# Personalized Reading Recommendations for *Saccharomyces* Genome Database

Ni Lao, William W. Cohen

Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
{nlao,wcohen}@cs.cmu.edu

**Abstract.** The rapid growth of research in biology, and the increasing degree to which different subareas of biology are connected, make it difficult to monitor the published literature effectively. To address this problem, we develop a reading recommendation system that requires no other input from users except their reading or citation history. This frees the users from the problem of expressing their information need using query languages. We use a graph representation for publication databases with rich metadata. With this representation, a path constrained random walk (PCRW) model is trained to discover effective recommendation strategies represented as edge paths on the graph. Experiments on both citation-based and history-based reading recommendation tasks show that by leveraging rich context information the PCRW-based approach outperforms random walk with restart based approaches as well as traditional content-based and collaborative filtering approaches. An online recommendation system for *Saccharomyces* Genome Database is available at http://hops.ml.cmu.edu:8080/strutsnies.

**Keywords:** Recommendation, Contextual Retrieval, Path Constrained Random Walk, *Saccharomyces* Genome Database (SGD)

## 1 Introduction

Working scientists rely heavily on rapid access to relevant recent research results. However, the rapid growth of research in biology, and the increasing degree to which different subareas of biology are connected, make it difficult to monitor the published literature effectively. From a practical point of view, there can be a very high cost of not being aware of relevant research methods and results, since weeks or months can be spent duplicating effort or pursuing approaches that are ultimately unproductive.

The primary mechanisms used to address this problem are information retrieval, generally coupled with detailed annotation of the research literature. Over time great strides have been made in information retrieval for biomedical applications, and also in automated methods for annotating biomedical documents (e.g., with predefined topics, or gene-protein entities). However,

information retrieval still requires a user to accurately formulate his or her "information need" as a search query, or series of search queries. This process is often straightforward, but can be difficult when the actual "information need" is a high-level one, for instance, "which papers are most important for me to stay current in my research area?" or "which papers are most relevant to the document (e.g., paper or grant proposal) that I am currently writing?" This problem is more acute when the underlining literature database has rich annotation, because structured queries need to be designed by domain experts in order to leverage the structured information.

An alternative approach to accessing the scientific literature is to formulate it as a *recommendation task*. The goal of a recommender system is to generate meaningful recommendations to a collection of users for items or products that might interest them [16, 1]. Recommendation systems are widely used for recommending potential purchases to consumers, and recommendation is a very well-studied problem in machine learning (with the NetFlix Challenge being only one of several widely-used benchmark problems). One interesting case is when recommendations are based on implicit feedback, e.g. past purchases history of the user. It completely frees users from the need of formulating search queries. The most commonly used algorithm is called the k-nearest neighborhood approach [20, 12, 22], in which an end user is first matched to others who have similar preference history, and some combination of their preferences is used to predict the future preference of this user. These systems are also called *collaborative filtering systems*. However, many existing recommendation methods are ill-suited to the task of recommendation of scientific papers, which include rich metadata. Effective recommendation of scientific papers also relies relating the paper's content and metadata to biological background knowledge, such as the relationships known to hold among genes mentioned in the paper (and its metadata) and genes of interest to the end user.

More recently, retrieval systems have been developed using graph based data representations, which can potentially leverage the rich metadata associated with scientific publications [6, 7, 2, 17, 26]. In this framework, a corpus, its associated metadata, and relationships between entities associated with the corpus are encoded as a (very large) graph. Retrieval tasks can be formulated as *typed proximity queries* in the graph, in which the user provides as input a set of *query nodes* and the type of the *answer nodes*, and receives as output a list of nodes of the desired answer type, ordered by *proximity* to the query nodes. For instance, traditional keyword-based ranked retrieval of documents can be formulated as a proximity query where the query nodes are term nodes, and the answer type is "document". In general, the appropriate notion of "proximity" may be task- or user-specific, and hence must be learned or engineered; however, there are also general-purpose graph proximity measures such as *random walk with restart* (RWR) (also called personalized PageRank [23]) which are fairly successful for many types of tasks. Additionally, several learning methods have been developed for tuning this proximity measure to a particular set of tasks [9, 8, 24].

In addition to supporting retrieval operations, the graph for a corpus also includes entities that correspond to the *authors* of a paper. This suggests a new type of proximity query, in which *recommendation* is performed: here the query node is an author $A$, and the desired answer nodes are items that should be recommended to that author. In this paper, we formulate two recommendation tasks depending on the availability of data. In one, called *history-based reading recommendation*, we assume that we have access to past reading behavior of an end user up to time $T$, as well as a graph representing the corpus up to time $T$. From this history, we learn a proximity model based on the past reading behavior. When new publications (and their metadata) are added to the graph, the learned proximity measure can be used to recommend new papers to read—i.e., the system addresses the information need "which papers are most important for me (the end user) to read?" In the second task, which is called *citation-based reading recommendation*, we only assume the publication history of an user. This setting is potentially useful for new users which we do not have access to their reading history.

Although our system assumes that users' past reading or citation histories are accessible, our system requires no other user input: it is not necessary to formulate an information need as a query. Importantly, the system learns models which are both personal (i.e., which use individual reading or citation history) and collaborative (i.e. trained from multiple users). We construct a graph representing *Saccharomyces* related publications with rich metadata from the *Saccharomyces* Genome Database(SGD), PubMed database, and the Gene Ontology (GO) database. Recommendation models are trained from either reading or citation histories from biologists. Then, as new publications come in, they can be selectively recommended to each user, based on his/her own behavior history. Not only users need not to formulate any search query, but the more they use the system, the better their information need can be modeled.

Technically, our system is based on the path constrained random walk (PCRW) model recently developed by Lao and Cohen [15]. This learning methods uses a richer "feature set" than other random walk based models. A proximity measure is defined by a weighted combination of simple "path experts", each of which corresponds to a particular labeled path through the graph. This kind of models can find useful paths on heterogenous graphs for accurate citation recommendation. Experiment has shown that this new learning method outperforms RWR-based models on several retrieval and recommendation tasks [15]. In this work, we extended the PCRW-based approach to include L1-regularizer on the parameters, which can help selecting features, and further improve the recommendation quality.

Our experiments on both tasks show that by leveraging rich context information the PCRW-based approach outperforms RWR-based approaches as well as traditional collaborative filtering and content-based approaches.

### 1.1 Related Work

Several other recommendation tasks for scientists have been considered previously. For instance, Basu et al. [5] recommend paper submissions to reviewers based on textual similarity between paper abstracts and reviewer profiles which are extracted from the Web. See [25] for a survey of systems recommending papers to reviewers. Arnold and Cohen [3] use proximity queries on co-authorship graph (including document-level metadata on entities associated with publications) to find "nearby" gene-protein entities, and showed that this nearby entities predicted new gene-protein entities that an author would publish papers about in the near future. He et al. [11] develope a non-parametric probabilistic model to measure the relevance of a document to a citation context. Recommendations are suggested to slots in a manuscript, where citations are needed. Our work is different by focusing on reading recommendation tasks, which requires no other input from users except their reading or citation history.

In the remainder of the paper, we first describe the tasks and the datasets to be used in our experiments in more details. We next briefly review the content-based and community-based recommendations approaches and path ranking algorithm by Lao and Cohen[15]. We then describe the experimental results comparing random walk based methods to traditional recommendation approaches and conclude.

## 2 Datasets and Tasks

We will introduce in this section the yeast publication datasets which have rich metadata, and describe two recommendation tasks based on users' citation and reading behaviors.

### 2.1 Datasets

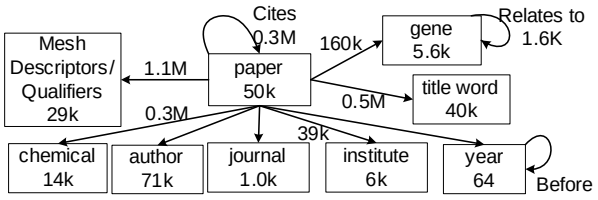


**Fig. 1.** Schema of yeast data with 0.21 billion nodes and 2.8 billion edges. Number of entities and edges of different types are indicated in the boxes and on the links.

Figure 1 shows the schema of the yeast corpus. *Saccharomyces* Genome Database (SGD)[1] is a database of various types of information concerning the

[1] www.yeastgenome.org

yeast organism *Saccharomyces cerevisiae*, including about 50K papers. Each paper is annotated with the genes and chemicals it mentions, and the MeSH[2] descriptors and qualifiers it is tagged with. We also extract gene-gene relations from *Gene Ontology* (GO)[3], which is a large ontology describing the *properties* of and *relationships* between various biological entities across numerous organisms. Paper content and metadata information are crawled from two resources: *PubMed*[4] is a free on-line archive of over 18 million biological abstracts for papers published since 1948; *PubMed Central* (PMC)[5] contains full-text and references to over one million of these papers.

### 2.2 Citation-based Reading Recommendation

We define *citation-based reading recommendation* as given the publication history of a user recommending new publications which might interest this user. In order to train path constrained model in a supervised manner, here we describe how to generate training data automatically from an existing publication database.

For each user in a publication database, we assume the knowledge of all the papers that he/she has published. From here on, we use the term user and author interchangeably. We also assume that each citation in a user's paper is somewhat related to his/her research interest. Therefore, predicting whether a paper is going to be cited by a particular user is approximately predicting whether the user is interested in this paper. We would like to analyze the importance of users' behavior information to the recommendation tasks, so we only consider users who are recorded to publish more than $M$ papers in the database, and ignore the rest who have few publications. This filtering process has the extra benefit of reducing the amount of training data and speeding up the training process.

For each year and each user, our task is to predict which papers of that year are going to be cited by the user. Each of the year-author pairs with non-empty citations is used to generate a query, and the set of papers actually cited are treated as the relevant targets. The papers written by the author him/herself are not included. We use these cited papers of a user to approximate his/her reading history, and we augment the graphs with one extra relation *Read*, which links a user to all the papers he/she has ever cited. We assume that the users always read a paper, if ever, the year after it was published. This way of generating training data is suitable for learning models which are good at suggesting the most recent publications to the users.

We set $M = 100$, and 1143 queries are generated. We randomly reserve 1/3 of the queries for testing purpose, and use the rest for parameter tuning (5-fold cross validations) and training. As a practical concern, we need to prevent the system from using information obtained later than a query when random walking

[2] http://www.nlm.nih.gov/mesh/introduction.html
[3] www.geneontology.org
[4] www.ncbi.nlm.nih.gov/pubmed
[5] www.ncbi.nlm.nih.gov/pmc

for that query. Therefore, we define a *time variant graph* in which each edge in the graph is tagged with a time tag (the year in which this edge is added to the graph). When random walk is performed for a particular query, we only consider edges that are earlier than the query's year.

### 2.3 History-based Reading Recommendation

In the case where users' reading activities are accessible we define the following *history-based reading recommendation* task—given the papers read by a user for the past years, recommending new publications which might interest this user.

In this study a biologist (Dr. W) whose major research interest is *Saccharomyces* volunteered to provide his reading information. We collect all the papers Dr. W has read during 1988-2008 (the data were collected during 2009). We have found 364 papers from his computer, and 265 of which can be matched in the yeast dataset. Since Dr. W cannot remember exactly in which year he read each of the papers, we assume that he read each paper the following year after its publication. This way of generating training data is suitable for learning models which are good at suggesting the most recent publications to the users. The yeast graph is augmented with two extra relations. One is the relation *Read*, which links a user (Dr. W) to all the papers he has read in the past (265 papers). Another, with some notation overloading, is the *Read* relation, which links each year to the set of papers read by the scientist at that year. Dr. W is recorded to have published 56 papers in the yeast database.

Our task is to predict for each year, which papers Dr. W actually read based on the papers he had read before that year. A query consisting of the user node (corresponding to Dr. W) and the year node is generated for each year, and its relevant target nodes are the papers which are actually read by Dr. W in that year. Therefore, we have 21 labeled queries. We tune parameters with leave-one-out cross validations on all 21 queries. We further evaluate the final model (trained on all queries) by Dr. W's judgement of recommended papers for him to read in 2009.

## 3 Approach

In this section, we summarizes the PCRW-based model, which is called Path Ranking Algorithm (PRA) by Lao and Cohen[15]. In order to compare the effectiveness of this random walk model to existing approaches, we first describe two commonly used recommendation approaches– content-based and community-based recommendations.

### 3.1 Content-based Recommendation

Recommendation systems are usually classified into *content-based* and *community-based* approaches [1]. Both of them have proved to be useful under different settings. The content-based recommendation approaches [4, 19, 14, 18]

are rooted in information retrieval settings, where descriptive features can be extracted from the items, but no rating information from users other than the current one is needed. The users are recommended with items similar to the ones the user preferred in the past.

More formally, each item $i$ is represented by an item profile $Item(i)$, which is a vector of features. In our case of publication recommendation, each item is an paper, and each feature is a piece of metadata information such as author, venue, topic category, title word, gene, chemical substance, etc. One of the best known weighting scheme for the features is the TF-IDF measure, in which the weight of the $j$-th feature to the $k$-th item is defined as

$$w_{k,j} = TF_{k,j} * IDF_j = \frac{f_{k,j}}{\max_z f_{k,z}} * \log \frac{N}{n_j}, \tag{1}$$

where $f_{k,j}$ is the number of times feature $j$ appears in item $k$, $N$ is the total number of items, and the $j$-th feature appears in $n_j$ of them. Similarly, each user $u$ is represented by a user profile $User(u)$, which is usually the average profile [4, 14] of all the items user $u$ have liked. In our case of publication recommendation, it is the set of all papers that a user have read in the past. Finally, the utility of item $i$ to user $u$ is estimated by the similarity between their profile vectors, often measured by cosine function:

$$U(u,i) = cos(User(u), Item(i)). \tag{2}$$

### 3.2 Community-based Recommendation

The community-based recommendation approaches (also called collaborative filtering) [21, 10, 13, 20, 12, 22] predict the utility of items for a particular user based on the items previously rated by a sub-community, and no content information is assumed. The users are recommended by items that people with similar tastes and preferences liked in the past.

More formally, the unknown rating $r_{u,i}$ for user $u$ and item $i$ is usually computed as an aggregation of the ratings of some other (usually, the $K$ most similar) users for the same item $i$:

$$Rate(u,i) = aggr_{u' \in N(u,K)} Rate(u', i),$$

where $N(u,K)$ is the set of $K$ users that are the most similar to user $u$ (and who have rated item $i$). In the simplest case, the aggregation can be the average:

$$Rate(u,i) = \frac{1}{K} \sum_{u' \in N(u,K)} Rate(u', i).$$

However, the most common approach is a weighted sum:

$$Rate(u,i) = \frac{1}{Z} \sum_{u' \in N(u,K)} Sim(u,u') Rate(u', i), \tag{3}$$

where $Sim(u,u')$ is a similarity measure between user $u$ and $u'$, and $Z$ is a normalization constant $\sum_{u' \in N(u,K)} Sim(u,u')$. In our case of publication

recommendation, we can only observe if a user read a document or not, instead of rating scores. Therefore, we choose the cosine-based approach when calculating the similarities between users:

$$Sim(u, u') = \cos(f_u, f_{u'}), \tag{4}$$

where $f_u$ is a binary vector indicating whether or not user $u$ has read each of all the documents.

Furthermore, the content-based model's prediction for the current user can be combined with the prediction to its neighbors. We use a parameter $\alpha$ to control their relative importance, and the final scoring function has the form

$$Rate(u, i) = \alpha U(u, i) + \frac{1}{Z} \sum_{u' \in N(u,K)} Sim(u, u') U(u', i). \tag{5}$$

### 3.3 Path Ranking Algorithm

Path Ranking Algorithm (PRA) introduced by Lao and Cohen [15] addresses classification and retrieval tasks using learned combinations of labeled paths through a graph. PRA treats a data base as a typed graph, with nodes representing documents, users, words, genes, etc., and edges representing relations among them. It learns to *rank* target nodes $t$ relative to a query node $s$. It uses random walk probabilities following different edge type sequences as features to summarize the relation between query and target nodes.

Following Lao and Cohen, we defined a *relation path* $\pi$ as a sequence of relations $R_1 \dots R_\ell$. In order to emphasize the types associated with each step, $\pi$ can also be written as $T_0 \xrightarrow{R_1} \dots \xrightarrow{R_\ell} T_\ell$, where $T_i = range(R_i) = domain(R_{i+1})$, and we also define $domain(P) \equiv T_0$, $range(P) \equiv T_\ell$. In this notation, the concept of "papers published in certain years" and "papers popularly cited in certain years" can be compactly expressed as the following two relation paths:

$$P1: \text{ year} \xrightarrow{\text{PublishedIn}^{-1}} \text{paper}$$
$$P2: \text{ year} \xrightarrow{\text{PublishedIn}^{-1}} \text{paper} \xrightarrow{\text{Cite}} \text{paper}$$

This notation makes it clear that the range of each relation path is *paper*, the desired type for our recommendation task. We use $^{-1}$ to denote the inverse of a relation. For example, if paper $a$ has relation $PublishedIn$ with year $y$, then year $y$ has relation $PublishedIn^{-1}$ with paper $a$.

Given a set of paths $\{\pi_1, \dots, \pi_m\}$, one could treat $P(s \to t; \pi_i)$, the probability of reaching target node $t$ starting from source node $s$ and following path $\pi_i$ as a relational feature for $t$. The following function gives a scoring of target nodes related to the query node $s$

$$score(s, t) = \sum_{\pi \in \mathcal{P}_\ell} P(s \to t; \pi) \theta_\pi, \tag{6}$$

where $\mathcal{P}_\ell$ is the set of relation paths with length $\leq \ell$, and $\theta_\pi$ are appropriate weights for the features.

Given a relation $R$ and a set of node pairs $\{(s_i, t_i)\}$ for which we know whether $R(s_i, t_i)$ is true or not, we can construct a training dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}$, where $\mathbf{x}_i$ is a vector of all the path features for the pair $(s_i, t_i)$—i.e., the $j$-th component of $\mathbf{x}_i$ is $P(s_i \to t_i; \pi_j)$, and $y_i$ is a boolean variable indicating whether $R(s_i, t_i)$ is true. The parameters $\theta$ can be estimated by a regularized logistic regression model. A biased sampling procedure selects only a small subset of negative samples as part of the training dataset (see [15] for detail).

## 4 Experiment

We report empirical results of comparing PRA with unsupervised random walk with restart model (RWR no training), and its supervised version (RWR). We use standard retrieval quality metrics like Mean Reciprocal Rank (MRR[6]) and Mean Average Precision (MAP[7]) to evaluate different models. While MRR focuses on top part of the result list, MAP evaluates the overall quality of the result list, or how easy it is to find all the relevant documents.

All experiments were run on a machine with 16 core Intel Xeon 2.33GHz CPU and 24Gb of memory. For parameter estimation of supervised RWR model, we use the same log-likelihood objective function and L-BFGS optimization procedure as for PRA. For community-based recommendation approach, we briefly tune both $K$ and $\alpha$ on the training set.

### 4.1 Citation-based Reading Recommendation

Figure 2 shows the effects of $\ell_2$- and $\ell_1$-regularization to recommendation quality measured by MRR on the tuning query set. More complex models ($L = 3, 4$) have better recommendation accuracies. $\ell_2$ regularizer is very important for preventing over fitting, and $\ell_1$ regularizer can slightly improve recommendation quality. Though not shown in Figure 2, $\ell_1$ regularizer can significantly reduce the number of features with non-zero weights. We report results for models up to maximum path length $L = 4$, because the random walk features of length 5 cannot be fit into memory.

Table 1 shows highest and lowest weighted features for the trained model of L=3. All the high positive weight features are related to the user's reading or publication history. Most of these features correspond to content-based recommendation strategies (feature 1, 3, 4, 6, and 7)—first find papers read by the current user before, then find similar paper through shared contents like authors, genes, chemicals, MeSH descriptors, or title words. Interestingly,

[6] MRR is defined as the inverse rank of the highest ranked relevant document in a set of results. If the the first returned document is relevant, than MRR is 1.0, otherwise, it is smaller than 1.0.

[7] MAP is defined as the average precisions at the position of each relevant documents in a sorted result list.

feature 2 takes a collaborative filtering approach—first find other users with similar reading history, and then find what they publish in the current year.

All the high negative weight features (9-14) are independent of the user's behaviors but related to the recency of a paper's topics, authors, mentioned genes or chemicals. Take feature 13 as an example, if an paper $p$ of the current year is written by an author who has published no paper in the past years, then a random walker starting from $p$ and following the path

$$\text{paper} \xrightarrow{\text{Write}^{-1}} \text{author} \xrightarrow{\text{Write}} \text{paper}$$

is very likely to return to paper $p$ therefore receive a relatively big negative weight. However, if the author has published many papers in the past years, the walker is very likely to end up in a paper in the past years, thus the paper would only receive a small negative weight in PRA scoring function.

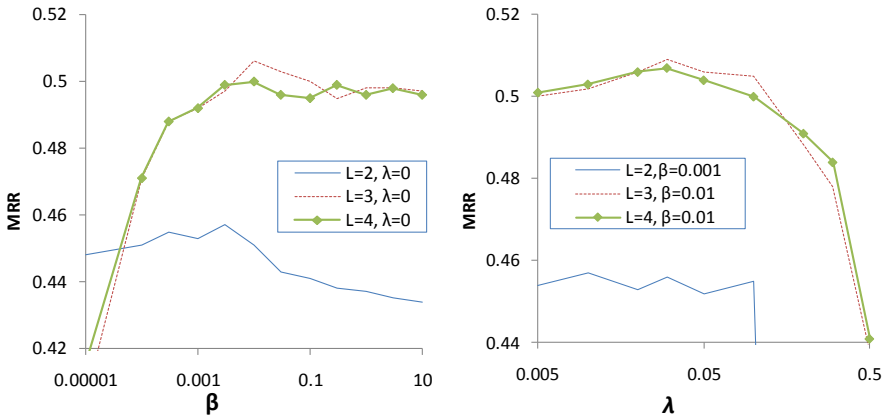


**Fig. 2.** Tuning $\ell_2$-regularizer parameter $\beta$ (left) and $\ell_1$-regularizer parameter $\lambda$ (right) for the citation-based reading recommendation on the yeast dataset. $L$ is the maximum length of PRA paths.

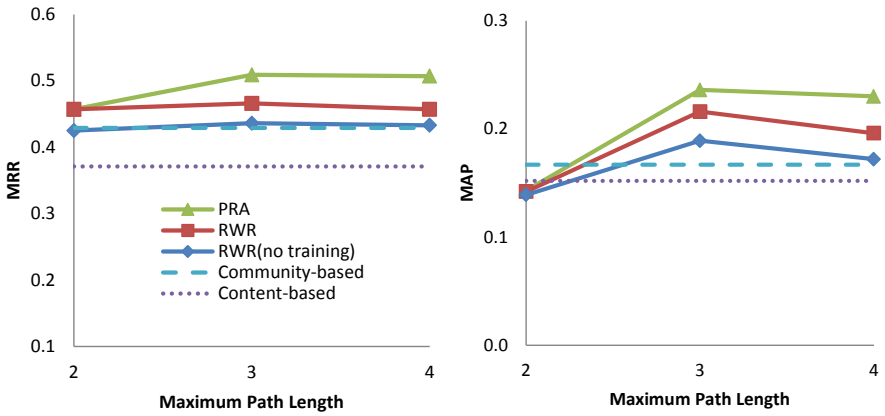


**Fig. 3.** Compare different approaches for the citation-based reading recommendation task on the yeast data. $L$ is the maximum length of relation paths. $K = 3, \alpha = 0$

**Table 1.** Highest and lowest weighted features from a PRA model with maximum path length $L = 3$ trained for the citation-based reading recommendation task on the yeast data. The $\theta_P$ column shows the weights.

| ID | $\theta_P$ | Feature | Comments |
|---|---|---|---|
| 1 | 4.8 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{Write}^{-1}}$ author $\xrightarrow{\text{Write}}$ paper | papers from my favorite authors |
| 2 | 3.6 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{Read}^{-1}}$ author $\xrightarrow{\text{Write}}$ paper | papers of scientist who read the same papers as I do |
| 3 | 2.1 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{HasGene}}$ gene $\xrightarrow{\text{HasGene}^{-1}}$ paper | papers about my favorite genes |
| 4 | 1.4 | author $\xrightarrow{\text{Write}}$ paper $\xrightarrow{\text{HasChem.}}$ chemical $\xrightarrow{\text{HasChem.}^{-1}}$ paper | papers about chemicals that I have worked on |
| 5 | 1.4 | author $\xrightarrow{\text{Write}}$ paper $\xrightarrow{\text{Read}^{-1}}$ author $\xrightarrow{\text{Write}}$ paper | papers from scientists who read my papers |
| 6 | 1.4 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{HasMajorMD}}$ topic $\xrightarrow{\text{HasMD}^{-1}}$ paper | papers about my favorite MeSH descriptors |
| 7 | 1.4 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{HasTitle}}$ word $\xrightarrow{\text{HasTitle}^{-1}}$ paper | papers with similar titles to what I read before |
| 8 | 1.3 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{Cite}^{-1}}$ paper | follow up papers to what I read |
| | ... | | |
| 9 | -0.3 | year $\xrightarrow{\text{In}^{-1}}$ paper $\xrightarrow{\text{HasMD}}$ topic $\xrightarrow{\text{HasMD}^{-1}}$ paper | papers of less established MeSH descriptors |
| 10 | -0.4 | year $\xrightarrow{\text{In}^{-1}}$ paper $\xrightarrow{\text{HasGene}}$ gene $\xrightarrow{\text{HasGene}^{-1}}$ paper | papers of less established genes |
| 11 | -0.6 | year $\xrightarrow{\text{In}^{-1}}$ paper $\xrightarrow{\text{HasMajorMQ}}$ topic $\xrightarrow{\text{HasMajorMQ}^{-1}}$ paper | papers of less established major MeSH qualifiers |
| 12 | -0.9 | year $\xrightarrow{\text{In}^{-1}}$ paper $\xrightarrow{\text{HasMQ}}$ topic $\xrightarrow{\text{HasMQ}^{-1}}$ paper | papers of less established MeSH qualifiers |
| 13 | -1.2 | year $\xrightarrow{\text{In}^{-1}}$ paper $\xrightarrow{\text{Write}^{-1}}$ author $\xrightarrow{\text{Write}}$ paper | papers of less established authors |
| 14 | -1.6 | year $\xrightarrow{\text{In}^{-1}}$ paper $\xrightarrow{\text{HasChem.}}$ chemical $\xrightarrow{\text{HasChem.}^{-1}}$ paper | papers of less established chemicals |

Figure 3 shows final evaluation results on the test queries. The community-based approach is significantly better than the content-based approach, but close to the untrained RWR model. The trained RWR model performs better than the untrained version, but only to certain extend. More complex versions (L=3 and 4) of PCRW-based models are significantly more accurate than RWR-based approaches and traditional recommendation approaches.

### 4.2 History-based Reading Recommendation

Figure 4 shows the effect of $\ell_2$- and $\ell_1$-regularization to recommendation quality measured by MRR on the tuning query set. Although $\ell_2$ regularizer is very important for preventing over fitting, $\ell_1$ is less useful in this task. We can see that if $\ell_2$ regularizer is too small or too big MRR is significantly lower. This indicates that with very small number of training queries (21 for this task) the model can easily overfit or under fit.

Table 2 shows top weighted features for models of different complexities. For the simplest type of model (L=2), only one feature (beside the bias term) has non-zero weight. It leverages the user's reading behaviors in the past. More complex models (L=4 and 5) are able to differentiate the user's reading information one, two, and three years back through the following paths

$$\text{author} \xrightarrow{\text{Read}} \text{paper}$$
$$\text{year} \xrightarrow{\text{After}} \text{year} \xrightarrow{\text{Read}} \text{paper}$$
$$\text{year} \xrightarrow{\text{After}} \text{year} \xrightarrow{\text{After}} \text{year} \xrightarrow{\text{Read}} \text{paper}$$

Figure 5 shows the final evaluation results. The community-based approach is significantly better than the content-based approach, but close to the untrained RWR model. Again the trained RWR model performs better than the untrained version but only to certain extend, and more complex versions (L=4 and 5) of PCRW-based models are significantly more accurate than RWR-based approaches and traditional recommendation approaches. However, it would be an interesting future work to apply efficient path finding and feature selection techniques to explore features which correspond to longer paths.

We further apply the PRA model trained on all the queries to predict which papers Dr. W would like to read for 2009. For each query, results with negative scores (the bias feature is ignored) are removed, because they are even less relevant judged by our model than most documents in the database which are not retrieved. We consider it meaningless to show these results to the user. We show top 50 results from 2009 to Dr. W, and Dr. W labels for each of them whether it is relevant to his research interest. The accuracy judged by Dr. W is p@50=0.82, which is reasonably high for such kind of recommendation task.

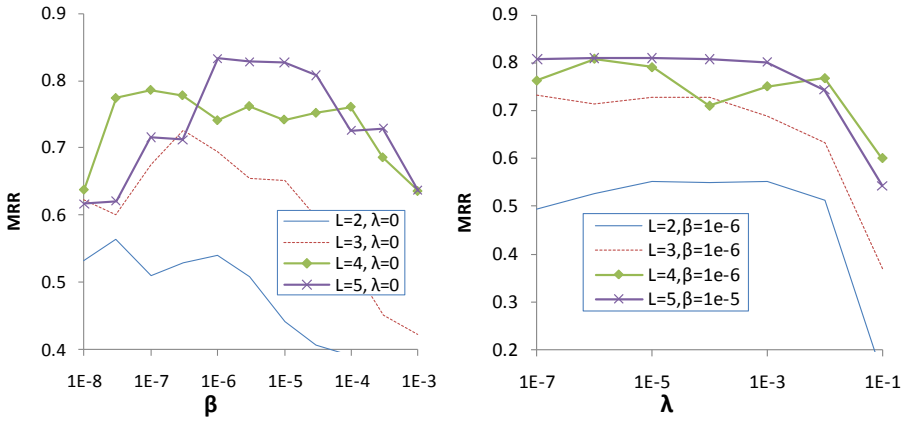


**Fig. 4.** Tuning $\ell_2$-regularizer parameter $\beta$ (left) and $\ell_1$-regularizer parameter $\lambda$ (right) for the citation-based reading recommendation on the yeast dataset. $L$ is the maximum depth of relation trees.

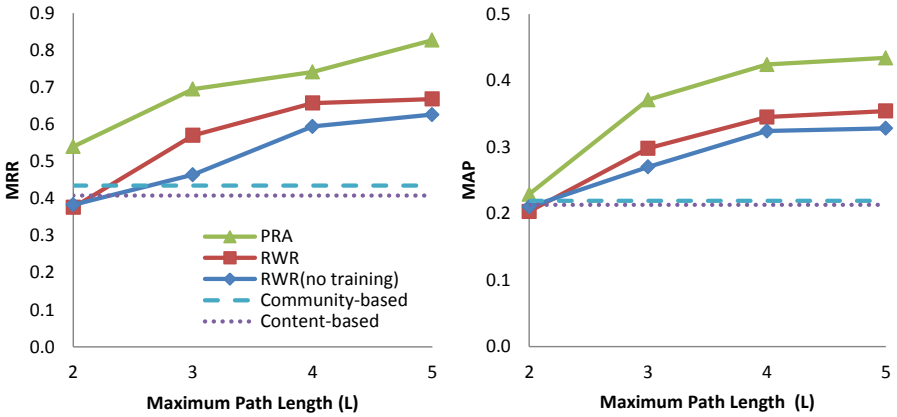


**Fig. 5.** Compare different approaches for the history-based reading recommendation task on the yeast data. $L$ is the maximum length of relation paths. $K = 30, \alpha = 0$.

**Table 2.** Top five weighted (non-zero) features for PRA models with different maximum path length $L$ trained for history-based reading recommendation task on the Yeast dataset. The $\theta_P$ column shows the weights.

| ID | $\theta_P$ | Feature | Comments |
|---|---|---|---|
| L=2 | | | |
| 1 | 803.8 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{Cite}^{-1}}$ paper | follow up papers to what I read |
| L=3 | | | |
| 2 | 645.9 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{Write}^{-1}}$ author $\xrightarrow{\text{Write}}$ paper | papers from my favorite authors |
| 3 | 590.4 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{Read}^{-1}}$ author $\xrightarrow{\text{Write}}$ paper | papers of scientist who read the same papers as I do |
| 4 | 464.4 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{HasMajorMQ}}$ topic $\xrightarrow{\text{HasMajorMQ}^{-1}}$ paper | papers about my favorite topics |
| 5 | 438.0 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{HasTitle}}$ word $\xrightarrow{\text{HasTitle}^{-1}}$ paper | papers with similar titles to what I read before |
| 6 | 357.1 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{Cite}}$ paper $\xrightarrow{\text{Cite}^{-1}}$ paper | papers which cite the same papers as what I read |
| L=4 | | | |
| 7 | 438.5 | year $\xrightarrow{\text{After}}$ year $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{Cite}}$ paper $\xrightarrow{\text{Cite}^{-1}}$ paper | papers which share citations with what I read last year |
| 8 | 382.0 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{Write}^{-1}}$ author $\xrightarrow{\text{Write}}$ paper | papers from my favorite authors |
| 9 | 345.8 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{HasMajorMQ}}$ topic $\xrightarrow{\text{HasMajorMQ}^{-1}}$ paper | papers about my favorite topics |
| 10 | 253.8 | year $\xrightarrow{\text{After}}$ year $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{HasMD}}$ topic $\xrightarrow{\text{HasMD}^{-1}}$ paper | papers which share MeSH descriptor with what I read last year |
| 11 | 237.8 | author $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{Read}^{-1}}$ author $\xrightarrow{\text{Write}}$ paper | papers of scientist who read the same papers as I do |
| L=5 | | | |
| 12 | 108.5 | year $\xrightarrow{\text{After}}$ year $\xrightarrow{\text{After}}$ year $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{HasMajorMQ}}$ topic $\xrightarrow{\text{HasMajorMQ}^{-1}}$ paper | papers which share MeSH descriptor with what I read 2 years back |
| 13 | 99.9 | year $\xrightarrow{\text{After}}$ year $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{Cite}}$ paper $\xrightarrow{\text{Read}^{-1}}$ author $\xrightarrow{\text{Write}}$ paper | papers by users who read what I cited last year |
| 14 | 87.2 | year $\xrightarrow{\text{After}}$ year $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{HasTitle}}$ word $\xrightarrow{\text{HasTitle}^{-1}}$ paper | papers which share title words with what I read last year |
| 15 | 81.1 | author $\xrightarrow{\text{Write}}$ paper $\xrightarrow{\text{HasMajorMQ}}$ topic $\xrightarrow{\text{HasMQ}^{-1}}$ paper | papers which share MeSH qualifier with what published |
| 16 | 80.6 | year $\xrightarrow{\text{After}}$ year $\xrightarrow{\text{After}}$ year $\xrightarrow{\text{Read}}$ paper $\xrightarrow{\text{HasTitle}}$ word $\xrightarrow{\text{HasTitle}^{-1}}$ paper | papers which share title words with what I read 2 years back |

## 5 Discussion

In this study, we use a graph representation for publication databases with rich metadata. With this representation, a path constrained random walk (PCRW) model is trained to discover effective recommendation strategies represented as edge paths on the graph. Experiments on both citation-based and history-based reading recommendation tasks show that by leveraging rich context information the PCRW-based approach outperforms RWR-based approach as well as traditional content-based and collaborative filtering approaches. We also extended the PCRW-based approach to include $\ell_1$-regularizer on the parameters, which is shown in our experiment to further improve the recommendation quality.

We are aiming at releasing this system as an online service, so that biologists of certain domains can use it during their daily research work. By having more users interact with the system, we can also acquire more training data for developing better recommendation systems.

In the future, it would be interesting to apply efficient path finding and feature selection techniques to explore features which correspond to longer paths.

## Acknowledgement

We gratefully acknowledge the support of NSF grant IIS-0811562 and NIH grant R01GM081293. We sincerely thank Dr. John Woolford for donating over 20 years of pdf data from his computer.

## References

1. Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.*, 17(6):734–749, 2005.
2. Alekh Agarwal, Soumen Chakrabarti, and Sunny Aggarwal. Learning to rank networked entities. *KDD*, pages 14–23, 2006.
3. Andrew Arnold and William W. Cohen. Information extraction as link prediction: Using curated citation networks to improve gene detection. *ICWSM*, pages 541–550, 2009.
4. Marko Balabanovic and Yoav Shoham. Content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, 1997.
5. Chumki Basu, Haym Hirsh, William W. Cohen, and Craig G. Nevill-Manning. Technical paper recommendation: A study in combining multiple information sources. *J. Artif. Intell. Res. (JAIR)*, 14:231–252, 2001.
6. Gaurav Bhalotia, Arvind Hulgeri, Charuta Nakhe, Soumen Chakrabarti, and S. Sudarshan. Keyword searching and browsing in databases using banks. *ICDE*, pages 431–440, 2002.
7. Matthew Brand. A random walks perspective on maximizing satisfaction and profit. In *SDM*, 2005.
8. Soumen Chakrabarti and Alekh Agarwal. Learning parameters in entity relationship graphs from ranking preferences. *PKDD*, pages 91–102, 2006.

9. Michelangelo Diligenti, Marco Gori, and Marco Maggini. Learning web page scores by error back-propagation. *IJCAI*, pages 684–689, 2005.
10. David Goldberg, David A. Nichols, Brian M. Oki, and Douglas B. Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70, 1992.
11. Qi He, Jian Pei, Daniel Kifer, Prasenjit Mitra, and C. Lee Giles. Context-aware citation recommendation. *WWW*, pages 421–430, 2010.
12. William C. Hill, Larry Stead, Mark Rosenstein, and George W. Furnas. Recommending and evaluating choices in a virtual community of use. In *CHI*, pages 194–201, 1995.
13. Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John Riedl. Grouplens: Applying collaborative filtering to usenet news. *Commun. ACM*, 40(3):77–87, 1997.
14. Ken Lang. Newsweeder: Learning to filter netnews. In *ICML*, pages 331–339, 1995.
15. Ni Lao and William W. Cohen. Relational retrieval using a combination of path-constrained random walks. *Machine Learning*, 2010.
16. Prem Melville and Vikas Sindhwani. Recommender systems. *Encyclopedia of Machine Learning, Claude Sammut and Geoffrey Webb (Eds), Springer*, 2010.
17. Einat Minkov, William W. Cohen, and Andrew Y. Ng. Contextual search and name disambiguation in email using graphs. *SIGIR*, pages 27–34, 2006.
18. Raymond J. Mooney and Loriene Roy. Content-based book recommending using learning for text categorization. In *ACM DL*, pages 195–204, 2000.
19. Michael J. Pazzani and Daniel Billsus. Learning and revising user profiles: The identification of interesting web sites. *Machine Learning*, 27(3):313–331, 1997.
20. Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *CSCW*, pages 175–186, 1994.
21. Elaine Rich. User modeling via stereotypes. *Cognitive Science*, 3(4):329–354, 1979.
22. Upendra Shardanand and Pattie Maes. Social information filtering: Algorithms for automating "word of mouth". In *CHI*, pages 210–217, 1995.
23. Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. Random walk with restart: fast solutions and applications. *Knowledge and Information Systems*, 14(3):327–346, 2007.
24. Kristina Toutanova, Christopher D. Manning, and Andrew Y. Ng. Learning random walk models for inducing word dependency distributions. *ICML*, 2004.
25. Fan Wang, Ben Chen, and Zhaowei Miao. A survey on reviewer assignment problem. *IEA/AIE*, pages 718–727, 2008.
26. Ding Zhou, Shenghuo Zhu, Kai Yu, Xiaodan Song, Belle L. Tseng, Hongyuan Zha, and C. Lee Giles. Learning multiple graphs for document recommendations. *WWW*, pages 141–150, 2008.