Research Paper Recommender Systems: A Random-Walk Based Approach

Marco Gori
Dipartimento di Ingegneria
dell'Informazione
University of Siena
Via Roma, 56 Siena, Italy
marco@dii.unisi.it

Augusto Pucci
Dipartimento di Ingegneria
dell'Informazione
University of Siena
Via Roma, 56 Siena, Italy
augusto@dii.unisi.it

Abstract

Every day researchers from all over the world have to filter the huge mass of existing research papers with the crucial aim of finding out useful publications related to their current work. In this paper we propose a research paper recommending algorithm based on the Citation Graph and random-walker properties. The PaperRank algorithm is able to assign a preference score to a set of documents contained in a digital library and linked one each other by bibliographic references. A data set of papers extracted by ACM Portal has been used for testing and very promising performances have been measured.

1. Introduction

Writing papers, technical reports, presentations and so on is a crucial activity for lots of people jobs all over the world. Technology already provided good means to write, but we need better means to collect the information we need for writing something interesting and meaningful. If we focus our attention on the production of research papers we have a very interesting applicative scenario where the knowledge refinement and the bibliographical search is a key part of the work. A researcher have to be well aware of most recent improvement in the topic he is studying and writing about, but the number of research paper published is exponentially growing, so he needs to choose and select papers from the huge quantity of potential candidates. This filtering process is generally tedious and time consuming. Typical tools adopted are keywords based search and paper citations following, unluckily both techniques are quite inaccurate and with a low coverage level. In this paper we propose a research paper recommending algorithm based on the citation graphs and random-walker properties. PaperRank is a support for the resource filtering process, in fact it requires a user to select an initial small subset of documents, relevant for the topic he is writing about. Then the algorithm can spread its boosting effect, due to selected papers, through the citation graph in order to discover other interesting and useful resources. The paper is organized as follows. Subsection 1.1 contains a review of paper recommender systems. In section 2 paper recommender system problem is formalized. Section 3 describe PaperRank algorithm in details. We describe the data set we build to test our algorithm in section 4 and we show performance measures in section 5. Finally in section 6 we summarize the main contribution of the present paper and we show some possible future extensions of the given algorithm.

1.1 Related Work

In literature some algorithms have been proposed attempting to overcome valuable research papers seeking. This class of tasks is usually defined as "Paper Recommender Systems". A simplified example of recommender algorithm based on text and citations can be found in [9], but this system just creates recommendations inside a single digital book. A more interesting example is [6], here the authors propose a collaborative filtering based approach to paper recommendation and they use the Citation Graph between papers to create the ratings, they also tried six different approaches on a subset of ResearchIndex 186,000 research papers contained in Research Index. Another interesting approach has been proposed in [1], the idea is that researchers from the same area tend to be interested to the same articles, so it is possible to improve search results by using recommendations based on previous searches performed by other people with similar interests. Paper recommender systems are just a special case of recommender systems research area. In general a recommender system makes personalized item suggestions by extracting knowledge from the previous user interactions with the system. Such services are particularly useful in the modern elec-

Proceedings of the 2006 IEEE/WIC/ACM International Conference
on Web Intelligence (WI 2006 Main Conference Proceedings)(WI'06)
0-7695-2747-7/06 $20.00 © 2006 IEEE

tronic marketplace which offers an unprecedented range of products. Several recommender systems have been developed that cope with different kind of items/products, see [8] for a review.

2. Paper Recommendation Problem

In this section we better formalize the paper recommender system problem scenario. We consider an active user u who is writing a paper containing references, we denote this "work in progress" document as $\bar{p}$. We suppose user u can access a digital library to consult materials related to the topic he is currently working on. The set of documents in a given digital library will be referred as $\mathcal{D} = \{p_i : i = 1, \cdots, n\}$, moreover, for every paper p_i, we consider set $\mathcal{R}_{p_i}$ which contains every paper cited by p_i so $\mathcal{D} \supseteq \mathcal{R}_{p_i} = \{p_j : p_j \textit{ is cited by } p_i\}$. We also suppose user u has already cited in its current paper reference list some papers belonging to $\mathcal{D}$, so $\mathcal{R}_{\bar{p}} \neq \emptyset$. The aim of a recommender system in this context is to compute a meaningful score $s(p_i)$ for every $p_i \in \mathcal{D}$, such that the higher the score of paper p_i, the higher has to be its relevance with respect to the topic of paper $\bar{p}$. It is also possible to have a graphical interpretation of the cross-citation structure of $\mathcal{D}$. We can introduce the *Citation Graph* $\mathcal{G}_{\tilde{\mathcal{C}}}$. This graph contains n nodes, one for each paper in $\mathcal{D}$. From the edges point of view, we consider graph $\mathcal{G}_{\tilde{\mathcal{C}}}$ to be undirected and it contains the edge pairs (p_i, p_j) and (p_j, p_i) if and only if paper p_i cites paper p_j or vice versa. So more formally:

$$((p_i, p_j), (p_j, p_i) \in Edge\{\mathcal{G}_{\tilde{\mathcal{C}}}\}) \Longleftrightarrow \left(p_i \in \mathcal{R}_{p_j} \vee p_j \in \mathcal{R}_{p_i}\right)$$

where $Edge\{\mathcal{G}_{\tilde{\mathcal{C}}}\}$ is the set of edges in $\mathcal{G}_{\tilde{\mathcal{C}}}$. Matrix $\tilde{\mathcal{C}}$ is the *Connectivity Matrix* corresponding to the Citation Graph, such that:

$$\tilde{\mathcal{C}}_{i,j} = \begin{cases} 1 & \text{if } (p_i, p_j) \in Edge\{\mathcal{G}_{\tilde{\mathcal{C}}}\} \\ 0 & \text{otherwise} \end{cases}$$

where $\forall i,\ \tilde{\mathcal{C}}_{i,i} = 0$ and $\tilde{\mathcal{C}}$ is a symmetric matrix. We normalize matrix $\tilde{\mathcal{C}}$ in order to obtain a stochastic matrix $\mathcal{C}_{i,j} = \frac{\tilde{\mathcal{C}}_{i,j}}{\omega_j}$ where ω_j is the sum of entries in $j - th$ column of $\tilde{\mathcal{C}}$. $\mathcal{C}$ is the *Correlation Matrix*, every entry contains the correlation index between paper pairs. The Correlation Matrix can be also considered as a weighted connectivity matrix for the *Correlation Graph* $\mathcal{G}_{\mathcal{C}}$. The weight associated to link (p_i, p_j) will be $\mathcal{C}_{i,j}$, note that while $\tilde{\mathcal{C}}$ is symmetrical, $\mathcal{C}$ is not, so the weight associated to (p_i, p_j) can differ from (p_j, p_i) weight.

3. PaperRank Algorithm

The idea underlaying the PaperRank algorithm is that we can use the model expressed by the Correlation Graph to find out valuable papers to suggest to an user, on the basis of papers already chosen to be part of the current paper bibliography ($\mathcal{R}_{\bar{p}}$). If we consider the set of papers in $\mathcal{R}_{\bar{p}}$, we can exploit the information present in graph $\mathcal{G}_{\mathcal{C}}$ in order to "spread" user preferences through the Correlation Graph. Obviously we have to properly control the *preference flow* in order to transfer high score values to papers that are strongly related to papers already present in $\mathcal{R}_{\bar{p}}$. The spreading algorithm we apply has to possess two key properties: propagation and attenuation. These properties reflect two key assumptions. First of all if a paper p_k is related to one or more good papers in $\bar{p}$ reference list, then paper p_k will also be a good suggestion for the user. If we analyse the Correlation Graph we can easily discover relationships between papers and also the strength of these connections, that is the weight associated to every link connecting two papers. The second important factor we have to take into account is attenuation. Good papers in $\mathcal{R}_{\bar{p}}$ have to transfer their positive influence through the Correlation Graph, but this effect decreases its power if we move further and further away from good papers, moreover if a good paper p_i is connected to two or more nodes, these have to share the boosting effect from p_i according to the weights of their connections as computed in matrix $\mathcal{C}$. PageRank algorithm (see [7]) has both propagation and attenuation properties we need, furthermore thanks to significant research efforts we can compute PageRank in a very efficient way (see [4]). Consider a generic graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes connected by directed links in $\mathcal{E}$, the classic PageRank algorithm computes an importance score $PR(n)$ for every node $n \in \mathcal{V}$ according to graph connectivity: a node will be important if it is connected to important nodes with a low out-degree. So the PageRank score for node n is defined as:

$$PR(n) = \alpha \cdot \sum_{q:(q,n)\in\mathcal{E}} \frac{PR(q)}{\omega_q} + (1 - \alpha) \cdot \frac{1}{|\mathcal{V}|} \quad (1)$$

where ω_q is the out-degree of node q, α is a decay factor[1]. Classic PageRank can be extended by generalizing equation 1:

$$\mathbf{PR} = \alpha \cdot \mathbf{M} \cdot \mathbf{PR} + (1 - \alpha) \cdot \mathbf{d} \quad (2)$$

where $\mathbf{M}$ is a stochastic matrix, its non-negative entries has to sum up to 1 for every column, and vector $\mathbf{d}$ has non-negative entries summing up to 1. Vector $\mathbf{d}$ can be tuned in order to bias the PageRank by boosting nodes corresponding to high value entries and matrix $\mathbf{M}$ controls the propagation and attenuation mode. Biased PageRank has been analysed in [5] and custom static score distribution vectors $\mathbf{d}$ have been applied to compute topic-sensitive PageRank

[1] A common choice for α is 0.85

Proceedings of the 2006 IEEE/WIC/ACM International Conference
on Web Intelligence (WI 2006 Main Conference Proceedings)(WI'06)
0-7695-2747-7/06 $20.00 © 2006 IEEE

[3] and for combating web spam [2]. We present the *PaperRank* algorithm, that is a biased version of PageRank designed to be applied to a recommender system. PaperRank equation can be easily derived from equation 2. We use graph $\mathcal{G}_\mathcal{C}$ to compute a PaperRank vector **IR**, where its $i-th$ component IR_i correspond to the PaperRank value of paper p_i, obviously these values also depend on the set of papers in $\mathcal{R}_{\bar{p}}$. In order to compute PaperRank, the stochastic matrix **M** in equation 2 will be the Correlation Matrix $\mathcal{C}$ and the dependencies between **IR** and $\mathcal{R}_{\bar{p}}$ can be modeled by simply building a **d** static score distribution vector depending on $\mathcal{R}_{\bar{p}}$. The resulting equation is:

$$\mathbf{IR} = \alpha \cdot \mathcal{C} \cdot \mathbf{IR} + (1-\alpha) \cdot \mathbf{d} \tag{3}$$

where **d** has been build by defining the unnormalized $\tilde{\mathbf{d}}$, with respect to the $i-th$ component, as:

$$\tilde{d}_i = \begin{cases} 1 & \text{if } p_i \in \mathcal{R}_{\bar{p}} \\ 0 & \text{if } p_i \notin \mathcal{R}_{\bar{p}} \end{cases}$$

So the normalized **d** vector will simply be $\mathbf{d} = \frac{\tilde{\mathbf{d}}}{|\tilde{\mathbf{d}}|}$. PaperRank, as defined in equation 3, can be computed also iteratively. The corresponding dynamic system has to be run for different $\mathcal{R}_{\bar{p}}$ in order to compute effective paper suggestion, luckily it only needs on average about 20 iterations to converge. The interpretation of **IR** score vector is straightforward, PaperRank scores induce a sorting of papers according to their expected liking for a given user. The higher is the PaperRank for a paper, the higher is the probability that it will be a valuable suggestion for a user.

4. Dataset

In order to test the effectiveness of algorithm proposed in section 3 we build a data set of papers to measure performances. The data set has been obtained by extensively crawling the ACM Portal web site[2] that is a wide collection of high quality papers in Computer Science. First of all we chose 9 different topics, identified by one or more keywords, then ACM Portal has been queried for each topic, the result is a ranked list of papers within the digital library that are good candidates to satisfy the query obtained by combining keywords related to a given topic. Now for every topic τ it is possible to select the first 20 papers as shown in the result list for that topic keywords by ACM Portal, we refer each of these sets as Q_τ. The crawler has to look for papers cited by papers contained in Q_τ and then the process continues iteratively until a maximum number of paper citations has been found (in this case the fixed maximum is 2000). The crawler output will be a citation graph $\mathcal{G}_\mathcal{C}{}^\tau$ (obviously we have a different citation graph for every topic) and Q_τ. Note that even if the maximum number of papers crawled is limited to 2000, that is the limit of papers downloadable by the crawler for each topic, but the number of identified cited documents is usually higher.

[2] http://portal.acm.org/

Topic	Number of Papers
Distributed Databases	4,961
Feature Extraction	3,463
Hidden Markov Model	3,978
Multiprocessors	4,292
Page Rank	5,225
Recommendation System	4,367
Relevance Feedback	6,278
Semantic Web	6,423
Support Vector Machine	4,206

Table 1. Graph dimension for every topic in the data set.

Table 1 shows the 9 selected topics and the number of nodes of corresponding Citation Graphs.

5. Experimental Results

Many different performance measure techniques have been adopted in research paper recommender system literature in order to proof the effectiveness of a proposed algorithm. There are two different class of approaches: online and offline experiments. In the online experiment human subjects are involved, they try an algorithm and it has to be collected user opinion on the quality of predictions made by a given method. In the offline experiments you can remove some citations from the test dataset and then attempting to predict those missing citations, otherwise you can focus on a cluster of documents you already know to be related one each other and you can test if an algorithm is able to "complete" this cluster if provided just a small part of it, that is the testing approach we decided to use in the present paper.

In this paper the data sets described in section 4 has been used to test PaperRank algorithm. Each data set τ is formed by the paper set Q_τ and graph $\mathcal{G}_\mathcal{C}{}^\tau$. It is possible to randomly choose a part of Q_τ, that can be used as the reference list $\mathcal{R}_{\bar{p}}$ of an hypothetical current paper $\bar{p}$ about topic τ. So the remaining part of Q_τ will be the "target set", and it can be denoted as T_τ. Due to the way we build Q_τ, the target set, obtained by splitting Q_τ, is composed by papers which are related to topic τ but not yet present in the current reference list $\mathcal{R}_{\bar{p}}$. A good scoring algorithm have to assign high scores to papers contained in T_τ. Following this philosophy we tested PaperRank algorithm on 9 data set we defined. We also compared performances with respect to different T_τ sizes, in particular, for every considered τ we randomly

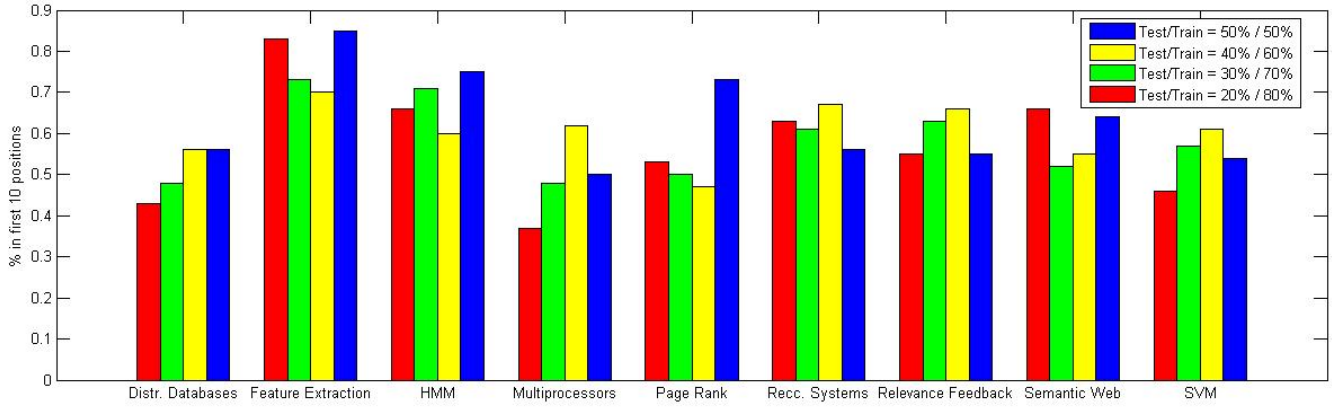

Figure 1. Percentage of interesting papers ranked in the first 10 positions for every data set and with increasing test set dimension.

chose 20%, 30%, 40% and 50% of papers contained in each Q_τ to be part of the corresponding T_τ. Moreover experiments have been done in a 10-fold cross-validation mode, in fact for every τ and for every target set/reference list proportion we randomly shuffled Q_τ 10 times and each time we obtained different T_τ (and $\mathcal{R}_{\bar{p}}$). Now we ran PaperRank for every experimental setup, then we measured the absolute ranking position of papers in every T_τ and counted the number of papers in T_τ which are present in the first 10, 20 and more positions. Results have been very promising and they are shown in figure 1 with respect to the first 10 positions, we do not report plots for the percentage of interesting papers ranked in the first 20 and more positions because PaperRank has been able to rank **100%** of the papers in T_τ within first 20 positions. On the y-axes there is the percentage (1 correspond to 100%) of papers in T_τ placed by PaperRank within first 10 positions, these values are obtained by averaging 10 fold-cross validation corresponding percentages, one for each trial. Note that in most of the cases we obtain values higher than 50%, that is true for the biggest T_τ too. It shows that we need just a small set of initial reference papers $\mathcal{R}_{\bar{p}}$ in order to rank properly the other papers in the Citation Graph.

6. Conclusions and Future Work

In this paper, we present a random–walk based scoring algorithm, which can be used to recommend papers according to a small set of user selected relevant articles. We tested the algorithm on a data set extracted by ACM Portal Digital Library and it performed very well, in fact target papers obtained a ranking within the first 20 result positions. Future research topics include the experimentation of the algorithm on other digital libraries. Moreover we wish to add to the new version of PaperRank the capability of using also negative examples, in this case a set of papers selected by the user that are not relevant for the topic, in fact it could be really valuable to introduce user feedback, after the first round of suggestions, into the paper recommendation task.

References

[1] N. Agarwal, E. Haque, H. Liu, and L. Parsons. Research paper recommender system: A subspace clustering approach. In *International Conference on Web-Age Information Management (WAIM) 2005*, 2005.

[2] Z. Gyongyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with trustrank. Technical report, Stanford University, 2004.

[3] T. Haveliwala. Topic-sensitive pagerank. In *Eleventh International Conference on World Wide Web*, 2002.

[4] S. Kamvar, T. Haveliwala, C. Manning, and G. Golub. Extrapolation methods for accelerating pagerank computations. In *Twelfth International Conference on World Wide Web*, 2003.

[5] A. Langville and C. Meyer. Deeper inside pagerank. *Internet Mathematics*, 1(3):335–380, 2003.

[6] S. McNee, I. Albert, D. Cosley, P. Gopalkrishnan, S. Lam, A. Rashid, J. Konstan, and J. Riedl. On the recommending of citations for research papers. In *ACM Conference on Computer Supported Cooperative Work*, 2002.

[7] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. *Technical report, Stanford University*, 1998.

[8] J. Schafer, J. Konstan, and J. Riedl. Electronic commerce recommender applications. *Journal of Data Mining and Knowledge Discovery*, January 2001.

[9] A. Woodruff, R. Gossweiler, J. Pitkow, E. Chi, and S. Card. Enhancing a digital book with a reading recommender. In *CHI '00*, 2000.