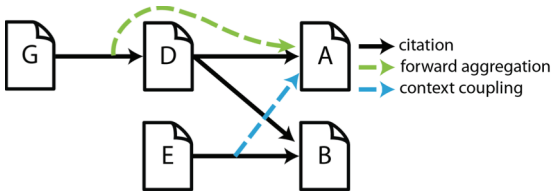

Figure 3. An illustration of the context-coupling approach. To enrich the citation context available for a paper, we impute citation contexts for low-citation papers by leveraging the contexts of similar papers

The cache warming subsystem (Figure 2 (h)) monitors various user-driven events and dynamically updates the cache. For example, when an author begins using CiteSight they enter their names and the names of their co-authors. For every author in the authors list, the cache warmer finds all papers that were cited at least three times in the past by that author are included. The content of the cache is updated as the paper is written. Whenever the title, abstract, or keywords of the current manuscript change, the system issues the new content in the background as a query to the larger index of papers and collects the hundred most relevant results to add to the cache. Finally, whenever the user chooses to add a paper to the bibliography, all the co-citations of that paper and its references are added to the cache as well

4.3 Retrieving Suggested Papers in CiteSight

We now look at how the CiteSight index and cache are used to retrieve inline and global recommendations.

4.3.1 Retrieving Inline Recommendations

When an author opens a citation bracket, CiteSight uses the 50 words before and after that bracket as the citation context. Words beyond the limits of a paragraph, as determined by line breaks, are ignored. The citation context is used to search the cache (line 2 in Figure 2, and, at the same time, to search the larger primary index (line 5). The cache-based recommendations offer an instant response while less familiar recommendations are being examined. Once the top 500 papers are retrieved from the primary index and the appropriate features are computed, they are ranked along with the cached candidates. As described above, there are a number of different possible strategies to integrate these results for display in the interface (e.g., replacement, interleaving, etc.). For the purposes of the prototype, we opted to use the replacement strategy.

The initial cached candidates and, later, the full set of candidates, are ranked using ranker learned using gradient boosted regression trees (GBDTs) [11][46]. GBDTs produce a prediction model in the form of a collection of decision trees. The approach is widely used, adaptable, interpretable, and produces highly accurate models. Additionally, in our experience they performed better than linear regression or support vector machines.

We utilized the following features as input to the learner:

Citation context similarity: This feature measures the similarity between the text an author has just entered into CiteSight and any text that has been used in the past to cite the candidate paper using a standard cosine similarity (over a TFIDF weighted term vector). The intuition behind this feature is that new citation contexts are likely to repeat old patterns when referring to the paper. For cached recommendation, a citation context similarity score is computed for all candidates, as they typically number less than 200. However, as the candidate set may be much larger when retrieved for the full corpus, only the top 500 relevant candidates (identified using Lucene's built-in TFIDF/vector-based, ranking scheme) are scored; all other candidates receive a score of zero.

Title/Abstract/Keywords similarity: To capture the textual similarity of a paper to the current manuscript we measure the similarity between the title, abstract and keywords of each. Each of these three fields is held in a separate Lucene field. As before, similarity is computed as cosine similarity over their TFIDF vector.

Citation count: As popular papers are likely to remain popular and be cited again, this feature counts the number of citations a paper attracted from all the papers in the corpus. Numerous citations often indicate the candidate paper is important and that authors should at least have awareness of it.

Author similarity: This feature measures the similarity between all authors of the manuscript and the authors of a candidate paper. The similarity is computed as the Jaccard index between the two author sets. A different way to think of this feature is as a self-citation indicator that boosts papers written by the authors of the current paper. This function can be biased so that the current author (i.e., the one editing) weighs more heavily.

Author history: For each candidate paper, we count the number of times it was cited in the past by the authors of the current manuscript. We set this feature to be the mean number of times the authors (if there are more than one) cited the candidate paper. This feature naturally boosts the score of papers that were cited by the authors in the past and reflects the tendency of an author to consistently cite the same papers [43].

Venue relevancy: This feature measures how relevant the venue is in which a candidate paper was published to the venue to which the currently edited manuscript is targeted. The value of this feature is set to be the frequency at which papers published in the targeted venue cite papers published in the venue of the candidate paper. This feature captures the tendency of authors to cater their results to specific outlets [13].

Citations from cited papers: To boost the scores given the current editing session, we measure the number of times the candidate paper was cited by papers that were already added as references to the manuscript (averaged over these references). This supports the chaining that sometimes happens as an author explores the references of references.

Co-citations with cited papers: Similarly to chaining, we measures the number of times the candidate paper was co-cited with previously selected references. This feature boosts candidates in the spirit of "authors who cited X also cited Y." We expect this feature to be superior in cases where new papers extend previous work, making the earlier work obsolete.

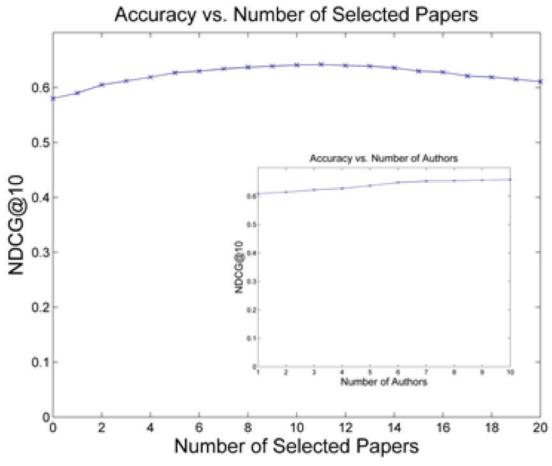

Figure 4. Recommendations accuracy vs. number of previously selected papers (top) and number of authors (inset). CiteSight performs better as authors and citations are added, up to a point. Having a bibliography larger than 11 papers does not lead to additional improvement

While some features are stable (e.g., author similarity, venue relevancy, etc.), others change as the author modifies the paper (e.g., citation context similarity or features that take into account other papers cited in the session). Practically, this means that while all features can be computed and cached, some do require recalculation if the paper changes. However, because the cache is relatively small, recalculating these invalidated features is not expensive.

4.3.2 Retrieving Global Recommendations

In addition to inline recommendations, CiteSight also suggests citations that are relevant to the entire paper. There are a number of ways global recommendations can be identified, including by searching the full index. However, given the papers in the cache are intended to be broadly relevant to the edited paper, we found that we could use the cached papers to identify global recommendations. To do this, we simply utilize the cache without the citation context similarity score feature. The remaining features are again used to rank the documents using the gradient boosted decision tree described above, and the top results are shown to the user at the bottom of the interface. Though we might lower the importance of different similarity metrics to provide more "serendipity" or "diversity" in these results, further work is necessary to find the balance of relevance to serendipity/diversity.

4.4 Summary of the CiteSight System

In summary, CiteSight indexes a corpus of academic papers using paper meta-data and citation context, and uses this index to identify citation recommendations. Although the citation context available for any given paper can be sparse, the system enriches what is available using context-coupling. This results in an index that contains valuable information, but that it is too large to provide the instantaneous response needed for a real-time text editing tool. For this reason, CiteSight maintains a cache that monitors various user-driven events and dynamically updates so that it can make the most relevant papers available on a moment's notice. This cache is further used to identify the most relevant global citations.

5. EVALUATION

Because the global citation problem is well studied in previous work, we focus on evaluating the quality of the inline recommendations. We show that citation context coupling significantly improves the quality of the inline recommendations the system provides, and that citation-related features are particularly valuable for ranking. Personalization appears very important, with the citation history of authors contributing significantly and performance improving with more authors. We also look at the influence of the cache on the differential search experience, finding the cache provides accurate recommendation in many cases with minimal latency. In the next section, we present feedback from users of the CiteSight system. The server used for both training and testing was a 12-core server (2 Six-Core AMD Opteron(tm) Processor 2431 2.4GHz with 48Gb of RAM).

5.1 Experimental Design

We evaluated the recommendations made by the CiteSight system by looking at how well it would have performed for existing papers where the set of citations is already known. Arguably, this serves as a reasonable ground truth as references from these papers were deemed relevant to the paper by the expert authors who chose to cite them. The reality, however, is that authors must often balance space requirements against the need of citing all relevant

Table 1. CitesSight performance given different features used for ranking, using ranking by citation context as a baseline. Text features tended to perform worse than features related to the author or citation structure. Each row below context signifies context + feature (e.g., Context + keywords, Context + title, etc.)

Features	Type	NDCG@10
Context (baseline)	Text	46.50%
+ keywords	Text	46.50%
+ title	Text	46.60%
+ authors similarity	Author	47.50%
+ abstract	Text	47.80%
+ citation count	Citation	48.60%
+ venue relevancy	Venue	49.20%
+ citations	Citation	53.00%
+ co-citations	Citation	56.70%
+ authors history	Author	57.60%
+ all	All	61.90%

papers. Because of this, many relevant and appropriate recommended citations would be judged as incorrect. We believe that in the absence of complete data, a fairer test is one that also includes possible "replacement" or "augmentation" citations as relevant. That is, a citation to paper A might be replaced or augmented with a citation to paper B. To identify these automatically, we use the idea that both replacement and augmented citations are likely to have been co-cited in the past with the "true" reference. We distinguish between "true" citations and "secondary" citations. True citations (reference A in the example above) are those citations that actually exist in the source paper. These receive a relevancy score of 1 (the max). Secondary citations (reference B) are those that co-cited with the true cite. They receive a relevancy score proportional to the number of times the secondary paper is co-cited with the true citation ($0 \leq \text{score} \leq 1$).

We created a test dataset of 1000 randomly selected CS papers published in 2011, the year after the last paper included in the CiteSight corpus. Only 0.1% of all citations in these papers referenced a paper published in 2011, and we omit these references. Papers were required to have between 20 and 40 references as a rough way to eliminate papers that were incorrectly parsed for citations. The median number of citations made by papers in the test set was 25. The mean and median length of the citation contexts for this dataset were 158.3 and 153 characters respectively. On average, all papers in the dataset had 4.3 citation contexts.

We used a 5-fold cross validation approach in which 80% of the papers were used for training and the remaining were used for testing. In the training phase we used the candidate papers of all the references of the training paper to train a gradient boosted regression trees model. We then applied the model on the candidates of each reference of the test papers and ranked the candidates based on the model's prediction.

To measure the performance of the system, each paper's references were considered independently using a normalized discounted cumulative gain metric. DCG and NDCG are defined as:

$$DCG_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log(i)}; NDCG_p = \frac{DCG_p}{IDCG_p}$$

where $IDCG_p$ is the ideal DCG_p, achieved by optimally ranking the retrieved documents and p is the number of documents used for evaluation (5 or 10 in our case).

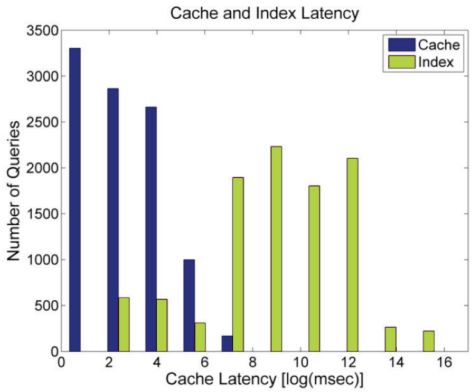

Figure 5. The log latency of the cache and index. Searching the cache is much faster than searching the index, with a median latency of 6.2 milliseconds for the cache, compared with 452 miliseconds for the index.

In calculating the gain, only one document (the true citation t) receives a gain of 1 (the perfect relevancy score, i.e., $rel_t = 1$). All secondary (s) citations—those co-cited with the true citation in the corpus and represent the replacement or augmentation described above—receive a relevancy score of:

$$rel_s = \frac{number\ of\ papers\ citing\ s\ \&\ t}{number\ of\ papers\ citing\ s}$$

5.2 Inline Recommendation Quality

Table 1 shows the performance of different models compared to the baseline model. Augmenting the paper using the three most relevant "coupled" contexts improves accuracy from 40.8% (not shown in the table) to 46.5% (the forward aggregation scheme only offers 41.8%). As past work has demonstrated the effectiveness of citation-contexts we utilize this feature (with context-coupling) as a more realistic "baseline." We compare the performance of models that use different features independently to understand the contribution of each feature. We found that the single most important feature is the context similarity, particularly after applying the context-coupling technique.

We also find the citation history of authors and co-citations with selected references lead to the biggest improvement in performance over the baseline model. The abstract, title and keywords similarities offer negligible contribution beyond the context similarity, supporting findings of previous work [16]. Another interesting observation is that the contribution of citations by selected references is notably higher than the contribution of global citation count. This suggests that global visibility is not as important as visibility within the topical community of the paper. The combination of all features yields better results—15.4% over baseline.

Our system aims to improve its recommendation as the user interacts with it. Most notably, it uses the previously selected references to rank future candidates. Figure 4 (large image) shows the accuracy of the recommendations as a function of the number of previously selected papers. There is a steady increase in accuracy as more papers are being cited up to the eleventh paper (peaking at a value of 64.2%, compared with 58% at the first recommendation). While the slope of the line appears small, the change is meaningful to the final result. Around that point, having larger bibliography does not lead to additional improvement. This finding suggests performance could be further improved if users could express the relevance of each selected reference (or this could be inferred given the way a citation is used or where it is placed in the paper). The number of authors also affects citation recommen-

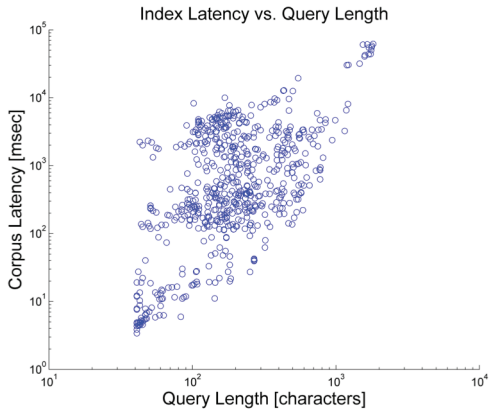

Figure 6. Index latency versus query length. While long queries did not affect cache response time, they resulted in higher latency when querying the full index.

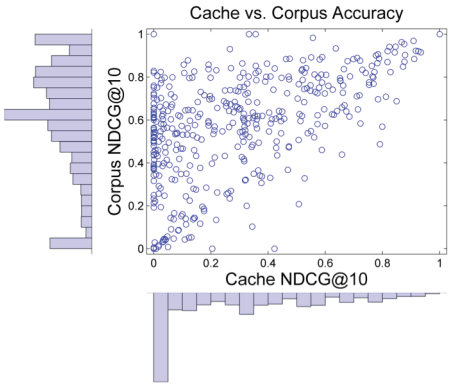

Figure 7. Recommendation accuracy of the cache and full index. The cache account for about half of the NDCG@10 achieved by the index, and in some cases the cache actually provides better recommendations.

dations as Figure 4 (bottom) shows. Having more authors also leads to better recommendations as the most predictive feature—citation history—becomes more reliable.

5.3 Impact of the Cache

Next we study the effect of the cache. First we measure the time it takes to retrieve the results. Figure 5 shows the latency of the cache and the index. Not surprisingly, the cache is much faster. The median cache latency is 6.2 milliseconds whereas for querying the index it is about 452 milliseconds. Note that these rates were achieved where both client and server were on the same local-network, with an unloaded server. However, while the cache results are well below the desired 100ms, additional optimization may be desirable as the system scales to more users.

As expected, long queries resulted in higher latency when querying the full index (see Figure 6) and the correlation between the two was 0.77. However, the length of the query did not appear to affect cache response time, we found no correlation between the two. This is likely due to the fact that the cache index was loaded into memory on the server and any differences were slight relative to network time. As we discuss later, long queries are another instance where slow search [39] might be beneficial.

Of course, the cache will be useless if the recommendations it generates are irrelevant. Figure 7 shows a scatterplot of the cache and corpus accuracy for each query, along with the histogram of cache and corpus accuracies. On average, the recommendations of the cache account for about 45.5% of the NDCG@10 achieved by the corpus. In about 9.4% of the cases the cache in fact provides better recommendations that match the actual citation more closely than the full corpus. These may be cases where authors preferred citing a familiar paper or "venue-suitable" rather than the best fit. Overall, the cache provides accurate recommendation in many cases with minimal latency.

5.4 Limitations

Our evaluation is focused exclusively on Computer Science papers. Other domains, with different citation behaviors, may have different results. We believe that the citation distributions and sparseness of contexts is ubiquitous in the scientific literature but additional work is needed to validate this.

Our evaluation does not extensively evaluate the impact of network latency as we test on high-speed, unloaded connection or local searches. However, as the amount of actual data transferred is often less than 2k we believe that we can stay below the 100ms requirement even on more significantly loaded networks. In terms of scaling, we note that while GBRTs are complex, training on the entire corpus (not only the 1000 documents) can be achieved in well under 24 hours on our modest hardware. Any scaling-related performance issues can likely be handled without extensive infrastructure but this requires additional validation.

Finally, while we believe that NDCG is an appropriate metric there are others suited for our evaluation task. Selecting the criteria to optimize will require more extensive real-world use that would allow us to identify desired service characteristics.

6. USER FEEDBACK

In addition to evaluating CiteSight's performance, we also asked five participants to use the system and provide semi-structured feedback. All participants were PhD students in the Computer Science department at the University of Michigan. Because the students did not necessarily have a sufficient number of their own papers for effective personalization we asked each participant to choose a paper written by someone else that they were familiar with. First each participant entered in the title, abstract, venue, keywords, and authors of the paper they selected. Next, each participant was instructed to type in a paragraph from the text that cited one or more papers they were familiar with, soliciting recommendations from the system. We asked participants to answer structured questions and provide open-ended reflection on their impressions of the system (e.g., would you be interested in a system/plugin that provides dynamic recommendations? What properties/features are most important in such a system? etc.).

Prior to using the system, participants expressed considerable interest in having a system that provides dynamic recommendations for citations. They stated that the most important properties would be "*accuracy of recommendations*" (P1, P3, P4) and "*speed of response*" (P2), highlighting the importance of the challenging latency and accuracy trade-offs that CiteSight's differential search approach is designed to support. Some participants also mentioned "*coverage of papers*" (P1, P2) and "*the capability of handling synonyms and higher-level semantics*" (P2) as secondary requirements. Such functionality is likely to require additional processing, and would fit well into the differential search session.

During use, CiteSight often correctly recommended the actual citation for the participant's selected paper as the top result or among the top results. The other recommendations, while not the actual cited paper, also tended to be relevant (e.g., "*The expected paper was not returned in my case. But other recommendations were highly relevant.*"). Participants identified these recommenda-

tions as relevant to the topic of the paper, and, sometimes, to the specific sentence being used as context. However, it is difficult to evaluate the quality of these alternatives because the participants can only make assumptions about what the authors would have done. In some cases CiteSight failed to find the actual citation used in the model text due to limited index coverage (we simply did not have the paper metadata).

After being shown the system, participants were asked what additional features or extensions they would like to see it support. Their replies highlight the variation in how people perform citation work, with different access points being important. For example, one participant (P3) asked for, *"A LaTeX file parser"* to automatically extract metadata and another (P4) wanted to *"be able to use [CiteSight] with SubLime text and Latex plugin."* Such extensions are possible using CiteSight's API.

One limitation to CiteSight that became apparent during use is that the system can only rely on context that has already been typed by the user. Sometimes this text was insufficient to make an accurate prediction, for example, where the author opened a bracket at the start of the sentence (e.g., "In […"). While the system does allow the author to leave an empty set of brackets and return to them, this may be unnecessarily disruptive. However, brackets may not be inherently necessary. One participant (P2) suggested a feature to provide recommendations for both papers and citation placement: "*I would add suggestions on places to insert citations, so that the user does not have to find and create a bracket by himself*." This could be done by matching global recommendations to particular citation contexts, or running a series of background queries using different subsets of the paper text.

7. DISCUSSION

The CiteSight system demonstrates a mechanism for disentangling recommendations given different tasks and work modalities. It attempts to provide fast and accurate responses for in-line queries and slower, contemplative recommendations when those are appropriate, either slowly populating the auto-complete dialog, or suggesting further reading. Based on user feedback and our own experience we have begun to explore other applications within the differential search framework. Many of these can continuously run in the background making unobtrusive or on-demand recommendations. For example, authors going through a self-audit before submission could benefit from background analysis that is surfaced to indicate possible missing references or updated results that have been found on the Web or through crowdsourcing. Alternatively, authors wishing to save space in their paper may benefit from an analysis that finds review articles that covers many of the cited articles or suggests possible cuts.

From an engineering perspective, we have also begun to explore how background tasks can be used to anticipate end-user requests. While an end-user may be willing to tolerate a delay for a more complex search, they would likely not complain if the response were fast. It may be possible to achieve this without significant costs in accuracy by pre-querying. For example, the system could query the index every 5 words typed by the user. When the user asks for local recommendation, the system will automatically show the recommendations retrieved from the last background search and update the recommendations once the "live" background search completes. This is particularly true for long queries that take a lot of time to be processed (see Figure 6). If implemented correctly, the user may not even need to type a bracket.

The notion of utility as expressed through cost/benefit analysis in mixed-initiative systems [17][35] may be a useful model for learning which search or recommender modalities are appropriate for which tasks. However, it is not always clear how different utility models may perform in our particular setting. Academics may find recommendations of value even if they are not immediately useful for the given context but that value may vary greatly depending on time pressures and other competing interests.

We believe that CiteSight is an exemplar solution of a larger set of problems on how human and search (or recommendation) systems can be integrated. Search engines are often engineered to be as fast and as responsive as possible and consequently users have become accustomed to search results being show up in a fraction of a second. Seemingly negligible increases of less than half a second in response time decreased user engagement dramatically [32]. However, our experience, and continued evidence from the IR community [39], is that "fast" is not the only pertinent criterion. Task requirements may demand stability (for refinding), novelty (for recommending a new movie to watch), freshness, appropriate synthesis (finding contrasting opinions or identifying consensus in search results), and context appropriateness. Notably, many of these tasks are interleaved by any single end-user requiring an adaptiveness that is not currently observed in bulk search or recommendation engines. Exploratory searches, medical research, vacation planning and scientific literature review are some examples for search instances where users might be more patient and would value higher quality results over search speed. In this world, latency is no longer the most pertinent constraint.

By architecting systems with defined tasks and expectations in mind, in our case by creating a dynamic cache, we believe it is possible to address multiple needs differentially while respecting existing work practice. Note that for the most part, we do not utilize different search systems or have over-fit each component to the task it is intended to support. Rather, we have opted for a single ranking strategy that is focused on different datasets (i.e., likely known versus likely unknown) and for which results are surfaced in different ways.

8. CONCLUSION

In this paper we presented CiteSight, a system that dynamically recommends citation to academics as they edit a manuscript. The system is designed to support both in-line, where instantaneous results are expected, and offline tasks that can be computed in the background. A critical component that enables the versatility of CiteSight is the cache, that maintains a personalized "session profile" with papers ready for instant retrieval. Context-coupling further enhances our index to support better recommendations for uncommon papers. In a preliminary user study we identified response time and recommendation accuracy as the two most important properties of the system. We empirically evaluate our system using a large dataset of papers, focusing on those aspects. We found that using the cache the system is able to provide personalized recommendations instantly (<10 ms) while more diverse results are retrieved in the background.

Acknowledgements

We would like to thank Kevyn Collins-Thompson for useful discussions and the Microsoft Academic team for providing us with data, advice, and suggestions. This work is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20155. The U.S. government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies

or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

References

[1] Adamic, L.A. and Adar, E., Friends and neighbors on the Web. *Social Networks.* (25)3:211—230, 2003.

[2] Adar, E., Teevan, J. and Dumais, S.T., Large scale analysis of web revisitation patterns. *CHI'08*, 1197—1206, 2008.

[3] Baldi, S. 1998. Normative versus social constructivist processes in the allocation of citations: A network-analytic model. *American Sociological Review* 63(6):829—846, 1998.

[4] Bennett, P.N., White, R.W., Chu, W., Dumais, S.T., Bailey, P., Borisyuk, F. and Cui X., Modeling the impact of short- and long-term behavior on personalized search. *SIGIR'12*, 185—194, 2012.

[5] Bethard, S. and Jurafsky, D., Who should I cite: learning literature search models from citation behavior. *CIKM'10,* 609—618, 2010.

[6] Bornmann, L. and Daniel, H.-D., What do citation counts measure? A review of studies on citing behavior. *Journal of Documentation*, 64(1):45—80, 2008.

[7] Budzik, J. and Hammond, K.J., Watson: Anticipating and contextualizing information needs. *ASIS'99,* 727-740, 1999.

[8] Clauset, A., Shalizi, C R., Newman, M.E.J., Power-law distributions in empirical data. *SIAM rev.* 51(4):661-703, 2009.

[9] Dumais, S.T., Cutrell, E., Sarin, R. and Horvitz, E., Implicit queries (IQ) for contextualized search. *SIGIR'04*, 594, 2004.

[10] Erosheva, E., Fienberg, S. and Lafferty, J., Mixed-membership models of scientific publications. *PNAS* vol. 101, suppl. 1, April 6, 2004.

[11] Friedman, J.H., Greedy function approximation: A gradient boosting machine. *Anl. of Statistics,* 29(5):1189-1536, 2001.

[12] Gipp, B., Beel, J. and Hentschel, C., Scienstein: A research paper recommender system. *ICETiC'09*, 309—315, 2009.

[13] Harwood, N., Publication outlets and their effect on academic writers' citations. *Scientometrics,* 77(2):253—265, 2008.

[14] He, J., Nie, J., Lu, Y. and Zhao, W., Position-aligned translation model for citation recommendation. *SPIRE '12*. 251—263, 2012

[15] He, Q., Kifer, D., Pei, J., Mitra, P. and Giles, C.L., Citation recommendation without author supervision. *WSDM'11,* 755—764, 2011.

[16] He, Q., Pei, J., Kifer, D., Mitra, P. and Giles, L., Context-aware citation recommendation. *WWW'10*, 421—430, 2010

[17] Horvitz, E., Principles of mixed-initiative user interfaces. *CHI'99*, 159—166, 1999.

[18] Huang, W., Kataria, S., Caragea, C., Mitra, P., Giles, C.L. and Rokach, L., Recommending citations: Translating papers into references. *CIKM'12*, 1910—1914, 2012.

[19] Kataria, S., Mitra, P. and Bhatia, S., Utilizing context in generative Bayesian models for linked corpus. *AAAI*, 36, 2010.

[20] Küçüktunç, O., Saule, E., Kaya, K. and Çatalyürek, Ü. V., Diversified recommendation on graphs: Pitfalls, measures, and algorithms. *WWW'13*, 715—726, 2013.

[21] Lu, Y., He, J., Shan, D. and Yan, H., Recommending citations with translation model. *CIKM'11*, 2017-2020, 2011.

[22] Lucene, https://lucene.apache.org/

[23] Marchionini, G. Exploratory search. *CACM*, 49(4):41—46, 2006.

[24] McFarlane, D.C., Latorella, K.A., The scope and importance of human interruption in human-computer interaction design, *Human-Computer Interaction,* (17)1:1—61, 2002.

[25] McNee, S.M. et al., On the recommending of citations for research papers. *CSCW'02*, 116—125, 2002.

[26] Metzler, D., Novak, J., Cui, H. and Reddy, S., Building enriched document representations using aggregated anchor text. *SIGIR'09*, 219—226, 2009

[27] Miller, R.B., Response time in man-computer conversational transactions. *AFIPS FJCC '68*, 267—277, 1968.

[28] Nallapati, R.M., Ahmed, A., Xing, E.P. and Cohen, W.W., Joint latent topic models for text and citations. *KDD'08*, 542—550, 2008.

[29] Pedregosa, F. et al., Scikit-learn: Machine Learning in Python . *J. of Machine Learning Research* 12:2825-2830, 2011.

[30] Pickens, J., Golovchinsky, G., Shah, C., Avarfordt, P and Back, M., Algorithmic mediation for collaborative exploratory search. *SIGIR'08*, 315—322, 2008.

[31] Rhodes, B.J. and Starner, T., Remembrance Agent: A continuously running automated information retrieval system. *PAAM'96*, 487—495, 1996.

[32] Schurman, E. and Brutlag, J., Performance related changes and their user impact. *Velocity'09*, 2009.

[33] Shah, C. and Gonzáles-Ibáñez, R., Exploring information seeking processes in collaborative search tasks. *ASIST*, 47(1):1—7, 2010.

[34] Shaoping, Z., ActiveCite: An interactive system for automatic citation suggestion. Master's Thesis, National University of Singapore, 2010.

[35] Shilman, M., Tan, D. and Simard, P., CueTIP: a mixed-initiative interface for correcting handwriting errors. *UIST'06*, 323—333, 2006.

[36] Strohman, T., Croft, W.B. and Jensen, D., Recommending citations for academic papers. *SIGIR'07*, 705—706, 2007.

[37] Tang, J. and Zhang, J., A discriminative approach to Topic-Based citation recommendation. *PAKDD'09*, 572-579, 2009.

[38] Teevan, J., Adar, E., Jones, R. and Potts, M.A.S., Information re-retrieval: repeat queries in Yahoo's logs. *SIGIR'07*, 151—158, 2007.

[39] Teevan, J., Collins-Thompson, K., White, R.W., Dumais, S.T. and Kim, Y., Slow Search: Information Retrieval without Time Constraints. *HCIR'10*, 2010.

[40] Teevan, J., Dumais, S. and Horvitz, E., Personalizing search via automated analysis of interests and activities. *SIGIR'05*, 449—456, 2005.

[41] Teevan, J., Morris, M. and Bush, S., Discovering and using groups to personalize search. *WSDM'09*, 15—24, 2009.

[42] Torres, R., McNee, S.M., Abel, M., Konstan, J. A. and Riedl, J., Enhancing digital libraries with TechLens+. *JCDL'04*, 228—236, 2004.

[43] White, H., Authors as citers over time. *JASIST*, 52(2):87—108, 2001.

[44] White, R.W., Drucker, S.M., Marchionini, G., Hearst, M. and Schraefel, m. c., Exploratory search and HCI. *CHI'07* , 2877—2880, 2007.

[45] Zhang, X., Qu, Y., Giles, C.L. and Song, P., CiteSense: supporting sensemaking of research literature. *CHI'08*, 677—680, 2008.

[46] Zheng, Z., Zha, H., Zhang, T., Chapelle, O., Chen, K. and Sun, G., A general boosting method and its application to learning ranking functions for web search. *NIPS'07*, 1697—1704, 2007.