ZeroRank: Ranking Newly Published Scientific Literatures Without Citations

ABSTRACT

Ranking scientific literatures is an important but challenging task. Current ranking algorithms aim to measure the prestige of each paper in a given academic network. Though the dynamic nature of citation network is considered, most of them are not specifically designed to rank nodes lying on the edge of the network, which are newly published papers without citation information, thus incur inaccurate ranking in such task. In this paper, we define *zero citation ranking* problem and propose *ZeroRank* to deal with this issue. *ZeroRank* is an algorithm combining random walk on heterogeneous network and learning to rank framework. We use random walk as a feature extractor to obtain the "author", "venue", "affiliation" features of each paper, and then apply learning to rank method to train a ranking model. To make our algorithm capable of handling huge network efficiently, we design a parallel random walk algorithm and implement it on *Spark*. We conduct experiments on *Microsoft Academic Graph* and the results show that our algorithm achieves at least 17.6% improvement in NDCG score compared with the state-of-the-art literature ranking and citation prediction algorithms. We experiment on 31 subfields of computer science and observe a different finding from previous work that "author" is a dominant feature for a paper to gain future citations compared with "venue" and "affiliation".

CCS Concepts

•Information systems → Data mining;

Keywords

Heterogeneous network, learning to rank, random walk, ranking, scholarly big data

1. INTRODUCTION

Ranking scientific literatures is helpful for researchers to find high quality papers, potentially promising research directions, and also plays an important role in academic reward system.

Traditional methods use the citation count as a metric. Yet they are too "democratic" in treating all citations as equal and ignoring differences in importance of citing papers [28]. With *PageRank* [23] and *HITS* [17], many graph based ranking methods were proposed to model the citation network as website network in order to measure the prestige of each publication. Nevertheless, the dynamic and evolving nature makes citation network different from *WWW*, because newly published papers are only able to cite earlier published ones. As a result, methods that do not consider this nature are likely to give bias to old papers.

Many efforts have been made to address this issue. Walker *et al.* [28] proposed *CiteRank* to leverage the publication time information by modifying *PageRank* with an exponential initial distribution. To utilize more information such as authors, Sayyadi and Getoor [25] presented the model *FutureRank*, involving a random walk on citation network and author-paper network. Wang *et al.* [30] defined a ranking algorithm integrating citations, authors, venues and publication time information. Wang *et al.* [29] designed *MRFRank* employing text-feature modeling innovativeness of a paper.

While the work mentioned above leverages different information, these methods are all designed to rank papers of the whole network, and therefore tend to neglect the ranking result of the latest papers without citation information, since these papers are only a small portion of the dataset and make little contribution to the performance measure function. Meanwhile, they lie on the edge of the citation network and have no indegree, making it hard for *PageRank* based methods to perform ranking.

However, we find that ranking such papers could benefit researchers in the community. Consider the situation: after *CIKM 2016* is held, hundreds of papers will be published in the proceedings. It is a natural question to ask, which one will receive the highest citation after 5-10 years among those papers. Answering this question enables providing researchers with potential hot papers and topics. Admittedly, scientific articles are not born equal [26]. We analyze 2552 papers[1] published in *CKIM* from 1992 to 2011. Figure 1 shows the number of their citations after 5 years, which presents a power distribution. The huge difference between citation numbers gives us a motivation to raise a new question *zero citation ranking*, *i.e.*, ranking newly published papers without citations.

In this work we choose "author", "venue", "affiliation" as

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

[1]The data is from *Microsoft Academic Graph [27]*.

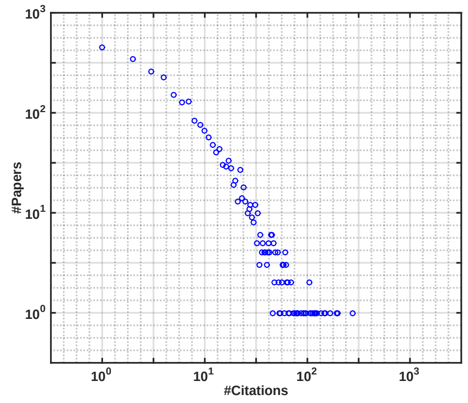


Figure 1: The 5-year citation numbers of 2552 *CIKM* papers published from 1992 to 2011.

features to characterize a paper's potential of being popular in the future, because there is a strong relation between these features and future citations. However, two challenges for such a scheme are: how to define these three features? And how to quantitatively measure their different contributions to highly cited papers?

To acquire the features, we leverage citations, authors, venues, affiliations and time information, and design a reinforced random walk as a feature extractor. Unlike previous random walk algorithms, we aim to obtain the potential of being popular instead of prestige or centrality, thus we apply an average function each iteration. For example, an author's score is based on the average score of all the papers he/she writes. To make our algorithm handle huge network efficiently, we design a parallel random walk algorithm and implement it on *Spark*.

To differentiate the weights, we use learning to rank technology instead of previous methods such as linear regression. Learning to rank enables us to use a training set which consists of many time slices obtained by shifting the current time point, where linear regression can not apply. It can also directly optimize the measure function we select to further refine the results obtained by random walk.

We conduct our experiments on *Microsoft Academic Graph* which contains more than 100 million papers. And the results show that our algorithm achieves at least 17.6% improvement in NDCG score compared with the state-of-the-art ranking and citation prediction methods. Furthermore, we run *ZeroRank* separately on 31 subfields of computer science. Unlike previous result [8] in which "venue" plays a major role in future citation number, our experiment shows that "author" is a dominant feature for future citation.

2. RELATED WORK

2.1 Scientific Literature Ranking

It has been a long time since people tried to give scientific literatures an order. In 1972, Garfield [12] introduced *Impact Factor(IF)*, a citation count based metric to rank journals. Although simple, this algorithm is widely accepted by researchers nowadays. Based on *IF*, *h-index* [15] and *g-index* [9] were proposed and *h-index* is currently used in *Google Scholar*.

With *PageRank* [23] and *HITS* [17], many researchers modeled citation network as *WWW* and used graph based methods for prestige discovering [3, 7, 20]. However, these solutions only involve homogeneous citation network, while ignoring other useful information. In the next period, lots of methods aimed to leverage information like authors or venues to construct a heterogeneous network. For example, Zhou *et al.* [34] proposed a method for co-ranking authors and their publications using co-author network, citation network and authorship network. Yan *et al* [32] introduced *P-rank* to rank scholarly prestige based on network of papers, authors and journals. Yet these methods ignore the dynamic nature of citation network, making them biased to old articles.

To address the dynamic nature of citation network, many efforts have been made. Walker *et al.* [28] presented a network traffic model *CiteRank* to mimic researchers starting their research from recent publications with probability proportional to:

$$\rho_i = e^{age_i/\tau_{dir}},$$

where age_i is the time between publication and current year, τ_{dir} is constant. By adding an exponential weight it reduces the aging effect. Moreover, Sayyadi and Getoor [25] introduced their model *FutureRank*, involving a random walk on citation network and author-paper network with a time-aware preference vector. Ghosh *et al.* [13] presented the idea of effective contagion matrix (ECM), which de-emphasized the older papers and can be efficiently computed. Wang *et al.* [30] defined a ranking algorithm utilizing citations, authors, venues, and charactered the publication time as time-aware edge. Wang *et al.* presented [29] *MRFRank* which uses words and words co-occurrence to model the innovativeness.

However, all the work mentioned above aims to rank each publication in a given dataset, and the network structure and optimization criteria thereby limit their ability to rank nodes lying on the edge of the network, *i.e.*, papers published just now and having no citations. Meanwhile, no ranking method leverages the affiliation information, to the best of our knowledge.

2.2 Citation Prediction

Another direction for ranking papers is based on predicting a paper's future citation number. This direction has gained increasing attention since the citation number is a common metric for a paper's quality and used frequently as a measurement for scholarly impact. The early work on citation number prediction, including [5], [6], and [18], extracts heuristic features and adopts simple models such as linear regression on relatively small datasets with the number of publications less than one thousand. Later, Fu and Aliferis [11] introduced the textual content and utilized it through identifying important and discriminative keywords in the text. Yan *et al.* [33] studied more sophisticated features such as authors' productivity, sociality, *h-index* and venue's centrality. They found the "author" and "venue" will make paper attractive while the content features in isolation are not predictive. Following this direction, Didegah and Thelwall [8] found that venue prestige plays a central role in determining a paper's future citation number. However, our experiment in Subsection 5.5 obtains a different result that "author" plays a dominant role for zero citation papers.

2.3 Learning to Rank

Learning to rank for Information Retrieval (IR) is a task to automatically construct a ranking model using training data [19]. It leverages machine learning technology to build effective ranking models, and can be divided into pointwise, pairwise and listwise categories.

The early methods are mainly pointwise and pairwise approaches. Herbrich *et al.* [14] proposed *RankSVM*, a pairwise method based on *SVM*, reducing the task of ranking to binary classification. Freund *et al.* [10] introduced *RankBoost*, a pairwise boosting approach where the weak classifier is chosen by maximizing the weighted rank loss. Based on neural network, Burges *et al.* [4] presented *RankNet* employing a probabilistic loss function. Yet the loss function of these algorithms can not be directly optimized by final ranked list, therefore many methods turned to focus on listwise approaches. Metzler and Croft [21] applied coordinate ascent with linear listwise model. And Xu and Li [31] modified *AdaBoost* and proposed a listwise boosting algorithm, *AdaRank*, which is able to optimize ranking metrics of any kind.

3. PRELIMINARIES

We model the academic network as a heterogeneous graph containing four kinds of nodes and four kinds of edges, and define *zero citation paper set* and *zero citation ranking* problem based on it.

3.1 Notations and Definitions

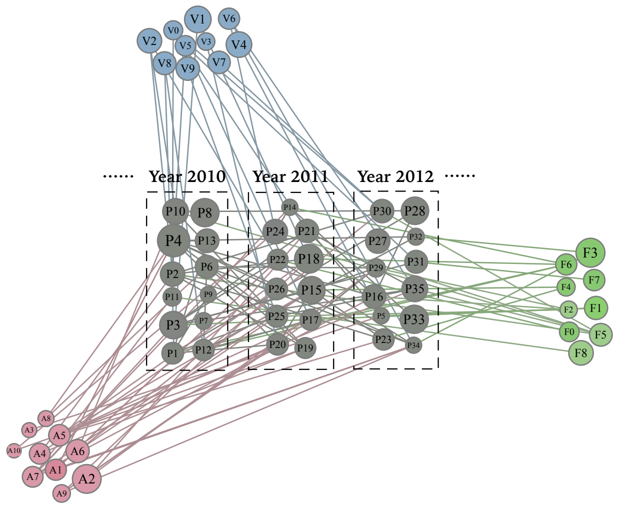


Figure 2: A demonstration of the heterogeneous graph in our method, which contains four kinds of nodes, *i.e.*, papers, authors, venues and affiliations.

DEFINITION 1. *We denote the heterogeneous graph containing papers, authors, venues (conferences and journals) and affiliations as*

$$G = (P \cup A \cup V \cup F, E^{PP} \cup E^{PA} \cup E^{PV} \cup E^{PF}), \qquad (1)$$

where P, A, V, F are the sets of nodes representing papers, authors, venues and affiliations. Each edge $(p_v, p_u) \in E^{PP}$ indicates a reference from paper v to paper u. E^{PA} denotes the authorship. $(p_v, v_u) \in E^{PV}$ denotes paper v is published on venue u. $(p_v, f_u) \in E^{PF}$ denotes paper v has an author from affiliation u. Note that here we model the affiliation to be one "attribute" of the paper instead of the author, in order to keep the centrality of paper nodes in the graph and perform the reinforced random walk in Algorithm 1. A demonstration of the network is shown in Figure 2.

DEFINITION 2. *Let the heterogeneous graph consist of papers published over time $t_0 < t_1 < \cdots < t_{crt}$, where t_0 is the publication time of the oldest paper in the network, t_{crt} is the current year. Let $t(p_i)$ be the publication year of paper p_i, we define zero citation paper set Z as*

$$Z = \{p_z \in P \mid t(p_z) = t_{crt}\}. \qquad (2)$$

Note that in our definition, *zero citation paper set* only contains papers published in the current year, instead of older papers without citations. We also assume a paper can only cite papers older than its publication year. In other words, *zero citation paper set* is equal to the set of current year papers.

3.2 Problem Formulation

Zero ranking problem aims to rank only newly published papers without citations. More formally, We define the *zero ranking problem* as following: we aim to obtain a ranking model r, such that for an academic graph G, it can output $r(G)$, a permutation of the *zero citation paper set* Z of G. In other words, $r(G)$ represents a ranked list of Z according to the predicted citation number after Δt years. Our optimization goal is:

$$\max_r E(r(G), \boldsymbol{y}_{\Delta t}), \qquad (3)$$

where $\boldsymbol{y}_{\Delta t}$ is the ranked list according to real citation numbers after Δt years. $E(\boldsymbol{l}_x, \boldsymbol{l}_y)$ denotes a performance measure function for an output list $\boldsymbol{l}_x$ with baseline list $\boldsymbol{l}_y$. In our work we set $\Delta t = 5$. Note that this problem is different from citation prediction, because here we do not care the exact citation number, but an order by predicted future citations.

4. ZERORANK ALGORITHM

Table 1: Notations.

Notation	Explanation
P, A, V, F	Papers/authors/venues/affiliations nodes in Graph G
$\boldsymbol{p}, \boldsymbol{a}, \boldsymbol{v}, \boldsymbol{f}$	Papers/authors/venues/affiliations scores in random walk
$\boldsymbol{x_t^a}, \boldsymbol{x_t^v}, \boldsymbol{x_t^f}$	"author", "venue", "affiliation" features for slice t
$\boldsymbol{y_t}$	Real citation rank list for slice t
$\{(\boldsymbol{x_t^a}, \boldsymbol{x_t^v}, \boldsymbol{x_t^f}, \boldsymbol{y_t})\}_{t=t_0}^{t_{crt}-1}$	Training set
r	Ranking model
$E(\cdot, \cdot)$	Performance measure function
$k_n \in \{k^A, k^V, k^F\}$	weaker ranker

In this work we choose "author", "venue", "affiliation" as features to characterize a paper's potential of being popular in the future, because intuitively a paper written by an influential author, from a top tier venue or affiliation is likely to receive more citations in that it is probably of high quality

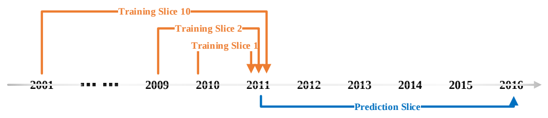


Figure 3: By shifting current time, we can construct a training set consisting multiple slices.

and is more likely to be read. However, giving a quantitative measurement to these "attributes" is difficult. If we directly apply citation based statistics such as IF to define an author or a venue, we are likely to ignore the difference of citing papers.

In fact, the measurement of papers, authors, venues and affiliations are correlated. For example, the measurement of an author is based on its publications. Therefore, in our algorithm, instead of using citation based statistics to define features, we design a reinforced random walk as a feature extractor.

To better use the information we have, we shift our current time t from t_0 to $t_{crt}-1$ to construct a training set, as shown in Figure 3. For each time we hide the information after the current time point t, and obtain a new *zero citation paper set*. We then perform our feature extraction algorithm on the network of this time and obtain a set of features. Moreover, this method enables us to obtain the citations between t to t_{crt} as a label for training, denoted by the orange arrows. We denote the set of features and citations when current time point is set to t as slice t. By applying this method we can obtain a training set containing $t_{crt}-t_0$ slices: $S = \{(\boldsymbol{x_t^A}, \boldsymbol{x_t^V}, \boldsymbol{x_t^F}, \boldsymbol{y_t})\}_{t=t_0}^{t_{crt}-1}$, where $\boldsymbol{x_t^A}, \boldsymbol{x_t^V}, \boldsymbol{x_t^F}$ are "author", "venue", "affiliation" features for slice t, and $\boldsymbol{y_t}$ is real citation rank list for slice t.

To train a ranking model combining different features, many citation prediction methods apply *linear regression* or *KNN*. However, these methods are not suitable for our problem due to its unique character: it is unreasonable to train one *linear regression* model using two papers from different slices because they are from different years and their citation numbers are influenced by their different historic situations. In addition, the real citation observation interval $t_{crt}-t$ decreases as t increases, making the smaller interval slice inevitably receive fewer citations. To deal with this issue, we design a learning to rank algorithm based on *AdaRank*. Each slice is handled as a query, and with boosting training, we optimize our ranking model iteratively by the performance measure function, and maintain the weights for each feature.

There are three phrases in our algorithms, which are random walk phrase, learning to rank phrase and deployment phrase. In the random walk phrase, we perform a reinforced random walk to get the authority scores of authors, venues and affiliations for each paper. In the training phrase, we use the authority scores as features and real future citations as labels to train a learning to rank model. In the deployment phrase, we use the trained learning to rank model with the authority scores to predict the zero citation rank.

4.1 Feature Extraction Phrase

The random walk algorithm, which leverages citations, authors, venues, affiliations and time information, is based on the following assumptions:

- Important papers are often cited by many important papers.
- Influential researchers are more likely to publish high quality papers, and distinguished articles increase their authors' influence.
- Top tier venues are more likely to publish high quality papers, and excellent publications increase venues' reputation.
- Top tier affiliations are more likely to publish high quality papers, and well-known literatures increase affiliations' fame.
- Recent papers are more convincing in showing the authority of authors, affiliation and venues, as well as the popularity of papers at present.

Algorithm 1 *ZeroRank* Random Walk

```
Require:
    Graph: $G$
    Parameter: $w_1, w_2, w_3, w_4, w_5, \rho, t_{crt}$
Ensure:
    Scores: $\boldsymbol{p}, \boldsymbol{a}, \boldsymbol{v}, \boldsymbol{f}$
 1: for all paper $i$ do
 2:     $p_i = \frac{1}{N}$
 3: while not converge do
 4:     for all author $i$ do
 5:         $a_i = AVG_{P_j \in neigh(A_i)}(p_j)$ ;
 6:     for all venue $i$ do
 7:         $v_i = AVG_{P_j \in neigh(V_i)}(p_j)$ ;
 8:     for all affiliation $i$ do
 9:         $f_i = AVG_{P_j \in neigh(F_i)}(p_j)$ ;
10:     for all paper $i$ do
11:
```

$$\begin{aligned} p'_i = \quad & w_1 \sum_{P_j \in in(P_i)} \frac{p_j}{|out(P_j)|} + \\ & w_2 \frac{1}{Z_A} AVG_{A_j \in neigh(P_i)}(a_j) + \\ & w_3 \frac{1}{Z_V} AVG_{V_j \in neigh(P_i)}(v_j) + \\ & w_4 \frac{1}{Z_F} AVG_{F_j \in neigh(P_i)}(f_j) + \\ & w_5 \frac{1}{Z_T} \exp(-\rho(t_i - t_{crt})). \end{aligned}$$

Based on these assumptions, we design our reinforced random walk algorithm shown in Algorithm 1. Initially each paper will be assigned a score of $\frac{1}{N}$, where N denotes the total number of papers. Then the algorithm performs an iterative computation until convergence when for any paper i, its scores of two consecutive iterations p_i and p'_i satisfy $|p'_i - p_i| < \epsilon$, where we set $\epsilon = 10^{-9}$.

Each iteration contains two steps:

- The scores of authors, venues and affiliations are computed by their related papers.
- The scores of papers are obtained by their related papers, authors, venues, affiliations as well as a time-aware constant.

In the first step, we compute author, venue, affiliation scores by averaging their related papers' scores. $AVG(\cdot)$ denotes the average value function. For example, for author i, his/her score will be the average score of all the publications belonging to him/her.

In the second step, a paper's score is obtained by linear combination of five parts: scores of papers citing it, authors, venue, affiliations it related, and a time constant. For the first part, a classic *PageRank* algorithm will be used. For computing the authors/affiliations, an average function will be used, and since one paper can only published on one venue, the venue part will be exactly the score of the venue a paper belonging to. Note that to make sure the algorithm converges, a normalization process is performed to make all the scores added from each part sum to 1 with normalization variables Z_A, Z_V, Z_F, Z_T. And for the last part, because old papers have more edges, which exaggerate their impact even if they are obsolete, we use a damping factor ρ to compensate newly published papers. $w1 \sim w5$ denote the weights of the five parts, and sum to 1.

Note that in real dataset, the information of authors, venues and affiliations is often incomplete. To tackle this problem, we bring the idea of virtual nodes. For example, if a paper u has no author, we will give it a virtual author whose publication contains only u.

Our random walk is in consistent with the five assumptions, and reveals the innate reinforcement relations between papers, authors, venues and affiliations. In particular, only the citation reinforcement is unidirectional, because papers only contribute to their references but not vice versa. Nevertheless, paper-author, paper-venue, paper-affiliation relationships are mutual reinforced.

Convergence

Here we prove the convergence of the random walk phrase. We rewrite Algorithm 1 into the matrix form: let A_P,A_A, A_F,A_V denote the normalized adjacent matrices of the graphs $G_P = (P, E^{PP})$, $G_A = (P \cup A, E^{PA})$, $G_V = (P \cup V, E^{PV})$, $G_F = (P \cup F, E^{PF})$. $\tilde{A_A}, \tilde{A_F}, \tilde{A_V}$ denote the normalized transpose adjacent matrices of the same graphs. The States 5,7,9 in Algorithm 1 can be rewritten as

$$\boldsymbol{a} = A_A \boldsymbol{p}, \tag{4}$$

$$\boldsymbol{v} = A_F \boldsymbol{p}, \tag{5}$$

$$\boldsymbol{f} = A_V \boldsymbol{p}. \tag{6}$$

Since $||\boldsymbol{p}||_1 = 1$, we can denote the last term in State 11 as $(\boldsymbol{d} \times \boldsymbol{e})$, where $\boldsymbol{d}$ is the time-aware vector, and $\boldsymbol{e} = (1 \cdots 1)$ is the vector consisting of all $1s$. Then iteration process for paper can be rewritten as:

$$\begin{aligned} \boldsymbol{p_{k+1}} = (&\omega_1 A_P + \omega_2 \tilde{A_A} A_A + \omega_3 \tilde{A_V} A_V \\ &+ \omega_4 \tilde{A_F} A_F + \omega_5 \boldsymbol{d} \times \boldsymbol{e}) \boldsymbol{p_k}. \end{aligned} \tag{7}$$

We denote

$$M = \omega_1 A_P + \omega_2 \tilde{A_A} A_A + \omega_3 \tilde{A_V} A_V + \omega_4 \tilde{A_F} A_F + \omega_5 \boldsymbol{d} \times \boldsymbol{e}.$$

According to [24] the sequence $\boldsymbol{p_k}$ will converge to the unique principal eigenvector of M.

4.2 Learning to Rank Phrase

After the random walk algorithm converges, we can obtain the "author", "venue", "affiliation" features of each paper i by averaging the authority scores related to P_i:

Algorithm 2 *ZeroRank* Ranking Model Training

```
Require:
    $S = \{(\boldsymbol{x_t^A}, \boldsymbol{x_t^V}, \boldsymbol{x_t^F}, \boldsymbol{y_t})\}_{t=t_0}^{t_{crt}-1}$
Ensure:
    Ranking model $r$
1: $P_1(t) = \frac{1}{t_{crt}-t_0}$
2: while performance increasing do
3:     create weak ranker $k_n \in \{k^A, k^V, k^F\}$ such that
           $$\max_{k_n} \sum_{t=t_0}^{t_{crt}-1} P_n(t) E(k_n(\boldsymbol{x_t^A}, \boldsymbol{x_t^V}, \boldsymbol{x_t^F}), \boldsymbol{y_t}). \tag{8}$$
4:     $\alpha_n = \frac{1}{2} \ln \frac{\sum_{t=t_0}^{t_{crt}-1} P_n(t)(1+E(k_n(\boldsymbol{x_t^A}, \boldsymbol{x_t^V}, \boldsymbol{x_t^F}), \boldsymbol{y_t}))}{\sum_{t=t_0}^{t_{crt}-1} P_n(n)(1-E(k_n(\boldsymbol{x_t^A}, \boldsymbol{x_t^V}, \boldsymbol{x_t^F}), \boldsymbol{y_t}))}$
5:     $r_n = r_{n-1} + \alpha_n k_n$
6:     $P_{n+1}(t) = \frac{\exp\{-E(k_n(\boldsymbol{x_t^A}, \boldsymbol{x_t^V}, \boldsymbol{x_t^F}), \boldsymbol{y_t})\}}{\sum_{t=t_0}^{t_{crt}-1} \exp\{-E(k_n(\boldsymbol{x_t^A}, \boldsymbol{x_t^V}, \boldsymbol{x_t^F}), \boldsymbol{y_t})\}}$
```

$$x_i^A = AVG_{A_j \in neigh(P_i)}(a_j), \tag{9}$$

$$x_i^V = AVG_{V_j \in neigh(P_i)}(v_j), \tag{10}$$

$$x_i^F = AVG_{F_j \in neigh(P_i)}(f_j). \tag{11}$$

Here note that the features computed by Equations 9, 10, 11 are innately of the same scale, which can be used in training without another scaling process.

In order to construct the training set, we shift our current time t from t_0 to $t_{crt} - 1$, each time we hide the graph information after the current time point t, obtaining a new *zero citation paper set*. We then perform our feature extraction algorithm on network of this time and get a set of features. By apply such method we can have a training set containing $t_{crt} - t_0$ slices: $S = \{(\boldsymbol{x_t^A}, \boldsymbol{x_t^V}, \boldsymbol{x_t^F}, \boldsymbol{y_t})\}_{t=t_0}^{t_{crt}-1}$, where $\boldsymbol{x_t^A}, \boldsymbol{x_t^V}, \boldsymbol{x_t^F}$ are "author", "venue", "affiliation" features for slice t, $\boldsymbol{y_t}$ is real citation ranking for slice t. In practice we observe that t_0 is not necessarily set to the first year of graph G, because time slice that is too old could help little to reveal the authority at present, thus in our experiment we set $t_0 = t_{crt} - 10$.

Then we modify the *AdaRank* algorithm to train a ranking model based on the training set. In Algorithm 2, a boosting algorithm is performed. Each iteration we select a weak ranker and finally we linearly compose all the weak rankers to obtain the ranking model r.

To be specific, a weight distribution for each slices is maintained. We denote it as $\boldsymbol{P_n}$ for nth iteration. After each iteration, $\boldsymbol{P_n}$ is modified to make those slices which have a "bad" ranking result take up a higher weight, and those having a "good" ranking result take up a lower weight. In this way, in the next iteration *ZeroRank* will try to select a weak ranker focusing on those "hard" slices. Furthermore, a weak ranker will be directly selected from the features. That is, we select a weak ranker which can maximize Equation 8, where $E(\cdot, \cdot)$ is the performance measure function. For example, weak ranker k^A will rank only according to the author feature. After k_n is selected, α_n will be computed as a measurement of effectiveness of k_n. And finally r is composed of the linear combination of k_n with weight α_n.

Note that in our algorithm we track the performance changing and stop the iterative process when the performance no longer increases. According to Theorem 1 in [31], there exists a lower bound for the accuracy of the training function. Meanwhile, since the performance continuously increases and has an upper bound 1, the algorithm must converge.

4.3 Parallel Random Walk

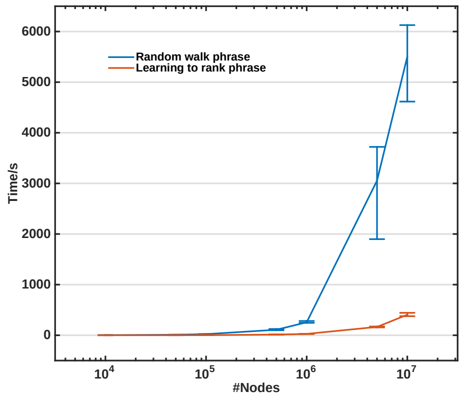


Figure 4: The running time of random walk phrase and learning to rank phrase among different sizes. The algorithm is implemented is *C++* and performed on a server with 2 Intel Xeon CPU E5-2650 v3 processors and 128 GB ram.

We first analyze the complexity of our algorithm. The random walk phrase time has the same time cost as *PageRank*, which is $O(M)$ [2], where M denotes the number of edges. And the learning to rank phrase will cost $O(TNlogN)$ [31], where N is the number of papers in the training set and T denotes the number of iterations.

Besides the time cost, the space cost, $O(M + N)$, makes it unacceptable to run this algorithm on a single node, thus implying a parallel algorithm is needed. In practice, experiment result in Figure 4 shows that the random walk phrase takes a major running time. Consequently, we propose a parallel solution for random walk and implement it on *Spark*. As shown in Algorithm 3, there are two kinds of nodes, which are nodes of authors, venues and affiliations and nodes of papers. The first category runs $RankAVF$ procedure while the second runs $RankP$ procedure. During each iteration, nodes get a message list as parameters and then update their values before they send messages to their adjacent nodes.

To be specific, for a P node, it first computes a score based on $Pmsgs$ (denoting messages from P nodes in the previous iteration) and $Amsgs$ (denoting messages from AVF nodes). After that, it computes and sends messages to its successor P nodes and adjacent AVF nodes. And an AVF node computes its authority score based on $Hmsgs$ and sends it to P nodes. If the score computed by one P node converges, the node will vote to stop, and the algorithm halts if all nodes vote to stop.

Algorithm 3 *ZeroRank* Parallel Random Walk

```
Require:
    Graph: $G$
    Parameters: $w_1, w_2, w_3, w_4, w_5, \rho, t_{crt}$
Ensure:
    Scores: $\boldsymbol{p}, \boldsymbol{a}, \boldsymbol{v}, \boldsymbol{f}$;
1: procedure RankAVF($v$:$AVFid$, $Hmsgs$:$List$)
2:     var $msgSum, msgNum = 0$
3:     for all $m \leftarrow msgs$ do
4:         $msgSum$ += $m$
5:         $msgNum$ ++
6:     $v.score = msgSum/msgNum$
7:     for all $j \leftarrow neighP(v)$ do
8:         $msg = v.score$
9:         $send\_msg(to{=}j,\ msg)$ ▷ send to adjacent papers

10: procedure RankP($v$:$Pid$, $Pmsgs$:$List$, $Amsgs$:$List$)
11:     var $msgSum.p, msgSum.a, msgSum.v, msgSum.f{=}0$
12:     var $msgNum.a, msgNum.v, msgNum.f = 0$
13:     for all $m \leftarrow Pmsgs$ do
14:         $msgSum.p$ += $m$
15:     $v.score = w1 * msgSum.p + w5 * v.timeScore$
16:     for all $m \leftarrow Amsgs$ do
17:         $msgSum.(m.type)$ += $m.value$
18:         $msgNum.(m.type)$ ++
19:     for $i$ in $\{a, v, f\}$ do
20:         $v.score$ += $w_i * Norm(msgSum.i/msgNum.i)$
21:     for all $j \leftarrow v.outP$ do
22:         $msg = v.score/|v.outP|$
23:         $send\_msg(to{=}j,\ msg)$ ▷ send to adjacent papers
24:     for all $j \leftarrow v.outAVF$ do
25:         $msg = (type{=}j.type, value{=}v.score)$
26:         $send\_msg(to{=}j,\ msg)$ ▷ send to adjacent AVFs
27:     if converged($v.score$) then
28:         $voltToHalt(v)$
```

5. EXPERIMENT

5.1 DataSet

We use *Microsoft Academic Graph(MAG)*[2] provided by *Microsoft* [27] as our experiment dataset. We first remove papers published in 2017 and edges pointing from past to future, obtaining 126, 908, 750 papers published from 1800 to 2016 as well as 526, 449, 409 edges. And the dataset also contains 114, 698, 004 authors, 23, 404 journals, 1, 283 conferences and 19, 843 affiliations. The distribution of papers over publication year is shown in Table 2.

In the following experiments, we extract subsets of MAG and test different features of our algorithm. The details about how we extract the subsets are described in related subsections. In the following subsections, we first choose the parameter ρ in our algorithm based on computer science field, and then perform 4 experiments. The first experiment shows that the random walk phrase does extract features leading to high citations, and the second presents that comparing to other scientific ranking and citation prediction methods, our algorithm has a higher accuracy on *zero citation ranking* problem. Then we compare the performance of *ZeroRank* and *FutureRank* based on varying parameters and observe our method has an average better result. Finally, we run our algorithms on 31 subfields of computer science to study the major feature for highly cited papers.

[2]http://research.microsoft.com/en-us/projects/mag/.

Table 2: Distribution of papers over publication year.

year	before 2000	2000	2001	2002	2003	2004	2005	2006	2007
# of papers	47,627,409	2,668,362	2,759,899	3,020,753	3,240,642	3,514,052	3,833,380	4,370,690	4,702,195
year	2008	2009	2010	2011	2012	2013	2014	2015	2016
# of papers	5,177,040	5,964,730	6,144,130	6,562,040	6,750,651	7,315,938	7,057,228	5,729,525	470,086

Aging Parameter Choosing

In the random walk phrase, the aging parameter ρ compensates scores of newly published papers, thus finding a best value of ρ can adequately model the aging effect. To achieve this, considering in the following experiments we mainly focus on CS field, we first extract a subset of papers whose keywords map to computer science, which contains $8,884,763$ papers. Then we plot their citation numbers over time after publication. Inspired by [25], we ignore the points for year 0, 1 after publication, and find the best exponential function which matches the figure is:

$$ce^{-0.124t}. \tag{12}$$

So we set $\rho = -0.124$. It is interesting to point out this is different from $\rho = -0.62$[25] for arXiv (hep-th) dataset, which implies these two datasets have different structures.

5.2 Feature Extraction

Table 3: Top 10 authors, venues, affiliations computed by random walk.

Rank	Author		Venue		Affiliation	
	Cits	CRank	Cits	CRank	Cits	CRank
1	15	4	15	1	0.5	369
2	28	3	3	5	0	1156
3	77	1	0	636	4.25	7
4	7.5	22	1	56	3.9	10
5	12	8	0	636	0	1156
6	41	2	0	636	0	1156
7	2	451	3	5	3	12
8	8	19	0.5	218	3	12
9	5	80	6	2	0.4	515
10	0	5572	0.5	218	0.5	369

In this experiment, we evaluate *ZeroRank's* ability to extract features related to high citations. We first extract the set of papers whose keywords map to both computer science and data mining, and obtain $20,320$ papers, $29,586$ authors, $1,738$ venues and $2,836$ affiliations. We then set the current time $t_{crt} = 2006$ and adjust the parameters to achieving the best prediction result in 2011. Next we set $t_{crt} = 2011$ to evaluate the feature extraction process.

Table 3 shows the top 10 authors, venues, affiliations computed by random walk, where "Cits" denotes the real average citation from 2011 to 2016 and "CRank" denotes ranking for "Cits". Note that we do not aim to select authors, venues, affiliations that have published a large quantity of papers, but those who are likely to publish highly cited papers. So we measure the results according to the average citation instead of total citations. We can observe that *ZeroRank* extracts promising features because of all the 30 items above, 15 of them are in the top 30 CRank. In addition, the performance of "author" feature extraction is the best and 7 of the authors are in top 30 CRank from 29586 in total. While the "affiliation" extraction seems the worst, but considering 59.3% affiliations have no citations from 2011 to 2016, many affiliations themselves are indistinguishable.

5.3 Zero Citation Ranking

In this experiment we evaluate *ZeroRank* for *zero citation ranking* problem by comparing with state-of-the-art ranking and citation prediction algorithms.

Dataset and Time Setup

We select two subsets from *MAG*: data mining(DM) and database(DB). For each dataset, we first collect all papers whose keywords map to the label, and then enlarge the set by adding other papers directly linked it. Finally, we obtain DM with $461,392$ papers, and DB with $434,442$ papers.

Similar to experiment in Subsection 5.2, we first adjust the parameters of all the following algorithms on the current time point 2006 with the future citation numbers from 2006 to 2011. After that we evaluate them on the current time point 2011 with future citation numbers from 2011 to 2016. The parameters $w_1 \sim w_5$ for data mining and database are $0.4, 0, 0.1, 0.1, 0.4$ and $0.4, 0, 0.3, 0.2, 0.1$.

Baseline

We compare our algorithm with *FutureRank* [25], *P-rank* [32], *CCP-CART*[33], and *ZeroWalk*. *CCP-CART* is a citation prediction method using Classification and Regression Tree, and we rank the paper list according to the predicted citations. Due to the limit of dataset, for *CCP-CART* we do not implement content based features. *ZeroWalk* denotes the random walk phrase of our algorithm without learning to rank. For *ZeroRank*, we set the performance measure function in learning to rank phrase as NDCG@10, because we would like to focus on the top highly cited papers.

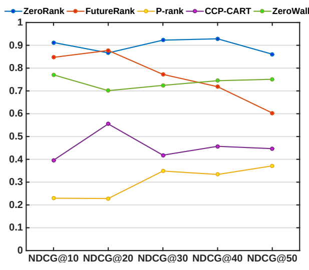
Evaluation Metrics

In order to evaluate the performance of these ranking algorithms, we introduce four kinds of metrics. Since a large part of the papers in *zero citation paper set* obtain no citations from 2011 to 2016 and researchers mainly care a small portion of papers, it's adequate to choose metrics that emphasize the top part of papers:

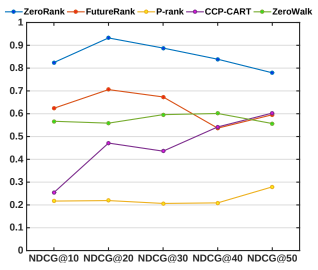
NDCG Normalized discounted cumulative gain [16] measures the performance based on the ground truth and gives top results high weights. The NDCG score is computed as

$$\text{NDCG@}k = Z_k \sum_{i=1}^{k} \frac{2^{r_i} - 1}{log_2(i+1)}, \tag{13}$$

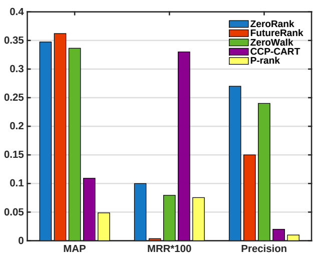
where r_i is the score of the ith paper, k is a constant and Z_k is the normalization constant to ensure that an ideal rank will get score 1. Since citation number is unsuitable to directly used in NDCG, We sort the papers according to their future citation numbers in descending order, and



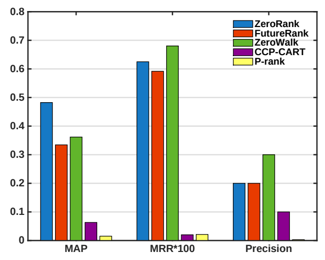
(a) *NDCG-DM*



(b) *NDCG-DB*



(c) *Others-DM*



(d) *Others-DB*

Figure 5: Figure 5a, 5b illustrate the NDCG scores of DM, DB, Figure 5c, 5d show MAP, MRR, Precision scores of DM and DB. All methods choose the best parameters adjusted from 2006 to 2011.

assign the $0\% - 10\%$, $10\% - 30\%$, $30\% - 60\%$, $60\% - 100\%$ papers with scores 3, 2, 1, 0 separately.

MAP Mean of the average precision scores [1] is defined as following:

$$\text{MAP}@k = \sum_{i=1}^{k} \frac{p_i}{d_i}, \tag{14}$$

where p_i is the ith relevant paper while d_i is the rank of the paper in the list, k is a constant, where we set $k = 100$. In our experiment, we set papers of top 1% future citations with $p_i = 1$, otherwise $p_i = 0$.

MRR: Mean reciprocal rank, which measure the performance of a ranking algorithm by top 1 paper in the real rank:

$$\text{MRR} = \frac{1}{x}, \tag{15}$$

where x is the position of real top 1 paper ranked by the algorithm.

Precision: Precision[25] is defined by the union of real top k papers with the top k papers returned by algorithm.

$$\text{Precision}@k = \frac{|\text{realTopk} \cap \text{rankTopk}|}{k}. \tag{16}$$

In the experiment we set $k = 100$.

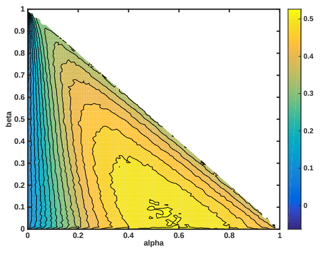
Experiment Result

Figure 5a, 5b demonstrate the NDCG scores of the five algorithms. Figure 5c, 5d show the MAP, MRR, Precision scores. To plot MAP, MRR, Precision scores on one figure, we scale the scores of different metrics.

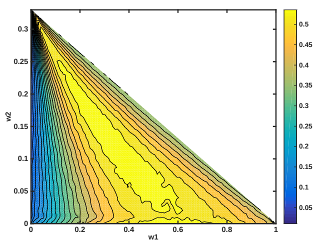
For NDCG, we can observe that in both datasets *ZeroRank* achieves the best scores, with average scores 0.898 and 0.853, while *FutureRank* obtains the average scores 0.764 and 0.627. The scores of *ZeroRank* is 17.5% and 35.9% higher than *FutureRank* separately. And comparing *ZeroRank* with *ZeroWalk*, we can observe that learning to rank refines the ranking result by adjusting the weights of each feature according to the training set.

For MAP, MRR and Precision, we can observe *ZeroRank* and *ZeroWalk* together achieve 4 best results of 6. And for the comparison between *ZeroRank* and *ZeroWalk*, the result shows that *ZeroWalk* outperforms *ZeroRank* in some cases. This can be explained by the NDCG@10 performance measure function we choose in learning to rank phrase. Different metrics are not ideally identical and there is a trade-off between them. The performance measure function help *ZeroRank* achieve a higher score in NDCG while leads a lower score in other metrics. However, if we choose a different performance measure function, we can achieve higher scores in the corresponding metric. This implies another advantage of learning to rank phrase that it enabling us to refine certain metric by selecting corresponding performance measure function.

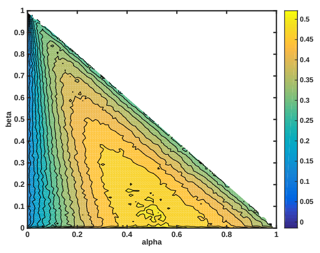
5.4 Performance Comparison Based on Varying Parameters



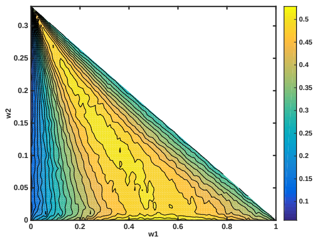
(a) *FutureRank-DM*



(b) *ZeroRank-DM*



(c) *FutureRank-DB*



(d) *ZeroRank-DB*

Figure 6: Spearman's rank correlation coefficient for *FutureRank/ZeroRank* on DM/DB. Comparing the left and right part, we can conclude that *ZeroRank*'s random walk phrase has a better score for different parameter settings.

In this Subsection we evaluate the performance of *ZeroRank* by varying parameters. We use DM and DB in Subsection 5.3 as the evaluation datasets, and compare *ZeroRank* with *FutureRank*. Considering the learning to rank phrase uses additional information and refines the features' weights, we only compare *FutureRank* with the *ZeroRank*'s random walk phrase.

We choose the Spearman's rank correlation coefficient [22] as our evaluation metric, which measures the similarity of two rank lists, and is defined as:

$$\rho = \frac{\sum_i (R_1(P_i) - \bar{R}_1)(R_2(P_i) - \bar{R}_2)}{\sqrt{\sum_i (R_1(P_i) - \bar{R}_1)^2 \sum_i (R_2(P_i) - \bar{R}_2)^2}},$$

where $R_1(P_i)$ and $R_2(P_i)$ are the rank positions of paper i in rank list 1 and rank list 2. $\bar{R}_1$ and $\bar{R}_2$ are the average rank positions of rank list 1 and 2.

FutureRank [25] has three parameters α, β, γ, and $\alpha + \beta + \gamma = 1$. We enumerate its parameter settings and record the results. For *ZeroRank*, it involves 5 parameters $w_1 \sim w_5$ and $\sum_{i=1}^{5} w_i = 1$, which makes it impossible to enumerate each parameter freely and plot the results on a 2-D plan. To deal with this issue we set $w_2 = w_3 = w_4$, considering $w_2 \sim w_4$ all denote the authority weights.

Figures 6a, 6b denote *FutureRank* and *ZeroRank* on DB, while Figures 6c, 6d denote on DM. The x, y axes in Figure 6a, 6c denote α, β, while in Figure 6b, 6d denote w_1, $w_2 \sim w_4$. The color in each point represents the Spearman's rank correlation coefficient, where brighter point indicates a higher score. From both pairs we can find *ZeroRank* outperforms *FutureRank* on most parameter settings. For a quantitative measurement *ZeroRank* achieves an average value 0.4238 in DM and 0.3988 in DB , while *FutureRank* achieves 0.3908 and 0.3759 separately. Based on these observations, we can draw the conclusion that ZeroRank achieves not only a higher maximum score but also a higher average score among all parameter settings.

Note that from Subsection 5.5 we can learn actually these features have significantly different weights, which indicates equal weight setting by all means decreases *ZeroRank*'s achievement. In other words, *ZeroRank* could achieve higher accuracy if non-equal settings are applied.

5.5 Feature Importance in CS Field

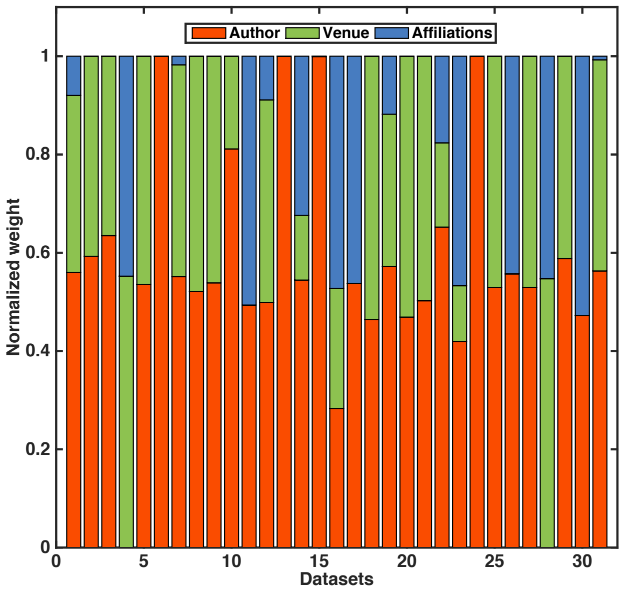


Figure 7: The normalized feature weights for *ZeroRank* on 31 subfields of computer science. We can observe "author" is a dominant feature.

Learning to rank algorithm enables us to quantitatively measure the importance of different features in determining whether a paper will receive more citations. We study this by experiments based on the "field of study" information in *MAG*. We focus on the papers belonging to computer science, which has 35 subfields[3].

We first perform preprocessing in the dataset. Papers containing author, venue, affiliation information and mapping to both CS and one of the subfields will be collected. After this step, we delete 4 subsets which are computational science, Internet privacy, management science, and theoretical computer science, because they have too few papers. Finally, we obtain a dataset containing 388,688 papers of 31 subfields, with each subfield containing 1,204 to 77,898 papers. Note that we choose a rather "strict" filtering restriction, causing the final qualified paper set small. A looser filtering method may get a larger dataset but it is also possible to involve many papers outside CS field.

We run *ZeroRank* separately on 31 subfields and the result is shown in Figure 7, where we can find the "author" feature dominates most subfields. We normalized the weights obtained from learning to rank and observe the average normalized weight of "author" is 56%. The "venue" feature follows with an average weight 29%, and the least significant feature is "affiliation". This is an interesting result which contradicts our intuition. However, the citation distribution in Figure 1 can give us some understanding why "venue" and "affiliation" seem not so promising.

6. CONCLUSION AND FUTURE WORK

In this paper, we propose a new ranking problem *zero citation ranking*, which means ranking newly published scientific articles without citation information. To deal with this issue, we introduce an algorithm *ZeroRank* integrating random walk and learning to rank. *ZeroRank* leverages the citations, authors, venues, affiliations and time information to construction a heterogeneous network, and uses random walk as a feature extractor. It also trains an efficient ranking model based on learning to rank technology. To ensure the algorithm's scalability, we design a parallel random walk and implement it on *Spark*. Experimental results show that our algorithm outperforms the state-out-art scientific ranking and citation prediction algorithms. We also conduct experiments and find that in computer science field, "author" is a dominant feature in determining a paper to gain more citations.

In the future, we will do experiments on other fields to verify whether they have the same major feature as computer science. Furthermore, we will add other features such as innovativeness of papers, popularity of topics, degree of difficulty to the learning to rank phrase and measure their importances. Finally, we will further study why "author" plays such a significant role while "venue" and "affiliation" do not.

7. REFERENCES

[1] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*, volume 463. ACM press New York, 1999.

[2] B. Bahmani, A. Chowdhury, and A. Goel. Fast incremental and personalized pagerank. *Proceedings of*

[3]Fields in *MAG* map to hierarchies from $L0$ to $L3$ from high to low. CS maps to $L0$ and the 35 subfields we choose map to $L1$.

the VLDB Endowment, 4(3):173–184, 2010.

[3] J. Bollen, M. A. Rodriquez, and H. Van de Sompel. Journal status. *Scientometrics*, 69(3):669–687, 2006.

[4] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96. ACM, 2005.

[5] M. Callaham, R. L. Wears, and E. Weber. Journal prestige, publication bias, and other characteristics associated with citation of published studies in peer-reviewed journals. *Jama*, 287(21):2847–2850, 2002.

[6] C. Castillo, D. Donato, and A. Gionis. Estimating number of citations using author reputation. In *String processing and information retrieval*, pages 107–117. Springer, 2007.

[7] P. Chen, H. Xie, S. Maslov, and S. Redner. Finding scientific gems with google's pagerank algorithm. *Journal of Informetrics*, 1(1):8–15, 2007.

[8] F. Didegah and M. Thelwall. Determinants of research citation impact in nanoscience and nanotechnology. *Journal of the American Society for Information Science and Technology*, 64(5):1055–1064, 2013.

[9] L. Egghe. Theory and practise of the g-index. *Scientometrics*, 69(1):131–152, 2006.

[10] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *The Journal of machine learning research*, 4:933–969, 2003.

[11] L. D. Fu and C. Aliferis. Models for predicting and explaining citation count of biomedical articles. In *AMIA Annual Symposium Proceedings*, volume 2008, page 222. American Medical Informatics Association, 2008.

[12] E. Garfield et al. Citation analysis as a tool in journal evaluation. American Association for the Advancement of Science, 1972.

[13] R. Ghosh, T.-T. Kuo, C.-N. Hsu, S.-D. Lin, and K. Lerman. Time-aware ranking in dynamic citation networks. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 373–380. IEEE, 2011.

[14] R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. *Advances in neural information processing systems*, pages 115–132, 1999.

[15] J. E. Hirsch. An index to quantify an individual's scientific research output. *Proceedings of the National academy of Sciences of the United States of America*, 102(46):16569–16572, 2005.

[16] K. Järvelin and J. Kekäläinen. Ir evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 41–48. ACM, 2000.

[17] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[18] A. V. Kulkarni, J. W. Busse, and I. Shams. Characteristics associated with citation rate of the medical literature. *PloS one*, 2(5):e403, 2007.

[19] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[20] N. Ma, J. Guan, and Y. Zhao. Bringing pagerank to the citation analysis. *Information Processing and Management*, 44(2):800–810, 2008.

[21] D. Metzler and W. B. Croft. Linear feature-based models for information retrieval. *Information Retrieval*, 10(3):257–274, 2007.

[22] J. L. Myers, A. Well, and R. F. Lorch. *Research design and statistical analysis.* Routledge, 2010.

[23] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[24] T. Sauer. *Numerical Analysis.* Pearson Addison Wesley, 2006.

[25] H. Sayyadi and L. Getoor. Futurerank: Ranking scientific articles by predicting their future pagerank. In *SDM*, pages 533–544. SIAM, 2009.

[26] J. Shen, Z. Song, S. Li, Z. Tan, Y. Mao, L. Fu, L. Song, and X. Wang. Modeling topic-level academic influence in scientific literatures. In *Workshops at the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[27] A. Sinha, Z. Shen, Y. Song, H. Ma, D. Eide, and K. Wang. An overview of microsoft academic service (mas) and applications. WWW - World Wide Web Consortium (W3C), May 2015.

[28] D. Walker, H. Xie, K.-K. Yan, and S. Maslov. Ranking scientific publications using a model of network traffic. *Journal of Statistical Mechanics: Theory and Experiment*, 2007(06):P06010, 2007.

[29] S. Wang, S. Xie, X. Zhang, Z. Li, S. Y. Philip, and X. Shu. Future influence ranking of scientific literature. In *SDM*, pages 749–757. SIAM, 2014.

[30] Y. Wang, Y. Tong, and M. Zeng. Ranking scientific articles by exploiting citations, authors, journals, and time information. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[31] J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 391–398. ACM, 2007.

[32] E. Yan, Y. Ding, and C. R. Sugimoto. P-rank: An indicator measuring prestige in heterogeneous scholarly networks. *Journal of the American Society for Information Science and Technology*, 62(3):467–477, 2011.

[33] R. Yan, J. Tang, X. Liu, D. Shan, and X. Li. Citation count prediction: learning to estimate future citations for literature. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1247–1252. ACM, 2011.

[34] D. Zhou, S. A. Orshanskiy, H. Zha, and C. L. Giles. Co-ranking authors and documents in a heterogeneous network. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 739–744. IEEE, 2007.