The Pennsylvania State University

The Graduate School

Department of Information Sciences and Technology

IMPROVING DIGITAL LIBRARY RANKING WITH EXPERT INFORMATION

A Thesis in

Information Sciences and Technology

by

Yuan-Hsin Chen

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

August 2015

The thesis of Yuan-Hsin Chen was reviewed and approved* by the following:

C. Lee Giles
Thesis Advisor
Interim Associate Dean of Research
Professor of Information Sciences and Technology
Graduate School Professor of Computer Science and Engineering, College of Engineering
Director, Intelligent Systems Research Laboratory

Xiaolong Luke Zhang
Associate Professor of Information Sciences and Technology
Human Computer Interaction, IST Faculty Council

David Reitter
Assistant Professor

Carleen Maitland
Interim Associate Dean of Undergraduate and Graduate Studies

*Signatures are on file in the Graduate School

ABSTRACT

The purpose of this research is to investigate whether expert information can improve the ranking function of academic search engines. We chose CiteSeerX as the testbed which is a well-known academic paper search engine where the ranking function takes the number of citations and the similarity between the query and the paper. Therefore, more cited papers easily get higher positions in the ranking list. Thus, adding more features to the ranking function may reduce effects on one or a few factors. Intuitively, if an author of a paper is an expert in the area, the paper should be more credible and also searchers should be more interested in. This research found that a document of which an author who is an expert has higher probability to be clicked than a document of which the author who is not an expert. Furthermore, we included expert information as another feature of CiteSeerX's ranking function. A supervised ranking approach that considers expert information was used. The evaluation shows that the two learned ranking functions perform better than the existing CiteSeerX's one.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

First, I want to thank my advisor, Dr. Lee Giles, who provides a lot of resources and freedom while I was pursuing my master's degree. Besides, I also want to thank all my committee members, Dr. Luke Zhang and Dr. David Reitter, who gave very helpful comments to improve my thesis.

While doing the research, Kyle Williams and Dr. Peifeng Yin spent a lot of time discussing with me and gave really helpful suggestions of research although they were very busy. I learned a lot from them. I really appreciate their efforts and patience. Also, Dr. Jian Wu and Madian Khabsa helped a lot while I was designing and running the experiments.

Finally, I want to thank my parents and my family for giving me unconditional love and endless tolerance.

Chapter 1

Introduction and Related Work

Introduction

CiteSeerX is a well-known digital library search engine [1] where its ranking function takes the number of citations and the similarity between the query and the paper into consideration. However, for some queries, the top most clicked documents which are possibly more relevant to the queries [2] are not shown on the top positions in the search results.

Table 1-1 A ranking list of query "cloud computing" ordered by CiteSeerX and the number of clicks.

rank	Ordered by CiteSeerX	Ordered by number of click	Number of click
1	10.1.1.149.7163	10.1.1.158.2549	1018
2	10.1.1.462.4311	10.1.1.148.1726	760
3	10.1.1.158.2549	10.1.1.149.2162	400
4	10.1.1.413.4719	10.1.1.178.1323	291
5	10.1.1.453.6574	10.1.1.173.686	231
6	10.1.1.352.6525	10.1.1.149.7163	125

Table 1-1 shows that a ranking list of the query 'cloud computing' ordered by CiteSeerX and by the number of clicks, which is obtained from the CiteSeerX's search

logs from 2009 to 2013, are different in order. The document gets the most number of clicks is 10.1.1.158.2549 which is at the rank 3 in the CiteSeerX's ranking list. Also, the top ranked document 10.1.1.149.7163 in the CiteSeerX's ranking list only has the sixth most number of clicks among clicked documents associated with the query. Based on the assumption that a document gets more clicks should be more relevant, the CiteSeerX's ranking has spaces to be improved.

We analyzed search results from CiteSeerX and investigated the correlation between the features extracted from a paper, such as title, author, venue, and so on, and the number of clicks. We narrowed down the scope of features and focus on author information since intuitively if an author of a paper is an expert in the area, the paper should be considered as more credential which might mean that it would get more clicks. Therefore, we aim to answer the following two research questions in the research:

Research Question 1:

How does expert information affect academic search engines?

Research Question 2:

Can expert information be used to improve the ranking function?

Related Work

The purpose of our work is to improve the ranking function of an academic search engine which combines knowledge of multiple domains in information retrieval including information integration, digital library, learning to rank and expert finding. In our work, in order to add more features into the ranking function, results from different resources are selected and integrated in a reasonable way. Since the research target is a digital library search engine, understanding other digital libraries and their ranking functions, which state the general concepts of what ~~are~~ features are used in terms of ranking functions in digital libraries, provides clues for improvement. Furthermore, in order to include expert information, one of expert finding algorithms among expert finding literatures is needed. From the search logs, we extracted user's search activities and use them as training dataset to learn a ranking function with higher retrieval efficiency.

Information integration

Information integration is to merge information from heterogeneous resources and present the integrated result in different perspectives. [3] is one of examples of information integration being used in digital libraries. The proposed academic search engine searches across journal publisher collections and merges returned result into a single search result page. Although the search results are merged from multiple resources, the contents of the search results are similar, that is, all are academic papers.

Our research aims to collect diverse information from different resources and integrate them to improve the ranking function.

Furthermore, another way to integrate information is to intersect results from search resources and return the intersection which is upon an assumption that each search resource has the same representation of collections. [4] uses the intersection as one of the features of learning to merge search results. However, in our case, intersection between four selected verticals is really small; hence, the methods are not applicable to the problem.

The search result blending methods proposed by [5] are built on an assumption that the relative order of documents coming from the same sources is not allowed to change since they believe that each specialized search engine generates the best ranking about their own documents. Thus, we assume that each search result from different verticals can help to refine the ranking function since all of them are in different nature and correlated in some ways.

Ranking functions of digital libraries

Academic search engines provide open and easily accessible information retrieval platforms for academic publications. The data a popular academic search engine collects is broadly representative of the impact of journals, conferences, and publications [6]. The number of citation is one of the important indicators to be a successful publication and in [7], the research discovered that there are positive

correlations between citation counts and the position of documents mostly. However, for some quires, no correlation existed which means some more factors affect ranking.

The age of the publication could be another feature in ranking functions. For Google Scholar, although [8] shows that no significant correlation between an paper's age and the ranking, older papers are ranked in top positions more often than recent ones. The possible reason is that Google scholar highly depends on citation counts for ranking. There are some more options for ranking academic documents such as h-index and g-index. [9] used a combination of g-index and h-index, known as hg-index, to rank marketing journals which have highly correlation in terms of ranking with Journal Impact Factor. Furthermore, [10] aims to provide guidelines on optimizing scholarly literature for academic search engines as well as discuss the concept of academic search engine optimization.

Learning to rank

Learning to rank approaches are used in rankers including abounding and diverse sets of features [4, 5, 11] and use a judged training set of query-document pair and apply machine learning techniques to learn complicated combination of features. Although the approaches are not readily applicable to use here since different search verticals have different feature spaces which must be merged somehow. In [5, 12], for every source, a copy of features for each source allows the learning to rank model to learn a relationship between relevance and features for each vertical. [13] proposed

SVM-based method to learn retrieval functions and illustrated the features and the results of weighting the features which would be a good baseline for reference of learning to rank.

[12, 14] focus on aggregated searches and learn to merge results in the search engine's final result page with block-based ranking which means search results of the same searching resources must be grouped together while presenting. The papers address the difficulty of the representation of different features across search resources as well as propose approaches to allow learning algorithm to learn across features.

There are three categories of learning to rank, pointwise, pairwise, and listwise according to [11]. [2, 13] proposed pairwise learning to rank algorithms and both of them used clickthrough data to optimize with different strategies where [2] involved human judgement to develop the learning. Interestingly, clickthrough data perform really well for learning, however, involvement of human judgement works less reliable and informative since large size of clickthrough data represent decisions of large amount of users.

Expert Finding

Expert finding is a common task and popular research area in digital libraries and in different perspectives as well. Basically, expert finding algorithm takes a query as an input and returns a list of experts of the area. A Citation-Author-Topic (CAT) model is proposed by [15] which models the linkage among cited authors, words, and

paper authors together. [16] proposes large-scale expert finding algorithms in a specific field with the data supplementation of DBLP biblography and Google Scholar. [17, 18] use CiteSeerX document collection as the corpus to build the models where [17] proposes a graph-based algorithm which accommodates multiple features extracted from documents and links to other documents and [18] generates keyphrases which are used to gather authors' expertise.

Chapter 2

How does expert information affect academic search engines?

Introduction

When a user searches an academic search engine, besides clicking on the top search results, the user looks for certain evidence to identify which document to click. Venues, published date, the number of citations, the relevance of the title or abstract, or the author's identification, such as name, affiliation and expertise, are part of consideration. A user who is familiar with an area might take venues and author's identification into consideration. Or a user who is a novice in an area might regard the number of citation or the relevance of the title or abstract.

An author's expertises, is an implicit feature for users to identify. Therefore, a research topic, expert finding, is under popular investigation. An Expert finding algorithm searches experts on a particular area such as [17] which proposed a PageRank-like model using features of papers and [18] combined the resources of CiteSeerX and Wikipedia to build a keyphrase based expert recommendation system.

Intuitively, if an author of an academic paper is an expert of an area, users may be more interested and more confident in the paper which means the paper should be placed at a higher rank. Fortunately, the web server of the academic search engine records transaction logs. We can analyze the search logs and answer the question on how expert information affects the academic search engine.

Methodology

The purpose of the experiment is to investigate whether expert information gives positive, negative or neutral effect to an academic search engine. Therefore, we design an experiment to show whether documents with experts as authors have higher probability of being clicked than documents without experts as their authors. In other words, we want to explore the ratio of the probability of documents without any experts, $P_{C_{\bar{e}}(r)}$, divided by the probability of documents with at least one author as an expert, $P_{C_e(r)}$, which is represented as $\frac{P_{C_{\bar{e}}(r)}}{P_{C_e(r)}}$, is. If the ratio of $\frac{P_{C_{\bar{e}}(r)}}{P_{C_e(r)}}$ is less than 1, documents with expert authors have a higher probability of being clicked which suggests that expert information has positive effect on the academic search engine.

Since the search logs of CiteSeerX do not record the documents of search results which were not clicked on, we cannot compute the probabilities of clicked documents with and without experts directly. Table 2-1 shows the representations and explanation of all the notations we are going to use to explain the methodology.

Table 2-1 the representation of the experiment

Notation	meaning
$C_e(r)$	$number\ of\ clicked\ documents\ with\ experts\ at\ rank\ r$
$C_{\bar{e}}(r)$	$number\ of\ clicked\ documents\ without\ an\ expert\ at\ rank\ r$
$D_e(r)$	$number\ of\ documents\ in\ a\ ranking\ list\ with\ experts\ at\ rank\ r$
$D_{\bar{e}}(r)$	$number\ of\ documents\ in\ a\ ranking\ list\ without\ experts\ at\ rank\ r$
$P_{C_e(r)}$	$Probability\ that\ an\ expert-document\ at\ rank\ r\ is\ clicked$

$P_{C_{\bar{e}}(r)}$	$Probability\ that\ a\ non-expert-document\ at\ rank\ r\ is\ clicked$
$P_{D_e(r)}$	$Probability\ that\ an\ expert-document\ at\ rank\ r\ is\ returned$
$P_{D_{\bar{e}}(r)}$	$Probability\ that\ a\ non-expert-document\ at\ rank\ r\ is\ returned$

As we have the number of clicks on documents with and without experts, $C_e(r), and\ C_{\bar{e}}(r)$, respectively, in order to formulate $P_{C_e(r)}$, we have $C_e(r)$ be divided by $D_e(r)$ and multiplied by $D_e(r)$ at the same time in both the numerator and the denominator.

$$\frac{C_e(r)}{C_{\bar{e}}(r)} = \frac{\frac{C_e(r)}{D_e(r)} \times D_e(r)}{\frac{C_{\bar{e}}(r)}{D_{\bar{e}}(r)} \times D_{\bar{e}}(r)}$$

And then divide by $D(r)$ for both numerator and denominator where $D(r) = D_e(r) + D_{\bar{e}}(r)$.

$$\frac{C_e(r)}{C_{\bar{e}}(r)} = \frac{\frac{C_e(r)}{D_e(r)} \times D_e(r)}{\frac{C_{\bar{e}}(r)}{D_{\bar{e}}(r)} \times D_{\bar{e}}(r)} = \frac{\frac{C_e(r)}{D_e(r)} \times \frac{D_e(r)}{D(r)}}{\frac{C_{\bar{e}}(r)}{D_{\bar{e}}(r)} \times \frac{D_{\bar{e}}(r)}{D(r)}}$$

Thus, we can derive $P_{C_e(r)}, P_{C_{\bar{e}}(r)}, P_{D_e(r)}, and\ P_{D_{\bar{e}}(r)}$ from

$$P_{C_e(r)} = \frac{C_e(r)}{D_e(r)}$$

$$P_{C_{\bar{e}}(r)} = \frac{C_{\bar{e}}(r)}{D_{\bar{e}}(r)}$$

$$P_{D_e(r)} = \frac{D_e(r)}{D(r)}$$

$$P_{D_{\bar{e}}(r)} = \frac{D_{\bar{e}}(r)}{D(r)}$$

The ratio of $\frac{C_e(r)}{C_{\bar{e}}(r)}$ is

$$\frac{C_e(r)}{C_{\bar{e}}(r)} = \frac{P_{C_e(r)} \times P_{D_e(r)}}{P_{C_{\bar{e}}(r)} \times P_{D_{\bar{e}}(r)}}$$

We move $P_{C_e(r)}$ and $P_{C_{\bar{e}}(r)}$ to the left-hand side of the equation, and obtain that the ratio of the probability of clicked documents without and with experts is the multiplication of the ratio of the number of clicks of documents without and with experts and the ratio of the probability of returned documents with and without experts.

$$\frac{P_{C_{\bar{e}}(r)}}{P_{C_e(r)}} = \frac{C_{\bar{e}}(r)}{C_e(r)} \times \frac{P_{D_e(r)}}{P_{D_{\bar{e}}(r)}}$$

We can easily obtain $C_e(r)$ and $C_{\bar{e}}(r)$ from the search logs, however, we still need $P_{D_e(r)}$ and $P_{D_{\bar{e}}(r)}$ which have to be computed by sending queries to CiteSeerX. We could submit the same set of queries to CiteSeerX, label each returned document as either with or without experts, and obtain the probabilities. However, the probabilities would be biased since the search engine keeps updating and the ranking list associated with the query might vary based on when the search logs were recorded. Therefore, we use sampling estimation to estimate the probabilities of returned documents with or without experts at rank r instead.

To estimate the probability of documents with or without experts in a ranking list, a series of queries are sent to CiteSeerX and a ranking list associated with the query

is retrieved. The probabilities of documents with and without experts are computed. Say, for instance, there are three queries, Q1, Q2, and Q3, and top three documents are retrieved in the order shown in Table 2-2. Each retrieved document is labeled either E meaning that at least one author is an expert, or NE meaning that no author is an expert. Therefore, for each rank, $P_{D_e(r)}$ is the number of documents labeled as E divided by the number of queries. In the example, $P_{D_e(1)}$ is $\frac{2}{3}$.

Table 2-2 an example of estimation of $P_{D_e(r)}$

rank	Q1	Q2	Q3	$P_{D_e(r)}$
1	E	E	NE	$\frac{2}{3}$
2	NE	NE	NE	0
3	E	NE	NE	$\frac{1}{3}$

Data

The dataset is the search logs of CiteSeerX from July 2014 and December 2014 which are the logs with ranking information to avoid ranking bias. As shown in Figure 2-1, the search logs are obtained from servers and the queries, document ID, rank and number of clicks are extracted. The number of clicks is accumulated for each pair of document ID and rank. In addition, expert labeling identifies experts among authors of each document. Finally, the probability of clicked documents with experts as well as the probability of clicked documents without experts are computed.

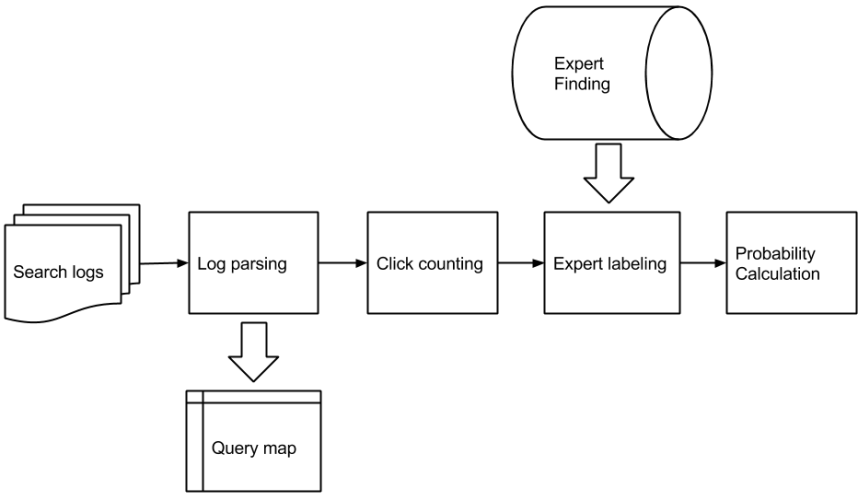

Figure 2-1 the flowchart of probability calculation

Experimental Results

In order to compute the ratio of the probability of clicked documents without expert to the probability of clicked documents with experts, we need to estimate the probabilities of returned documents with and without experts by sending queries to CiteSeerX. We selected the queries randomly from the cleaned search logs and the queries are identical. Figure 2-2 shows the probabilities of documents with experts for each rank by sending the queries to CiteSeerX to estimate $P_{D_e(r)}$. The x-axis is the number of queries sending to CiteSeerX divided by 10 and the y-axis is the probability of documents with experts in ranking lists. We observe that while more queries are sent

to CiteSeerX, at a certain amount of sent queries, the probabilities of documents with experts converge which is the sampling estimation of $P_{D_e(r)}$ and $P_{D_{\bar{e}}(r)}$.

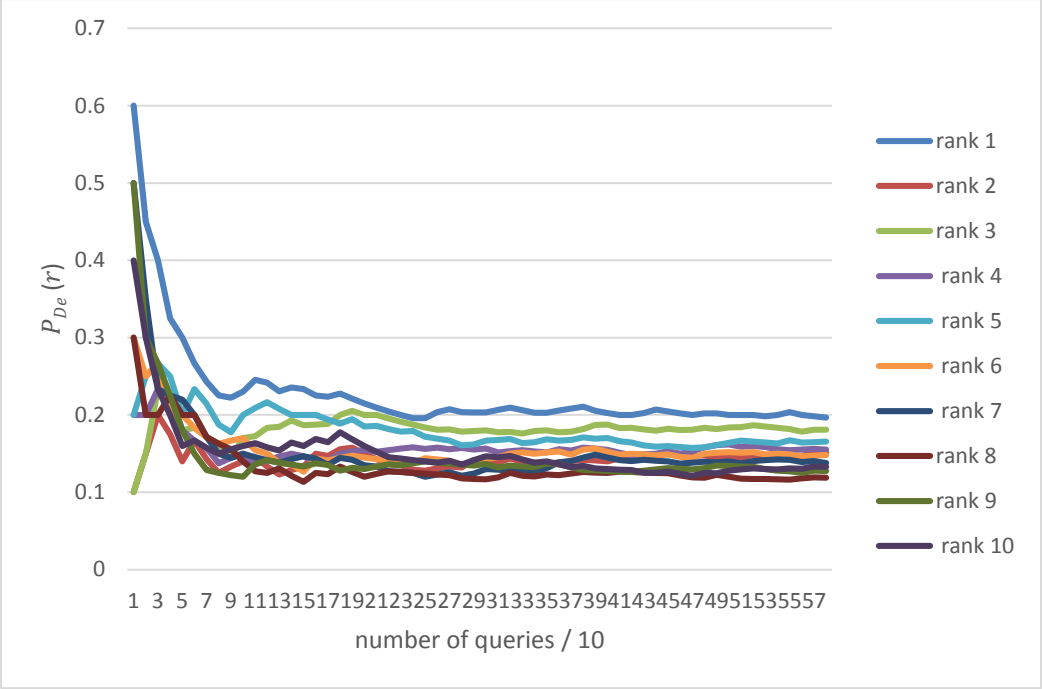

Figure 2-2 The probabilities of documents with experts at rank 1 to rank 10 by sending queries to CiteSeerX to estimate $P_{D_e(r)}$

Figure 2-3 shows that the probabilities of clicked documents with and without experts. The ratio of $\frac{P_{C_{\bar{e}}(r)}}{P_{C_e(r)}}$ at each rank from 1 to 10 is less than 1 which means the probability of documents with experts being clicked, $P_{C_e(r)}$ is higher than the probability of documents without experts being clicked, $P_{C_{\bar{e}}(r)}$. Therefore, expert information has positive effect on the academic search engine.

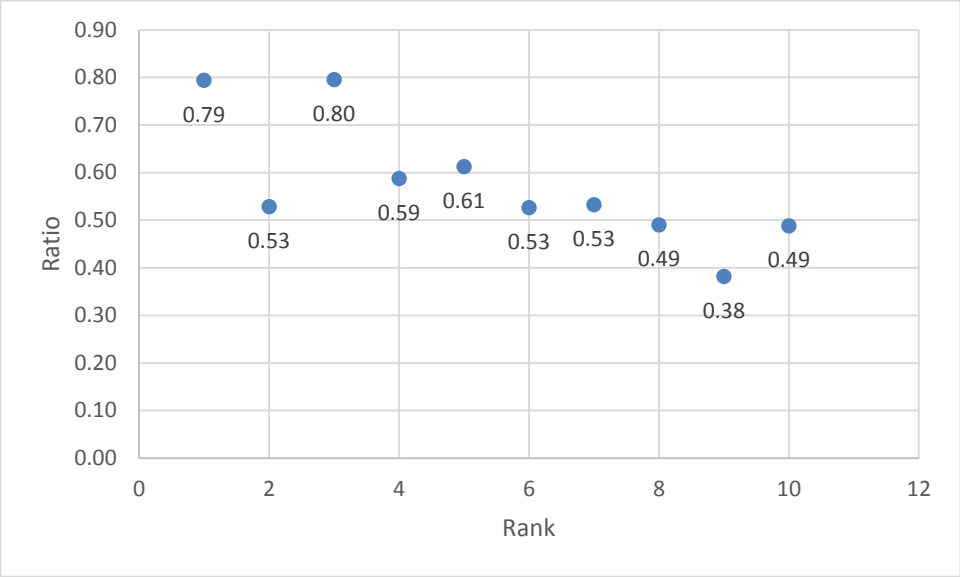

Figure 2-3 the ratio of $P_{C_{\bar{e}}(r)}$ to $P_{C_e(r)}$ at rank 1 to rank 10

Conclusion

In this chapter, we aim to investigate how expert information affects the academic search engine. We formulated the research question to investigate the ratio of the probability of clicked documents without experts to the probability of clicked documents with experts, $\frac{P_{C_{\bar{e}}(r)}}{P_{C_e(r)}}$. Although the probabilities of clicked documents with and without experts cannot be directly obtained by the search logs, we estimated the probabilities of returned documents with and without experts and extracted the number of clicks from the search logs. As a result, all the ratios at rank 1 to rank 10 are less than 1 which means expert information has positive effect on boosting the probability of

documents being clicked. Therefore, we can conclude that expert information affects the academic search positively.

Chapter 3

Can expert information be used to improve the ranking function?

Introduction

We have shown that expert information has positive effects on the academic search engine where documents with experts have higher probability of being clicked than those without experts. In this chapter, we are going to propose a new ranking function for CiteSeerX which integrates expert information as a feature.

CiteSeerX's ranking function is a combination of the number of citations, the similarity between a query and the title, abstract, and the text of a document. The score $score(q, d)$, where q is a given query, for a document d is given by [19]:

$$score(q, d) =$$

$$coord(q, d) \times queryNorm(q) \times \sum_{t\ in\ q} (tf(t\ in\ d) \times idf(t)^2 \times Boost() \times norm(t, d))$$

where $coord(q, d)$ is a score factor reflects the similarity and $queryNorm(q)$ is a normalizing factor to make score comparable between queries. $tf(t\ in\ d) \times idf(t)^2$ is term's frequency and inverse document frequency which gives higher score for more

occurrences of a given term in a document and rarer terms among documents. $Boost()$ and $norm(t, d)$ contain boost functions including the boost function of the number of citations in search time and indexing time. Furthermore, we have a weight of 4 for title and 2 for abstract to boost the score of documents.

Methodology

In this study, a learning to rank approach is used to learn new ranking functions. Figure 3-1 is the flowchart of learning to rank which has two stages: training and prediction. In the training stage, a set of training data is prepared which contains queries, q1 to qn, where for each query, there are a list of CiteSeerX's score, $CiteseerScore(q, d)$ where q is the query and d is the document among the associated documents, author's score, $AuthorScore(q, d)$, and labels, $LearnedScore(q, d)$. Note that the author's score is generated by the expert finding module. The structured training data is the input for the learner and the learner produces a model afterwards, which is the learned ranking function. The learned ranking function would be a part of ranker which will be used in the prediction stage. In the prediction stage, the ranker aims to order documents on the basis of the learned ranking function. The output of the ranker is a list of ranked documents ordered by $LearnedScore(q, d)$:

$$LearnedScore(q, d) = CiteseerScore(q, d) + \beta \times AuthorScore(q, d)$$

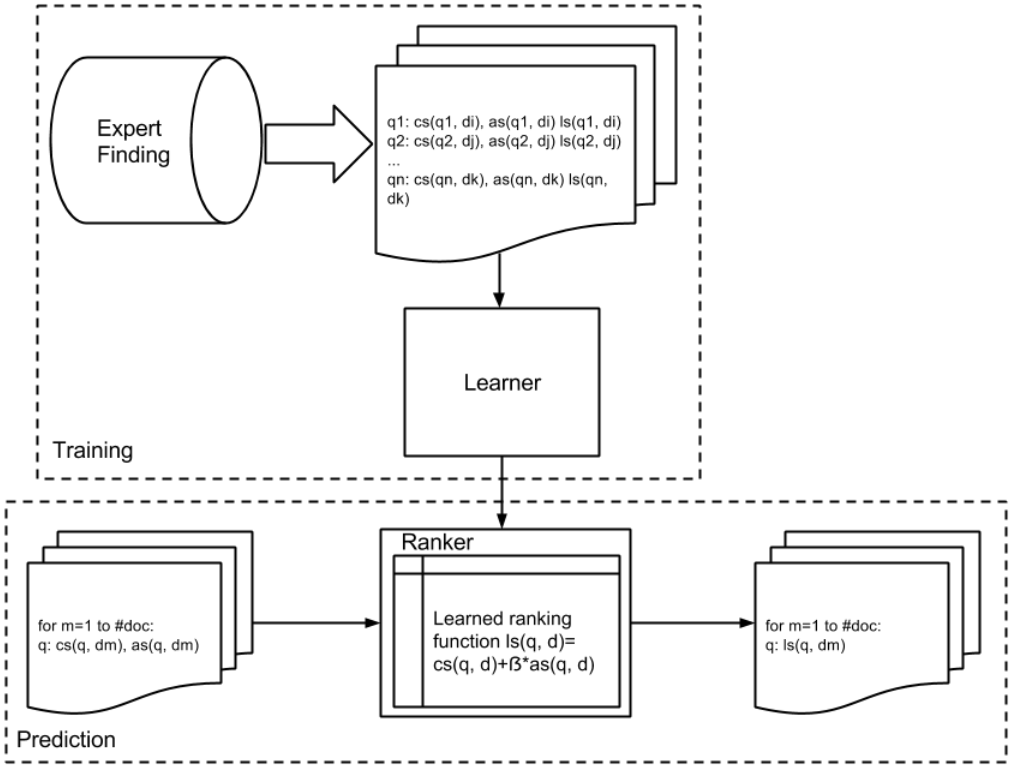

Figure 3-1 The flowchart of learning to rank

From CiteSeerX' search logs, the number of clicks associated with a query and a document can be extracted. The assumption is made that more clicked documents are considered as more relevant among the returned documents of a query. Therefore, two pointwise learning strategies are proposed:

Learning Strategy 1. (Clicks)

$$click(q,d) = CiteseerScore(q,d) + \beta \times AuthorScore(q,d)$$

where the label, $click(q,d)$, is the number of clicks for each pair of training data, $(CiteseerScore(q,d), AuthorScore(q,d))$. The reason to use the number of click is

that the size of training dataset is small and an absolute-score based learning strategy results in a large amount of training data that can be used.

Learning Strategy 2. (Levels)

$$level(q,d) = CiteseerScore(q,d) + \beta \times AuthorScore(q,d)$$

where the label, $level(q,d)$, is the pre-calculated level of relevance for each query. Each document of a query is assigned a level of relevance based on the number of clicks compared to that of other documents. The level of relevance has predefined value and limited number which is supposed to has better learning efficiency compared to the strategy 1.

We convert the number of clicks to the level of relevance, $LevelofRelevance(q,d,l)$, for each document per query in advance. To decide the level of relevance, we basically normalize the number of clicks to a range of level per query which captures the relative relevance among documents for each query. The level of relevance, $LevelofRelevance(q,d,l)$, is defined as

$$LevelofRelevance(q,d,maxL) = \frac{clicks(q,d) \times maxL}{\max\limits_{d_i \in doc\ in\ q}(click(q,d_i))}$$

where given a query, q, the level of relevance, $LevelofRelevance(q,d,maxL)$, for a document, d, with a predefined arbitrary maximum level, $maxL$, is the product of the number of clicks and the maximum level divided by the maximum of the number of

clicks within the query. And then we assign $LevelofRelevance(q, d, maxL)$ to each label $level(q, d)$.

Table 3-1 is an example of labeling documents with level of relevance which is acquired from search logs containing documents and the number of clicks for a query. The maximum level l is 3 meaning that there are four levels of relevance from 0 to 3 which are the levels between 'not relevant' to 'highly relevant'. With the level l , 3, d3 gets the most clicks which is assign the most relevant level 3 in the case, whereas, d2 gets 20 clicks but is still relatively irrelevant to the query and therefore it is assigned the least relevant level, 0.

Table 3-1 An example of labeling documents with level of relevance 3 for a query

Document d	# clicks $clicks(q, d)$	Level of relevance $level(q, d, l)$
d1	247	1
d2	20	0
d3	534	3
d4	1	0
d5	430	2

Data

We use two datasets from CiteSeerX which are comprised of 434 and 2943 queries from the periods of April 2014 to December 2014 and 2009 to 2013, respectively. In order to produce inversed score of CiteSeerX, $CiteseerScore(q, d)$,

rank information is necessary to be recorded in search logs. However, only the search logs after April 2014 contains rank information which are used as the training data. Besides, since the size of the search logs consisting of rank information is too small for testing, the larger dataset from 2009 to 2013 is used as the ground truth which is the relevance judgements for evaluation.

Table 3-2 Basic statistics of datasets

	# Queries	# Documents	# Clicks
Training data (clicks)	434	935	3256
Training data (level)	42	253	1283
Test data	2943	70409	165754

Table 3-2 shows the basic statistics of the datasets. For the training dataset, we have 434 queries and 42 queries for training strategy 1 and 2, respectively. To clean training data, we exclude the data with only 1 click which is too noisy. Furthermore, for the training data for level, in order to capture the relative relevance among documents more accurately, only the queries with more than 9 documents are involved. This is why the size of training data for level is smaller compared to that for clicks.

Evaluation Metrics

We evaluate the proposed ranking functions over a series of well-known information retrieval metrics, i.e., Precision and Normalized Discounted Cumulative Gain (NDCG).

Precision at k

Precision at k reports the fraction of retrieved documents that are relevant and ranked in the top k results. In our setting, if a document has been clicked which means it appears in the search logs given the query, it is considered relevant.

NDCG at k

For a given query q, NDCG at position k is defined as [20]:

$$NDCG_k = \frac{DCG_k}{IDCG_k}$$

where $DCG_k = rel_1 + \sum_{i=2}^{k} \frac{rel_i}{log_2(i)}$, rel_i is the relevance score. We have 3 levels of relevance score which interpret as 'not relevant', 'mildly relevant', and 'highly relevant' according to the number of clicks a document receives compared to other documents in a query. $IDCG_k$ is also computed from the DCG_k formula with monotonically decreasing sort of the relevance judgments rel_i where I stands for ideal.

Experimental Results

We proposed the two learned ranking functions: learned-clicks and learned-level and learned the weights on $AuthorScore(q, d)$ of 0.45 and 0.17 by Ridge Regression, respectively. We compare the two learned ranking functions with the CiteSeerX's ranking function 'citeseerx' by evaluation metrics, NDCG, precision and recall.

The test data is split into ten folds randomly to computed NDCG. Figure 3-2 shows mean and standard deviation of NDCG at k where k is 5, 10, 20, 30, and 50 for CiteSeerX, learned-level, and learned-click. All learned models show better performance in terns of NDCG than citeseeerx. Comparing two learned models to citeseerx respectively, learned-click has more than 1% improvement of NDCG at top 5 to top 30 and the learned-level has around a 1% improvement.

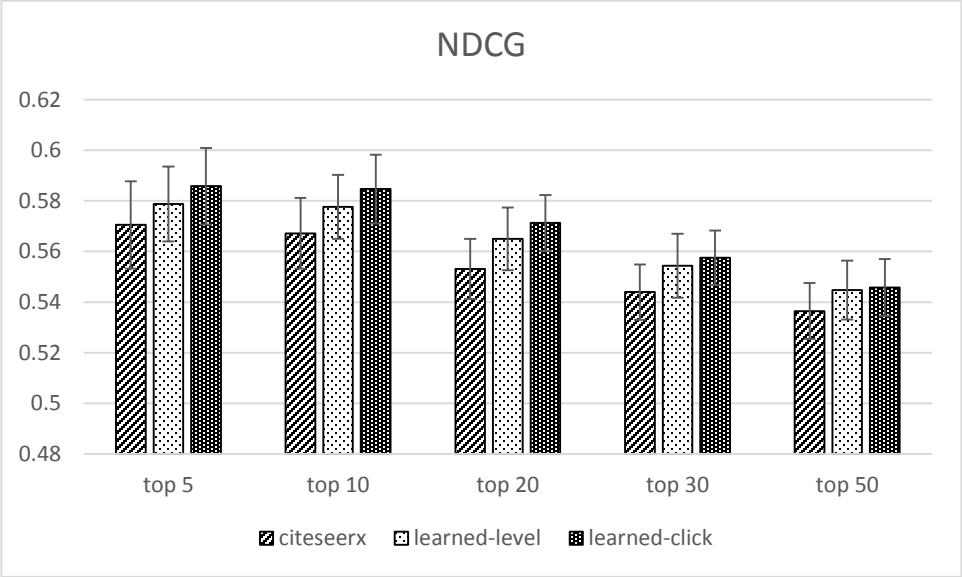

Figure 3-2 NDCG at top k for citeseerx, learned-level, and learned-click

Since the distribution of each evaluation is not normal, Kruskal-Wallis H-test which is a non-parametric method to test whether two or more samples are the same is used to approximate p-value. Table 3-3 shows the improvement associated with the p-value of NDCG at k if the improvement is statistically significant. Only the learned-click strategy has a statistically significant improvement at top 20 at 0.01. At 10%

significant level so does learned-level. Although either learned-level or learned-click has around 1% improvement, the improvement is not statistically significant.

Table 3-3 The improvement of NDCG at k and p-value

NDCG	Top 5	Top 10	Top 20	Top 30	Top 50
learned-level	0.9%	0.8%	1.1% (p=0.05)	1%	0.9%
learned-click	1.3%	1.4%	1.4% (p=0.01)	1.2% (p=0.05)	0.9%

Figure 3-3 shows that two learned ranking functions have higher precision than the standard ranking function in CiteSeerX. At top 5, 10, and 20, learned-click has more than 2 % statistically significant improvement at 0.01 as shown in Table 3-4. Comparing two learned models, learned-click works 1% better than learned-level in precision.

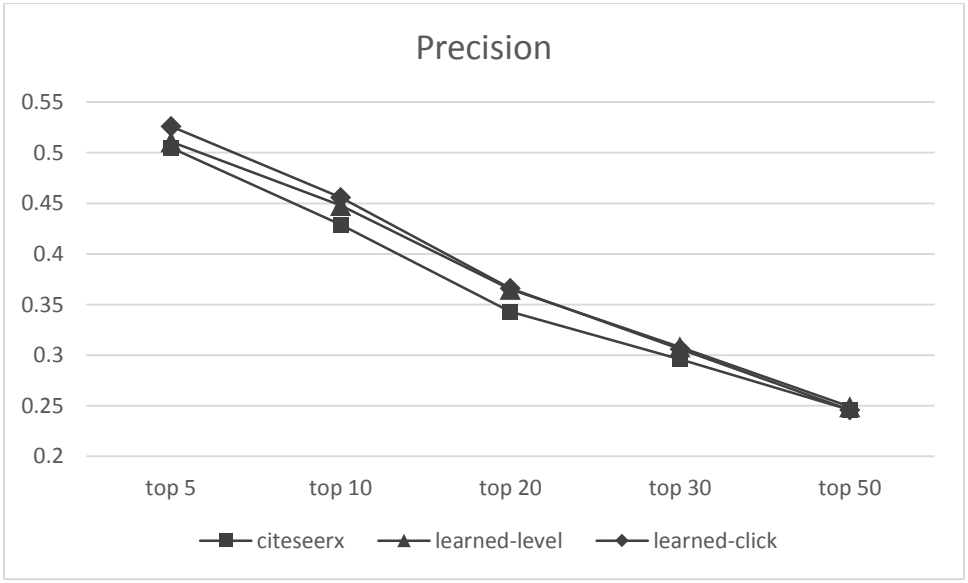

Figure 3-3 Precision at k for citeseerx, learned-level, and learned-click

Table 3-4 The improvement of precision at k and p-value

Precision	Top 5	Top 10	Top 20	Top 30	Top 50
learned-level	0.6%	1.9% (p=0.01)	2.2% (p=0.01)	1.2%	0.3%
learned-click	2.1% (p=0.01)	2.7% (p=0.01)	2.3% (p=0.01)	1%	0%

From Figure 3-4, while recall increases, the precision drops from 50% to only 25% for three ranking functions. The recall from 0 to 0.15, both precision and recall increase since the precision of top 1 to top 4 raises. Still, we observe that learned-level and learned-click clearly work better than citeseerx while recall is less than 40%. The two learned ranking functions perform evenly since learned-click has better precision while recall is less than 0.2, and the other way around.

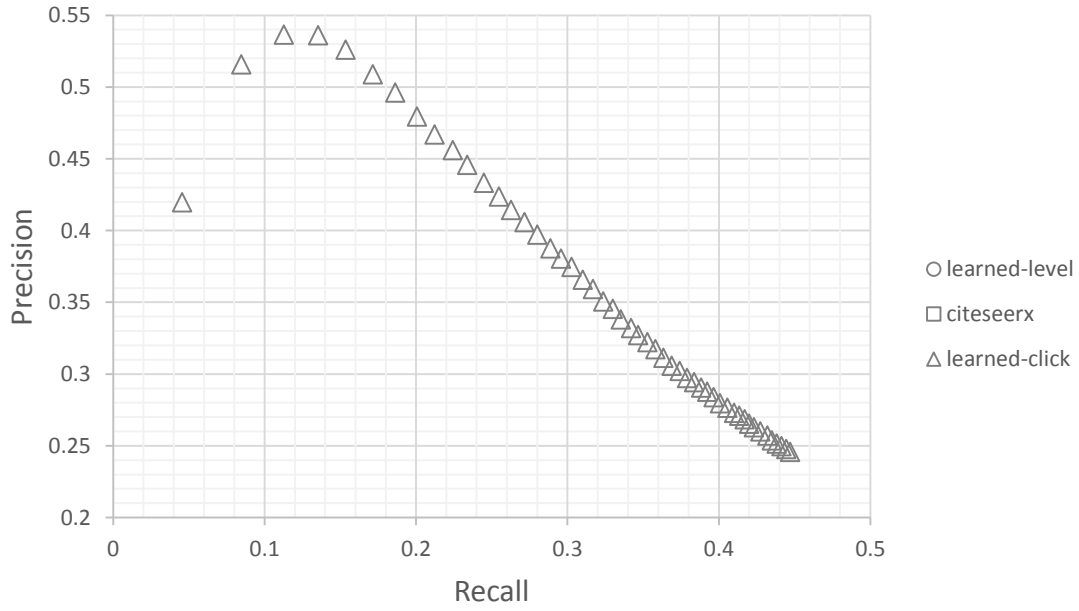

Figure 3-4 Precision/recall plot of citeseerx, learned-level, and learned-click

Conclusion

Based on the finding in Chapter 2 that expert information has a positive effect on the academic search engine, a new supervised ranking function that considers both expert information and CiteSeerX's score was proposed. Two pointwise learning strategies were used where one was based on the number of clicks and the other based on the level of relevance.

We reported the weights of the learned-click and learned-level are 0.45 and 0.17 learned by Ridge Regression which means comparing to the CiteSeerX's score, the Author's score has 45% and 17% influence on the ranking score respectively. Unsurprisingly, both of the weights are positive number since we have already discovered the expert information has an assured effect on academic search engine. Interestingly, there is almost 30% difference between the weights on the Author's score of two learned ranking function even though we use the same training data but different learning strategies.

2943 queries were used for testing with evaluation metrics precision/recall and NDCG. The evaluation showed that the two proposed ranking functions have better retrieval quality than the existing CiteSeerX one. Comparing the two proposed ranking functions, the learned-click one works slightly better than the learned-level one since the size of training data for learning learned-level ranking function might be too few to build the model. The learned-click ranking function has around 1.4% statistically significant improvement at 0.01 at top 20 for NDCG as well as has almost a 3% statistically significant improvement at 0.01 at top 10.

In general, the two proposed ranking functions have better performance the existing ranking function in CiteSeerX according to the evaluation. The finding responds to the suggestion from Chapter 2 that the expert information has positive effect on the academic search engine.

Reference

[1] JianWu, K.W., Hung-Hsuan Chen, Madian Khabsa, Cornelia Carageay, Alexander Ororbia, Douglas Jordan, C. Lee Giles, *CiteSeerX: AI in a Digital Library Search Engine*, in *the Twenty-Eighth AAAI Conference on Artificial Intelligence*. 2014.

[2] Zhicheng Dou, R.S., Xiaojie Yuan, and Ji-Rong Wen, *Are Click-through Data Adequate for Learning Web Search Rankings?*, in *the 17th ACM conference on Information and knowledge management*. 2008.

[3] Schatz, B.M., W. ; Cole, T. ; Bishop, A. ; Harum, S. ; Johnson, E. ; Neumann, L. ; Hsinchun Chen ; Dorbin Ng, *Federated search of scientific literature*, in *IEEE Computer*. 1999. p. 51 - 59.

[4] D. Sheldon, M.S., M. Szummer, and N. Craswell. *LambdaMerge: Merging the Results of Query Reformulations*. in *WSDM '11,*. 2011. ACM.

[5] Lefortier D, S.P., Romanenko F, de Rijke M, *Blending vertical and web results: A case study using video intent*, in *36th European Conference on Information Retrieval (ECIR'14)*. 2014.

[6] Anne-Wil Harzing, R.v.d.W., *Google Scholar: the democratization of citation analysis?* Ethics in Science and Environmental Politics, 2007.

[7] Beel, J., Gipp, B., *Google Scholar's ranking algorithm: The impact of citation counts (An empirical study)*, in *Research Challenges in Information Science*. 2009.

[8] Jöran Beel, B.G., *Google scholar's ranking algorithm: The impact of articles' age (an empirical study)*, in *Information Technology: New Generations*. 2009.

[9] Mourad Touzani, S.M., *Ranking marketing journals using the Google Scholar-based hg-index*. Journal of Informetrics, 2009. **4**(1): p. 107-117.

[10] J Beel, B.G., E Wilde, *Academic Search Engine Optimization (ASEO): Optimizing Scholarly Literature for Google Scholar & Co.* Journal of Scholarly Publishing, 2010. **41**(2): p. 176-190.

[11] Liu, T.-Y., *Learning to Rank for Information Retrieval*. 2008: Springer.

[12] Jaime Arguello, F.D., Jamie Callan. *Learning to aggregate vertical results into web search results*. in *the 20th ACM international conference on Information and knowledge management*. 2011. New York, NY, USA.

[13] Joachims, T., *Optimizing search engines using clickthrough data* in *KDD*. 2002.

[14] Ashok Kumar Ponnuswami, K.P., Qiang Wu , Ran Gilad-bachrach , Tapas Kanungo. *On composition of a federated web search result page: using online users to provide pairwise preference for heterogeneous verticals*. in *the fourth ACM international conference on Web search and data mining*. 2011.

[15] Yuancheng Tu, N.J., Dan Roth, Julia Hockenmaier, *Citation Author TopicModel in Expert Search*, in *the 23rd International Conference on Computational Linguistics*. 2010.

[16] Hongbo Deng, I.K., Michael R. Lyu, *Formal Models for Expert Finding on DBLP Bibliography Data*, in *Data Mining, 2008. ICDM '08. Eighth IEEE International Conference on*. 2008, IEEE.

[17] Sujatha Das, P.M., C. Lee Giles. *Ranking Authors in Digital Libraries*. in *the 11th annual international ACM/IEEE joint conference on Digital libraries*. 2011.

[18] Hung-Hsuan Chen, P.T., Prasenjit Mitra, C. Lee Giles. *CSSeer: an expert recommendation system based on CiteseerX*. in *the 13th ACM/IEEE-CS joint conference on Digital libraries*. 2013. ACM New York, NY, USA.

[19] *Class TFIDFSimilarity*. Available from: https://lucene.apache.org/core/4_0_0/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html#formula_queryNorm.

[20] *Discounted cumulative gain*. Available from: http://en.wikipedia.org/wiki/Discounted_cumulative_gain#cite_note-stanfordireval-2.