Learning Reputation in an Authorship Network

Charanpal Dhanjal, Stéphan Clémençon

LEARNING REPUTATION IN AN AUTHORSHIP NETWORK

CHARANPAL DHANJAL AND STÉPHAN CLÉMENÇON

Télécom ParisTech, 46 rue Barrault, 75634 Paris Cedex 13, France

Abstract. The problem of searching for experts in a given academic field is hugely important in both industry and academia. We study exactly this issue with respect to a database of authors and their publications. The idea is to use Latent Semantic Indexing (LSI) and Latent Dirichlet Allocation (LDA) to perform topic modelling in order to find authors who have worked in a query field. We then construct a coauthorship graph and motivate the use of influence maximisation and a variety of graph centrality measures to obtain a ranked list of experts. The ranked lists are further improved using a Markov Chain-based rank aggregation approach. The complete method is readily scalable to large datasets. To demonstrate the efficacy of the approach we report on an extensive set of computational simulations using the Arnetminer dataset. An improvement in mean average precision is demonstrated over the baseline case of simply using the order of authors found by the topic models.

1. Introduction

Identifying experts is a valuable task for finding coauthors for a new research project or grant, assigning reviewers for the peer-review of an article or employing consultants. In so-called Reputation Systems [23] one has explicit ratings of reputation such seller feedback provided on the eBay online auction site. Here we address the more challenging problem of estimating the reputation of authors in a network of authors and their publications. In a general sense, one must first evaluate the domain(s) of authors and then grade their expertise by the number and quality of publications in peer-reviewed journal and conferences.

The particular problem under study is stated in a more formal setting as follows. An undirected graph $G = (V, E)$ is composed of vertices $\{v_1, \ldots, v_n\} = V$ and edges $E \subseteq V \times V$ in which vertices represent authors and edges represent connections between the authors, for example common mediums of influence such as coauthorship or citation. Each vertex has a list of articles associated with it, representing an author's publications. The first question is how can one find all authors who have worked in a given domain $D_i \subset V$ based on their publications. Next, consider a class of scoring functions over the vertices in D_i, $f \in \mathcal{F}$, and an unordered set of top k vertices $S_i = \{v_{x_1}, u_{v_2}, \ldots u_{v_k}\} \subset D_i$. Our task is to find a ranking function close to the *oracle* f^* for which the top k ranked elements are

E-mail address: {charanpal.dhanjal, stephan.clemencon}@telecom-paristech.fr.

Date: November 25, 2013.

Key words and phrases. reputation assessment, expert finding, graph centrality, rank aggregation.

identical to S_i. Thus, the learning problem is to identify the characteristics of a reputable author in domain D_i in the space of functions $\mathcal{F}$.

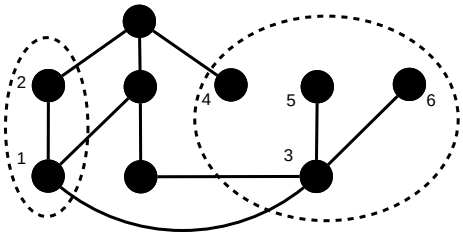

FIGURE 1. A graph of authors and connections, with the authors in the domain of interest circled. If we take the number of edges incident to a vertex as a measure of its expertise then we can see that author 3 has the most expertise as he/she has 3 edges to other authors in the same domain.

To tackle this problem we first study the titles and abstracts corresponding to the articles written by each author using Latent Semantic Indexing (LSI, [6]) and Latent Dirchlet Allocation (LDA, [4]). These are two popular and effective topic modelling algorithms, and readily scalable to the large dataset typically in use. This step identifies authors within the field of interest. We then construct a coauthorship graph using the authors found in this step (see Figure 1), and use a variety of centrality measures to score and rank vertices in that particular domain. Furthermore, to leverage the rankings we examine rank aggregation for the task of expert prediction. The chief novelty of this paper is the use of efficient topic modelling approaches and state-of-the-art graph-based algorithms in combination with rank aggregation to study this problem.

We start by describing our approach to the domain identification of the authors in Section 2. Following, we outline a number of centrality measures in graphs and how the ranking of vertices of these measures can be aggregated in Sections 3 and 4 respectively. Section 5 reviews related work in this area, and then we present computational results on a large author-publication dataset in Section 6. A summary and some perspectives are given in the final section.

2. Domain Identification

It is a common problem in information retrieval to recover documents corresponding to a particular subject and here we briefly review LSI and LDA and their online variants for this purpose.

Imagine that one has a set $T = \{d_1, \ldots, d_m\}$ of documents and we wish to discover the subset of those documents in a particular domain. The first step is to preprocess the words using a Porter Stemmer [28] to amalgamate words with the same base such as "learning" and "learned". One then finds a *bag of ℓ-grams* representation of the documents which is essentially a count of each sequence of ℓ consecutive words in the documents. This enables the identification of important word concurrences such as "singular values" which would be lost if a bag of words (1-gram) representation was used. This representation can be improved by removing stop words such as "and" and "the" which are common and do not convey

information useful for discrimination. At the end of this process we have a set of terms (n-grams) and documents, and occurrences of terms within the documents.

Rather than use this data directly it is often useful to represent documents using the *term frequency/inverse document frequency* (TF-IDF, [24]) representation. Assume that there are m documents and k_j appears in n_i of them so that $\mathbf{F}_{ij}$ is the number of times k_i appears in document d_j. The normalised term frequency is defined as:

$$\mathrm{TF}_{i,j} = \frac{\mathbf{F}_{ij}}{\max_z \mathbf{F}_{zj}},$$

where the maximum is computed over all frequencies $\mathbf{F}_{zj}$ for document d_j. Frequent keywords may not be useful and hence one also uses inverse document frequency, defined for a keyword k_i as

$$\mathrm{IDF}_i = \log \frac{|T|}{n_i}.$$

The TF-IDF weight for a keyword k_i in document d_i is then $\mathbf{X}_{ji} = \mathrm{TF}_{i,j} \times \mathrm{IDF}_i$ and each document can be represented using a vector of keyword weights. In this way similar vectors correspond to similar documents.

In LSI, a partial Singular Value Decomposition (SVD, [11]) is performed on the TF-IDF matrix $\mathbf{X}$ to determine relationships between the terms and semantic concepts represented in the text. The SVD of $\mathbf{X} \in \mathbb{R}^{m \times \ell}$ is the decomposition

$$\mathbf{X} = \mathbf{P}\mathbf{\Sigma}\mathbf{Q}^T,$$

where $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_r]$, $\mathbf{Q} = [\mathbf{q}_1, \ldots, \mathbf{q}_r]$ are respective matrices whose columns are left and right singular vectors, and $\mathbf{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_r)$ is a diagonal matrix of singular values $\sigma_1 \geq \sigma_2 \geq, \ldots, \geq \sigma_r$, with $r = \min(m, n)$. The matrix $\mathbf{P}$ can be thought of a mapping from documents to a semantic concept, $\mathbf{Q}$ is a mapping from terms to concepts and $\mathbf{\Sigma}$ is a scaling of the columns and rows respectively of these matrices. By taking the partial SVD one chooses the singular values and vectors corresponding to the largest k singular values, denoted by $\mathbf{P}_k, \mathbf{\Sigma}_k$ and $\mathbf{Q}_k$ respectively. This truncation has the effect of retaining the important concepts whilst removing noise in the concept space of $\mathbf{X}$. Notice that $\mathbf{X}$ is typically a sparse matrix and hence one can use efficient methods for computing the SVD such as Lanczos or Arnoldi (e.g. PROPACK [17]) or randomised methods [12]. In the later experiments we use the multipass stochastic online LSI algorithm presented in [20].

LDA is a generative model that explains a set of documents using a small set of topics. It assumes a set of k topics about the set of documents T. Each topic is drawn from a Dirichlet distribution $\beta_\ell \in \mathrm{Dirichlet}(\eta)$. For each document d_j one draws a distribution over topics $\theta_{d_j} \in \mathrm{Dirichlet}(\alpha)$. For each word t_i in the document one draws a topic index $z_{d_j,t_i} \in \{1, \ldots, k\}$ with weights $z_{d_j,t_i} \in \theta_{d_j}$. The observed word is then drawn from $t_{ji} \in \beta_{z_{d_j,t_i}}$. To infer the distributions in this model, one uses a variational Bayes approximation of the posterior distribution. In our later computational work, we use the online variant of LDA given in [14].

One approach to evaluate the similarity of a query to the training documents is to map the query and documents to the LSI or LDA space and find the highest cosine of the angle between them, known as *cosine similarity*. Note that the cosine of the angle between two vectors $\mathbf{a}$ and $\mathbf{b}$ is given by $\cos(\theta) = \mathbf{a}^T\mathbf{b}/(\|\mathbf{a}\|\|\mathbf{b}\|)$. If we

fix a threshold γ and find all documents with $\cos(\theta) > \gamma$ and then the corresponding authors, we have two effective methods of identifying authors in the query domain.

3. Ranking Experts

Once we have a collection of authors who have published under a particular domain, we can extract the corresponding coauthorship graph and use this graph to rank authors by their expertise. To do so we draw upon state-of-the-art results in graph structure analysis and rank aggregation. The key idea is to construct $\mathcal{F}$ in such a way that we encapsulate the main characteristics of reputation. For that reason, we consider six measures: *influence maximisation* [15], *PageRank*, *hub score*, *closeness centrality*, *degrees*, and *betweenness*, and motivate their use.

3.1. **Influence Maximisation.**

Influence maximisation is an intuitive way to measure the reputation in an authorship graph. To find the most influential vertices we first introduce the concept of *graph percolation* in which vertices within a graph have a binary state: either active or inactive. A percolation process decides how activation spreads within the graph. The problem of influence maximisation is to find the k vertices which result in the largest total spread of activation at the end of the process[1]. In epidemic spread, for example, finding the most influential vertices may help to devise effective control strategies.

A binary percolation process P computes in an iterative manner which vertices will be active in the next iteration based on the edges and those that are currently active, and continues until no more activations occur. Let $\sigma_P(G, A)$ be the number of active vertices at the end of a percolation process defined by P, over graph G and with an initial set of active vertices A. Figure 2 demonstrates a percolation process within a simple graph. A commonly studied percolation process is the *Independent Cascade model*. For this model, there is a probability p_{ij} on an edge from v_i to v_j which allows a random decision to be taken for the activity of v_j given that v_i is active. The percolation then proceeds as follows: at time step t when a vertex v_i first becomes active it is given a single chance to activate each of its neighbours $n(v_i)$ according to the edge probabilities. If v_i succeeds then the corresponding vertices become active in the next time step. If not then no further attempts are made in subsequent rounds.

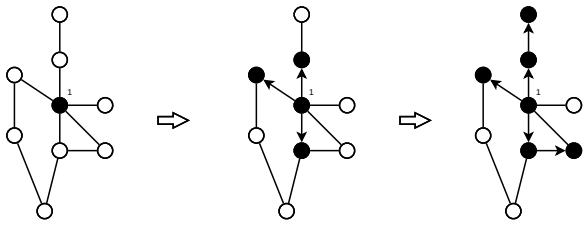

Figure 2. Percolation within a graph. Vertices in black are active and activation spreads to other vertices in an iterative manner. The initial active set is $A = \{1\}$ and $\sigma_P(G, A) = 6$ in this case.

[1]A percolation process can be said to be concluded when there are no additional activations/disactivations.

The problem of computing the maximal influence is: given a graph G and process P which subset $A \subseteq V$ of $k = |A| < n$ vertices should be chosen to ensure maximal activation $\sigma_P(G, A)$ at the end of the process? This optimisation corresponds to a combinatorial problem that is NP-hard in general. The authors of [15] tackle this problem for the Linear Threshold and the Independent Cascade models, and provide greedy algorithms to maximise influence. The key observation is that the influence function $\sigma_P(G, A)$ is *submodular* for these percolation models. A submodular function is one in which there are diminishing returns. Provided that $\sigma_P(G, A)$ is submodular and monotone, a simple $O(nk)$ greedy algorithm exists for choosing the most influential vertices, see [15] for more details. A faster variant of this algorithm, known as Cost Effective Lazy Forward selection (CELF), is given in [18] in which computational savings are made by using the submodularity property and the previous influences of each vertex at each stage.

3.2. **Graph Centrality Measures.** Closely related to the influence of vertices within a graph is the idea of graph centrality, and here we outline several useful measures. We begin with PageRank which was designed to rank Web pages using the graph of hyperlinks, however since then has been applied to many other types of graph.

The key intuition of PageRank is that a hyperlink to a page counts as a vote of support, and hyperlinks from "important" pages are weighted higher than unimportant ones. In this sense Pagerank is defined recursively and depends on the PageRank metric of all pages that link to it (incoming links). A page that is linked to by many pages with high PageRank receives a high rank itself. The idea is linked closely to the concept of performing a random walk on a graph. For a directed graph G PageRank is defined as follows:

$$P(u) = \frac{1 - |V|}{D} + D \sum_{n_{in}(u)} \frac{P(v)}{|n_{out}(v)|},$$

where $n_{in}(u)$ is the set of all vertices with edges directed towards u, $n_{out}(u)$ is a set of vertices with edges directed from u and D is a damping factor between 0 and 1 which is used to enable the random walker to jump out of cycles.

A precursor to PageRank is the hub score (HS, [16]) of the vertices in a graph. In short, a hub is a catalogue of information that points to authority pages. A highly rated hub points to many authority pages, and a good authority is referenced by many hubs. To compute the hub score we initialise two scores $h(v) = 1$ and $a(v) = 1$ for all $v \in V$. To update these scores one performs mutual recursion as follows:

$$a(v) = \sum_{u \in n_{in}(v)} h(u),$$

and

$$h(v) = \sum_{u \in n_{out}(v)} a(u),$$

and one normalises uses the 2-norm of the corresponding scores after each iteration to allow convergence.

Another useful measure which we shall use for our analysis is betweenness, which is the number of times a shortest path passed through a certain vertex. Intuitively,

this quantifies the importance of the vertex in terms of linking other vertices. It is defined more formally as

$$B(u) = \sum_{v,w \in V \setminus u} \frac{\sigma_{vw}(u)}{\sigma_{vw}},$$

where $\sigma_{vw}(u)$ is the number of shortest paths from v to w that pass through u and σ_{vw} is the total number of paths between v and w.

Next we look at closeness centrality [10] which is a measure of how close all other vertices are to the current one. One way of considering this type of centrality is how long it would take for information to spread to other vertices in the network, and hence it makes sense for the type of network considered. The closeness centrality is defined as follows:

$$C(u) = \frac{1}{\sum_{v \in V \setminus u} d(u,v)},$$

where $d(u,v)$ is the distance between u and v. Hence closeness centrality is the inverse of the average length of the shortest paths to all other vertices in the graph.

Finally we also use the degree of vertices as a measure of their centrality where the degree is simply the number of edges incident to a vertex.

4. Aggregating Rankings

In this section we show how to combine the rankings given by the above centrality measures. Rank aggregation has been studied using Borda count [1], median rank aggregation [9] and Markov Chains [8]. Here we detail the popular Markov chain method of [8]. The principal advantages of Markov Chain based rank aggregation methods is that they can work with partial lists, are efficient, and shown to outperform other methods in [22].

The setup for rank aggregation is described as follows. Consider a set of elements D and an ordered list τ whose elements are a subset of the elements of D, $\tau = [x_1 \geq x_2 \geq \ldots \geq x_{|\tau|}]$ with $x_i \in D$, where $\geq$ is an ordering relation on D. If τ contains all the elements in D it is called a *full list* otherwise it is a *partial* or *top-k list* for which only the first k elements are present. In the case of rank aggregation we have a number of ranked lists $\tau_1, \ldots \tau_n$ and we also have an ideal ranking τ^*. The goal is to find an aggregation function $\phi : \tau_1, \ldots, \tau_\ell \mapsto \mathbf{x}$, where $\mathbf{x}$ is a score vector for all entries, such that the ordering according to $\mathbf{x}$ is as close to τ^* as possible.

In the MC_2 model of [8] we construct a Markov chain which is a state transition machine in which a transition to a new state is dependent only on the current one. Each item $x_i \in D$ is represented by a state and then a ranking list τ_j is selected randomly such that x_i is an element of τ_j. One then selects a random state uniformly from the elements in τ_j which are not ranked lower than x_i. More formally, define the kth *transition matrix* as $\mathbf{P}^{(k)}$ such that $\mathbf{P}^{(k)}_{ij}$ is the conditional probability of state x_j given state x_i and ranking list τ_k. We have

$$\mathbf{P}^{(k)}_{ij} = \begin{cases} \frac{1}{q} & x_j \geq x_i \\ 0 & \text{otherwise,} \end{cases}$$

where $q = |\{x_j | x_j \geq_{\tau_k} x_i\}|$. The final transition matrix is given by the mean of the individual matrices for each ranked list, $\mathbf{R} = \frac{1}{\ell} \sum_{i=1}^{\ell} \mathbf{P}^{(i)}$. The score vector is

then computed as the stationary distribution $\mathbf{x} = \mathbf{R}^T\mathbf{x}$ such that $\sum_{i=1}^{|D|} \mathbf{x}_i = 1$ and $\mathbf{x}_i > 0$ for $i = 1, \ldots, |D|$.

5. Related Work

A key driver of expert recommendation in recent years has been the expert finding task in the TREC Enterprise track in 2005 [5]. The data present in this task includes email on public mailing lists, code and web pages extracted from the World Wide Web Consortium (W3C) sites in June 2004. The task consisted of ranking 1092 experts from the 331,037 documents available. Two types of model appeared from this task [2, 3]: candidate and document models. In candidate models, one builds a textual representation of the experts and ranks them based on a query. In document based models, one first finds documents relevant to the query and then locates associated experts. In our work we use a mixture of these two ideas.

For academic networks, the topic of discovering experts using graphs has been studied in conjunction with the Arnetminer [27] academic database and social network in [25]. Unlike our work which incorporates topic learning as part of the process, the authors label each member of the social network with a pre-assigned topic vector, and then try to measure influence in the network. A Topical Affinity Propagation (TAP) model is proposed which optimises the topic-level social influence on a network.

In [7] which focuses on the expert seeking task on the Digital Bibliography and Library Project (DBLP) dataset, three models are proposed, namely a Bayesian statistical language model, a topic-based model and a hybrid one. One of the key parts of the model is computing prior probabilities of authors using citation data which is used in conjunction with the language-based ranking of authors. Note that each article is augmented with similar documents extracted from Google Scholar which we do not use in our experiments. The experts are manually graded on a scale from $0 - 3$ and the learning system is tested against these ratings with favourable results to related algorithms in [19, 29]. In [19], the authors augment the DBLP data with Google search results as well as publication rankings from Citeseer.

A more scientometric approach is given in [13] which uses measures such as bibliographic coupling (two authors A1 and A2 are linked if they cite the same references) and author cocitation to recommend similar authors. In this paper there is not a focus on finding the most influential authors. Along the same lines, there are a number of other metrics one could use to rate the reputation of authors in a particular domain such as h-index and impact factor. The h-index is the largest number h such that h publications have at least h citations and impact factor[2] of a journal is the average number of citations in the two preceding years.

6. Simulations

In this section we evaluate the expertise ranking algorithms by comparing them to the baseline case of using the author order given solely using topic modelling and not any graph-based ranking scheme. The Arnetminer dataset [26] which is based on DBLP, is used. This dataset is a list of articles in computer science, along with their authors, the publication venue, year and paper abstracts and citations for

[2]Note that although this is the generally accepted definition of impact factor, alternative definitions exist.

Abbreviation	Category	Experts
BS	Boosting	57
CV	Computer Vision	215
CRY	Cryptography	174
DM	Data Mining	351
IE	Information Extraction	91
IA	Intelligent Agents	30
ML	Machine Learning	76
NLP	Natural Language Processing	54
NN	Neural Networks	122
OA	Ontology Alignment	56
PL	Planning	26
SW	Semantic Web	412
SVM	Support Vector Machines	111

TABLE 1. Summary of the information about experts over the Arnetminer dataset.

some articles. We use version 5 of the dataset which contains 1,572,277 papers with 529,499 abstracts and is generated on 21/2/2011. We want to observe the accuracy of our expert finding approach on this dataset in conjunction with experts in 13 fields suggested on the Arnetminer web site. The expert lists are generated using the Program Committee members of well know conferences/workshops and the members on sites specific to a particular field, for example on `www.boosting.org`. The lists are unordered and "noisy" due to their nature, however still useful for the purposes of evaluation. We use the experts in the fields listed in Table 1. All experimental code is written in Python and we use the Gensim library [21] for topic modelling.

Before predicting a set of experts, we perform model selection for our learning algorithm and hence split the experts into a 50:50 training/test set. The word vectoriser is set up as described above on the title and abstracts of articles for 1 and 2-grams with term counts included if the term frequency is in at least a proportion $\rho \in \{10^{-3}, 10^{-4}\}$ of the total number of documents. Since the documents being processed are typically small we use binary indicators for terms. For LSI we take the SVD of this matrix using the randomised SVD method of [12] with an exponent of $q = 2$ and oversampling of $p = 100$ and take $k \in \{100, 200, \ldots, 600\}$. After this stage, we find similar documents to a query term (the field) using the method outlined above and a cosine similarity threshold of $\gamma \in \{0.0, 0.1, \ldots, 0.9\}$. Each author in this set is then scored by summing the cosine similarity of their articles and we take the first x authors according to their score (denote this set of authors as U). In this case x is 10 times the number of training experts. For LDA we choose the number of topics in $k \in \{100, 200, \ldots, 600\}$ and otherwise use an identical process. The optimal model is selected by choosing parameters which result in the largest number of training experts across the complete set of 13 fields.

After model selection, the authors in U are positioned in a coauthorship graph in which an edge exists only if two authors $u, v \in U$ have collaborated. Edges in this graph are weighted according to the number of articles written by the corresponding pair of authors. We compute each centrality metric over the weighted graph and

use the inverse of the weights for the computation of betweenness and closeness centrality. This implies for example that two authors who have collaborated 5 times have an edge weight between them of 1/5 and thus are more likely to be on a shortest path than adjacent authors who have collaborated less frequently. For influence maximisation we obtain the 100 most influential authors using 100 repetition of the independent cascade model with transition probability $p = 0.05$. We also record the order of authors given by the topic modelling approaches and that given by sorting authors according to the total number of citations for articles in U.

The rankings are evaluated using the test set with the Mean Average Precision (MAP) at N metric. The *precision* at N is the number of experts in the first N items of the ranked list of authors divided by N or equivalently

$$p@N = \frac{tp}{tp + fp},$$

where tp is the number of true positives and fp is the number of false positives. Precision falls within the range $[0, 1]$ with 1 signifying that all items at the top of the list are experts. The *average precision* is the average of all precisions for all of the experts:

$$ap@N = \frac{\sum_{i=1}^{N} p@i \times \text{rel}(i)}{R},$$

where $\text{rel}(i)$ is an indicator function which is 1 for relevant experts and R is the number of experts. MAP is simply the average precision over all the queries. We look at MAP for $N \in \{5, 10, \ldots, 50\}$. Note that to compute these precisions for the test experts we remove the training experts from the rankings, and vice versa. After computing the graph and topic-based rankings, we use the MC2 algorithm of [8] to aggregate rankings from each field in a greedy fashion: using the training experts we pick the ranking with the best $ap@20$ score then choose additional rankings that give the best marginal gain until no improvement is obtained.

The proportion of training experts covered by LSA topic modelling is approximately 0.381 using $k = 500$, $\rho = 10^{-4}$ and $\gamma = 0.3$. This indicates the difficulty of finding relevant authors using this dataset. It is worth noting that amongst the complete set of experts, a mean proportion of 0.4 of their articles also have abstracts and we believe results could be improved with a higher proportion of abstracts. LDA was less effective than LSI at recovering the training experts during model selection with a mean coverage of 0.318 over all the fields, using $k = 400$, $\rho = 10^{-3}$ and $\gamma = 0.4$.

Table 2 shows the MAP values on the test experts in conjunction with the authors returned using LSI. In this table we see that the strongest single method is betweenness followed by the citation and topic orders. A possible reason for the efficacy of betweenness is that reputable authors are also social and attract collaborations and hence participate in many shortest paths in the coauthorship graph. Citation is a good indication of reputation since citations are often positive votes about the value and quality of a paper. We observed that the relative performances of the rankings varied between fields. In Ontology Alignment for example the $ap@50$ score for the citation ranking was 0.08 versus 0.163 for betweenness. A particularly challenging field was Neural Networks which was ranked best using influence with $ap@5 = 0.05$ and $ap@50 = 0.058$. A possible reason is that Neural Networks covers a large range of topics both in biology and machine learning. In contrast, we

N	Topic	Cit.	Bet.	Cls	PR	Dgr	Inf.	HS	MC_2
5	0.152	0.170	0.196	0.130	0.129	0.065	0.112	0.000	0.250
10	0.109	0.133	0.122	0.094	0.098	0.046	0.077	0.004	0.176
15	0.092	0.138	0.108	0.095	0.090	0.038	0.075	0.007	0.165
20	0.087	0.141	0.110	0.092	0.086	0.036	0.077	0.008	0.161
25	0.088	0.139	0.107	0.093	0.084	0.033	0.080	0.009	0.157
30	0.087	0.143	0.109	0.092	0.086	0.032	0.083	0.013	0.162
35	0.086	0.146	0.118	0.096	0.085	0.033	0.085	0.013	0.165
40	0.088	0.147	0.122	0.098	0.086	0.033	0.085	0.018	0.163
45	0.087	0.150	0.124	0.098	0.088	0.034	0.085	0.019	0.166
50	0.089	0.152	0.126	0.099	0.089	0.034	0.085	0.022	0.168

TABLE 2. MAP values at each value of N for the rankings using LSI for topic modelling. Abbreviations: Topic (LDA order), Cit. (citation order), Bet. (betweenness), Cls (closeness), PR (PageRank), Dgr (degree), Inf. (influence), HS (hub score) and MC_2 is the Markov chain model of [8].

obtained $ap@5 = 0.76$ with Data Mining using betweenness and $ap@5 = 0.76$ for Information Extraction using the citation ranking. A significant improvement is gained by aggregating the rankings of the topic modelling order, citation ranking and betweenness ranking. We see that $ap@5$ improves from 0.196 using betweenness to 0.250 with MC_2 and $ap@50$ improves from 0.152 with citations to 0.168.

Table 3 shows the corresponding results using LDA for topic modelling. The best performing rank methods were closeness and PageRank with $ap@5$ scores of 0.137 and 0.134 respectively. They improve significantly over the baseline topic order score (denoted "Topic" in the table) of 0.111. Interestingly, the citation-based ranking does not perform well in this case because more irrelevant, but highly cited, authors are found in U relative to LSI. The rank aggregate of our greedy MC_2 algorithm gives a slight improvement over the using closeness centrality. When considering individual fields, the comparison to LSI is more complicated. In the case of Semantic Web for example, PageRank gives $ap@5$ of 0.483 using LSI and 0.8 using LDA. As with LSI however, LDA scores poorly when the domain is Neural Networks.

7. Conclusions

We proposed an approach for finding experts in a set of authors and their publications. The method uses well-known topic modelling algorithms LSI and LDA to identify authors within the query domain, and then construct a coauthorship graph using these authors. In turn, the graph is used for the extraction of expert rankings using a number of centrality measures. Furthermore, we explore the use of a rank aggregation approach to leverage the orderings and improve rankings. Computational results on the large Arnetminer dataset show that the citation ranking and betweenness in conjunction with LSI for topic modelling provide the most precise single-rank estimates of experts, however these rankings are improved significantly using aggregations.

N	Topic	Cit.	Bet.	Cls	PR	Dgr	Inf.	HS	MC_2
5	0.111	0.065	0.108	0.137	0.134	0.033	0.082	0.000	0.142
10	0.076	0.060	0.086	0.101	0.084	0.028	0.066	0.006	0.109
15	0.074	0.058	0.075	0.098	0.083	0.027	0.061	0.009	0.108
20	0.070	0.057	0.070	0.105	0.081	0.026	0.057	0.014	0.108
25	0.074	0.057	0.072	0.102	0.077	0.028	0.057	0.016	0.104
30	0.076	0.061	0.071	0.104	0.077	0.028	0.058	0.021	0.107
35	0.078	0.061	0.074	0.106	0.076	0.030	0.057	0.023	0.111
40	0.080	0.061	0.077	0.110	0.075	0.033	0.059	0.025	0.111
45	0.078	0.062	0.076	0.111	0.075	0.033	0.059	0.028	0.111
50	0.080	0.065	0.077	0.111	0.077	0.034	0.060	0.032	0.110

TABLE 3. MAP values at each value of N for the rankings using LDA for topic modelling. Abbreviations: Topic (LDA order), Cit. (citation order), Bet. (betweenness), Cls (closeness), PR (PageRank), Dgr (degree), Inf. (influence), HS (hub score) and MC_2 is the Markov chain model of [8].

ACKNOWLEDGEMENTS

This work is funded by the Eurostars ERASM project.

REFERENCES

[1] J. A. Aslam and M. Montague. Models for metasearch. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 276–284. ACM, 2001.

[2] P. Bailey, N. Craswell, A. P. de Vries, and I. Soboroff. Overview of the trec 2007 enterprise track. 2007.

[3] K. Balog, I. Soboroff, P. Thomas, P. Bailey, N. Craswell, and A. P. de Vries. Overview of the trec 2008 enterprise track. 2008.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[5] N. Craswell, A. P. de Vries, and I. Soboroff. Overview of the trec-2005 enterprise track. In *TREC 2005 conference notebook*, pages 199–205, 2005.

[6] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[7] H. Deng, I. King, and M. R. Lyu. Formal models for expert finding on dblp bibliography data. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 163–172. IEEE, 2008.

[8] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622. ACM, 2001.

[9] R. Fagin, R. Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 301–312. ACM, 2003.

[10] L. C. Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1979.

[11] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHUP, 2012.

[12] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[13] T. Heck, O. Hanraths, and W. G. Stock. Expert recommendation for knowledge management in academia. *Proceedings of the American Society for Information Science and Technology*, 48(1):1–4, 2011.

[14] M. Hoffman, F. R. Bach, and D. M. Blei. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, pages 856–864, 2010.

[15] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146, 2003.

[16] J. M. Kleinberg. Hubs, authorities, and communities. *ACM Computing Surveys (CSUR)*, 31(4es):5, 1999.

[17] R. M. Larsen. Lanczos bidiagonalization with partial reorthogonalization. *DAIMI Report Series*, 27(537), 1998.

[18] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 420–429. ACM, 2007.

[19] J. Li, J. Tang, J. Zhang, Q. Luo, Y. Liu, and M. Hong. Eos: expertise oriented search using social networks. In *Proceedings of the 16th international conference on World Wide Web*, pages 1271–1272. ACM, 2007.

[20] R. Řehuřek. Subspace tracking for latent semantic analysis. In *Advances in Information Retrieval*, pages 289–300. Springer, 2011.

[21] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. `http://is.muni.cz/publication/884893/en`.

[22] M. E. Renda and U. Straccia. Web metasearch: rank vs. score based rank aggregation methods. In *Proceedings of the 2003 ACM symposium on Applied computing*, pages 841–846. ACM, 2003.

[23] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman. Reputation systems. *Communications of the ACM*, 43(12):45–48, 2000.

[24] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of*. Addison-Wesley, 1989.

[25] J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 807–816. ACM, 2009.

[26] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998. ACM, 2008.

[27] J. Tang, J. Zhang, D. Zhang, L. Yao, C. Zhu, J.-Z. Li, et al. Arnetminer: An expertise oriented search system for web community. In *Semantic Web Challenge*, 2007.

[28] C. J. Van Rijsbergen, S. E. Robertson, and M. F. Porter. *New models in probabilistic information retrieval*. Computer Laboratory, University of Cambridge, 1980.

[29] J. Zhang, J. Tang, and J. Li. Expert finding in a social network. In *Advances in Databases: Concepts, Systems and Applications*, pages 1066–1069. Springer, 2007.