2010 IEEE International Conference on Data Mining Workshops

Reviewer Profiling Using Sparse Matrix Regression

Evangelos E. Papalexakis*, Nicholas D. Sidiropoulos*, Minos N. Garofalakis*
*Dept. of ECE, Tech. Univ. of Crete, 73100 Chania - Crete, Greece
Email: (vagelis, nikos)@telecom.tuc.gr, minos@acm.org

***Abstract*—Thousands of scientific conferences happen every year, and each involves a laborious scientific peer review process conducted by one or more busy scientists serving as Technical/Scientific Program Committee (TPC) chair(s). The chair(s) must match submitted papers to their reviewer pool in such a way that i) each paper is reviewed by experts in its subject matter; and ii) no reviewer is overloaded with reviews or under-utilized. Towards this end, seasoned TPC chairs know the value of reviewer and paper profiling: summarizing the expertise / interests of each reviewer and the subject matter of each paper using judiciously chosen domain-specific keywords. An automated profiling algorithm is proposed for this purpose, which starts from generic / noisy reviewer profiles extracted using Google Scholar and derives custom conference-centric reviewer and paper profiles. Each reviewer is expert on few sub-topics, whereas the pool of reviewers and the conference may collectively need many more keywords for appropriate specificity. Exploiting this sparsity, we propose a sparse matrix factorization approach in lieu of classical SVD-based LSI or NMF-type approaches. We illustrate the merits of our approach using real conference data, and expert scoring of the assignments by a seasoned TPC chair in the area.**

***Keywords*-Reviewer profiling, text mining, latent semantic indexing, singular value decomposition, non-negative matrix factorization, sparse regression, lasso**

I. Introduction

Thousands of scientific conferences happen every year and each involves a laborious scientific peer review process conducted by one or more busy scientists serving as Technical / Scientific Program Committee (TPC) chair(s). The chair(s) must match submitted papers to the reviewer pool in such a way that i) each paper is reviewed by experts in its subject matter, preferably covering all methodological and application-oriented aspects of the submission; and ii) no reviewer is overloaded with reviews, or under-utilized. Usually there are pre-agreed quotas on the maximum number of papers that a volunteer is willing to review for the given conference. Seriously under-utilizing a reviewer is unfair with respect to reviewing credit, and it may discourage junior reviewers.

The paper to reviewer assignment requires prior knowledge about each reviewer's expertise and current research interests. TPC chairs often work in an 'assign-as-you-go' fashion, inspecting each paper on-the-fly and trying to find reviewers with spare capacity in a sequential fashion. This process gets more and more complicated, demanding significant trial-and-error as one moves on and reviewers become loaded. A low- to mid-size conference with several hundred submissions may require a week of hard work from the TPC chair - let alone major events with several thousand submissions.

As an intermediate step, experience shows that it helps to summarize the expertise / interests of each reviewer and the subject matter of each paper using judiciously chosen domain-specific keywords. Reviewer and paper profiling is a domain-sensitive process that requires considerable expertise and serious work by the domain expert - the busy TPC chair. Papers often come tagged with keywords selected by the authors, albeit these are not always sufficiently specific for the purposes of review assignment, and they may be biased. Reviewer profiling is even more difficult. Even if the chair is very familiar with the long-term track record of each reviewer, keeping tabs of recent activity and customizing profiles for the given conference is tedious. The chair can ask the reviewers to select from a list or suggest keywords to build their own profiles, but people tend to ignore their long-term expertise and over-emphasize current interests. The reason is that they prefer to sneak-preview what is happening in the areas they currently work on - this is one of the fringe benefits of peer reviewing. The net effect is that there is considerable work involved in producing fair and accurate reviewer and paper profiles that can really help in the review assignment process.

Bidding systems, in which reviewers are invited to bid on / score papers they are interested in reviewing, offer an alternative way of matching papers to reviewers (e.g., Microsoft's Academic Conference Management Service). Bidding is prone to the same bias as self-profiling, for obvious reasons; and it requires each reviewer to browse a long list of submitted papers. Many reviewers simply skip bidding for this reason, effectively passing this chore to the TPC chair.

In this contribution, we consider the problem of automatically building relevant reviewer and paper profiles for a given conference, without human expert intervention. A natural starting point for reviewer profiling is a reviewer's CV, but CVs are often outdated and come in many different formats, making them hard to analyze; and a person's 'weight' on a particular topic is hard to measure directly from the person's CV without additional information (e.g., citation counts). For this reason, we decided to use Google Scholar [2]. We developed a simple and robust parser to

978-0-7695-4257-7/10 $26.00 © 2010 IEEE
DOI 10.1109/ICDMW.2010.87

automatically retrieve information (publication titles, plus year and citation data) for each reviewer from Google Scholar. We then employ KEA [9], a TF/IDF-based tool, to extract an initial list of T terms from the *joint list* of reviewer publication titles and submitted paper titles. This way, each reviewer (or submitted paper) is initially represented by a $1 \times T$ row vector of zeros and ones, with ones indicating terms present in the reviewer's publications (resp. submitted paper's title). This first filtering step must be loose / forgiving (i.e., T must be large) to include terms for every entity. As a result, the extracted profiles are very noisy. Beyond the already implemented linguistic stemming, we need domain-specific stemming of terms into concepts. This is usually done using latent semantic indexing (LSI) [1]. That is, we stack all these row profiles in a matrix with T columns, and reduce the resulting matrix to $r << T$ components using singular value decomposition (SVD).

The *key methodological contribution* of this paper is the following. Experience dictates that each domain-specific concept should be formed from relatively few raw terms; i.e., any given concept should *not* contain most terms. It follows that each concept should be *sparse* in term space. Similarly, each reviewer (or paper) can only relate to a few latent concepts. That is because each reviewer is expert on a few sub-topics, whereas the pool of reviewers and submitted papers collectively need many more terms for appropriate specificity. It follows that each reviewer (or paper) should ideally be sparse in concept space - i.e., a linear combination of a few concepts. This *double sparsity* should be exploited in the reduction to latent space, yet SVD-based LSI [1] and other popular indexing schemes do not account for sparsity. We thus propose using a sparse matrix factorization approach in place of LSI or existing alternatives, such as non-negative matrix factorization [3]. We illustrate the merits of our approach using real conference data, and expert scoring of the assignments by a seasoned TPC chair in the area.

Related work to date addresses the reviewer assignment problem, see [8], [6] and references therein. Automated reviewer and paper profiling, which we deal with in this paper, can be viewed as an intermediate step towards optimized reviewer assignment. The method introduced in [6], for example, assumes the availability of proximity/affinity estimates between reviewers and papers; these can be obtained by computing inner products or Euclidean distances between reviewer and paper profile vectors, respectively.

The rest of this paper is structured as follows. In section II, we recall some basic matrix factorizations and introduce sparse matrix regression, a tool that we will use as a core component of our profiling algorithm in the sequel. In section III we present the proposed automatic profiling algorithm, starting from a program we developed for retrieving initial (generic / noisy) reviewer profiles from on-line resources, and explaining the various processing stages step by step. In section IV we illustrate the performance of the proposed algorithm using real conference data, and compare sparse matrix regression to non-negative matrix factorization in this context. Conclusions are drawn in section V.

II. Latent Factorization Using Sparse Matrix Regression

Latent Semantic Indexing [1] approximates high-dimensional data in a low-dimensional latent space by approximating the data matrix using a low-rank bilinear model

$$\mathbf{M} \approx \mathbf{A}\mathbf{B}^T$$

where rank($\mathbf{A}$) = rank($\mathbf{B}$) << rank($\mathbf{M}$). The columns of $\mathbf{A}$ are often called *loadings*, whereas those of $\mathbf{B}$ are called *scores*. Note that, for our particular application, $\mathbf{M}$ will be composed of two parts:

$$\mathbf{M} = \begin{bmatrix} \mathbf{R} \\ \mathbf{P} \end{bmatrix}$$

where $\mathbf{R}$ is the reviewer-by-term matrix, and $\mathbf{P}$ is the paper-by-Term matrix.

The above approximation is highly non-unique, because

$$\mathbf{M} \approx \mathbf{A}\mathbf{B}^T = \mathbf{A}\mathbf{Q}\mathbf{Q}^{-1}\mathbf{B}^T = (\mathbf{A}\mathbf{Q})\left(\mathbf{B}\mathbf{Q}^{-T}\right)^T,$$

for any invertible $\mathbf{Q}$. Uniqueness of factorization is not necessarily of interest, however, when the objective is to filter noise and bring out the essential latent structure. An optimal approximation in the least squares sense for a given rank is provided by the truncated Singular Value Decomposition (SVD), as originally proposed in [1]. The full SVD is given by $\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ where $\mathbf{U} \in \mathbb{R}^{I \times I}$ and $\mathbf{V} \in \mathbb{R}^{J \times J}$ are orthonormal matrices and $\mathbf{\Sigma} \in \mathbb{R}^{I \times J}$ is 'diagonal' ($\mathbf{\Sigma}(i,j) = 0, \forall j \neq i$) containing the non-negative *singular values* of $\mathbf{M}$ in non-increasing order on its diagonal. Truncating $\mathbf{U}$,$\mathbf{\Sigma}$,$\mathbf{V}$ to the first r singular components and arbitrarily splitting the singular values between the left and right factors yields an optimal rank-r approximation of $\mathbf{M}$ in the least-squares sense.

The SVD is an orthogonal decomposition that optimally compresses the data for a given rank. Uniqueness is important when we worry about interpretability of the latent structure, in which case we often seek meaningful constraints (e.g., element-wise non-negativity) on the matrix factors that can ensure unique or quasi-unique decomposition under certain conditions. Non-negativity does not ensure uniqueness in all cases (one can easily cook counter-examples), but it often yields interpretable results. For this reason, Non-Negative Matrix Factorization (NMF) [3] has recently become very popular in a number of applications, including latent semantic indexing.

Another property that is often plausible is *latent sparsity*, i.e., the latent loading and/or score vectors are sparse in the sense of having few non-zero elements. Sparsity is a

useful constraint when it reflects prior knowledge about the 'true' factors. In our particular context, for example, the reviewer profiles are expected to be sparse, as each reviewer can only be an expert on a few topics. Data collected for the reviewer from the web will typically be noisy, yet by imposing sparsity we hope to suppress the noise and keep only the important terms that match each reviewer's expertise. One-sided sparsity (where loadings *or* scores are sparse, but not both) has recently drawn much attention and found diverse applications in the literature; cf. the so-called lasso in [7]. Two-sided sparsity (both scores and loadings are sparse) is considered here and in somewhat different form in [5]. Sparse Matrix Regression (SMR) is a factorization that introduces additional ℓ_1 penalty terms to the standard least squares cost function. The role of these penalty terms is to steer the solution towards sparsity. Sparsity is naturally measured using the number of non-zero elements - the Hamming weight or ℓ_0 'norm'. The use of such counting measures, however, leads to NP-hard problems. In [7] and follow-up work it was shown that the ℓ_1 norm is a good substitute that is also amenable to efficient computation.

In its basic form, the one-sided (linear) SMR problem can be stated as follows:

$$\min_{\mathbf{B}}\{\| \mathbf{M} - \mathbf{A}\mathbf{B}^T \|_F^2 + \lambda|\mathbf{B}|_1\} \tag{1}$$

where $\mathbf{A}$ is assumed known or given,

$$\|\mathbf{M}\|_F = \sqrt{\sum_{i=1}^{I}\sum_{j=1}^{J} m_{i,j}^2}$$

and

$$|\mathbf{B}|_1 = \sum_{j=1}^{J}\sum_{\ell=1}^{k} |b_{j,\ell}|$$

A coordinate-descent method for solving (1) is listed as Algorithm 1, where $\mathbf{M}(:,j)$ ($\mathbf{M}(i,:)$) denotes the j-th column (resp. i-th row) of $\mathbf{M}$. Algorithm 1 implements lasso coordinate descent on a row-wise basis, and it is guaranteed to converge to the global optimum. Element-wise non-negativity can be easily incorporated by disabling the second (**else if**) assignment - it can be shown that this is optimal. In our application context, both $\mathbf{A}$ and $\mathbf{B}$ are presumed to be sparse, and neither is given. Each column of $\mathbf{B}$ (row of $\mathbf{B}^T$) corresponds to a latent group of keywords. Relatively few keywords should be used to form each latent group - thus $\mathbf{B}$ should be sparse. Each row of $\mathbf{A}$ is the representation of a reviewer or paper on the latent basis. Each reviewer can only be an expert in relatively few topics, therefore $\mathbf{A}$ should be sparse as well.

The two-sided (bilinear) SMR problem with a double sparsity penalty can be posed as

$$\min_{\mathbf{A},\mathbf{B}}\{\| \mathbf{M} - \mathbf{A}\mathbf{B}^T \|_F^2 + \lambda|\mathbf{A}|_1 + \lambda|\mathbf{B}|_1\} \tag{2}$$

Algorithm 1 Element-wise Coordinate Descent for one-sided (linear) SMR

Input: $\mathbf{M}$ of size $I \times J$, $\mathbf{A}$ of size $I \times \hat{k}$, λ
Output: $\mathbf{B}$ of size $J \times \hat{k}$

```
while convergence criterion is not met do
  for j = 1 : J do
    for f = 1 : k̂ do
      d = M(:, j) − AB(j, :)^T + A(:, f)B(j, f)
      a = A(:, f)
      if (a^T d − λ/2) > 0 then
        B(j, f) = (a^T d − λ/2) / ||a||_2^2
      else if (a^T d + λ/2) < 0 then
        B(j, f) = (a^T d + λ/2) / ||a||_2^2
      else
        B(j, f) = 0
      end if
    end for
  end for
end while
```

One may additionally impose non-negativity constraints when appropriate

$$\min_{\mathbf{A},\mathbf{B}}\{\| \mathbf{M} - \mathbf{A}\mathbf{B}^T \|_F^2 + \lambda|\mathbf{A}|_1 + \lambda|\mathbf{B}|_1\} \tag{3}$$
$$\text{subject to } a_{i,j} \geq 0 \text{ and } b_{i,j} \geq 0$$

The parameter λ is tunable and controls the sparsity vs. fidelity trade-off. Problems 2 and 3 can be solved iteratively, in a block coordinate-descent fashion: fixing $\mathbf{A}$ and solving for $\mathbf{B}$ using Algorithm 1, then fixing $\mathbf{B}$ and solving for $\mathbf{A}$ (using Algorithm 1 on the transposed system), until convergence of the cost function. The iteration is guaranteed to converge monotonically, albeit reaching the global minimum is not guaranteed. We note that using a common λ to penalize both factors is important for convergence - if different penalty factors are used then additional measures are needed to ensure stability of the overall algorithm.

III. Profiling Algorithm

A. Data acquisition - Google Scholar Miner

We developed Google Scholar Miner (GSM), a light custom Java application to query Google Scholar [2] for a specific researcher, parse the resulting list of publications, and produce a generic / 'noisy' reviewer profile that will serve as the starting point for subsequent analysis. The idea for GSM was inspired by a citation analysis application called *Publish or Perish* [4]. GSM queries Google Scholar using the full name of the researcher and the scientific field in order to disambiguate as much as possible. Parsing the resulting list, it records the citation count for each publication and then hops to the BL-Direct page to retrieve the title and year of publication. This way, GSM is more robust

and yields cleaner output than meta-analysis of the results of generic parsers. Apart from the title (which is essential for obvious reasons), the number of citations and the year of publication are useful in determining the weight of each publication (and keywords present in it). Highly cited and more recent papers carry higher weight, because they are indicative of expertise and current interests, respectively.

B. The Profiling Algorithm

The proposed algorithm comprises the following steps:

1) Extract a (large) set of raw terms from the list of submitted paper titles and the publication history of all reviewers in a joint manner. This set is the set of index terms of size T. We used KEA [9] for this initial extraction. For each reviewer, we produce weights for each publication according to citation count and timeliness. These weights are used for the KEA internal TF/IDF term weights. The easiest way to do this in KEA is via replication.
2) Form the Reviewer-by-Term matrix $\mathbf{R}$ of size $R \times T$, where R is the number of reviewers. Form the Paper-by-Term matrix $\mathbf{P}$ of size $P \times T$, where P is the number of submitted papers. Both matrices are binary, where a value of 1 in the (i, j) element denotes that the i-th reviewer or paper is matched by the j-th term.
3) Create the data matrix $\mathbf{M}$ as the concatenation
$$\mathbf{M} = \begin{bmatrix} \mathbf{R} \\ \mathbf{P} \end{bmatrix}$$
4) Factor $\mathbf{M}$ in a lower rank $\hat{k}$ as
$$\mathbf{M} \approx \mathbf{A}\mathbf{B}^T$$
using double-sided (bilinear) SMR with non-negativity constraints.
5) Reconstruct the data matrix as
$$\hat{\mathbf{M}} = \mathbf{A}\mathbf{B}^T$$
and extract $\hat{\mathbf{R}}$ and $\hat{\mathbf{P}}$.
6) From $\hat{\mathbf{R}}$ take the highest scoring terms for each reviewer. Then, filter out all terms that don't appear on submitted paper titles. The resulting list is the final set of index terms, with cardinality orders of magnitude lower than the initial index term set.
7) Build the new profile vectors for all papers and reviewers using only the set of index terms produced in the previous step. Here it is possible that some reviewer(s) or paper(s) end up with nil profiles over the restricted set. In such cases, we obtain the similarity matrix of each entity ($\hat{\mathbf{P}}\hat{\mathbf{P}}^T$ for the submitted papers and $\hat{\mathbf{R}}\hat{\mathbf{R}}^T$ for the reviewers) and substitute any empty profiles with the most similar non-empty profile.

By concatenating $\mathbf{R}, \mathbf{P}$, we treat reviewers and submitted papers jointly, aiming to extract terms that represent best both sets of entities. The low rank approximation of the data matrix compacts the noisy high-dimensional term space to a low dimensional topic/concept latent space. Note that the latent grouping of similar terms to a topic remains even after we expand the profile vectors to the original dimension in step 5. In step 6 we take full advantage of non-negativity *and* the inherent sparsity of reviewer and paper profiles to reduce noise in the results - this is a key difference relative to classical LSI or non-negative matrix factorization.

Note that we used submitted paper titles instead of the full texts in step 1. Even though full texts are available to the TPC chair, we focus on the titles because they have been chosen by the authors to convey the distilled 'essence' of the paper. Experienced authors will likely do better filtering than any machine, hence we chose to trust authors in this regard. Everything discussed in this paper, however, can also be applied to abstract or even full text indexing.

IV. Experimental results

We used data from two recent IEEE conferences to validate and benchmark our approach. Each dataset included a list of reviewer names and a list of submitted papers titles. In both cases, a domain expert with considerable TPC chair experience was asked to evaluate the results, as detailed in the sequel.

A. Precision

Using the first dataset, we executed the proposed algorithm for 3 different configurations of the GSM/KEA stage, chosen to extract

- 10 terms per reviewer & 5 terms per paper $\rightarrow T = 1251$;
- 20 terms per reviewer & 5 terms per paper $\rightarrow T = 1844$;
- 30 terms per reviewer & 5 terms per paper $\rightarrow T = 2431$.

We also executed a variation of the same algorithm, using non-negative matrix factorization (NMF) [3], which is nowadays widely used in text mining, in place of our non-negative SMR with a double sparsity penalty. This is designed to assess what SMR can offer over NMF in our context. In the case of NMF, we inserted an additional step of sorting and thresholding the intermediate profile vectors at step 6 of the algorithm, because these tend to be dense and very noisy.

We asked the domain expert to mark the terms produced by each algorithm as relevant or irrelevant. Then we computed the precision of each set of terms, according to the following formula:

$$\text{Precision} = \frac{|\text{Relevant} \cap \text{Retrieved}|}{|\text{Retrieved}|} \tag{4}$$

where $|\cdot|$ denotes cardinality, 'Relevant' is the set of relevant terms and 'Retrieved' is the complete set of extracted terms.

Table I
PRECISION FOR NON-NEGATIVE MATRIX FACTORIZATION

		$\hat{k}=20$	$\hat{k}=30$	$\hat{k}=40$
$T=1251$	precision	0.58065	0.51613	0.47170
		$\hat{k}=20$	$\hat{k}=30$	$\hat{k}=40$
$T=1844$	precision	0.5	0.42857	0.42309
		$\hat{k}=20$	$\hat{k}=30$	$\hat{k}=40$
$T=2431$	precision	0.6	0.59091	0.53846

In table I, we report precision as a function of approximation rank $\hat{k}$ for NMF and all three values of T.

In table II, we report precision as a function of approximation rank and λ for non-negative SMR with a double sparsity penalty. Recall that λ controls sparsity - the higher λ is the fewer entries are non-zero. For each combination of T and λ, we also report the average number of non zero coefficients in the resulting profiles. The latter is denoted as $\overline{\neq 0}$ and can be viewed as a measure of sparsity.

Table II
PRECISION FOR SPARSE MATRIX REGRESSION

			$\hat{k}=20$	$\hat{k}=30$	$\hat{k}=40$
$T=1251$	$\lambda=0.1$	precision	0.34615	0.30189	0.31373
		$\overline{\neq 0}$	51	44	45
	$\lambda=0.3$	precision	0.44731	0.35714	0.35714
		$\overline{\neq 0}$	32	30	31
	$\lambda=0.6$	precision	0.84615	0.64	0.64
		$\overline{\neq 0}$	19	19	21
	$\lambda=0.9$	precision	0.8	0.8	0.8
		$\overline{\neq 0}$	10	11	12
			$\hat{k}=20$	$\hat{k}=30$	$\hat{k}=40$
$T=1844$	$\lambda=0.1$	precision	0.22222	0.21260	0.21138
		$\overline{\neq 0}$	225	198	160
	$\lambda=0.6$	precision	0.4	0.44828	0.44828
		$\overline{\neq 0}$	70	66	50
	$\lambda=0.9$	precision	0.38095	0.44	0.44444
		$\overline{\neq 0}$	50	47	49
	$\lambda=1.3$	precision	0.46667	0.52632	0.5
		$\overline{\neq 0}$	18	20	21
			$\hat{k}=20$	$\hat{k}=30$	$\hat{k}=40$
$T=2431$	$\lambda=0.6$	precision	0.52778	0.53846	0.525
		$\overline{\neq 0}$	100	93	82
	$\lambda=0.9$	precision	0.6	0.58065	0.52941
		$\overline{\neq 0}$	51	57	47
	$\lambda=1.3$	precision	0.64706	0.57143	0.57143
		$\overline{\neq 0}$	28	30	31
	$\lambda=1.6$	precision	0.57143	0.52941	0.5
		$\overline{\neq 0}$	19	23	26

The results are plotted in figures 1-4. Figures 1-3 illustrate the precision of SMR as reported in table II. Figure 4 is a comparison between NMF and SMR for $lambda$ set to the value that yields the highest precision for each T.

B. Evaluation of review assignment

Apart from assessing the precision of the proposed algorithm, we also evaluated the quality of the optimized review assignment produced using a variation of [6] with affinity scores computed by taking the inner product of the resulting reviewer and paper profiles. For this we used the second dataset, from a relatively smaller workshop.

In order to have a baseline, the following back-of-the-envelope calculation is useful. Consider the case of a random reviewer assignment. To make the point, assume that each assignment consists of only 4 papers per reviewer, and each reviewer's expertise covers $\frac{1}{7}$-th of the broad scientific field of the conference. Let us also agree that a *bad* assignment is when at least 3 out of 4 papers assigned to the reviewer are outside the reviewer's expertise. Then the probability of a bad random assignment is

$$Pr\{\text{bad}\} = \binom{4}{3}\frac{1}{7}\left(\frac{6}{7}\right)^3 + \left(\frac{6}{7}\right)^4 \approx 0.9 \qquad (5)$$

This is in fact rather optimistic, because the probability of bad assignment grows with the number of assigned papers.

A *good* assignment is one that the TPC chair would normally do manually; a *very good* assignment is one above the average that a reviewer would expect from the TPC chair. The call was again made by the domain expert. The number of submitted papers was 78 and the number of registered reviewers was 64. The number of initially extracted terms was $T = 2288$. We used SMR with $\lambda = 0.1$ and $\hat{k} = 20$. The number of very good assignments was 24, the number of good assignments was 22, and the number of bad assignments was 18. This yields $Pr\{bad\} = 0.2812$ for the fully automated solution, as opposed to 0.9 for random assignment - not bad, but how does the automated solution compare to a manual expert assignment?

In parallel to the above experiment, we also asked authors and reviewers to produce their own custom profiles from a carefully selected list of keywords drawn from past editions of conferences and workshops in the same area. Then the resulting profiles were used to compute inner products, yielding affinity measures which were subsequently used to optimize the review assignment. The resulting assignment was again graded by the domain expert, yielding a probability of bad assignment equal to 0.047. This is considerably better than that of the fully automated solution, but it requires careful planning ahead of the conference, and cooperation / diligence on the part of authors and (more importantly) reviewers - which are hard to ensure. In a (laborious) fully manual assignment by the domain expert, the number of bad assignments turned out to be 7 out of 64 - in between the aforementioned assignments. The fully automatic assignment can be relatively easily improved by manual swapping, thus getting close to optimal results at a fraction of the effort.

V. CONCLUSIONS

Our results suggest that SMR (more specifically, non-negative two-sided SMR with a double sparsity penalty)

outperforms NMF in our present context, where both non-negativity and sparsity are meaningful constraints. There are reasons to believe that this advantage will also manifest in other applications of latent semantic indexing where it is reasonable to assume sparsity - e.g., customer-product or student-course profiling. The computational complexity of SMR is $\mathcal{O}((R+P)T\hat{k}^2)$ per iteration, whereas the computational complexity of NMF is $\mathcal{O}((R+P)T\hat{k})$ per iteration. The difference should not matter much, because $\hat{k}$ is usually small.

REFERENCES

[1] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[2] Google scholar: `http://scholar.google.com/`

[3] D. Lee and H. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[4] Publish or perish: `http://www.harzing.com/pop.htm`

[5] I.D. Schizas, G.B. Giannakis, and N.D. Sidiropoulos, "Exploiting Covariance-domain Sparsity for Dimensionality Reduction," *in Proc. of 3rd Intl. Workshop on Comp. Advances in Multi-Sensor Adapt. Proc.*, Aruba, Dec. 13-16, 2009.

[6] C.J. Taylor, "On the optimal assignment of conference papers to reviewers," University of Pennsylvania, Dept. of Comp. & Information Sci. Tech. Report No. MS-CIS-08-30, 2008.

[7] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[8] F. Wang, B. Chen, and Z. Miao, "A survey on reviewer assignment problem," *Lecture Notes in Computer Science*, vol. 5027, pp. 718–727, 2008.

[9] I. Witten, G. Paynter, E. Frank, C. Gutwin, and C. Nevill-Manning, "KEA: Practical automatic keyphrase extraction," in *Proceedings of the fourth ACM conference on Digital libraries*, pp. 254–255, ACM New York, NY, USA, 1999.

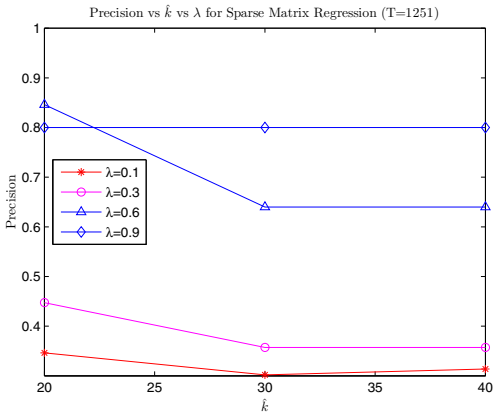

Figure 1. Precision vs $\hat{k}$ for SMR ($T = 1251$)

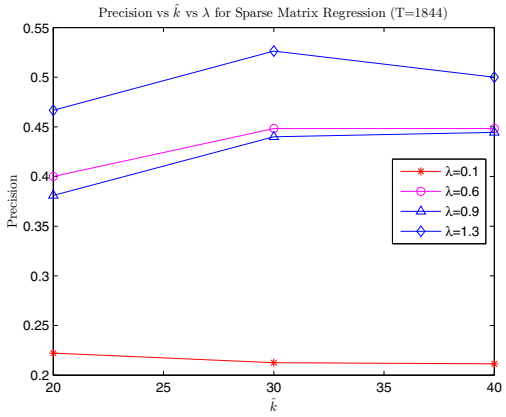

Figure 2. Precision vs $\hat{k}$ for SMR ($T = 1844$)

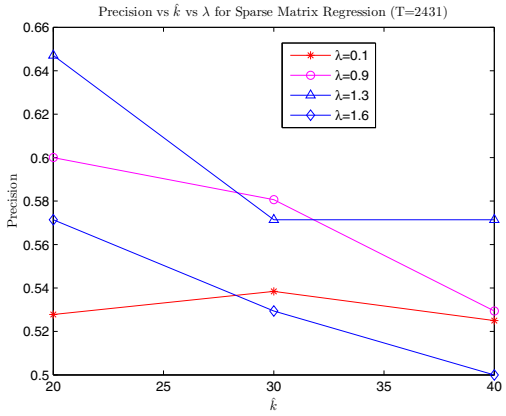

Figure 3. Precision vs $\hat{k}$ for SMR ($T = 2431$)

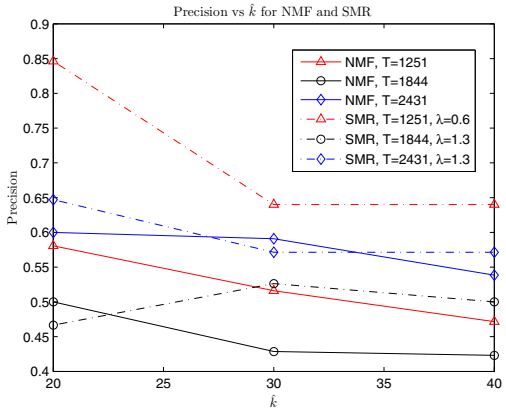

Figure 4. Comparison of NMF & SMR