Novel Tools To Streamline the Conference Review Process: Experiences from SIGKDD'09

Peter A. Flach, Sebastian Spiegler, Bruno Golénia, Simon Price
Dept. of Computer Science
University of Bristol
United Kingdom
{flach, spiegler, goleniab, price}@cs.bris.ac.uk

John Guiver, Ralf Herbrich, Thore Graepel
Microsoft Research
Cambridge
United Kingdom
{joguiver, rherb, thoreg}@microsoft.com

Mohammed J. Zaki
Dept. of Computer Science
Rensselaer Polytechnic Institute
Troy, NY, USA
zaki@cs.rpi.edu

ABSTRACT

The SIGKDD'09 Research Track received 537 paper submissions, which were reviewed by a Program Committee of 199 members, and a Senior Program Committee of 22 members. We used techniques from artificial intelligence and data mining to streamline and support this complicated process at three crucial stages: bidding by PC members on papers, assigning papers to reviewers, and calibrating scores obtained from the reviews. In this paper we report on the approaches taken, evaluate how well they worked, and describe some further work done after the conference.

1. INTRODUCTION

The workflow of a typical computer science conference is that, once the submitted papers have been received, the Program Committee (PC) chairs assign each paper to several (typically three) PC members for review. After a suitable period the reviews are collected and the PC chairs consolidate their recommendations into an accept/reject decision for each paper. For a large and premier conference like the *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (SIGKDD), it is customary that the PC chairs are assisted by a Senior Program Committee, the members of which each overview the reviewing process of a small subset of the papers.

For the last ten years or so these review processes are usually supported by some conference management tool. SIGKDD'09, like its predecessors, used the aptly named CMT system developed and supported by Microsoft Research[1]. Such tools allow a paper bidding phase prior to the actual paper assignments, during which PC members can express their interest in reviewing particular papers, based on the paper abstracts. There is also the opportunity to register conflicts of interests, if a PC member's relations with the authors of a particular paper are such that the PC member is not a suitable reviewer for that paper.

As the PC chairs for SIGKDD'09, the first and last author of this paper (PAF and MJZ) realised early-on that, in order to get a high-quality selection of submitted papers for inclusion in the conference proceedings and presentation at the conference, we needed to ensure that papers were reviewed by the right PC members, and the bidding phase was clearly an important instrument for that. However, for PC members it can be very time-consuming to go through more than five hundred abstracts in order to select about thirty papers on which to place a high bid. So the first step was to find a way to assist PC members in the bidding process. A team at the University of Bristol (authors PAF, SS, BG, and SP, with SS as the main architect) implemented a way to initialise the bids based on matching PC members' publications against the submitted papers (Section 2).

Secondly, once the bids were in, it was important to honour them as much as possible in the assignments. The same Bristol team (mainly BG) implemented this by means of a set of constraints represented as an integer program (Section 3).

After all the reviews were submitted by the PC members, the paper acceptance/rejection stage involved selecting the top-k papers, where k is about 100. The key problem here is one of calibration: individual reviewers may rate papers on slightly different scales, and individual papers may be in more or less fashionable subject areas. We experimented with a Bayesian model for calibrating reviewer scores, proposed and implemented by the team (authors JG, RH, and TG) at Microsoft Research Cambridge (Section 4).

Some of these tools are being developed into a publicly available service, which is described in Section 5. Our conclusions appear in Section 6.

2. BID INITIALISATION

The first step was to devise an effective way to aid PC members to bid on the submitted papers. Rather than change the bidding process *per se*, our approach was to give each PC member a personalised set of initial bids, which they could then change as they saw fit.

To achieve this, we calculated a similarity score for each PC member pc_i and each submitted abstract a_j based on the former's publications. We used a vector space model borrowed from information retrieval [6]. However, instead of scoring documents on a query we compared a bag of words occurring in the abstract to a bag of words in the titles of previous publications of the PC member using an approach similar to that in [8]. The dimensionality of the vector space

[1] `http://cmt.research.microsoft.com/cmt/`

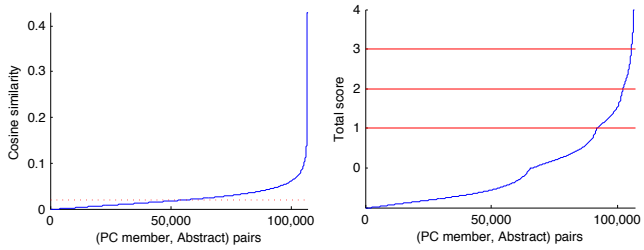

Figure 1: (Left) Calculated cosine similarities for each PC member and abstract. The dotted line denotes the median (0.0185). (Right) Total score including the shared keywords. The horizontal lines indicate the thresholds above which a bid would be initialised to 1 (in a pinch), 2 (willing) or 3 (eager). About 4.5% of the abstracts received a bid of at least 2 (on average 24 abstracts per PC member).

thus corresponds to the joint vocabulary of all PC member's publication titles taken from the DBLP Computer Science Bibliography[2] and the words from all abstracts without stop words. Initially, we assigned a vector v to each PC member and abstract containing counts of each word type used and then normalised these counts based on term frequency-inverse document frequency (tf-idf) [7]. The ranking of abstracts for each PC member was then done by applying cosine similarity which corresponds to the angle between an abstract vector and a given PC member vector.

$$s_1(pc_i, a_j) = \frac{v_{pc_i} \cdot v_{a_j}}{||v_{pc_i}|| \cdot ||v_{a_j}||}$$

Figure 1 (Left) plots the cosine similarities calculated for each of the $537 * 199 = 106,863$ (PC member, abstract) pairs. As can be seen, the values found are generally quite low, with a median of 0.0185.

We also had, for each PC member and each abstract, a set of subject areas. These were selected by the PC members and submitting authors from a pre-compiled list of about 65 subject areas which was based on the session titles from the SIGKDD conferences over the last ten years. We then calculated a second similarity measure $s_2(pc_i, a_j)$ which simply counts the number of subject areas a PC member and a paper had in common. The overall score of a PC member and a paper pair was then computed as

$$s(pc_i, a_j) = \alpha \cdot s_1(pc_i, a_j) + s_2(pc_i, a_j) - 1$$

where $\alpha = 15$ was manually found to give a good bid distribution (Figure 1, Right).

We converted these scores into initial bids by thresholding. Specifically, thresholds at 1, 2 and 3 gave initial bids at level 1 (in a pinch), level 2 (willing) and level 3 (eager). So, for instance, conditions under which a bid would be initialised to 3 include

- four shared subject areas;
- three shared subject areas and a cosine similarity of at least 0.067;
- two shared subject areas and a cosine similarity of at least 0.133.

[2] `http://dblp.uni-trier.de/`

On average, a PC member would have about 7.5 papers (1.4% of 537) initialised at bid level 3, 16.6 (3.1%) at level 2, and 52 (9.7%) at level 1.

We then gave the PC members the opportunity to revise these initial bids. The CMT system allows one to enter bids for individual papers as well as all the papers in particular subject areas. Ranked lists of papers on each of the three scores (s_1, s_2 and s) were made available to the PC members to aid them in the final bidding step. Comments we received from PC members at this stage ranged from a surprised "There are already non-default bids on my papers in the system (and they are not bad)" via a contented "I reviewed my automatically assigned papers a few days ago and was very happy with them so I did not put any bids in" to an excited "as I go thru my paper assignments, I am extremely impressed by quality of your initial automated assignment!"

We compared our initial bids to the final bids after PC members revised them. Disregarding the level of the bid, we calculated precision, recall and F-measure for each reviewer. Here, precision is the proportion of non-zero actual bids among the non-zero initial bids; recall is the proportion of non-zero initial bids among the non-zero actual bids; and F-measure is the harmonic mean of precision and recall. Over all reviewers the median F-measure is 72.7%; median precision 88.2%; and median recall 80.0% (we use the median because of the skew of the distribution). While of course both 'unbidding' (resetting non-zero initial bids to zero) and bidding on additional papers happen in practice, these results suggest that the latter happens more often than the former.

We also calculated mean squared error (MSE) for each PC member, which does take the bid levels into account. The median MSE over all reviewers is 0.20. All in all, these results are very encouraging and demonstrate that there is a lot of scope for automated support during the bidding process driven by text analysis. Notice that it might be possible to learn the various parameters such as α and the bid thresholds used, from the actual bids provided by the PC members.

3. PAPER ASSIGNMENT

The next step was to assign papers to PC members, taking their bids into account as much as possible. Our approach was inspired by the work of Smolka and Walser for the *Third International Conference on Principles and Practice of Constraint Programming* [1; 9]. The principle is to transform the allocation problem into an integer programming problem that can be solved using a standard solver. An integer program is defined by a set of boolean constraints that need to be satisfied when optimising an objective function.

Let r be the number of PC members and p the number of papers. As we have seen, each PC member bids on a couple of dozen papers from 1 to 3: 1 is lightly interested, 2 interested and 3 highly interested. A *bid matrix* $B^{r \times p}$ is an integer matrix which defines the preferences of each PC member $1 \leq i \leq r$ for each paper $1 \leq j \leq p$ such that $0 \leq B_{i,j} \leq 3$; 0 means no bid. (In fact, we added 2 to the non-zero entries in B to decrease the chance of assigning papers that a PC member did not bid on.)

PC members and submitting authors could also register conflicts of interest. A *conflict of interest matrix* $C^{r \times p}$ is a binary matrix in which 1 refers to a conflict of interest and

0 to no conflict of interest. Hereafter, we write r_r for the number of PC members per paper (i.e., the number of reviews required per paper; 3 for SIGKDD'09) and r_p for the number of papers per PC member ($r_p \approx \frac{r_r p}{r}$, about 8 for SIGKDD'09).

Now, we introduce the *integer assignment problem* as follows: Given $B^{r\times p}$, $C^{r\times p}$, r_r and r_p, an *integer assignment problem* consists in finding a binary matrix $A^{r\times p}$ (binary matrix) maximising the Frobenius inner product of A and B under the constraints r_r, r_p and $C^{r\times p}$. Specifically, r_r constraints are represented as:

$$\sum_{i=1}^{r} A_{i,j} = r_r,\ \forall\ 1 \le j \le p$$

Secondly, r_p constraints are defined as follows:

$$r_p \le \sum_{j=1}^{p} A_{i,j} \le r_p + 1,\ \forall\ 1 \le i \le r$$

As can be seen here, we allow ourselves a slight margin on the assigned number of papers per reviewer. Finally, for each i and j we add the constraint $A_{i,j} = 0$ if $C_{i,j} = 1$, which deals with the conflicts of interest.

The objective function is defined as follows:

$$max : \sum_{i=1}^{r}\sum_{j=1}^{p} B_{i,j} A_{i,j}$$

This integer program was solved with $r_r = 3$ and $r_p = 8$ using lp_solve [2] in less than 2 minutes.

The quality of the found solution is indicated by the fact that 94.3% of the assigned papers were bid on at levels 2 (willing) or 3 (eager). This compares favourably with the result found by CMT's built-in assignment tool, which resulted in 82%.

We also looked at the extent to which PC members achieved their optimum. We calculated the *assignment quality* as the total bids of the assigned papers as a percentage of the top r_p bids by the PC member. For example, if the PC member bid 3 on 10 papers and was assigned 5 of those, as well as two at bid level 2 and one at level 1, the assignment quality is $\frac{5\cdot 3+2\cdot 2+1\cdot 1}{8\cdot 3} = 83.3\%$. However, if the same assignment happened for a PC member who only bid 3 on 5 papers and 2 on at least 3 others, the assignment quality is higher: $\frac{5\cdot 3+2\cdot 2+1\cdot 1}{5\cdot 3+3\cdot 2} = 95.2\%$. The median assignment quality over all PC members was 95.8%; this corresponds to a PC member who has bid high on sufficiently many papers and gets assigned 7 of those papers s/he is eager to review, plus one at bid level 2.

After the paper assignment was calculated, very little manual 'tweaking' was needed for some special cases, e.g., authors who were on the PC for both the Research Track and the Industrial and Government Applications Track and hence needed a lower reviewing load for either track, or to handle previously unnoticed conflicts of interest. For this manual tweaking we used a by-product of the bidding phase: a list of papers with for each paper the ten PC members with highest cosine similarity. The list itself was sorted on decreasing average cosine similarity over all PC members, with the more 'dissimilar' papers at the top of the list. All in all, the automated bidding and assignment of papers contributed significantly to the high quality of reviews we received for SIGKDD'09.

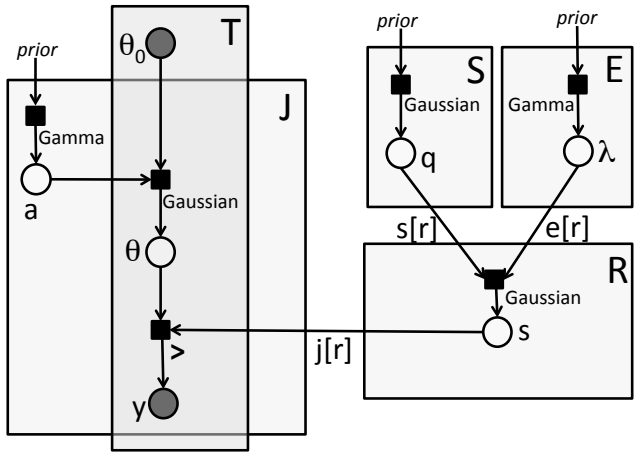

Figure 2: Factor graph of the generative probabilistic model consisting of judges J, submissions S, expertise levels E and reviews R. T denotes the number of judge-specific thresholds that are used to convert continuous scores into discrete recommendations from 1 to 6.

4. REVIEWER SCORE CALIBRATION

The final and most challenging step in the review process was for the PC chairs to accept or reject papers. The PC chairs assembled all the information in front of them, asking where necessary for further clarification from the Senior PC member handling the paper or the PC members who reviewed it; occasionally soliciting further reviews or taking a look at papers themselves; and so on. Finally, decisions were taken taking into account a range of factors, including not just the Senior PC member's recommendation or the average score assigned by the reviewers, but also the spread in those scores, the expertise of individual reviewers, points raised during the discussion phase, and so on. This year, an additional factor we took into account was the output of a probabilistic model developed by the team at Microsoft Research Cambridge.

The model used to calibrate reviewer scores is a generative probabilistic model which addresses variation in reviewer accuracy, and self-assessed differences in reviewer expertise level. The model relies on the fact that each paper has several reviewers, and each reviewer scores several papers. This enables us to disentangle (up to uncertainty) the reviewer's reviewing standards and the quality of the papers. Similar forms of model have been used for assessing NIPS reviews, for rating the skills of Xbox Live players, and also for the Research Assessment Exercise for UK computer science, 2008. A factor graph of the model is shown in Figure 2; arrows are included to show the generative flow. We now describe the model in detail.

Assume R reviews, S submissions, E reviewer expertise levels, and J reviewers — we use J (for judges) to distinguish reviewers from reviews. For each review $r \in [1, R]$, there is a corresponding submission $s[r] \in [1, S]$, expertise level $e[r] \in [1, E]$, and judge $j[r] \in [1, J]$; these dependent indices are used to label the appropriate edges in the factor graph to indicate the sparse connection topology between the review plate and the other plates.

We assume that there is a latent submission 'quality' $q_s \sim \mathcal{N}(\mu_q, v_q)$ for each submission s; this is the primary set of

variables that we want to infer. In addition, we assume a precision $\lambda_e \sim \mathcal{G}(k_\lambda, \beta_\lambda)$ (a Gamma distribution parameterised by shape and rate) for each expertise level; we also infer this set of precisions. The expertise levels for the SIGKDD'09 review process were *informed outsider*, *knowledgeable*, and *expert*. For each review r, a review 'score' $s_r \sim \mathcal{N}(q_s, \frac{1}{\lambda_e})$ is then derived from the submission quality and expertise precision associated with that review.

We use a cumulative threshold model for the ordinal scale of the ratings. More specifically, the review score is compared to a set of judge-specific thresholds $\theta_{jt} \sim \mathcal{N}(\theta_{0t}, \frac{1}{a_j})$, where θ_{0t} is a nominal set of thresholds, and $a_j \sim \mathcal{G}(k_a, \beta_a)$ is the 'accuracy' of judge j. The observed data is in the form of a set of ratings $z_r, r \in [1, R]$; these ratings take on a set of integer values $1, \ldots, 6$ corresponding respectively to *strong reject*, *reject*, *weak reject*, *weak accept*, *accept*, and *strong accept*. The number of thresholds T is one less than the number of rating labels representing the score boundaries between the different labels; the nominal θ_{0t} are set to $\{1.5, 2.5, 3.5, 4.5, 5.5\}$ to reflect this.

In order to keep the graphical model succinct, we convert each observed rating z_r into an array of boolean observations y_r corresponding to the assertion that $s_r > \theta_{jt}$. This allows all thresholds to be treated equally, and they can therefore be represented by a plate in the factor graph, and treated collectively by the inference algorithm. So, for example, an observation of *weak accept* is converted to $\{T, T, T, F, F\}$.

SIGKDD'09 had 537 submissions, 199 judges (reviewers), and 1610 reviews each consisting of a rating and a self-assessed expertise level. The model has $J + 2JT + S + E + R$ variables, giving a total of 4339 variables. Three priors need to be set – those for the quality $q_s \sim \mathcal{N}(\mu_q, v_q)$, expertise precision $\lambda_e \sim \mathcal{G}(k_\lambda, \beta_\lambda)$, and accuracy $a_j \sim \mathcal{G}(k_a, \beta_a)$. These were set as follows: $\mu_q = 3$, $v_q = 1$, $k_\lambda = 10$, $\beta_\lambda = 10$, $k_a = 10$, $\beta_a = 10$. (In hindsight, $\mu_s = 3.5$ at the mid-range of the nominal thresholds would have been a better choice.)

Inference is done using Expectation Propagation [3; 4] which is a fast deterministic algorithm for approximate Bayesian inference. It is a message-passing algorithm which seeks to find the fully factorised approximation whose K-L (Kullback-Leibler) distance is closest to the true joint probability distribution. The model was implemented using Infer.NET [5], which is a .NET framework for inference. This framework provides a programming interface for defining a probabilistic model; the model code can be written in any .NET program and can be combined with non-model code. When performing inference, the model code is compiled into a message passing algorithm tailored to the specific model, leading to very efficient inference; execution time for the SIGKDD'09 dataset was a few seconds.

Marginal posteriors for the q_s, the θ_{jt}, the λ_e, and the a_j were obtained from the model. These posteriors allow one not only to obtain calibrated scores per paper, but also to assess the level of uncertainty in those scores. We can also assess, for instance, the level of consistency of individual reviewers (through the posterior on λ_e). The PC chairs used these calibrated scores not in an algorithmic fashion, but rather to highlight areas where further scrutiny was needed. The code for the reviewer calibration model is available as an example in the Infer.NET user guide at the Infer.NET website [5].

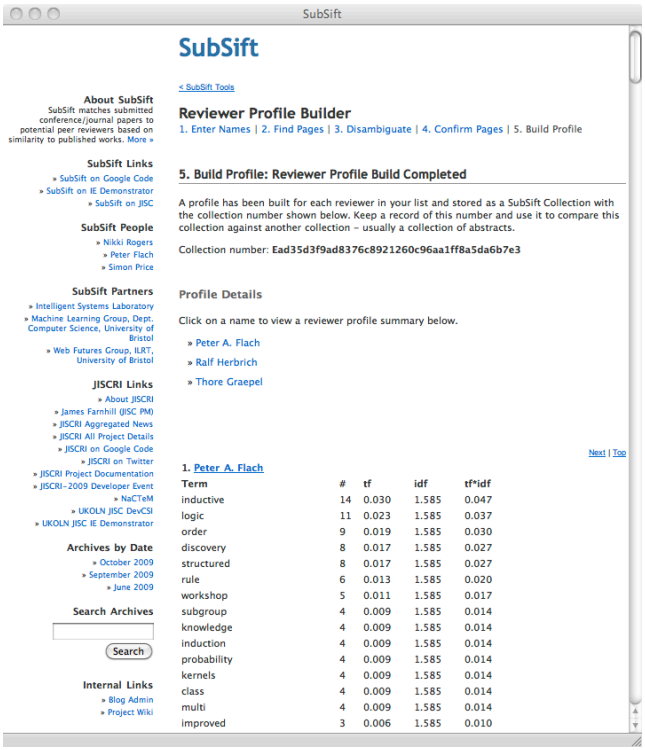

Figure 3: SubSift builds a profile of individual reviewers by constructing a tf-idf vector from the most frequent words in the titles of their publications as recorded on DBLP.

5. THE SUBSIFT TOOLS

The organisation of an academic conference or workshop usually takes place under quite strong time constraints, and the development of reusable tools is not necessarily very high on the agenda. The software described above was implemented in a variety of programming languages, environments and platforms, including Java, C++, Prolog and Matlab, and is probably not usable without direct involvement of the developer. Once the conference was over, however, the Bristol team recognised the potential for the general approach to be applied in a much wider range of academic peer review activities.

A project proposal was therefore submitted to the UK Joint Information Services Council (JISC) under the Rapid Innovation strand of the Information Environment Programme 2009-11, and subsequently granted. The project, which is called SubSift (short for 'submission sifting'), is being undertaken by the Institute for Learning and Research Technology (ILRT) at the University of Bristol, and runs until January 2010. A short description of the anticipated outcomes follows.

- SubSift will document, package and release the bid initialisation and paper assignment software as an Open Source application via the Google Code website. (At present there are no plans to open source the reviewer score calibration.) SubSift will repackage the software as both a website and as a family of web services. The latter enables re-use in bespoke applications or in "mash-ups".

- SubSift will allow customisation of the software through well documented configuration settings and through support for mash-ups with tools like Yahoo! Pipes. By adopting standards-based protocols and data formats, SubSift's web services will enable ad hoc workflows, allowing "pick and mix" of the most useful features on an application-by-application basis.

- SubSift will accept input from different bibliographic data sources in addition to DBLP, such as Citeseer, Google Scholar, eprints, news and blogs. It will also accept abstracts from a wider range of sources (it will not, however, directly integrate with conference management systems).

Figure 3 shows a screenshot of the current prototype after building a tf-idf profile of individual reviewers.
The SubSift project also seeks to evaluate the application of these tools to other conferences and in novel contexts elsewhere. Please contact us via the project website at `http://subsift.ilrt.bris.ac.uk/` if you would like to become involved.

6. CONCLUDING REMARKS

While the academic peer review system is well-established and quite well understood, our experiences from SIGKDD'09 indicate that there is considerable scope for technological innovations. These are not just fancy add-ons: we strongly believe that the tools we used and developed have had significant impact on the quality of the reviews, and ultimately, the scientific program of the conference. By documenting our experiences and developing some of the tools as open source, we hope to have provided a useful service to the scientific community.

Acknowledgements

We would like to thank the CMT support team at Microsoft Research for their unfailing support in all things CMT; the SIGKDD'09 General Chairs Françoise Soulié-Fogelman and John Elder and the (previous) SIGKDD Chair Gregory Piatetsky-Shapiro for their encouragement and suggestions, and the SIGKDD'08 PC chairs Bing Liu and Sunita Sarawagi for providing the inspiration for some of this work.

7. REFERENCES

[1] G. Smolka and J. Walser. Review Assignment for CP97, 1997. `http://www.ps.uni-sb.de/~walser/reviewing.ps`.

[2] M. Berkelaar, K. Eikland and P. Notebaert. lp_solve : Open source (Mixed-Integer) Linear Programming system. `http://lpsolve.sourceforge.net/`.

[3] T. Minka. Expectation propagation for approximate bayesian inference. In J. S. Breese and D. Koller, editors, *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pages 362–369. Morgan Kaufmann, 2001.

[4] T. Minka. Divergence measures and message passing. Technical report, Microsoft Research, 2005.

[5] T. Minka, J. Winn, J. Guiver, and A. Kannan. Infer.NET 2.3, 2009. Microsoft Research Cambridge. `http://research.microsoft.com/infernet`.

[6] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, 1975.

[7] K. Spärck-Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21, 1972.

[8] S. Spiegler. Comparative study of clustering algorithms on textual databases - clustering of curricula vitae into comptency-based groups to support knowledge management. Master's thesis, Technical University Ilmenau, Ilmenau, Germany, February 2007.

[9] J. Walser. *Integer Optimization by Local Search*, volume 1637 of *Lecture Notes in Computer Science*, pages 81–93. Springer, 1999.