RecAuth: Recommending Authors Based on Keywords

Ang Li
angli5@illinois.edu

Yaning Liang
liang33@illinois.edu

Abstract

Ample references should be collected prior to conducting research and writing academic papers. The references should be of high quality, which are often produced by authors who are highly reputable in their respective field of research. Less experienced researchers would benefit greatly from a tool that gives them suggestions on the authors that share their research interests. A framework, called RecAuth, was developed to recommend a list of authors based on an input set of keywords. The recommended authors have high relevance to the key phrases and are highly ranked. The algorithm leveraged the propagation of ranking over venues and authors to generate author rankings. It was found that the recommend authors are highly consistent with the top ranking authors found on Google Scholar.

1. Introduction

It is critical for researchers to have a good understanding of the work that has already been done in their field of research. Reading the influential work done by highly reputable authors allows both extensions and innovations to be made. However, there are many very well-known authors across all disciplines, which makes it difficult for those who are relatively inexperienced to determine which reputable authors' work they should read. A system that can automatically recommend authors based on series of keywords would be very beneficial for those who are looking for some guidance.

Current search engines such as Google Scholar are only capable of returning papers as results. Sometimes much additional effort is required to filter out irrelevant papers. In addition, these engines' functionality to search papers by authors require the user to know the authors' names ahead of time. A system that makes author recommendations can effectively return a small number of authors who have most likely has produced highly relevant work.

There is currently no author recommendation systems using keyword-based queries. Most of the relevant work deals with citation recommendation. For example, ClusCite is an information network-based clustering citation system[1]. Although it is able to effectively make citation recommendations, it does not solve the aforementioned problem completely.

We developed a framework, called RecAuth, that answers keyword-based queries and recommends reputable authors who have done work closely related to the query terms. From a DBLP dataset, key words were first extracted from paper abstracts. A heterogeneous bibliographic network was then built using the extracted terms, authors and venues for each paper. Through the network, authors were associated with venues and coauthors. Similarity scores for each author was then calculated based on the relationship between the query and the papers the authors have written. The authors were then ranked by performing iterations, where authority was propagated between authors and venues and between authors and coauthors. Upon convergence, the highest ranked authors were returned as the recommendation result.

Experiments were performed to tune various input parameters. The performance of the framework was examined via a series of sample inputs. The same queries were also executed on Google Scholar. The authors recommended by RecAuth was compared against the authors of the papers returned by Google Scholar. It was found that authors recommended by our framework were some of the most influential and reputable authors who have produced work in fields related to the input query. Google Scholar, on the other hand, was only able to return a small list of papers published by highly reputable authors.

2. Background and Related Work

2.1. Background Information

Some concepts related to text and phrase mining. While developing our framework, phrase mining techniques are performed to extract terms from abstracts because when calculating similarities between papers and input keywords, it is essential to have an accu-
rate representation of papers using a concise list of key phrases. Possible techniques include TurboTopics, KERT, ToPMine and SegPhrase+\cite{3, 5, 4}. Having explored these techniques to perform phrase mining, SegPhrase+ had excellent performance and was incorporated RecAuth.

SegPhrase+ is an algorithm used to extract high quality phrases from a large set of input documents. Since this algorithm requires very little supervision, and the quality of result produced outperformed any other phrase mining methods like TopMine, KERT, TurboTopics. In addition, SegPhrase+ does not require any kind of NLP methods, thus providing the possibility of extending our algorithm to other languages in the future. SegPhrase+ follows a four (4) step process when mining quality phrases which are: 1. Frequent Phrase Detection; 2. Phrasal Quality Estimation based on popularity, concordance, informativeness and completeness; 3. Phrasal Segmentation; 4. Feedback as Segmentation Features. SegPhrase+ also incorporates a very small set of user specified labels to perform training. Thus allowing high quality phrases to be mined from a huge text corpus.

Similarity measurements are also explored in our research. Jaccard Coefficient calculates similarity between finite sets of samples, and the main principal is to calculate how much overlap intersection of the input sets over the union of input. Cosine Similarity measurement is inspired by using Euclidean dot product to calculate similarity. In order to convert text documents to a vector form, We can leverage Text Frequency (TF) and Inverse Document Frequency (IDF) to convert text streams to vectors.

PathSim is used to perform similarity measurements in heterogeneous network over different meta paths\cite{10}. The main intuition of PathSim is that it explores how different types of paths in the heterogeneous network, and the number of paths that connect the two components. The result PathSim produces favors peers other than a hierarchical relationship.

### 2.2. Related Work

There are some existing systems that make citation recommendations based on queries. ClusCite is an algorithm that recommends relevant papers to cite given a set of query manuscripts. It greatly leverages the nature of heterogeneous network, and explores different meta paths to perform recommendation. The framework makes the assumption that instead of hard-clustering citations, soft clusters of different interest groups would produce better results. Having obtained the different interest groups, the framework predicts citations by using separate models for each group. More specifically, the framework uses group membership information and infers authority and relevance of each group. The authority scores are propagated via iterations, which is a joint optimization problem\cite{11}. Though the system is effective at recommending papers, it is not capable of recommending authors.

Many other citation recommendation systems also exist, many of which depend on different types of information, such as paper content, known citations, paper venues, and paper authors. A particular effective technique, developed by Yu et al., extracts meta-path based features from heterogeneous bibliographic networks\cite{15}. This technique is capable of capturing text-based similarity, conceptual relevance, and different types of social relatedness. Using this information, accurate citation recommendation can be achieved.

Another system, developed by Bethard et al., utilizes a linearly weighted model that considers both relevance and authority features\cite{2}. This approach addresses the problem that critical information such as the paper importance and quality not being considered.

Authority ranking on graphs is a critical step for such recommendation systems and has been studied extensively. A system, developed by Sun et al., propagates paper authority scores bibliographic networks by considering the paper citation frequency and the prestige of the published venue\cite{12}. If some supervision can be applied to the ranking process, the performance can usually be improved. These methods, however, do not consider the authority bias involved with changes in query topic or interests\cite{11}.

A personalized PrankRank algorithm, developed by Haveliwala et al., derives authority scores specifically for each query by considering query topics\cite{7}. This idea allows the representation of different classes using features obtained from object relative authority, which is employed when performing clustering and classification in heterogeneous information networks\cite{8, 14}.

### 3. Framework

The algorithm we developed followed two main intuitions.

1. Use papers as linkage between input keywords and authors

Since there is no direct correlation between authors and the input set of keywords, we used papers the authors publish to link the authors and input set of keywords. Thus similarity between authors and the input set of keywords can be generated by calculating similarities between papers.
this specific author publish and the set of keywords. Techniques on how to calculate similarities between papers and the set of keywords will be further illustrated in the following sections.

2. Authority score of authors can be inferred by venues this author publish papers at and his co-authors

The authority score of authors can be divided into two components. The first component is based on the authority score of venues this author publish papers at. A highly reputable author should publish papers at highly ranked venues, and highly reputable venues attract highly ranked authors. The second component of an author’s authority score come from people this person co-authors with. The rank of an author can be enhanced if the people he co-authors with on publications are highly reputable. Thus an author’s authority score can be calculated by integrating the authority score from venues and authority score form his co-authors.

The algorithm we implemented can be divided into two stages. The first stage is an offline process. The offline stage is independent of the input keywords or any user specified parameters. The offline stage involves running SegPhrase+ for all the abstract from papers in the DBLP dataset. The second step of the offline stage is used to generate relationship matrices between Author Paper Venue and Author Paper Author. The online process includes calculating similarities between input keywords and key phrases from papers in the dataset. The second step of the online process is to update similarity matrices between Author Paper Venue and Author Paper Author based on the similarity scores calculated.

3. Offline Stage

3.1. Represent Paper Abstracts using SegPhrase+

We selected SegPhrase+ among all the phrase mining techniques to mine key phrases from the abstract of all the papers. The reason why SegPhrase+ is selected is because it uses very little supervision, and the result it produced are of high quality. Also, the scope of SegPhrase+ can be applied to a large text corpora, thus only the abstract section from all the papers in the DBLP dataset is also time efficient. SegPhrase+ does not use any Natural Language Processing (NLP) methods either. Thus the possibility of exploring our algorithm to other languages can also be explored. After running the abstract section of each paper through SegPhrase+, we generate a list of key phrases to represent each paper. For example, some key phrases generated are: For the paper Recommending user generated item lists published by Yidan Liu, Min Xie, and Laks V.S. Lakshmanan in 2014, the key phrases returned are: Collaborative Filtering, Model Learning, Collective Matrix Factorization. Another example is for the paper Mining high-speed data streams published by Pedro Domingos and Geoff Hulten in year 2000, the key phrases mined from the abstract are: Decision Trees, Incremental Learning, Disk-based algorithms, subsampling. As we can see from the examples, SegPhrase+ generates very high quality phrases representation of the papers in the dataset.

3.1.2 Generate APV and APA Relations

Of all the papers, authors and venues in the dataset, we used papers as linkage to generate APV and APA matrices, as shown below. The intersection between authors and venues represent the set of papers this specific author published at this certain venue. Similarly, an APA matrix is generated from the dataset representing co-author relationships. Each cell within this matrix represent the set of papers these two authors co-authored on. Different from the APV matrix, this matrix is diagonally symmetric. Since the relationship between a person co-authoring with himself is not applicable, and we treat every author equally for the co-author relationship regarding a specific paper. Thus APA matrix is diagonally symmetric.

3.2. Online Stage

The offline process is directly performed on the DBLP dataset, and is invariant to the input keywords. As for the online process. The online stage will involve...
calculations of the similarity scores between papers and input keywords, and updating authority scores until converges.

### 3.2.1 Generate APV and APA Similarity Matrices

Based on the APV and APA matrices generated from Step 2, and key phrases that represent the papers from Step 1, we perform similarity assessment between the input keywords and the phrases. Multiple similarity measurements were explored including Jaccard Coefficient, PathSim and Cosine Similarity. Cosine Similarity is chosen here because it excels in calculating text similarities, and is very time efficient. From the set of key phrases for each paper, we first parse these phrases to unigram words, and perform cosine similarity between the words following the equation below:

\[
\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}
\]

Here \(A_i\) and \(B_i\) represent the input keywords and keyphrases representing each paper respectively.

An example of the APV similarity matrix is shown below. As we can infer from Figure 2, each cell in this matrix represent a score of the authority of a specific author corresponding to the specific venue. This score is a sum of similarity scores of all the papers published at this venue corresponding to the input keywords. When we see a 0 in this matrix, it’s either because of this author does not have any publication at this specific venue, or because of the papers this author published at this venue have a similarity score of 0 with the input keywords.

![Figure 2: APV Similarity Matrix](image)

### 3.2.2 Calculate authority scores for both authors and venues

Based on APV and APA matrices generated above, we calculate authority scores for both venues and authors. We follow the three rules specified below:

1. Highly ranked authors publish papers at highly ranked venues

   Authority scores of authors can be inferred by authority scores of venues these authors publish papers at. This rule can be represented with the equation below:

   \[
r_Y(j) = \sum_{i=1}^{m} W_{YX}(j,i) r_X(i)
   \]

   Here we have \(Y\) represent type author, and \(r_Y(j)\) represent the authority score of a specific author \(j\).

2. Highly ranked venues attract highly ranked authors

   Authority score of venues can be inferred by authority scores of authors. This rule can be represented with the equation below:

   \[
r_X(i) = \sum_{j=1}^{n} W_{XY}(i,j) r_Y(j)
   \]

   Here \(X\) represent type venue, and \(r_X(i)\) represent the authority score of a specific venue \(i\).

3. Rank of an author is enhanced if it co-authors with many highly ranked authors
Leveraging the APA matrix, if an author co-authors with authors that are highly reputable, his authority score could be enhanced. This rule can be represented with the equation below:

\[ r_Y(j) = \alpha \sum_{i=1}^{m} W_{X}(j,i)r_x(i) + (1 - \alpha) \sum_{j=1}^{n} W_{Y}(i,j)r_y(j) \]  

(4)

The first part of this equation is the same as the equation for rule 1. The second part of the equation is generated from the APA Similarity matrix.

Based on these three rules specified, we generated our mechanism to calculate authority scores of papers and venues. Authority scores for venues are calculated directly from the APV Similarity matrix shown in Figure 2. In order to calculate authority scores of authors, we need to leverage both APV Similarity matrix shown in Figure 2 and APA Similarity Matrix shown in Figure 3. This is a sum of authority score of all the venues this author has published papers at and all the authors this author has co-authored with on any paper.

3.2.3 Update ranking until converges

Based on the methodology introduced in the previous section to calculate authority scores of authors and venues, we use an iterative approach to update authors’ authority scores and venues’ authority scores during each iteration. We conclude our iterations when one of the two conditions specified below is met:

1. Maximum number of iterations \( \tau \) reached

   The user is allowed to specify a cap of the number of iterations he would like to run. The use is recommended to select a threshold that takes both time efficiency and quality of results into account.

2. Ranking of the top \( k \) authors remain unchanged over \( \sigma \) iterations

   Since we are only concerned about the top ranked authors, which the user can specify the top \( k \) author he wants to output. The authority scores of authors and venues would change over iterations, but as long as top \( k \) authors’ relative rankings stay the same over a threshold of \( \sigma \) iterations, we can terminate the algorithm.

Algorithm 1 outlines the entire algorithm.

### Algorithm 1: RECAUTH algorithm

**Data:** abstracts, venues, and authors of DBLP papers

**Input:** query keywords, \( \tau \), \( k \), \( \sigma \)

**Output:** top \( k \) ranked authors

1. perform key phrase extraction from abstracts;
2. construct heterogeneous bibliographic network;
3. construct APV, APA matrices;
4. construct APV, APA similarity matrices based on cosine similarity between input keywords and key phrases from abstracts;
5. calculate authority scores for authors and venues;
6. initialize rankings;
7. **repeat**
   8. calculate authority scores for authors and venues;
   9. rank authors based on the new authority scores;
8. **until** maximum iteration \( \tau \) is reached OR ranking for top \( k \) authors remain unchanged for the last \( \sigma \) iterations;

4. Experiments

The performance of RECAUTH is evaluated. The algorithm was tested on a real-world dataset and the quality of the author recommendation was assessed.

4.1. Data Preprocessing

The experiment performed involved the DBLP dataset\(^1\). In addition to containing information such as authors, venues and titles for each paper, much of the data contained in this particular set of data also carries each paper’s abstract information. Paper data that do not carry abstracts were filtered out.

4.1.1 Key phrase Extraction

Instead of just using the words contained in the title of each paper, which is not always representative of the content of the paper, we used SegPhrase+ to extract keyphrases from the abstract. Combining the extracted key phrases with the existing key words produced vectors that are a fairly complete representation of each paper.

4.1.2 Heterogeneous Bibliographic Network

Using the constructed keyword and key phrase vector, the title, authors, and venues associated with each paper, a heterogeneous bibliographic network was con-

\(^1\)http://dblp.uni-trier.de/db/
structured for the entire dataset. The network facilitates the association of authors with venues and co-authors, as mentioned earlier. The schema of the network is shown in Figure 4.

![Figure 4: Heterogeneous network constructed from DBLP data](image)

### 4.2. Experimental Parameters

ReCAUTH contains many parameters that can be tuned to influence the performance. During our experiments, trial and error was performed to determine the values of these parameters that would yield optimal recommendation results. We started by setting $\alpha$ value in Equation 4 to 1, which represents that the authority score of each author completely relies on the venues they publish papers at, decremented $\alpha$ by 0.1 during each step, and continued this process until $\alpha$ reaches 0. $\alpha$ score equals to 0 represents that the authority score of an author is constructed entirely based on the people he co-authors with. It was found that setting $\alpha$ value in Equation 4 to 0.6 resulted in reasonable authority ranking scores. The ranking produced by $\alpha$ value of 0.6 is also very consistent with the domain knowledge of highly reputable authors in each field. This means that the venues each author publishes paper at hold slightly more weight than each author’s coauthors. This is reasonable because certain conferences, such as KDD and ACM, are highly prestigious and only accept truly high quality publications.

The number of iterations for propagating ranking between authors and venues play a large role in the quality of the recommended authors. It was also found that setting the maximum number of iterations $\tau$ produced a nice balance between the running time and the quality of the output.

The parameter $k$ simply controls the number of authors recommended, which does not actually affect the overall quality of the algorithm. Setting $k$ to too high a value, however, often defeats the purpose of recommending the most reputable authors. The parameter $\sigma$ influences the running time of the algorithm quite significantly. Yet a high enough value is required to ensure the algorithm does not terminate too early. It was found that setting $k$ and $\sigma$ to 10 and 20, respectively, resulted in excellent recommendations.

A summary of all parameter values used during the experiments is shown in Table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>0.6</td>
</tr>
<tr>
<td>$\tau$</td>
<td>500</td>
</tr>
<tr>
<td>$k$</td>
<td>10</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 1: Summary of all parameters used during the experiments

### 4.3. Framework Performance

Various key phrases were used as input to test the quality of the author recommendation made by the ReCAUTH framework. The results were compared against the results returned by entering the identical queries into Google Scholar.

Table 2 shows the result of querying the words “data mining.” Our framework returned some of the most well-known authors in the data mining community, such as Jiawei Han, Philip S. Yu, and Christos Faloutsos. Compared to the search result returned from Google Scholar, which is shown in Figure 5, ReCAUTH’s results are considered superior. Although Google Scholar returned some papers whose authors are relevant and reputable, the number is far fewer.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Author Name</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Jiawei Han</td>
<td>2.0175E+28</td>
</tr>
<tr>
<td>2</td>
<td>Philip S. Yu</td>
<td>1.6769E+28</td>
</tr>
<tr>
<td>3</td>
<td>Jian Pei</td>
<td>1.2933E+28</td>
</tr>
<tr>
<td>4</td>
<td>Rakesh Agrawal</td>
<td>1.1215E+28</td>
</tr>
<tr>
<td>5</td>
<td>Christos Faloutsos</td>
<td>1.0325E+28</td>
</tr>
<tr>
<td>6</td>
<td>Wei Wang</td>
<td>8.1145E+27</td>
</tr>
<tr>
<td>7</td>
<td>Michael Stonebraker</td>
<td>6.7353E+27</td>
</tr>
<tr>
<td>8</td>
<td>Charu C. Aggarwal</td>
<td>6.6373E+27</td>
</tr>
<tr>
<td>9</td>
<td>Haixun Wang</td>
<td>6.3833E+27</td>
</tr>
<tr>
<td>10</td>
<td>Divesh Srivastava</td>
<td>5.8780E+27</td>
</tr>
</tbody>
</table>

Table 2: Author recommendation made by ReCAUTH for the query “data mining”

The results for additional queries, “database system” and “reinforcement learning”, for ReCAUTH are shown in Table 3. Similar to the previous query, our framework produced many of the most reputable authors in their respective fields of research. The results
for Google Scholar, shown in Figure 5, only returned papers that are a very small subset of the most well-known authors.

5. Future Work

Interesting future work include incorporating Latent Keyphrase Inference (LAKI) to perform paper representation, ranking authors based on smaller areas, and allowing user to input paragraphs of text before further refining to keywords.

5.1. Incorporate LAKI to represent documents

LAKI is an algorithm that extracts a vector of keyphrases, which can be used to recommend an input document [9]. Since in our current work, we only used abstract of papers to generate key phrases that represent each paper, there still resides the possibility of key phrases being omitted from our research. Also, currently, we treated every key phrase generated from SegPhrase+ with equal weight. Leveraging the weighting factor of LAKI could better help us match similar papers to the input keywords. Plus, during the offline stage of LAKI, it leverages the entire text corpora to generate domain keyphrase extraction and keyphrase silhouetting. Thus with existing research, we think LAKI could best represent each document with a vector of key phrases.

5.2. Perform clustering while ranking

RankClus is an Expectation Maximization (EM) styled algorithm that integrates clustering while performing ranking [13]. In order to better recommend authors specific to one area, clusters are formed while ranking is performed. RankClus excels in locating authors who are more specific to one cluster other than on a global scale. There exists the case when an author doesn’t have as many publications, but are very reputable in one area instead of a global scale outlined in this work.

5.3. Extend input keywords to text paragraphs

The current scope of our work limit users to input a set of keywords that best describe the research they are starting to perform. However, there are times a new researcher doesn’t have a refined idea of his research. Thus allowing the possibility for a user to input text paragraphs instead of very concrete keywords would be beneficial in the future. We can leverage SegPhrase+ to mine key phrases from input text paragraphs that represent comprehensively of a user’s intention and research goal.
5.4. Explore automation techniques as dataset is updated

Since the dataset we used in our research was static, and the DBLP dataset is updated regularly with new authors and papers. An automation technique can be leveraged to perform synchronization whenever the dataset is updated. Thus providing users with the most up-to-date result will be guaranteed.

With these future work performed, this algorithm could be better leveraged to help new researchers acquire authors that are highly reputable in their respective field, that could help them with their upcoming research. The output will also be up-to-date as the user’s research goes.

6. Conclusion

In this paper, we proposed a method to first find similarity between authors and keywords and then perform ranking on the authors. A framework, called RecAuth, was developed that offers custom author recommendation based on a set of input keywords. By propagating ranking over venues and authors, the algorithm determines the most highly ranked authors among the ones who are similar to the input query. Experimental results show that RecAUTH is able to recommend authors that are highly consistent with the results returned by Google Scholar. For certain queries, RecAUTH outperforms Google Scholar.

Additional work can be done to further improve the performance of our framework. Terms can be extracted from DBLP papers directly instead of just abstracts so each paper can be more accurately represented by terms. Input queries can be extended to text, such as whole abstracts, instead of just keywords.

References


