The Toronto Paper Matching System: An automated paper-reviewer assignment system

Laurent Charlin LCHARLIN@CS.TORONTO.EDU
University of Toronto

Richard S. Zemel ZEMEL@CS.TORONTO.EDU
University of Toronto

Abstract

One of the most important tasks of conference organizers is the assignment of papers to reviewers. Reviewers' assessments of papers is a crucial step in determining the conference program, and in a certain sense to shape the direction of a field. However this is not a simple task: large conferences typically have to assign hundreds of papers to hundreds of reviewers, and time constraints make the task impossible for one person to accomplish. Furthermore other constraints, such as reviewer load have to be taken into account, preventing the process from being completely distributed. We built the first version of a system to suggest reviewer assignments for the NIPS 2010 conference, followed, in 2012, by a release that better integrated our system with Microsoft's popular Conference Management Toolkit (CMT). Since then our system has been widely adopted by the leading conferences in both the machine learning and computer vision communities. This paper provides an overview of the system, a summary of learning models and methods of evaluation that we have been using, as well as some of the recent progress and open issues.

1. Introduction

Conference organizers are faced with the difficult task of determining their conference's program. In many computer science conferences this involves asking fellow experts to evaluate papers submitted to the conference. Obtaining high-quality reviews is of great importance to the quality and reputation of a conference. Further, conference organizers typically need to assign reviewers within a couple of days after the conference submission deadline. Typical conferences in our fields now routinely receive more than one thousand papers which have to be assigned to reviewers from a pool which often consists of hundreds of reviewers. The assignment of each paper to a set of suitable reviewers requires knowledge of both the topics explored in the paper as well as knowledge about reviewers' expertise. For a typical conference it will therefore be beyond the ability of a single person, for example the program chair, to assign all submissions to reviewers. Decentralized mechanisms are also problematic since global constraints, such as reviewer load, conflicts of interest, and the fact that every paper must be reviewed by a certain number of reviewers, have to be fulfilled. The main motivation for automating the reviewer assignment process is to reduce the time required to (manually) assign submitted papers to reviewers.

A second motivation for an automated reviewer assignment system concerns the ability of finding suitable reviewers for papers, and to expand the reviewer pool and overcome research cliques. Particularly in rapidly expanding fields, such as machine learning, it is of increasing importance to include new reviewers into the review process, and automated systems offer the ability to learn about new reviewers as well as the latest research topics.

In practice conferences often adopt a hybrid approach where a reviewers' interest with respect to a paper is first independently assessed either by allowing reviewers to bid on submissions or, for example, by letting members of the senior program committee express their expertise assessments of reviewers. Using either of these assessments the problem of assigning reviewers to submissions can then be framed and solved as

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

an optimization problem (see Section 4.3). Such a solution still has important limits. Reviewer bidding requires reviewers to assess their preferences over the list of all papers. Failing to do so, for example if reviewers search for specific keywords, will naturally introduce noise into the process. On the other hand, asking the senior program committee to select reviewers is still a major time burden.

Faced with these limitations when Richard Zemel was the co-program chair of NIPS 2010, he decided to build a more automated way of assigning reviewers to submissions. The resulting system that we have developed aims at properly evaluating the expertise of reviewers in order to yield good reviewer assignments while minimizing the time burden on the conferences' program committees (reviewers, area chairs, and program chairs). Since then the system has gained adoption in both the machine learning and computer vision conferences and has now be used (repeatedly) by: NIPS, ICML, UAI, AISTATS, CVPR, ICCV, ECCV, ECML/PKDD, ACML, ICGVIP.

2. Overview of the framework

In this section we first describe the functional architecture of the system including how several conferences have used it. We then briefly describe the system's software architecture.

Our aim is to determine reviewers' expertise. Specifically we are interested in evaluating the expertise of every reviewer with respect to each submission. Given these assessments, it is then straightforward to compute optimal assignments (see Section 4.3).

The workflow of the system works in synergy with the conference submission procedures. Specifically for conference organizers the busiest time is typically right after the paper submission deadline since at this time the organizers are responsible for all submissions and several different tasks, including the assignment to reviewers, have to be completed within tight time constraints. For TPMS to be maximally helpful reviewers' expertise assessment could then be computed ahead of the submission deadline. With that in mind we note that an academic's expertise is naturally reflected through his work, and most easily accessed by examining his published papers. Hence we use a set of published papers for each reviewer participating in a conference. Throughout our work we have used the raw text of said papers. It stands to reason that other features of a paper could be modelled: for example one could use a citation or co-authorship graphs built from papers' bibliography and co-authors respectively.

Reviewers' previously published papers have proven to be very useful to assess one's expertise. However we have found that we can further boost our performance with another source of data: reviewer's self-assessed expertise about the submissions. We will refer to such assessments as *scores*. We differentiate scores from more traditional *bids*: scores represent expertise rather than interest. We use assessed scores to predict missing scores, and then use the full reviewer-paper score matrix to determine assignments. Hence a reviewer may be assigned to a paper for which s/he did not provide a score.

To summarize, although each conference has its own specific workflow, it usually involves the following sequence of steps (Figure 1). First, we collect reviewers' previous publications (note that this can be done before the conference's paper submission deadline). Using those publications we build reviewer profiles which can be used to estimate each reviewer's expertise. These initial scores can then be used to produce paper-reviewer assignments, or to refine our assessment of expertise by guiding a score elicitation procedure (e.g., using active learning to query scores from reviewers). Elicited scores, in combination with our initial unsupervised expertise assessments, are then used to predict the final scores. Final scores can then be used in various ways by the conference organizers (e.g., to create per-paper reviewer rankings that will be vetted by the senior program committee or directly in the matching procedure).

Below we describe the high-level workflow of several conferences that have used TPMS.

NIPS 2010: For that conference most of the focus went toward modelling the area chairs' (senior program committee) expertise. We were able to evaluate area chairs's initial expertise using their previously published papers (there were an average of 32 papers per area chair). We then used these initial scores to perform elicitation. The exact way in which we picked which reviewer paper pairs to elicit is described in the next section.We did the elicitation in two rounds. In the first round we kept about two-thirds of the papers selected as the ones our system was most confident about (estimated as the inverse entropy of the distribution across area chairs per paper). Using these elicited score we were then able to run a supervised learning model and proceeded to elicit information from the remaining one-third of the papers. For the reviewers, we used the initial scores and asked each reviewer to bid on about 8 papers. These two sources of information were used to provide a ranked list of reviewers to each area chair.

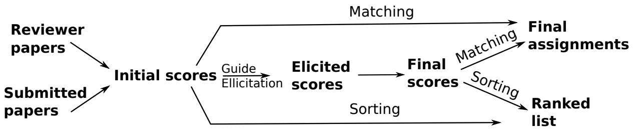

Figure 1. A conference's typical workflow.

ICCV-2013: ICCV used author suggestions (each author could suggest up to 5 area chairs that could review their paper) to restrict area chair score elicitation. The elicited scores were used to assign area chairs. Area chairs then suggested reviewers for each of their paper. TPMS initial scores were used to present a ranked list of candidate reviewers to each area chair.

ICML 2012: Both area chairs and reviewers could assess their expertise for all papers. TPMS initial scores were used to generate personalized ranked list of candidate papers which area chairs and reviewers could consult for help. TPMS then used recorded scores, for both reviewers and area chairs, in a supervised learning model. Predicted scores were then used to assign area chairs and one reviewer per paper (area chairs got to assign the two other reviewers). [1]

2.1. Active expertise elicitation

As mentioned in the previous section initial scores can be used to guide active elicitation of reviewer expertise. The direction we have followed is to run the matching program using the initial scores. That is, we use the initial scores to find an (optimal) assignment of papers to reviewers. Then, the reviewer expertise for all assignments (assigned reviewer-paper pair) are queried. Intuitively, these queries are informative since according to our current scores, reviewers are queried about papers they would have to review (a strong negative assessment of a paper is therefore very informative). By adapting the matching constraints, conference organizers can tailor the number of scores elicited per user (in practice it can be useful to query reviewers about more papers than warranted by the final assignment). We have more formally explored these ideas in (Charlin et al., 2011). Note that our elicited scores will necessarily be highly biased by the matching constraints (in other words scores are not missing at random). In practice this does not appear to be a problem for this application (i.e., assigning papers to a small number of expert reviewers).

2.2. Software view

For the NIPS-10 conference the system was initially made up of a set of Matlab routines that would operate on conference data. The data was exported (and re-imported) from the conference website hosted on Microsoft's Conference Management Toolkit (CMT). [2] This solution had limitations since it imposed a high cost on conference organizers that wanted to use it. Since then, and encouraged by the ICML 2012 organizers, we have developed an online version of the system which interfaces with CMT and can be used by conference organizers through CMT (see Figure 2).

The system has two primary software features. One is to store reviewers' previously published papers. We refer to these papers as a reviewer's archive. In order to populate their archive reviewers can register and login to the system through a web interface. Reviewers can then provide URLs containing their publications to the system that will automatically crawl them in search of their PDFs. There is also a functionality to allow reviewers to upload PDFs from their local computer. Conference organizers can also populate reviewers' archives on their behalf. An option also allows our system to crawl a reviewer's Google Scholar profile. [3] The ubiquity of the PDF files has made it our default format for the system. The interface is entirely built using the python-based Django web framework [4] (except for the crawler which is written in PHP and relies heavily on wget [5]).

The second main software feature is one that permits communication with Microsoft's CMT. Its main purpose is to allow our system to access some of the CMT data as well as allow organizers to call our system's functionalities through CMT. The basic workflow works as follows: organizers, through CMT, send TPMS the conference submissions, then they can send us a *score request* which asks our system to compute scores for a set of reviewers and papers. This request contains the paper and reviewer identification for all the scores that should be returned. Additionally, the request can contain elicited scores (bids in CMT terminology). After having received such a request, our system processes the data, that may include processing pdf submissions and reviewer publications, and computes scores according to a particular model. TPMS scores can then be retrieved through CMT by the conference organizers.

[1] The full ICML 2012 process has been detailed by the conference's program chairs: http://hunch.net/?p=2407

[2] http://cmt.research.microsoft.com/cmt/

[3] http://scholar.google.com/

[4] https://www.djangoproject.com/

[5] http://www.gnu.org/software/wget/

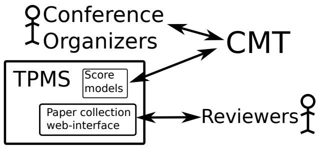

Figure 2. High-level software architecture of the system.

Technically speaking our system acts as a paper repository where submissions and meta-data (both from CMT) can be deposited. Accordingly, the communication protocol being used is SWORD based on the Atom Publishing Protocol (APP) version 1.0 [6]. SWORD defines a format to be used on top of HTTP. The exact messages allow CMT to: a) deposit documents (submissions) to the system; b) send the system information about reviewers such as their names and publication URL; c) send the system reviewer bids (in the CMT terminology). On our side, the SWORD API was developed in Python and is based on a simple SWORD server implementation. [7] The SWORD API interfaces with a computations module written in a mixture of Python and Matlab (we also use Vowpal Wabbit [8] for some of training some of the learning models).

Note that although we interface with CMT, TPMS runs completely independently (and communicates with CMT through the network); thus other conference management frameworks could easily interact with TPMS. Further, CMT has their own matching system which can be used to determine reviewer assignments from scores. CMT's matching program can be used to combine several pieces of information such as TPMS scores, reviewer suggestions and subject area scores. Hence, we typically return scores from CMT and conference organizers then run CMT's matching system to obtain a set of (final) assignments.

3. Related Work

We are aware that other conferences, namely SIGGRAPH, KDD and, EMNLP, have previously used a certain level of automation for the task of assigning papers to reviewers. The only conference management system that we are aware of that has explored machine learning techniques for paper reviewer assignments is MyReview [9] and some of their efforts are detailed in (Rigaux, 2004).

On the scientific side several authors have been interested in similar problems. We note the work of (Conry et al., 2009) which uses a collaborative filtering method along with side information about both papers and reviewers to predict reviewer paper scores. (Mimno & McCallum, 2007) developed a novel topic model to help predict reviewer expertise. (Rodriguez & Bollen, 2008) have built co-authorship graphs using the references within submissions in order to suggest initial reviewers. Finally, (Balog et al., 2006) utilize a language model to evaluate the suitability of experts for various tasks.

4. Learning and testing the model

As mentioned in Section 2, at different stages in a conference's workflow we may have access to different types of data. Accordingly we will differentiate models based on the type of data that they use. We first describe models that can be used to evaluate reviewer expertise using reviewers' archive and the submitted papers. Then we describe supervised models that have access to ground truth expertise scores. The same dichotomy will be used to report experimental results.

We first introduce some notation that will be useful to describe the various learning models using a common framework. P: the set of all submitted papers. Individual submissions are indexed by p. A: the set of archive (reviewers' previously published papers). Reviewer r's archive is denoted A_r. Note we will assume that a reviewer's papers are concatenated into a single document to create that reviewer's archive. $f(\cdot), g(\cdot)$: represent functions which map papers (either submitted or archive) to a set of features. Features can be word counts associated in the bag-of-word representation, or possibly, higher-level features such as the ones learned with LDA (Blei et al., 2003).

4.1. Initial scoring models

Language Model: This model predicts a reviewer's score as the dot product between a reviewer's archive representation and a submission:

$$s_{rp} = f(A_r)f(P_p)' \qquad (1)$$

There are various possible incarnations of this model. The one that we have routinely been using consists in using the word count representation of the submissions (i.e., each submission is encoded as a vector where the value of an entry corresponds to the number of times that word associated with that entry appears in the submission). For the archive we use the normalized

[6] The same protocol with similar messages is used by http://arxiv.org to allow users programmatic submissions of papers

[7] https://github.com/swordapp/Simple-Sword-Server

[8] http://hunch.net/~vw/

[9] http://myreview.lri.fr/

word count for each word appearing in the reviewer's published work. By assuming conditional independence between words given a reviewer, working in the log-domain, the above is equivalent to:

$$s_{rp} = \sum_{w \in P_p} \log f(A_{rw}) \tag{2}$$

In practice we Dirichlet smooth the reviewer's normalized word counts to better deal with rare words:

$$f(A_{rw}) = \left(\frac{N_{A_r}}{N_{A_r} + \mu} \right) \frac{|w_{A_r}|}{N_{A_r}} + \left(\frac{\mu}{N_{A_r} + \mu} \right) \frac{|w|}{N} \tag{3}$$

with N_{A_r} the total number of words in r's archive, N is the total number of words in the corpus, $|w_{A_r}|$ and $|w|$ the number of occurrences of w in r's archive and in the corpus respectively, and smoothing parameter μ.

Since papers have different lengths, scores will be uncalibrated. Namely, shorter papers will receive higher scores than longer papers. Depending on how scores are used this may not be problematic. For example, this will not matter if one wishes to obtain ranked lists of reviewers for each paper. We have obtained good matching results with such a model. However, normalizing each score by the length of its paper has turned out to also be an acceptable solution. Finally, in the above language model, the dot product between archive and submission representation is used to measure similarity; other metrics could also be used such as the KL-divergence.

Latent Dirichlet Allocation (LDA): LDA is a unsupervised probabilistic method used to model documents. Specifically we can utilize the topic proportions as found by LDA in order to represent documents. Equation 1 can then naturally be used to calculate expertise scores from the LDA representations of archives and submissions.

4.2. Supervised score prediction models

Once scores are available, supervised regression methods can be used.

Linear Regressions: The simplest regression model learns a separate model for each reviewer using the word counts of the submissions as features:

$$s_{rp} = \theta_r f(P_p)' \tag{4}$$

where θ_r denotes user specific parameters. This method has been shown to work well in practice, particularly if many scores have been elicited from each reviewer. We have explored a number of variants of this simple regression model, including nonlinear and ordinal forms. These did not offer any significant performance gains.

One issue is that in a conference setting it is more typical for some reviewers to have very few or even no observed scores. It may then be beneficial to allow for parameter sharing between users and papers. Furthermore, re-using information from each reviewer's archive may also be beneficial. One method of sharing parameters in a regression model was proposed by John Langford, co-program chair of ICML:

$$s_{rp} = b + b_r + (\theta + \theta_p) f(P_p)' + (\omega + \omega_r) g(A_r)' \tag{5}$$

where b is a global bias, θ and ω are parameters shared across reviewers and papers, which encode weights over features of submissions and archive respectively. b_r, θ_p and ω_r are parameters specific to each papers or reviewer. For ICML-12 $f(P_p)$ was paper p's word counts while $g(A_r)$ was the normalized word count of reviewer r's archive (same as used in the language model). For that conference, and since, that model is trained in an online fashion, using Vowpal Wabbit, with a squared loss and L_2 regularization. In practice, since certain reviewers have few or no observed scores, one also has to be careful to properly weight the regularizers of the different parameters such that the shared parameters are learned at the expense of the individual parameters.

Probabilistic Matrix Factorization: Predicting reviewer paper scores can be seen as a collaborative filtering task. PMF is probabilistic extension of SVD which has proved very successful for certain canonical collaborative filtering tasks (Salakhutdinov & Mnih, 2008). Scores are (probabilistically) modelled by the dot product between two low-rank matrices: $s_{rp} = \theta_r \omega_p'$. Since PMF does not use any information about either papers or reviewers its performance suffers in the cold-start regime. Nonetheless it remains an interesting baseline to compare against.

We have also explored several other learning methods, including a discriminative Restricted Boltzmann Machine (Larochelle et al., 2012).

4.3. Matching formulation

Matching is the process of assigning papers to reviewers. This assignment can be formulated as an optimization problem with the following constraints and goal: a) each paper must be reviewed by a certain number of reviewers R_{target}; b) reviewers have a limit on the number of papers that they can review $P_{\max}$; c) while satisfying the above constraints, conference organizers would like to assign the best reviewers for each paper. Such desiderata can be expressed by the following optimizing problem (Taylor, 2008):

$$\begin{aligned} \text{maximize} \quad & J(y) = \sum_r \sum_p s_{rp} y_{rp} \qquad (6) \\ \text{subject to} \quad & x_{rp} \in \{0, 1\}, \quad \forall r, p \\ & \sum_r y_{rp} = R_{target}, \quad \forall p \\ & \sum_p y_{rp} \leq P_{\max}, \quad \forall r. \end{aligned}$$

In this objective a reviewer's expertise is the sole factor determining a his reviewing quality. There are likely other facets that affect the quality of a reviewer (e.g., how much time a reviewer can allocate to his reviews), however expertise is the only one we can readily evaluate from this data. We have also explored other settings where the utility is a non-linear function of expertise (Charlin et al., 2011).

4.4. Evaluations

A machine learning model is typically assessed by evaluating its performance on a task of interest. For example, we can evaluate how well we do at the score prediction task by comparing the predicted scores to the ground truth scores. However, the task we are most interested in is the one of finding good paper reviewer assignments. Ideally we would be able to compare the quality of our assignments to some gold standard assignment. Such an assignment could then be used to test both different score prediction models as well as different matching formulations. Since a ground truth assignment is unavailable we explore different metrics to test the performance of the overall system, including method for comparing the score prediction models.

4.4.1. Datasets

NIPS-10: This dataset consists of 1251 papers submitted to the NIPS 2010 conference. Reviewers consist of 48 area chairs. The submission and archive vocabulary consists of 22,535 words. *ICML-12:* This dataset consists of 857 papers and 431 reviewers from the ICML 2012 conference. The submission and archive vocabulary consists of 21,409 words.

4.4.2. Initial score quality

We first examine the quality of the initial scores; those estimated solely by comparing the archive and the submissions without access to elicited scores. We will compare the performance of a model which uses the archive and submission representation in word space to one which uses their representation in topic space. The method that operates in word space is the language model as described by Equation 2. For comparison purposes we further normalized these scores using the submission's length. We refer to this method as *word-LM*. For learning topics we used LDA to learn 30 topics over both archives and submissions. For the archive we learned topics for each reviewer's paper and then averaged a reviewer's papers in topic space. This method is denoted *topic-LM*.

	NIPS-10	ICML-12
NDCG@5	0.926	0.867
NDCG@10	0.936	0.884

Table 1. Evaluating the similarity of the top-ranked reviewers for word-LM versus topic-LM on the NIPS-10 and ICML-12 datasets.

We first compare the two methods to one another by comparing the top ranked reviewers for each paper according to both methods. Table 1 reports the average similarity of the top 10 reviewers using NDCG where topic-LM is used to sort. The high value of NDCG indicates that both rankings are very similar on average.

We can get a better appreciation of the rankings of each model by plotting the model's (top) scores for each paper. Each datum on Figure 3 shows the score of one of the top 40 reviewers for a particular paper. For visualization purposes points corresponding to the same reviewer ranking across papers are connected. [10] For topic-LDA (top two figures) the model is good at separating a top-few reviewers from the rest where as word-LDA tends to concentrate all reviewers after the top one or two. The behavior of topic-LDA seems sensible: for a typical paper there are a few very qualified reviewers followed by numerous reviewers with decreasing expertise. One possible explanation for these discrepancies is that working in topic-space removes some of the noise present in word-space. Specifically elements like writing style of individual authors may be abstracted away by moving to topic space.

We could also compare word-LM and topic-LM on matching results. However such results would be biased toward word-LM since, for both datasets, it was used to produce initial scores which guided the elicitation of scores from reviewers (we have validated experimentally that word-LM slightly outperforms topic-LM using this experimental procedure). Using the same experimental procedure word-LM also outperforms matching based on CMT subject areas.

In-vivo experiments are a good way to measure the quality of TPMS scores and ultimately the usefulness

[10] This methodology was suggested and first experimented with by Bill Triggs, the program chair for ICGVIP 2012.

of the system. ICML 2012's program chairs experimented with different initial scoring methods using a special interface which showed ranked candidate papers to reviewers. [11] The experiment had some biases: the papers of the three groups were ranked using TPMSscores. The poll, which asked after the fact if reviewers had found the ranked-list interface useful, showed that reviewers who had used the list based on word-LM were slightly more likely to have preferred the list than the regular CMT interface (the differences were likely not statistically significant).

ICCV-2013 asked its authors to suggest area chairs that would be suitable for reviewing their submissions. We will use this data to compare word-LM and topic-LM to CMT's subject areas.

4.4.3. Final score quality

The final scores can be evaluated straightforwardly. First, we can run score prediction experiments where performance of the different models is measured on a held-out set of scores. Similarly matching performance for different models and different number of observed scores per user can also be compared.

Table 2 reports the RMSE test performance of three models, *LR* (Equation 4), *LR-shared* (Equation 5), and PMF. Each dataset was split into five folds and hyper-parameters were tuned using a single fold. For PMF rank 3 for NIPS-10 and 5 for ICML-12 did best in validation. LR and LR-shared methods were trained with VW using BFGS. PMF was trained with gradient descent. The two datasets vary in terms of the number of observed scores per user. While the NIPS-10 data has an average 86 observed scores per user at training (min. 57, std. 10), ICML-12 has an average of 29 observed scores per user (min. 5, std. 15). NIPS-10 is representative of the particlar eliciation process that was used for this conference's area chairs. ICML-12 is somewhat more representative of typical conference reviewer bidding. Results show that the way LR-shared shares parameters is useful for both datasets. Further we note that although the NIPS-10 data had many observed scores for each user, each paper only has an average of 3 observed scores which partly explains the bad performance of PMF.

Matching results, which we do not report here, are similar in the sense that better score predictions lead to better matching performance.

	NIPS-10	ICML-12
LR	$0.97 \pm 2.5 \times 10^{-4}$	$1.01 \pm 1.4 \times 10^{-4}$
LR-shared	$0.94 \pm 2.6 \times 10^{-4}$	$0.99 \pm 1.1 \times 10^{-4}$
PMF	$1.02 \pm 3.3 \times 10^{-4}$	$1.09 \pm 7.7 \times 10^{-5}$
Constant	$1.05 \pm 7.3 \times 10^{-5}$	$1.07 \pm 1.11 \times 10^{-5}$

Table 2. Evaluating the score prediction performance (RMSE) of three different methods on the NIPS-10 and ICML-12 datasets.

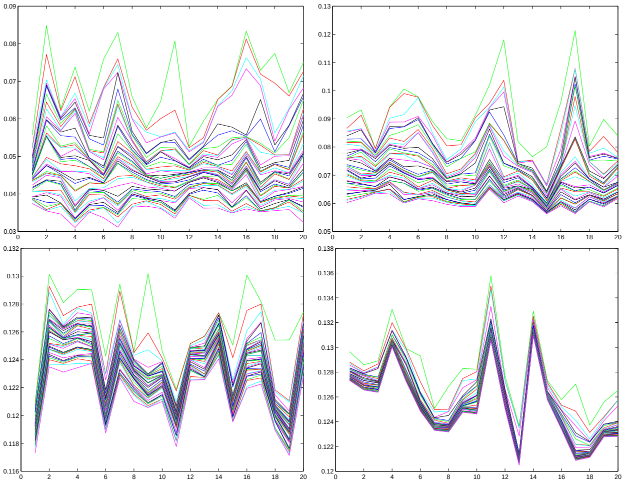

Figure 3. Score of top 40 reviewers for 20 randomly selected submitted papers. Top: topic-LM, Bottom: word-LM, Left: NIPS-10, Right: ICML-12.

5. Plans & Sustainability

We are evaluating a couple options for what to do with TPMS going forward. Currently we ask conferences to help with some of the costs of the system, namely the time commitment. Unless additional manpower is devoted to TPMS we do not see the current model as being sustainable. The main problem is that tailoring the system to each conference's workflow requires considerable time and effort, and the main worker is graduating and moving on to other adventures.

One possibility we are considering is to provide a weakly supported public release that conference organizers could download and use locally. This has some drawbacks such as requiring extra time from the conference organizers as well probably extra support from Microsoft's CMT. Further a central system that has data from multiple conferences can be advantageous. For example, TPMS has been able to store and re-use reviewers' archive from conference to conference. On the other hand it allows interested conference organizers and system developers to improve and extend the current system.

[11] http://hunch.net/?p=2407

Another possibility would be to further open up our platform but keep it, along with all harvested data, in a central location. The aim is to allow conference organizers to use the base system and implement any other features that could be useful to their specific conference with minimal help from us.

6. Current & future directions

There are two main directions that we are currently pursuing. The first is a study of more thorough collaborative filtering with textual side-information models. We are especially interested in exploring methods that perform across a full range of missing data per reviewer. That is, the model should handle a mixed set of reviewers, ranging from cold-start (i.e., reviewers with no scores) all the way up to reviewers with many scores.

The second area of interest is to explore the active elicitation at different abstraction levels. Specifically, we are interested in having the possibility of querying users about their subject areas of expertise, in addition to their self-assessed expertise about particular papers. For this we have elaborated a topic model over both subject areas, words and scores and are developing a principled active learning approach that can choose between querying about subject areas and papers.

On the software side we aim at further automating the system in order to reduce the per conference cost (both to us and to conference organizers) of using the system. That implies providing automated debugging and inquiry tools for conference organizers. We are also trying to ease the interaction for reviewers with their archive, and to continue to improve the automated archive development (the latest addition is able to harvest the data in one's Google Scholar profile. [12]

We have also identified a few more directions that will require our attention in the future:

A) Re-using reviewer scores from conference to conference: Currently reviewers' archives are the only piece of information that are re-used in-between conferences. It possible that elicited scores could also be used as part of one reviewer's profile that gets shared across conferences.

B) Score elicitation before the submission deadline: Conferences often have to adhere to strict and short deadlines when assigning papers to reviewers after the submission deadline. Hence collecting additional information about reviewers before the deadline may be able to save further time. One possibility would be elicit scores from reviewers about a set of representative papers from the conference (e.g., a set of papers published at the conference's previous edition).

C) Releasing the data: The data that we gathered through TPMS has opened different research opportunities. We are hoping that some of this data can be properly anonymized and released for use by other researchers.

D) Better integration with conference management software: Running outside of CMT (or other conference organization packages) has had advantages but the relatively weak coupling between the two systems also has disadvantages for conference organizers. The more conferences use our system and the better position we will be in for further developing links between TPMS and CMT.

Finally, we are actively exploring ways to better evaluate the accuracy, usefulness and impact of TPMS. The number of conferences that have expressed an interest in the system is a good argument for the usefulness of the system. Now we would like to obtain more detailed data about its accuracy. Of particular interest we would like to evaluate the possible impact that a system like TPMS may have. A few conferences (such as ICML12 and ICCV13) have carried out experiments that provide some evaluation, and additional experiments that can be carried out within conferences' workflows may be valuable in this direction.

Acknowledgments

We thank members of the Machine Learning group at the University of Toronto for their help in developing the system (notably Hugo Larochelle and Amit Gruber) and for valuable feedback on the system. We also thank several chairs of conferences: the NIPS-09 Chairs John Lafferty and Chris Williams (for providing NIPS-09 dataset); the NIPS-10 Co-Chair John Shawe-Taylor (for putting up with the delays and headaches of the initial development and deployment); the ICML-12 Chairs John Langford and Joelle Pineau, and Mahdi Milani Fard (for the experiments and model development); the ICGVIP-12 chair Bill Triggs (for experimental ideas); and the CVPR-12 co-Chair Robert Collins (for debugging).

References

Balog, Krisztian, Azzopardi, Leif, and de Rijke, Maarten. Formal models for expert finding in enterprise corpora. In Efthimiadis, Efthimis N., Dumais, Susan T., Hawking, David, and Järvelin, Kalervo

[12] http://scholar.google.com/

(eds.), *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and development in Information Retrieval (SIGIR-06)*, pp. 43–50, Seattle, Washington, USA, 2006. ACM. ISBN 1-59593-369-7.

Blei, David M., Ng, Andrew Y., Jordan, Michael I., and Lafferty, John. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:2003, 2003.

Charlin, Laurent, Zemel, Richard, and Boutilier, Craig. A framework for optimizing paper matching. In *Proceedings of the Proceedings of the Twenty-Seventh Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-11)*, pp. 86–95, Corvallis, Oregon, 2011. AUAI Press.

Conry, Don, Koren, Yehuda, and Ramakrishnan, Naren. Recommender systems for the conference paper assignment problem. In *Proceedings of the Third ACM Conference on Recommender Systems (RecSys-09)*, pp. 357–360, New York, New York, USA, 2009. ACM. ISBN 978-1-60558-435-5.

Larochelle, Hugo, Mandel, Michael, Pascanu, Razvan, and Bengio, Yoshua. Learning algorithms for the classification restricted boltzmann machine. *JMLR*, 13:643–669, March 2012.

Mimno, David M. and McCallum, Andrew. Expertise modeling for matching papers with reviewers. In Berkhin, Pavel, Caruana, Rich, and Wu, Xindong (eds.), *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 500–509, San Jose, California, 2007. ACM. ISBN 978-1-59593-609-7.

Rigaux, Philippe. An iterative rating method: application to web-based conference management. In *SAC*, pp. 1682–1687, 2004.

Rodriguez, Marko A. and Bollen, Johan. An algorithm to determine peer-reviewers. In *Proceeding of the 17th ACM Conference on Information and Knowledge Management (CIKM-08)*, pp. 319–328, Napa Valley, California, USA, 2008. ACM. ISBN 978-1-59593-991-3.

Salakhutdinov, Ruslan and Mnih, Andriy. Probabilistic matrix factorization. In Platt, J.C., Koller, D., Singer, Y., and Roweis, S. (eds.), *Advances in Neural Information Processing Systems 20 (NIPS)*, pp. 1257–1264. MIT Press, Cambridge, MA, 2008.

Taylor, Camillo J. On the optimal assignment of conference papers to reviewers. Technical Report MS-CIS-08-30, UPenn, 2008.