A Source Independent Framework for Research Paper Recommendation

Cristiano Nascimento[1] Alberto H. F. Laender[1] Altigran S. da Silva[2]
Marcos André Gonçalves[1]

[1]Universidade Federal de Minas Gerais
Departamento de Ciência da Computação
Belo Horizonte, MG, Brazil
{crist, laender, mgoncalv}@dcc.ufmg.br

[2]Universidade Federal do Amazonas
Departamento de Ciência da Computação
Manaus, AM, Brazil
alti@dcc.ufam.edu.br

ABSTRACT

As the number of research papers available on the Web has increased enormously over the years, paper recommender systems have been proposed to help researchers on automatically finding works of interest. The main problem with the current approaches is that they assume that recommending algorithms are provided with a rich set of evidence (e.g., document collections, citations, profiles) which is normally not widely available. In this paper we propose a novel source independent framework for research paper recommendation. The framework requires as input only a single research paper and generates several potential queries by using terms in that paper, which are then submitted to existing Web information sources that hold research papers. Once a set of candidate papers for recommendation is generated, the framework applies content-based recommending algorithms to rank the candidates in order to recommend the ones most related to the input paper. This is done by using only publicly available metadata (i.e., title and abstract). We evaluate our proposed framework by performing an extensive experimentation in which we analyzed several strategies for query generation and several ranking strategies for paper recommendation. Our results show that good recommendations can be obtained with simple and low cost strategies.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*information filtering, retrieval models, search process*; H.3.7 [**Information Storage and Retrieval**]: Digital Libraries

General Terms

Algorithms, Experimentation

Keywords

Content-Based Filtering, Recommender Systems, Information Seeking, Digital Libraries

1. INTRODUCTION

Bibliographical search is an essential task performed daily by thousands of researchers around the world. This task consists of finding and analyzing many scientific papers in order to find a set of works that might be helpful to develop a specific research. This set of papers might contain, among others, the formalization of a given research problem, a technique useful to be applied to a specific facet of a problem, information to help to understand the context of the research, the history of development of the field or a comparative study among existing techniques.

A typical strategy used by many researchers in this task is to follow a list of references in papers they already possess. Despite potentially effective in some cases, this is a quite limited approach since there is no guarantee of full coverage and important references may not be cited, besides the fact that only papers published before the current paper can be traced. Finally, even if the researcher is able to find a good set of references she may not be able to actually access those papers if they are not publicly available.

Today there is an increasing amount of research papers available on the Web, thus a significant portion of the bibliographical search task is currently performed in the digital environment. In this environment, researchers make common use of tools such textual search services that allow them to look for interesting papers according to their research interests which are usually expressed by means of textual queries (set of keywords). After that, they are presented with a result list that contains papers that may be related to their search. One disadvantage of this approach is that it is responsibility of the users to translate their information needs and goals in the "best" possible query with the most suitable terms trying to retrieve all and only all papers they may actually need. In other words, it is not a trivial task at all, even disregarding the number of different existing sources, query form syntaxes, and so on.

An alternative approach that has been proposed in the literature advocates the use of research paper recommender systems [2, 8, 10, 15, 20, 27]. These systems automatically suggest research papers by using some user provided information which is usually a little bit more elaborated than a few keywords. Examples of such input include a list of papers authored by an author [8], a single paper [25], a list of citations [15], part of the text of a paper [11], and so on. Based on this initial information, these systems create a profile to represent the user and then search for items or other profiles similar to the one provided for that user. In the first case, the most similar papers can be suggested, whereas in the second one items absent in the current user profile but present in profiles of other "similar" users are recommended.

The main problem with existing paper recommendation approaches is that they assume the existence of a lot of "privileged" information which may include private document collections, sets of citations (references), sets of (restricted) user profiles, etc. For example, the use of citation graphs, like in [15], depends on the system possessing a collection of documents and, consequently, their citations. This means that recommendations are made based on the particular "view" of the system provider, based solely on the information this provider has. Thus, cross-system or cross-application recommendations are rarely supported. If users want to find papers in different information sources they need to access each one individually because those sources probably belong to different owners with their own recommender systems. Moreover, approaches based on user profiles (e.g., [8]) can work well only when the system already has a number of registered users, a major obstacle for the construction of new recommender systems.

In this paper, we propose a novel source independent framework for research paper recommendation that does not require any privileged information stored a priori. This framework exploits a search approach to create a set of candidate papers for recommendation on demand by querying search forms made available by Web information sources. The framework requires as input only a single research paper and generates several potential queries by using terms in that paper, which are then submitted to existing Web information sources (e.g., digital libraries) that hold research papers. As it submits only a few top-ranked queries, we expect that the framework does not cause any damage to the queried sources; it would appear like a normal user accessing those sources. Once a set of candidate papers for recommendation is generated, the framework applies content-based recommending algorithms to rank them in order to recommend the ones most related to the input paper. Moreover, this framework uses only publicly available metadata (i.e., title and abstract) to perform the recommendation.

Although public services such as Google Scholar[1] and CiteSeer[2] can be used for paper recommendation, they require all collections to be indexed, thus limiting the recommendation results to a predefined set of sources. On the other hand, our proposed framework is extensible and allows the plug-in of any existing Web information source that provides a form-based interface.

In order to evaluate our proposed framework, we considered a specific scenario in which we tested two strategies for query generation and two strategies for query weighting - the queries are weighted so that we can choose the top-k to be submitted. We also assessed the capability of the three main fields of a research paper, i.e., *title*, *abstract* and *body*, in generating good candidate queries. We performed our evaluation by submitting to three well known Web information sources, ACM Digital Library[3], IEEE Xplore[4] and ScienceDirect[5], queries generated according to four strategy combinations. Then, in a second part of our experimental evaluation, we addressed three content-based strategies for recommending research papers.

Our results indicate that the best query generation strategy is one that combines a simple n-gram strategy to generate candidate queries and a term-frequency-based weighting scheme to rank them. They also show that the abstract is the best field to provide terms for query generation, since it usually provides a good summary of the content of a paper, and that we can improve recall by submitting queries that include terms from different fields (although with higher cost). We also observed that using a pool of information sources provides better results than using a single source. Finally, although we use only publicly available metadata for recommendation, our content-based strategies are able to provide fairly good results considering that they use only title and abstract to match papers.

The rest of this paper is organized as follows. We start by discussing related work in Section 2. In Section 3, we define our problem. In Section 4, we present our proposed framework for research paper recommendation. In Section 5, we present our recommendation scenario, and discuss our query generation and content-based recommendation strategies. In Section 6, we describe the experimental setup and discuss the results of our experiments in detail. Finally, in Section 7, we conclude the paper.

2. RELATED WORK

Recommender systems proposed in the literature have been successfully applied in several different domains including news [6], movies [12], audio [24], and so on. They have also been applied to recommending research papers, our focus in this work. Most of these systems can be classified based on the underlying method exploited. In [23], the authors consider six main categories: user-to-user correlation, item-to-item correlation, raw retrieval, manual selection, statistical summarization, and attribute-based. As the most successful research paper recommender systems fall into the first two categories, we focus on them. A survey of the main techniques used by recommender systems can be found in [1].

The user-to-user method is also known as the *collaborative filtering approach*. In such an approach a user profile is created using information that reflects user preferences and then the profile is compared to other user profiles. Based on the preferences of similar profiles, new items are suggested to a user. As examples of the use of the collaborative filtering approach for research paper recommendation we have [2] and [18]. In [2], the authors assume that researchers in the same or similar fields tend to be interested in similar papers. Thus, they exploit search information from similar users to improve search results. In [18], the authors use access information (e.g., 'user X has downloaded document Y') to predict if two papers are related to each other. The bX recommender service[6], which is based on [7], also use user assessment records to suggest related papers. However, in that system data from many sources and communities can be aggregated in order to enrich recommendations.

The approaches described above suffer from a typical prob-

[1]http://scholar.google.com
[2]http://citeseer.ist.psu.edu
[3]http://portal.acm.org/
[4]http://ieeexplore.ieee.org/

[5]http://www.sciencedirect.com/
[6]http://www.exlibrisgroup.com/category/bXOverview

lem of collaborative filtering called *cold-start* [21]. In the beginning, there are very few users using the system and no user preference information, implying in poor performance due to the lack of information. Obtaining a significant amount of direct user ratings or access information might take a long time what can be critical to new applications. In order to overcome this problem some collaborative filtering approaches try to generate user preferences or ratings implicitly. In [15], the authors create a citation graph using the reference list in research papers; citations correspond to votes and ratings are generated without user explicit evaluation. In a similar way, the work in [10] proposes a *PaperRank* algorithm that generates ratings by performing random-walks in the citation graph.

Although the last two approaches show good results, they also present disadvantages. The first one refers to new items or, more specifically, new papers that require some time to acquire enough citations to be regarded as good recommendations. A second disadvantage, perhaps more important, refers to the availability of citations in Web sources. That information is not largely available and it might not be feasible to rely on it to make recommendations.

The item-to-item correlation method is also known as the *content-based approach*. In this approach, a user profile is created using features of a user's preferred items, while an item profile is also created for each item using its own features. After that, user and item profiles are compared using some similarity function and the most similar items are recommended. It is an alternative to collaborative-filtering because it can be applied using only data available at any stage of the system.

A typical strategy used in content-based systems is to rank the most important keywords using TF-IDF [3]. As an example of this, in [16] the authors propose a strategy that matches stemmed words from a document against the whole collection to recommend research papers. However, notice that the use of TF-IDF requires global statistics on term frequencies from the target document collection, which can also be considered as privileged information. Another content-based approach [8] creates profiles based on papers authored by the users, in that case, authors with papers indexed by CiteSeer[7]. It then compares the generated profiles against concepts extracted from the target documents and the most related ones are then recommended. In [26] the authors also create a profile based on authored papers but they include contextual information through citation between papers. In [20], a profile is created for general users by using past accessed documents rather than authored papers.

Finally, *hybrid systems* combine collaborative filtering and content-based techniques. In [27], a content-based strategy that uses the cosine metric is applied to find similar papers, and then a collaborative filtering algorithm that exploits the *k-nearest neighbors* is used to suggest citations. In [13], a bipartite-graph is built using information on users, books and transactions, and the recommendation problem is seen as a graph searching.

Our framework for research paper recommendation differs from the previous approaches in three main aspects: (1) we consider only a single paper to generate statistics and infer the user's interests; (2) we only rely on existing Web information sources whereas other works use proprietary document collections; and (3) we use only publicly available information whereas other approaches take advantage of privileged and private information such as citations and user profiles.

[7]http://citeseer.ist.psu.edu/

3. PROBLEM DEFINITION

We define our problem as follows:

Given an input paper p and a set S of sources S_j from a specific knowledge area A, find a set of papers P which are the most related to p considering a given criterion c. Here, each paper p_i belonging to any source S_j is represented by a set of metadata M.

We include a criterion c in the above definition because different users with different backgrounds may have different goals when asking for recommendations. For example, one might want a paper to complement an existing list of references or a paper to cite in a specific context [11], or a paper with a certain level of novelty [29], or a paper that addresses a subject in a more specific or general way, and so on. Thus, the problem of recommending research papers should be addressed from different perspectives.

In [16] the authors discuss exactly this and claim that different goals would require different approaches. Consequently, no specific algorithm will solve all the different aspects of research paper recommendation and, therefore, any proposed framework should be able to be adapted to the users' needs.

4. PROPOSED FRAMEWORK

Differently from existing approaches, before providing any recommendation, our framework generates a set of candidate papers on demand. This set is specific for each input paper and only publicly available metadata is used for recommendation.

As many sources provide access to research papers on the Web (for free or through payment), we focus our attention on generating the set of candidate papers from these specific sources. Thus, we restrict the context of our framework to a specific knowledge area and a specific type of document, research papers. Moreover, as most of these sources provide an interface with a search form, we rely on this mechanism to retrieve documents in the same way users would do.

The idea of using documents as a query in search interfaces was also applied in similar problems. In [9] the authors apply this approach to the covering test problem where the goal is to find out if there is a near-duplicate of a certain document in a corpus of documents. In [28], the authors address the problem of cross referencing on-line information in the context of blogs.

The architecture of our framework is shown in Figure 1. The framework takes as input a paper provided by the user. This has the advantage of requiring low effort from the user and providing rich information about the subject of interest. Notice that users do not need to manually generate queries and the framework is strictly guided by contextual information present on the input paper, which is not the case when short queries are directly submitted by the user.

After the input paper has been provided, a structural parser acts to extract the required information. In general, research papers share a common structure that includes title, abstract, introduction, related work, conclusion, refer-

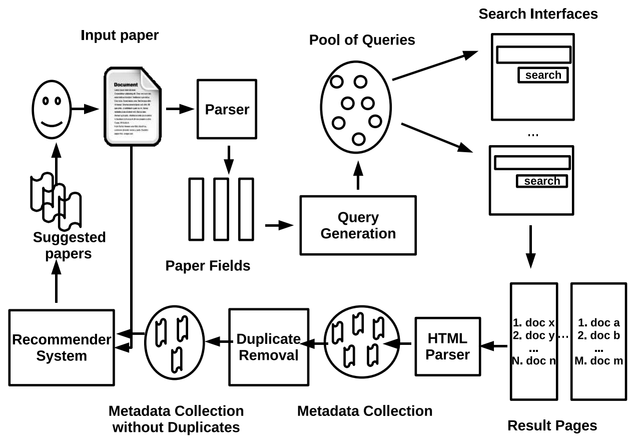

Figure 1: The Architecture of the proposed framework for research paper recommendation

ences, and so on. In order to make the problem simpler and, therefore, easier to generalize, we consider here only the three main fields present in most research papers: *title*, *abstract* and *body*.

It is out of the scope of this paper to propose any algorithm to extract fields from research papers. As such, in this work we use solutions usually adopted for PDF files - one of the most common formats for research papers available on the Web - and convert them to XML by using the pdftohtml tool[8]. Then, we extract fields by observing usual formatting patterns (e.g., bold, font size) and keywords like *abstract* and *introduction* by using a traditional SAX parser. Once the fields from the paper have been extracted, they are sent to a module responsible for generating queries. This module first generates a set of candidate queries and then rank them. Next, the top-k queries are selected to be submitted to a Web form.

We consider here the case in which a Web form provides a single search interface with a general input text field. This means that we do not deal with cases where different fields need to be automatically filled. Thus, we submit the selected queries through this general input text field to specific information sources. We advocate to use several information sources so that we can generate a large and diverse set of candidate papers for recommendation. The list of results returned from each source needs to be parsed so that we can extract metadata for the next step. Currently, we are not concerned in having a general parser, thus we have built a specific parser for the result pages of the sources used in our experiments.

In the next step we remove duplicates or near-duplicates from the set of candidate papers as shown in Figure 1, so that only one version of a paper is included in the final list. Simply matching titles on lower case solves most of the cases. Finally, after building the set of candidate papers, we are ready to make recommendations by ranking them according to a predefined criterion.

Notice that the proposed framework is extensible and can be easily adapted to a specific user community without depending on a particular information source to provide the desired service. For example, one can add a new query generation or recommendation strategy to the framework by simply implementing herself those strategies or by using some implementation available on the Web (see Figure 1).

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

5. A RECOMMENDATION SCENARIO

As discussed in Section 3, the problem of recommending research papers can be seen from different perspectives. For each one it might be necessary to adopt a different strategy to accomplish the recommendation task. In the problem definition stated in Section 3, the criterion c is used to define a particular perspective to the problem.

In this section, we describe a possible scenario for the problem of research paper recommendation and show how to generate queries and recommend papers within this scenario by using our framework. Additionally, by describing this scenario we give some insights on how to apply our framework to solve other research paper recommendation tasks.

Specifically, in the case of this work we are interested in the following situations: (1) a reviewer has received a paper that addresses a subject she is not a specialist on and wants to learn more about it; (2) a student has received a paper from her advisor to start a research in the topic covered by it; and (3) an ordinary user, after some initial searches, finds an interesting paper and wishes to find other related papers. In all these situations, the user needs to find additional related work. Thus, the criterion in focus here is "the most similar papers", in which similarity is considered in terms of the subject content.

In this recommendation scenario, we deal with the following constraints: (1) only a single input paper that indicates the user's preferences is available; (2) all information available to generate statistics is present in the input paper; (3) only title and abstract fields are publicly available on the considered information sources.

Thus, based on the above scenario we next present strategies for query generation and paper recommendation.

5.1 Query Generation Strategies

A major task in our proposed framework is to generate a set of candidate papers for recommendation. This task is accomplished by submitting queries to form-based search interfaces available on the Web and retrieving metadata available on the candidate papers. Here we discuss some key points of this query generation process.

The task of generating queries can be divided into two steps. The first one is the generation of potential query candidates. In this step, we take as input the text of a paper provided by the user and apply some criteria to compose queries with representative terms taken from that paper. The second step concerns the ranking of candidate queries so that only those considered as the most promising ones are actually submitted to the query forms. We rank these queries because query submission is a time-resource-consuming task and, since the number of candidate queries may be very high, we need to limit the number of possible candidate queries in order to make the whole framework practical.

5.1.1 Generating Candidate Queries

Roughly, the module responsible for generating candidate queries works as follows: it first receives a text segment as input and then, by using a predefined strategy, it generates a set of queries composed only of terms (words) present in that text that are considered as representative of its contents.

The simplest strategy would be to use the entire text of the input paper as a single query and then submit it through Web forms. In this case, we have what is called a "long query". However, once search engines in general do not perform well with this type of query, a better strategy is to generate several shorter queries.

The great challenge in generating such short queries is to discover the key "concepts" present in the long and verbose query. A concept can be a word, a free combination of words, a phrase, an expression, and so on [4, 5]. It is easy to see that the simplest case where concepts are represented as single words cannot be used in this context, since the generated queries would be too generic to be useful in most cases.

Thus, due to the large number of alternatives and, mainly, the difficulty in analyzing them, we here adopt two main strategies for query generation that have proved to be effective in other contexts, namely: (1) n-grams, a simple strategy that can be easily implemented with low cost; and (2) noun-phrases, a more sophisticated strategy that is meant to discover key concepts in a text.

N-grams: Using the n-gram strategy, we generate queries by extracting subsequences of n words from the input text. In this approach, we define a window size and slide this window over a segment of text. We start in the first word of the input text and finish when the end of the window reaches the last word of the text. Each word takes one place in the window. Thus, queries are generated by taking terms within the window.

Before applying the n-gram strategy, we preprocess the input text. We remove special characters (.,;,!,?,:,#,etc.), transform all terms to lower case, remove stopwords and apply stemming using Porter's algorithm [19]. By stemming, we group similar words avoiding duplicate queries. However, we transform the stemmed query back to its original version before submitting it because stemmed queries usually do not perform well.

As an example, by using a 2-gram strategy the text "A source independent framework for research paper recommendation" would generate the following queries: "source independent", "independent framework", "framework research", "research paper" and "paper recommendation".

Noun-phrases: Part-of-speech tagging (POST) is a technique that has been successfully used in past work for discovering key concepts in information retrieval applications [4, 28]. It applies a pre-trained classifier that labels each word in an input text to its part-of-speech, i.e., a verb, an adjective, an article, and so on. In this work we are concerned with concepts that are expressed by subsequences of nouns called *noun-phrases*. Thus, we extract noun-phrases in the same way that is done in [28]. For this task, we used an implementation provided by [17].

Distinctly from n-grams, we do not preprocess the input text when using POST. The POS-tagger differentiate sentences with and without punctuation, lower cases, and so on. Thus, we use the raw text extracted from the input paper. However, we stem candidate phrases to remove duplicates and similar queries, returning them to the original form at submission time.

As an example, consider the same text used before: "A source independent framework for research paper recommendation". The POS-tagger would classify each word in the following way: "A/DT source/JJ independent/JJ framework/NN for/IN research/NN paper/NN recommendation /NN", where DT identifies an article, NN a singular noun, IN a preposition and JJ an adjective[9]. A noun-phrase might be a sequence of two or more nouns or a sequence of adjectives followed by a sequence of nouns. In the previous example, the generated queries would be "source independent framework" and "research paper recommendation".

5.1.2 *Weighting Candidate Queries*

In order to make our framework practical, we focus only on information existing in the text provided as input and do not take into account any type of global information, such as TF-IDF that requires a collection of documents, when setting weights for candidate queries. Thus, the only information exploited to weight candidate query are: (1) term frequency and (2) term position. We describe next these two weighting schemes.

Term Frequency: The intuition behind this weighting scheme is that the more times a query (i.e., a list of terms that form a query) appears in the input paper the more likely this query captures the main subject of that paper. Titles and abstracts are usually very short; as such they do not provide much frequency information. On the other hand, the body is usually quite long and can provide us with richer information about the frequency of certain query expressions.

We define the weight w for a query q in terms of term frequency as

$$w(q) = \gamma \times \frac{freq(q)}{max\_frequency} + (1-\gamma) \times \frac{word\_frequency(q)}{max\_word} \quad (1)$$

where $freq(a)$ counts how many times an expression a appears in the text, $max\_frequency$ is the highest frequency among all the queries, q is a query, γ sets the importance of the terms in the equation, max_word is the highest value returned by function $word\_frequency$, which is defined by Equation 2.

$$word\_frequency(q) = \sum_k \frac{freq(i_k)}{max\_word\_frequency} \quad (2)$$

where $max\_word\_frequency$ is the highest frequency of a single word in the text and i_k represents the k-th word in the query q.

In Equation 1 the first term is for the whole query and the second one is for the words within the query. Since a multi-word query does not occur very frequently, we have added that second term that counts the frequency of each individual word that composes a query so that we can decide which query is better when there is a draw.

Term Position: In this weighting scheme, we take into account in which field of the input document the terms that form a query appear. We consider that some fields are more important than others and thus, if terms in a query are present in those fields, they should be better ranked.

We consider the three main fields of a paper: title, abstract and body. As the title and abstract usually contain a set of selected terms carefully chosen to summarize the content of a paper, we think they should imply higher weights than the body.

[9]http://bioie.ldc.upenn.edu/wiki/index.php/POS_tags

We use Equation 1 to establish the query weight but we calculate the term frequency as follows:

$$freq(i) = \theta \times title(i) + \phi \times abstract(i) + \omega \times body(i) \quad (3)$$

where *title*, *abstract* and *body* are binary functions that indicate the presence of an expression i in the respective fields, and θ, ϕ and ω define the importance of each function.

5.2 Paper Recommendation Strategies

For the paper recommendation task, we explore a content-based approach for recommendation as we assume that the only information available is the input document and some publicly available metadata (title and abstract) about the papers returned by the sources as result of a query. We leave for future work to incorporate collaborative filtering in our framework. The content-based approach we adopt simply matches the input document metadata against metadata extracted from each paper in the set of candidates returned by the sources.

As already stated, we target a quite specific scenario: we want to find the "most similar" papers and we are only provided with textual information. Thus, we use a traditional document similarity metric as our recommending strategy, more specifically, the cosine similarity metric.

With the cosine similarity metric, documents are represented as vectors where each position in the vector represents the frequency of a term in the document. The highest similarity value between two documents is when they are identical. The metric is defined by the following equation [3]:

$$Cosine(i, j) = \frac{\sum_k w_{ik} w_{jk}}{(\sum_k w_{ik}^2 \times \sum_k w_{jk}^2)^{1/2}} \quad (4)$$

where w_{ab} is the weight of the term b in document a. We calculate the term weights as the normalized frequency without the IDF factor. This is our first strategy for recommending research papers.

As we know that research papers are normally structured documents, we try to use this fact to get more relevant recommendations. We argue that each structural part contributes with a different importance when we calculate the document similarity. Thus, in our second strategy, we do not apply the cosine similarity metric to the whole text. Instead, we apply it to each field (in this case title and abstract) and make a linear combination of the values produced by each one as below:

$$L{-}Cosine(i, j) = \alpha \times Cosine(t_i, t_j) + \beta \times Cosine(a_i, a_j) \quad (5)$$

where i is the input document, j is a retrieved document, t_i is the title of document i, t_j is the title of document j, a_i is the abstract of document i, a_j is the abstract of document j, α is a parameter that sets the importance of the title and β is a parameter that sets the importance of the abstract.

In [22], the authors point out some disadvantages of using a linear combination of fields and propose a linear combination of term weights instead. Thus, to calculate the weight w of term i we adopt a strategy similar to [22], which is defined by the equation

$$w(i) = \alpha \times freq(i, t) + \beta \times freq(i, a) \quad (6)$$

where $f(a, b)$ denotes the frequency of the term a in the field b, t is the title, a is the abstract, α is a parameter that sets the importance of the title and β is a parameter that sets the importance of the abstract.

In their original work, those authors apply this strategy to the BM25 formula, but as it only changes the way weights are calculated, we have applied the same strategy to the cosine similarity metric. This is the third similarity metric that we use for recommendation.

6. EXPERIMENTS

In this section, we describe the experiments we performed to evaluate our proposed framework for research paper recommendation. The goals of our experiments were to verify: (1) what is the better combination between the query generation and ranking strategies; (2) what is the best field to use for query generation; (3) if the use of different information sources improve the recommendation; and (4) if certain combinations of strategies and sources would perform better than others.

We argue that a direct comparison with any previously discussed research paper recommendation approach would not be fair. Here we show how to build a framework that faces real world constraints and propose specific strategies to deal with several limitations. On the other hand, other works are not supposed to face the same constraints and they can not be easily implemented if they do not have privileged access to the information sources. Our goal is not to provide a more effective approach for research paper recommendation but raise some issues that should be considered when implementing a practical research paper recommender system. We should note, however, that we do have a simple baseline that simulates the most expected behavior of a typical user in the described scenario (see Section 6.3.1).

In our experiments we compared our proposed strategies by means of user evaluations. In order to verify the statistical significance of the results, we performed a paired *t-test* with a confidence interval of 90%. Any claim of superiority of one strategy over the other is only reported if the statistical significance test indicates so. Some decisions had to be made empirically due to the large number of parameters and experiments required. These experiments focus on the recommendation scenario described in Section 5.

6.1 Experimental Setup

Ten subjects, all of them Computer Science graduate students, participated in our experiments, which were performed as follows: (1) we asked the subjects to provide as input a paper of their interest and that would also be related to their area of expertise; (2) we submitted each provided paper to our framework to obtain a ranked list of recommended papers; (3) each subject evaluated her list of recommended papers as being *strongly related*, *related*, or *not related* at all to the paper given as input. The papers in the list to be evaluated were presented in a random order so that we avoided any bias (users tend to trust top-ranked documents).

The choice of Computer Science as the target knowledge area is justified by the difficulty in getting enough subjects from other areas. Thus, in these experiments we chose as our sources ACM Digital Library and IEEE Xplore, two well known Computer Science digital libraries, and ScienceDirect, a well known service provided by an international publisher that also allows searching for publications in Computer Science. Notice that any other bibliographic Web source with a generic search interface could have been used.

In these experiments, we used the following configuration: for the first query generation strategy, we considered

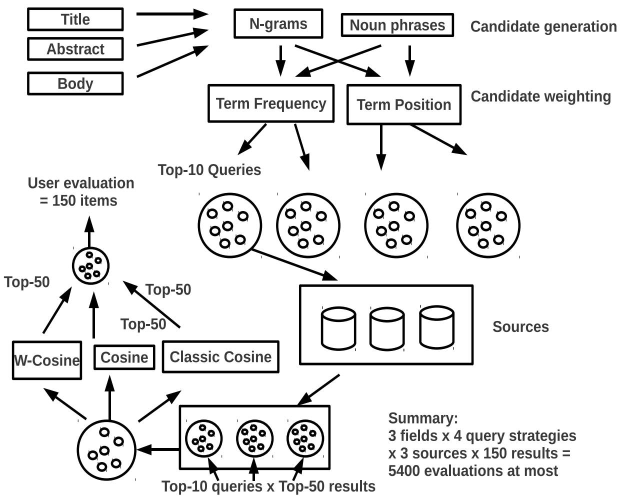

Figure 2: Overview of the experiments

n-grams of size 2 - that normally capture most concepts; for the parameter of Equation 1, we used $\gamma = 0.66$, thus giving more importance to the query frequency; for the parameters θ, ϕ and ω of Equation 3, we set them to 3, 2 and 1 respectively, thus assuming that terms in the title are the most important ones, followed by those in the abstract and in the body. We should stress that the choice of these parameters is based on common sense and that we did not perform any more complex parameter tuning procedure.

Figure 2 presents an overview of our experiment design. For each field, we have two query candidate generation strategies and two query ranking strategies. Thus, we have four strategies for query generation. For each one, we created a pool with top-k queries. We arbitrarily chose k = 10.

Thus, for each of the four query generation strategies, we sent all 10 queries to the selected sources. For each source we took the top-50 results and then created a pool of results. For each generated pool, we applied the three ranking strategies. We then created a new pool for each source, holding the top-50 results, so the new pool contains, in the worst case, 150 documents. The subjects evaluated each recommended paper in this pool as *strongly related*, *related* and *not related* to the original input paper.

In sum, we have three fields, four query generation strategies and three information sources. As the final pool contains 150 recommended papers, each subject would have to evaluate in the worst case $3\times4\times3\times150 = 5400$ papers. However, in most cases this number was significantly reduced by removing duplicates, and, in average, the subjects evaluated 1000 papers.

6.2 Evaluation Metrics

We adopted two well and widely used metrics for evaluating ranking methods in information retrieval: recall [3] and NDCG [14].

Recall is the fraction of positive (related or relevant) items retrieved by a query (considering that an input document is a query in our scenarios):

$$Recall = \frac{Positive\ Items\ Retrieved}{Total\ of\ Positive\ Items} \tag{7}$$

As mentioned before, in our experiments, the recommended papers were evaluated considering three levels of relevance. We define *strongly related* and *related* as positive and *not related* as negative when calculating recall. Indeed, the recall generated is a relative recall where we consider the sum all positive items retrieved by all queries as the total universe of positive items.

Discounted Cumulated Gain (DCG) is usually applied when relevance values are graded - as in our case. DCG is able to measure how good a ranking is by considering if highly relevant items appear on the top of the rank and penalizing when they appear lower. DCG is calculated as

$$DCG_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{log_2 i} \tag{8}$$

where p is the position where DCG is calculated and rel_k is the relevance value of the item in position k. We set the relevance value in the following way: items evaluated as *strongly related* get 2 points, as *related* 1 and *not related* 0.

NDCG is the Normalized DCG and it is calculated as

$$NDCG_p = \frac{DCG_p}{IDCG_p} \tag{9}$$

where $IDCG_p$ is the DCG calculated over the ideal ranking in which the most relevant items are ranked on the top and p is the position where the DCG is calculated. Thus, we can observe how close the obtained ranking is to the ideal one.

6.3 Results

Here we present and analyze the results of our experiments. We first evaluate the query generation strategies and then the recommendation ones.

6.3.1 Query Generation

In order to comparatively evaluate our strategies for query generation, we propose a baseline that focus on the user's behavior. When a typical user holds a single paper that represents her current interest (see our recommendation scenario in Section 5), a simple strategy that she could adopt to query a digital library (without reading the entire paper and formulating queries by herself) would be to use the paper title as a query. Thus, we take this approach as a baseline for the query generation task. Table 1 shows the relative recall obtained by this baseline for each information source, ACM Digital Library (ACM), IEEE Xplore (IEEE) and ScienceDirect (SD), as well as for a pool composed of all recommended papers retrieved by each source individually. Column *2-pos* means we consider 2 levels of positive items, *strongly related* and *related*, and column *1-pos* means we consider only one level of positive items, *strongly related*. We use Table 1 when comparing our strategies.

Table 1: Relative recall for the baseline

	Source	2-pos	1-pos
Baseline	ACM	0.09	0.16
	IEEE	0.02	0.03
	SD	0.03	0.02
	All	0.12	0.19

Table 2 shows the relative recall of the query generation strategies. The first column specifies which part of the input paper was used to generate the queries: title, abstract or body. There is a fourth option that we called *All fields* in which we included all queries generated by each field individually in a joint pool and then submitted all those queries to the search interfaces. By using this alternative, we would like to check whether we could improve the results by using all fields together and by increasing the number of queries. In the second column we show the information source to which we submitted the queries. The next columns show recall values resulted from using each query strategy: *ng-freq* corresponds to the n-gram query generation strategy combined with the term frequency weighting scheme, *ng-pos* corresponds to the n-gram strategy combined with the term position weighting scheme, *np-freq* is the noun-phrase query generation strategy combined with the term frequency weighting scheme and *np-pos* is the noun-phrase strategy combined with the term position weighting scheme. Finally, the last line, *All combined*, shows recall values for the aggregation of all fields and sources combinations over the n-gram and noun-phrase strategies.

Table 2: Relative recall for the query generation strategies: *strongly related* and *related*

	Sources	Query Generation Strategies			
		ng-freq	ng-pos	np-freq	np-pos
Title	ACM	0.12	0.12	0.04	0.04
	IEEE	0.04	0.04	0.03	0.03
	SD	0.04	0.04	0.02	0.02
	All	0.16	0.16	0.06	0.06
Abstract	ACM	0.21	0.14	0.18	0.18
	IEEE	0.15	0.10	0.13	0.12
	SD	0.08	0.07	0.06	0.06
	All	0.32	0.23	0.30	0.28
Body	ACM	0.17	0.19	0.17	0.18
	IEEE	0.12	0.13	0.12	0.13
	SD	0.07	0.09	0.06	0.06
	All	0.28	0.28	0.28	0.29
All fields	ACM	0.29	0.22	0.23	0.21
	IEEE	0.19	0.15	0.17	0.15
	SD	0.12	0.11	0.09	0.08
	All	0.40	0.32	0.35	0.34
All combined		0.69		0.52	

We first analyze the query generation strategies, n-grams and noun-phrases. When only the title is used to generate queries, we can see that the n-grams present higher recall than noun-phrases in ACM Digital Library, but they perform quite similarly on other sources. There is also a statistical difference when all sources are put together, but this is clearly influenced by the ACM results. When only the abstract is used, there is no statistical difference between the best results from using n-grams or noun-phrases. The same fact can be observed when we use the body field and when we use all fields together. This is an interesting result because n-grams provide a very simple and low cost strategy when compared to noun-phrases, thus being the preferred strategy to be applied in our context. This is further corroborated when we look at the *All combined* recall values in the last line of Table 2.

Comparing the n-gram strategies against the corresponding results obtained by the baseline (Table 1, column *2-pos*), we can see a draw when we use only the title for query generation. For the other cases, our proposed strategies obtain a large superiority over the baseline, for example Abstract-All 0.32 versus Baseline-All 0.12. Thus, if a user does not want to read the whole paper and generate her own queries, our strategies are a good choice to bring more related papers than using the entire title as a query.

Next we analyze the query weighting strategies: term frequency and term position. We compare the combination strategies *ng-freq* against *ng-pos* and *np-freq* against *np-pos*. When using only the title, we can see that for both query generation strategies there is no difference in using term frequency or term position as weighting strategy. As we limit the number of submitted queries to 10 and the title on average generates less than 10 queries, the weighting strategies make no difference in the ranking. For the other fields, the only statistically difference between frequency and term position is for n-gram on abstract and when we use all fields.

As the weighting strategies were applied to the same candidate queries and performed quite similarly, we thought they were returning the same papers. However, the aggregated values (see All fields-All) are low, which might indicate that they actually returned different relevant documents. In order to confirm this hypothesis, we aggregated the frequency and term position strategies. For n-grams, we got a recall of 0.40 with frequency and 0.32 with term position, with the aggregation we got a 0.69 recall - in this case we had almost no intersection. For noun-phrases, we got a recall of 0.35 with frequency and 0.34 with term position, with the aggregation we got a 0.52 recall - almost 50% higher. Notice, particularly, that the combination of the ng-freq and ng-pos strategies generated a recall value close to the maximum that could be obtained if the the sets of papers returned by both strategies were completely disjoint (0.72). This confirms our hypothesis that although frequency and term position have similar recall they bring different relevant papers, due to the very distinct query rankings generated by the two strategies. In fact, we believe that our best results obtained with the strategy *All combined*, absolutely or in comparison with the simple baseline, are very good as we are able to retrieve almost 70% of all relevant papers coming from all sources.

In Table 2 we can also see that the information sources perform differently from each other, with a clear tendency: the ACM Digital Library is the best one, followed by IEEE Xplore and Science Direct. In any case, the most important result we want to emphasize about the information sources is the fourth line in each field group. We can see that using all sources together has the potential of returning a more relevant and diverse set of papers than when using any single source. Actually, we observed an average gain of 42% using all sources over the best results obtained by using only the ACM Digital Library. As we have stated in Section 4, one might want to receive recommendations from more than one source and those results are a good evidence that this may enrich the experience when using a recommender system.

At last, we analyze the relative performance of each field as a source for query generation. If we contrast each line in the title, abstract, and body groups we can see that, using only the abstract to generate queries, we obtain the highest recall values. The results obtained by using the body are also good but they demand a higher cost. These results confirm our intuition that the abstract, being a synthesis of the paper, usually contains the most important keywords and, therefore, is a good source for query generation. Finally, as expected, using all fields produces the highest recall values but at the highest cost. Notice that in this case the number of queries increased more than 200% and the average gain obtained over the abstract is only 41%.

Finally, any research paper recommendation task is expected to return few *strongly related* papers and a large

Table 3: Relative recall for the query generation strategies: *strongly related* only

	Sources	Query Generation Strategies			
		ng-freq	ng-pos	np-freq	np-pos
Title	ACM	0.13	0.13	0.03	0.03
	IEEE	0.04	0.04	0.04	0.04
	SD	0.02	0.02	0.01	0.01
	All	0.16	0.16	0.07	0.07
Abstract	ACM	0.29	0.13	0.21	0.19
	IEEE	0.17	0.11	0.14	0.13
	SD	0.04	0.05	0.02	0.02
	All	0.43	0.24	0.34	0.31
Body	ACM	0.24	0.25	0.22	0.22
	IEEE	0.15	0.17	0.16	0.15
	SD	0.03	0.05	0.03	0.02
	All	0.38	0.36	0.37	0.35
All fields	ACM	0.39	0.27	0.25	0.23
	IEEE	0.21	0.18	0.17	0.16
	SD	0.07	0.07	0.04	0.03
	All	0.57	0.39	0.43	0.38
All combined		0.69		0.48	

amount of *related* papers. In order to verify if the amount of *related* papers were not distorting recall results, we present in Table 3 results that consider only the *strongly related* papers. The results are quite similar to those in Table 2 which means that our strategies are not biased by the amount of *related* papers. However, unlike in Table 3, the aggregated values are not so low when compared to the *All combined* ones, thus suggesting that the weighting strategies return the same *strongly related* papers most of the time.

6.3.2 Ranking/Recommendation

In this section, we focus our analysis on how well the proposed ranking strategies can order the retrieved papers in such a way that the most relevant ones stay on the top of the rank. In this analysis, we use NDCG as an evaluation metric for each combination of the query generation and ranking strategies.

As the abstract field presents the best recall results, we chose this field to analyze the ranking strategies. Table 4 shows the results. The first column indicates the information source used, the second the ranking strategy applied to the returned papers and the following ones the NDCG obtained for each query generation strategy. We use the same labels as in previous tables: *ng* corresponds to n-grams, *np* to noun-phrases, $freq$ to term frequency and *pos* to term position.

As we can see, simple strategies based on the cosine similarity metric are able to put relevant documents on the top of the raking. In most of the cases, we have an NDCG over 0.7, which is very good considering that we used only title and abstract to determine similarity. We can also see that the differences among the three evaluated ranking strategies, Classic Cosine, Linear-Cosine (L-Cosine) and Weighted-Cosine (W-Cosine) in all three information sources considered are, on average, less than 10% and actually are not statistically significant. The hypothesis that title and abstract used separately could improve the ranking was not observed. On the other hand, the results show that Classic Cosine without any parameter setup is a good option. Table 4 also shows that the best performance in general was obtained on the ACM Digital Library (which usually returns more relevant documents), followed by IEEE Xplore and ScienceDirect. Finally, we should emphasize the results obtained when all sources were used. Results in Table 2 showed that using all sources together increases recall values due to the larger number of options to work with. However, as can been seen in Table 4, our ranking strategies do not benefit from having more documents. Thus, it seems that for taking full advantage of using several information sources it is necessary to adopt more elaborate ranking strategies.

Table 4: NDCG when using only abstract to generate queries

	Ranking Strategies	Query Generation Strategies			
		ng-freq	ng-pos	np-freq	np-pos
ACM	Classic	0.76	0.74	0.72	0.73
	L-Cosine	0.77	0.74	0.75	0.75
	W-Cosine	0.81	0.74	0.81	0.81
IEEE	Classic	0.71	0.63	0.67	0.66
	L-Cosine	0.73	0.72	0.68	0.67
	W-Cosine	0.75	0.71	0.69	0.68
SD	Classic	0.56	0.58	0.54	0.55
	L-Cosine	0.59	0.55	0.51	0.51
	W-Cosine	0.60	0.54	0.51	0.51
All	Classic	0.75	0.71	0.71	0.73
	L-Cosine	0.76	0.72	0.73	0.73
	W-Cosine	0.79	0.76	0.77	0.76

All reported results were obtained by using the best weighting parameter value for title (α) and abstract (β). We varied these two parameters from 0 to 1, using a step of 0.05. In any case, our best results are very promising as we are able to produce, on average, ranks of papers that are 80% close to the best possible ranks that can be obtained.

7. CONCLUSIONS

Recommender systems have been used to help researchers to find research papers that are related to another paper they have read or to a specific topic of interest, thus reducing the amount of manual effort that is required when undertaking a bibliographical search. The main problem with current paper recommendation approaches is that they assume the existence of a lot of privileged information which is usually available only under very restricted conditions.

In order to solve that problem, we have proposed in this paper a novel source independent framework for research paper recommendation. The proposed framework takes a single research paper as input, generates queries using terms present in some fields of the paper and then submits the generated queries through a search form available on existing Web information sources. Once a set of candidate papers for recommendation is generated, the framework applies content-based recommending algorithms to rank candidate papers in order to recommend those most related to the input one. An interesting and very relevant aspect of this framework from a practical viewpoint, is that it only uses publicly available metadata, title and abstract, to perform the recommendation.

We evaluated our proposed framework by testing four strategies for query generation and found out that using a simple n-gram strategy to generate candidate queries and term frequency as a weighting scheme to rank them is sufficient to obtain good queries to submit to information sources.

We observed that the abstract is better than the body and title to provide good terms for query generation, although the body also provides good results. Our results also show that using more information sources it is possible to retrieve more relevant papers and that submitting queries generated by using the three fields (title, abstract and body) together increases recall. In order to recommend related papers, we tested three ranking strategies based on the cosine similarity metric: Classic Cosine, L-Cosine, that performs a linear combination of the cosine metric applied to title and abstract separately, and W-Cosine, a variation of the cosine similarity metric that weights terms differently. There was no significant difference among them but they present reasonable results considering they are simple strategies and that we used only the title and abstract to do the matching. In any case, the overall results look very promising.

As future work, we would like to test this framework with more sophisticated approaches for paper recommendation. We would also like to use additional fields to generate queries as, for instance, the list of references. Since in this paper we consider the worst scenario where only title and abstract are publicly available for supporting the recommendation task, we would also like to use citation metadata that is present in some sources to improve the quality of the recommendation results as well as to expand our content-based approach by considering additional information, like citation figures, to rank the papers.

8. ACKNOWLEDGMENTS

This research is partially funded by InWeb - The National Institute of Science and Technology for the Web (MCT/CNPq/ FAPEMIG grant number 573871/2008-6), by the Amanajé project (CNPq grant number 479541/2008-6), and by the authors's individual research grants from CNPq and FAPEMIG.

9. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, 2005.

[2] N. Agarwal, E. Haque, H. Liu, and L. Parsons. Research paper recommender systems: A subspace clustering approach. In *Proc. Intl. Conf. on Web-Age Inf. Management*, pages 475–491, 2005.

[3] R. A. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[4] M. Bendersky and W. B. Croft. Discovering key concepts in verbose queries. In *Proc. Intl. ACM SIGIR Conf. on Research and Devel. in IR*, pages 491–498, 2008.

[5] K. Bharat and A. Broder. A technique for measuring the relative size and overlap of public web search engines. In *Proc. Intl. Conf. on WWW*, pages 379–388, 1998.

[6] K. Bharat, T. Kamba, and M. Albers. Personalized, interactive news on the web. *Multimedia Syst.*, 6(5):349–358, 1998.

[7] J. Bollen and H. Van de Sompel. An architecture for the aggregation and analysis of scholarly usage data. In *Proc. Joint Conf. on Dig. Libraries*, pages 298–307, 2006.

[8] K. Chandrasekaran, S. Gauch, P. Lakkaraju, and H. P. Luong. Concept-based document recommendations for CiteSeer authors. In *Proc. Intl. Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 83–92, 2008.

[9] A. Dasdan, P. D'Alberto, S. Kolay, and C. Drome. Automatic retrieval of similar content using search engine query interface. In *Proc. Conf. on Inf. and Knowl. Management*, pages 701–710, 2009.

[10] M. Gori and A. Pucci. Research paper recommender systems: A random-walk based approach. In *Proc. Intl. Conf. on Web Intelligence*, pages 778–781, 2006.

[11] Q. He, J. Pei, D. Kifer, P. Mitra, and L. Giles. Context-aware citation recommendation. In *Proc. Intl. Conf. on WWW*, pages 421–430, 2010.

[12] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proc. Intl. ACM SIGIR Conf. on Research and Devel. in IR*, pages 230–237, 1999.

[13] Z. Huang, W. Chung, T.-H. Ong, and H. Chen. A graph-based recommender system for digital library. In *Proc. Joint Conf. on Dig. Libraries*, pages 65–73, 2002.

[14] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

[15] S. M. McNee, I. Albert, 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 *Proc. ACM Conf. on Computer Supported Cooperative Work*, pages 116–125, 2002.

[16] S. M. McNee, N. Kapoor, and J. A. Konstan. Don't look stupid: avoiding pitfalls when recommending research papers. In *Proc. Conf. on Computer Supported Cooperative Work*, pages 171–180, 2006.

[17] X.-H. Phan. CRFTagger: CRF English POS Tagger, 2006. http://crftagger.sourceforge.net/.

[18] S. Pohl, F. Radlinski, and T. Joachims. Recommending related papers based on digital library access records. In *Proc. Joint Conf. on Dig. Libraries*, pages 417–418, 2007.

[19] M. F. Porter. An algorithm for suffix stripping. In *Readings in Information Retrieval*, pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.

[20] A. K. Pudhiyaveetil, S. Gauch, H. Luong, and J. Eno. Conceptual recommender system for CiteSeerX. In *Proc. Conf. on Recommender Systems*, pages 241–244, 2009.

[21] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: an open architecture for collaborative filtering of netnews. In *Proc. ACM Conf. on Computer Supported Cooperative Work*, pages 175–186, 1994.

[22] S. Robertson, H. Zaragoza, and M. Taylor. Simple BM25 extension to multiple weighted fields. In *Proc. ACM Intl. Conf. on Inf. and Knowl. Management*, pages 42–49, 2004.

[23] J. B. Schafer, J. A. Konstan, and J. Riedl. E-commerce recommendation applications. *Data Min. Knowl. Discov.*, 5(1-2):115–153, 2001.

[24] U. Shardanand and P. Maes. Social information filtering: algorithms for automating "word of mouth". In *Proc. Conf. on Human Factors in Comp. Systems*, pages 210–217, 1995.

[25] T. Strohman, W. B. Croft, and D. Jensen. Recommending citations for academic papers. In *Proc. Intl. ACM SIGIR Conf. on Research and Devel. in IR*, pages 705–706, 2007.

[26] K. Sugiyama and M.-Y. Kan. Scholarly paper recommendation via user's recent research interests. In *Proc. Joint Conf. on Dig. Libraries*, pages 29–38, 2010.

[27] R. Torres, S. M. McNee, M. Abel, J. A. Konstan, and J. Riedl. Enhancing digital libraries with TechLens+. In *Proc. Joint Conf. on Dig. Libraries*, pages 228–236, 2004.

[28] Y. Yang, N. Bansal, W. Dakka, P. Ipeirotis, N. Koudas, and D. Papadias. Query by document. In *Proc. ACM Intl. Conf. on Web Search and Data Mining*, pages 34–43, 2009.

[29] Y. Zhang, J. Callan, and T. Minka. Novelty and redundancy detection in adaptive filtering. In *Proc. Intl. ACM SIGIR Conf. on Research and Devel. in IR*, pages 81–88, 2002.