An Exploratory Study on Content-Based Filtering of Call for Papers

Germán Hurtado Martín[1,2], Steven Schockaert[3], Chris Cornelis[2,4], and Helga Naessens[1]

[1] Faculty of Applied Engineering Sciences, University College Ghent, Ghent, Belgium
[2] Dept. of Applied Math., Comp. Science and Statistics, Ghent University, Belgium
[3] School of Computer Science & Informatics, Cardiff University, UK
[4] Dept. of Computer Science and Artificial Intelligence, University of Granada, Spain

Abstract. Due to the increasing number of conferences, researchers need to spend more and more time browsing through the respective calls for papers (CFPs) to identify those conferences which might be of interest to them. In this paper we study several content-based techniques to filter CFPs retrieved from the web. To this end, we explore how to exploit the information available in a typical CFP: a short introductory text, topics in the scope of the conference, and the names of the people in the program committee. While the introductory text and the topics can be directly used to model the document (e.g. to derive a tf-idf weighted vector), the names of the members of the program committee can be used in several indirect ways. One strategy we pursue in particular is to take into account the papers that these people have recently written. Along similar lines, to find out the research interests of the users, and thus to decide which CFPs to select, we look at the abstracts of the papers that they have recently written. We compare and contrast a number of approaches based on the vector space model and on generative language models.

Keywords: Recommendation, Call for papers, Information retrieval, Language models, Vector space model.

1 Introduction

Nowadays many conferences are organized, resulting in a high number of calls for papers (CFPs). This increasing number of CFPs, however, means for the researchers a substantial amount of time spent looking for potentially interesting conferences. The problem has been addressed in several ways, the most popular being the use of domain-specific mailing lists (e.g. DBWorld[1]), or organizing CFPs per subject on dedicated websites (e.g. WikiCFP[2], CFP List[3],

[1] `http://research.cs.wisc.edu/dbworld/`
[2] `http://www.wikicfp.com`
[3] `http://www.cfplist.com`

or PapersInvited[4]). However, these solutions still require users to spend part of their time searching for CFPs, and the results do not always match their specific interests.

A number of recent techniques have been proposed for recommending scientific resources such as research papers, with the study and emergence of research paper recommenders [1,5], citation recommendation [9], or applications to find experts in a specific research area [2]. However, to our knowledge, CFP recommendation remains unexplored.

Recommenders typically rely on collaborative filtering approaches [10], content-based methods [6], or hybrid methods. It can be expected that a CFP recommender would be most effective when content-based methods are combined with other techniques. However, before such a recommender can be developed, we feel that a number of content-based aspects need to be understood better, including how the research interests of a user can be induced from his publication history and how these interests could be matched to CFPs. The aim of this paper is to explore which methods may be most suitable for this task. In particular, we consider the textual content of the CFP such as the introductory text or the list of topics, and we complement that information with the abstracts of the papers recently written by the members of the program committee who are named in the CFP. This latter idea has already been used to address the review assignment problem [4,11]. Similarly, we use the abstracts of the papers that the users have previously written to discover their research interests.

The paper is structured as follows. First we discuss in more detail what types of information are at our disposal, and how this information can be used. Subsequently, in Section 3 we introduce different methods to effectively model and compare CFPs and user profiles. In Section 4 we present our experimental results. Finally, in Section 5 we summarize the conclusions of the paper.

2 Available Information

2.1 User Representation

To represent the research interests of users we exploit the papers they have written. Since research interests might change, only recent papers are considered. In our experiments we have considered papers written in the last five years as being recent, although more advanced methods could be envisaged to analyze how the research interests of a user are changing over time. Alternatively, in the case of users with few or no papers (e.g. a beginning researcher) users could specify those papers which represent their interests best. Since getting access to the full text of research papers is not always possible, we only use the papers' abstracts. We then consider, for each user, a document consisting of the concatenation of the abstracts of his papers. For the sake of clarity, we further refer to this document as d_{abs}.

[4] `http://www.papersinvited.com`

What we can also learn from an author's publication profile is which authors he frequently cites. This information can be valuable if we consider that authors are more likely to be interested in conferences whose program committee (PC) contains several people who are working in the same field and whose papers they sometimes cite. To take this into account, we will use a document consisting of the concatenation of the abstracts of the papers written by the authors usually cited by the user. In our experiments, we considered an author to be usually cited if at least 3 different papers written by him have been cited by the user in 3 different works. We refer to this document as d_{aut}.

2.2 CFP Representation

For this work we have used CFPs available from DBWorld. Although there is no standard format for writing CFPs, they usually include similar information: an introductory text about the conference, an indicative list of topics that are within the scope of the conference, and the names of the members of the program committee (or at least the organizers). They usually also include important dates and location, but we will disregard that information.

The introductory text usually consists of a short description about the conference which might contain terms that describe the scope of the conference and are therefore important. However, this description often also refers to past conferences, the proceedings, etc., which means that many terms are mentioned that are not representative of the topics of the conference. We try to compensate this by concatenating this text with the list of topics that are within the scope of the conference. We use the resulting document, which we further refer to as d_{txt}, to model a CFP document.

The names of the members of the program committee are also potentially useful. An option to use them directly could be trying to match them to the names cited in the papers of the users, but the results of initial experiments along these lines were not positive. However, these names can be used indirectly too. In particular, for the experiments reported in this paper, we associate each CFP with a document d_{con}, consisting of the concatenation of the abstracts of all papers that have been written in the last two years by its PC members.

Finally, if we want to consider both types of information simultaneously, we can concatenate d_{txt} and d_{con}; we refer to this document as d_{tot}.

3 Matching CFPs and Users

In this section we review some methods to model and compare users and CFPs, based on the documents defined in the previous section.

3.1 Tf-idf

To measure the similarity between a CFP and a user profile we compare them in the vector space model: each profile is represented as a vector, with one component for every term (unigram) occurring in the collection. A CFP is encoded as a vector as follows. Stopwords[5] are first removed, no stemming is used. Then, the weight for each term w_i in the CFP profile is calculated by using the tf-idf scoring technique [8]:

$$tfidf(w_i, d_{txt}) = \frac{n(w_i, d_{txt})}{|d_{txt}|} \cdot log(\frac{|\mathcal{C}_{txt}|}{|\{d_j : w_i \in d_j\}|}) \quad (1)$$

where $\mathcal{C}_{txt}$ is the collection of CFPs made from the concatenation of introductory text and scope topics (i.e., of documents of the form d_{txt}).

As mentioned in the previous section, CFP profiles can be represented in different ways. If the documents of the form d_{con} are used instead of those of the form d_{txt}, d_{txt} and $\mathcal{C}_{txt}$ in Eq. (1) are replaced by d_{con} and $\mathcal{C}_{con}$ respectively, where $\mathcal{C}_{con}$ is the collection of CFPs made from the concatenation of the abstracts of the papers written by the PC members (documents of the form d_{con}). On the other hand, if the documents of the form d_{tot} are used, d_{txt} and $\mathcal{C}_{txt}$ in Eq. (1) are replaced by d_{tot} and $\mathcal{C}_{tot}$ respectively, where $\mathcal{C}_{tot}$ is the collection of CFPs made from the concatenation of both textual content and abstracts of the papers written by the PC members (documents of the form d_{tot}).

Since user and CFP profiles belong to different collections, we consider user profiles as queries, and therefore the process to convert a user profile into a vector is slightly different. As with CFP profiles, stopwords are removed and no stemming is used; however, only those terms that occur in the CFP collection are considered, and the rest are ignored. Then the weight of each term in the user profile is calculated by replacing d_{txt} and $\mathcal{C}_{txt}$ in Eq. (1) by d_{abs}^{txt} and $\mathcal{C}_{txt}$, d_{abs}^{con} and $\mathcal{C}_{con}$ or d_{abs}^{tot} and $\mathcal{C}_{tot}$, depending on the type of information used, where d_{abs}^{txt}, d_{abs}^{con} and d_{abs}^{tot} are obtained from the user profile d_{abs} after removing all terms that do not occur in $\mathcal{C}_{txt}$, $\mathcal{C}_{con}$ and $\mathcal{C}_{tot}$ respectively.

Two vectors $\mathbf{d_1}$ and $\mathbf{d_2}$ corresponding to different profiles can then be compared using a standard similarity measure; we use the cosine similarity, defined by

$$sim_c(\mathbf{d_1}, \mathbf{d_2}) = \frac{\mathbf{d_1} \cdot \mathbf{d_2}}{\|d_1\| \cdot \|d_2\|} \quad (2)$$

where $\mathbf{d_1} \cdot \mathbf{d_2}$ denotes the scalar product and $\|.\|$ the Euclidean norm. In Section 4.2 we refer to the method that combines tf-idf with the cosine similarity measure as *tfidf-txt*, *tfidf-con* and *tfidf-tot*, depending on the information used.

3.2 Language Modeling

A different approach is to estimate unigram language models [7] for each document, and determine their divergence. A user profile or CFP d is then assumed to be generated by a given model D. This model is estimated from the terms that occur in d and in the other CFPs from the considered collection. Using

[5] The list of stopwords we have used for the experiments was taken from `http://snowball.tartarus.org/algorithms/english/stop.txt`, expanded with the following extra terms: *almost*, *either*, *without*, and *neither*.

Jelinek-Mercer smoothing, the probability that model D corresponding to a CFP generates term w is estimated as:

$$P^*(w|D) = \lambda P(w|d_{txt}) + (1-\lambda)P(w|\mathcal{C}_{txt}) \tag{3}$$

where $\mathcal{C}_{txt}$ is the collection of CFP profiles as defined in Section 3.1, and the probabilities $P(w|d)$ and $P(w|\mathcal{C})$ are estimated using maximum likelihood, e.g. $P(w|d)$ is the percentage of occurrences of term w in profile d. Alternatively, d_{txt} and $\mathcal{C}_{txt}$ in Eq. (3) can be replaced by d_{con} and $\mathcal{C}_{con}$ if the documents of the form d_{con} are used. To estimate the probability that the model of a user profile generates a given term w we simply replace d_{txt} in (3) by d^{txt}_{abs} or d^{con}_{abs} (as defined in Section 3.1).

Once the models D_1 and D_2 corresponding to a user profile d_1 and a CFP d_2 are estimated, we measure their dissimilarity using the Kullback-Leibler divergence:

$$KLD(D_1||D_2) = \sum_w D_1(w) log \frac{D_1(w)}{D_2(w)} \tag{4}$$

We further refer to these methods as *lm-txt* and *lm-con*.

However, if we want to consider both kinds of information jointly (i.e. the information from the documents of the form d_{txt} and that from the documents of the form d_{con}), language model interpolation is used:

$$P^*(w|D) = \lambda_1 P(w|d_{txt}) + \lambda_2 P(w|d_{con}) + \lambda_3 P(w|\mathcal{C}_{txt}) + \lambda_4 P(w|\mathcal{C}_{con}) \tag{5}$$

with $\sum_i \lambda_i = 1$ and where

$$\lambda_3 = \begin{cases} \frac{1-\lambda_1-\lambda_2}{2}, & \text{if } \lambda_1, \lambda_2 > 0 \\ 1-\lambda_1, & \text{if } \lambda_2 = 0 \\ 1-\lambda_2, & \text{if } \lambda_1 = 0 \end{cases} \qquad \lambda_4 = \begin{cases} \frac{1-\lambda_1-\lambda_2}{2}, & \text{if } \lambda_1, \lambda_2 > 0 \\ 1-\lambda_1, & \text{if } \lambda_2 = 0 \\ 1-\lambda_2, & \text{if } \lambda_1 = 0 \end{cases}$$

To estimate the probability for the user profile, we replace d_{txt} and d_{con} in Eq. (5) by d^{txt}_{abs} and $d^{con}_a bs$. In Section 4.2 we refer to this method as *lm-tot*.

3.3 Feature Selection

As mentioned in Section 2, the introductory texts of the CFPs often contain information about past editions of the conference or brief submission guidelines. This leads to the use of a number of relatively common terms, which are irrelevant for characterizing the scope of a conference. To eliminate such unwanted terms, we use the term strength method from [12]. This method is based on the idea that terms shared by closely related documents are more informative than others. The strength of a term w is thus computed by estimating the probability that a term w occurs in a document d_1 given that it occurs in a related document d_2:

$$strength(w) = P(w \in d_1 | w \in d_2) \tag{6}$$

If there is no information available regarding the relatedness of the documents, as in our case, the pairs of related documents (d_i, d_j) must first be identified. To this end, we have simply used method *tfidf-txt* from Section 3.1, and the pairs with a similarity degree above a certain value v are considered as related. We calculate v iteratively using a threshold γ, which represents the average number of documents we want each document to be related to (i.e., the average number of pairs (d_i, d_j) for each d_i). First, a random value between 0 and 1 is set as initial value of v, and all documents are compared. If the average number of related documents per document is above γ (i.e., each document is related to too many documents), the value of v is raised, and the process is repeated until the average number of related documents is below γ. Since a too small number of related documents is not desirable either, a second threshold γ' can be used to prevent that. In our case we set $\gamma = 20$, $\gamma' = 10$ as satisfactory performance is achieved in that range [12].

After calculating $strength(w)$ for every term w in the CFP collection, the N strongest terms are selected, ignoring the rest. For our experiments in Section 4 we have used $N = 500$ and $\mathcal{C}_{txt}$, since that combination performed well in early tests. The documents are then modelled as in Sections 3.1 and 3.2. When referring to particular methods in Section 4.2 below, we indicate when feature selection was used by adding the suffix *-fs* to the name of the method.

3.4 Related Authors

To reflect users' interest for those conferences whose PC members they are familiar with we propose to calculate extra models exclusively based on papers and compare them. Specifically, we compare the CFP model based on the concatenation of the abstracts of the papers written by the PC members (d_{con}) with a user model based on the concatenation of the abstracts of the papers written by the researchers usually cited by that particular user (d_{aut}). Depending on the type of model, the CFP model based on d_{con} is made according to method *tfidf-con* or *lm-con*. For the user model based on d_{aut} we simply replace d_{con} by d_{aut}^{con} in the definitions of those methods.

The method used to create and compare these extra models is always analogous to that used to calculate the original result, e.g. if the original result is obtained with method *lm-txt*, method *lm-con* is used to calculate these extra models, and they are then compared using the Kullback-Leibler divergence.

The idea is to use these models to complement the result obtained with the methods seen in the previous sections. In particular, once the models are created and compared, we simply combine the result with that of the original comparison by means of the weighted average. For example, to compare CFP *cfp* and user u with method *tfidf-txt*, the result was given by $sim_c(\mathbf{cfp_{txt}}, \mathbf{u_{txt}})$. However, if we take into account these extra models based on d_{con} and d_{aut} (in this case, cfp_{con} and u_{aut}), the result is now given by:

$$\alpha \cdot sim_c(\mathbf{cfp_{txt}}, \mathbf{u_{txt}}) + \beta \cdot sim_c(\mathbf{cfp_{con}}, \mathbf{u_{aut}}) \tag{7}$$

where $\alpha + \beta = 1$. Based on preliminary experiments, we use $\alpha = 0.8$ and $\beta = 0.2$ for the experiments in Section 4.2. We indicate that these extra models are used by adding the suffix *-nam* to the name of the method.

3.5 Related Authors and Feature Selection

Finally, both previously introduced variations can be combined: first, feature selection is applied, which also reduces the number of terms in the extra models based on the frequently cited authors, and then, as explained in the previous subsection, the models are compared separately, to finally combine the results. We indicate that this variation is used by adding the suffix *-fn* to the name of the method.

4 Experimental Evaluation

4.1 Experimental Set-Up

To build a test collection and evaluate the proposed methods, we downloaded 1769 CFPs from DBWorld, which reduced to 1152 CFPs after removing duplicates. Additionally, those CFPs lacking an introductory text or an indicative list of topics were removed too, which further reduced the total number to 969 CFPs. Each of these CFPs has a text part (union of introductory text and topics) and a concatenation of the abstracts of the papers written by the PC members in the last 2 years[6], where available.

On the other hand, 13 researchers from a field which relates to the scope of DBWorld took part in our experiments as users. In order to profile them, we downloaded the abstracts of the papers they wrote in the last 5 years. The ground truth for our experiments is based on annotations made by these 13 users. In a first experiment, each user indicated, for a minimum of 100 CFPs, whether these were relevant or not (relevance degree of 1 or 0 respectively). Then, using each of the studied methods, the CFPs annotated by the users were ranked such that ideally the relevant CFPs appear at the top of the ranking.

In a second experiment, we considered only CFPs assessed as highly relevant by at least one of the methods. To this end, we selected for each user and each of the 24 studied methods the top-5 CFPs of the rankings obtained in the first experiment. This resulted in 120 CFPs, which reduced to an average of about 50 CFPs per user due to overlap between the top-5 CFPs returned by each method. Each of those CFPs was then rated by the user, who gave them a score between 0 ("totally irrelevant") and 4 ("totally relevant"). Again, using each of the studied methods, these CFPs were ranked such that ideally the most relevant CFPs appear at the top of the ranking.

To evaluate the rankings resulting from both experiments, for each user and each method we use normalized discounted cumulative gain (nDCG) [3] to measure the relevance of each CFP according to its position in the ranking. The idea

[6] All the information regarding research papers was retrieved from the ISI Web of Science, `http://apps.isiknowledge.com`

of this measure is that the greater the ranked position of a relevant document, the less valuable it is for the user, as users tend to examine only those documents ranked high, except if those documents do not satisfy their information needs, in which case it is more likely that they still consider lower ranked documents. This is reflected by the discounted cumulative gain of the document ranked in position r:

$$DCG_r = rel_1 + \sum_{i=2}^{r} \frac{rel_i}{log_2 i} \tag{8}$$

The relevance rel_i of the document ranked in position i is the relevance indicated by the user, i.e. 0 or 1 for the first experiment, and 0, 1, 2, 3 or 4 for the second experiment.

Since the number of CFPs annotated by each user might be different, the length of the obtained rankings varies. In order to compare the DCG values we need to calculate the normalized DCG:

$$nDCG_r = \frac{DCG_r}{iDCG_r} \tag{9}$$

where $iDCG_r$ is the ideal DCG at position r: the DCG obtained at position r in the ideal case where all documents are perfectly ranked, from most to least relevant, according to the users' annotations.

For both experiments in Section 4.2 we work with the nDCG of the CFP ranked in the last position, i.e. $\mathrm{nDCG_r}$ where r is the total number of CFPs in the ranking, as this value reflects the gains of all the CFPs throughout the whole ranking.

4.2 Results

Tables 1 and 2 summarize the results of the first and second experiment respectively. In particular, for each method we show the average $\mathrm{nDCG_r}$ for the 13 users, where r is the number of CFPs in the ranking for each user as indicated in the previous section. For the sake of simplicity we have used some fixed values for the λ parameters of (5) in the methods based on language modeling. In particular, we use $\lambda_1 = 0.9$ and $\lambda_2 = 0$ for the *lm-txt* method (i.e. analogously to *tfidf-txt*, it only uses the information from the text parts of the CFPs); $\lambda_1 = 0$ and $\lambda_2 = 0.9$ for the *lm-con* method; and $\lambda_1 = 0.4$ and $\lambda_2 = 0.4$ for the *lm-tot* method.

First we compare the different kinds of information that can be used: introductory text plus topics, concatenation of the abstracts of the papers recently written by the PC members, or the concatenation of both. Figures 1 and 2 show that using the abstracts alone (*con*) does not suffice to outperform the methods based on the textual content (*txt*), while those based on the concatenation of abstracts and textual content (*tot*) seem to perform comparably or slightly better than the *txt* methods, except for the language model based methods without feature selection. If we fix the method and the use of feature selection or abstracts written by frequently cited authors, we can see that the differences,

Table 1. Ranking of methods for the first experiment, nDCG values

Method	nDCG	Method	nDCG	Method	nDCG
tfidf-tot-fs	0.606	tfidf-con-fn	0.553	lm-txt	0.51
tfidf-tot-fn	0.599	tfidf-con-nam	0.551	lm-tot	0.493
lm-tot-fs	0.575	tfidf-con-fs	0.549	lm-con-fs	0.493
tfidf-tot	0.563	tfidf-con	0.544	lm-txt-nam	0.482
tfidf-txt-fn	0.563	tfidf-txt	0.542	lm-con-fn	0.469
tfidf-txt-nam	0.562	lm-txt-fs	0.529	lm-con	0.44
tfidf-tot-nam	0.561	lm-tot-fn	0.516	lm-tot-nam	0.436
tfidf-txt-fs	0.555	lm-txt-fn	0.512	lm-con-nam	0.421

Table 2. Ranking of methods for the second experiment, nDCG values

Method	nDCG	Method	nDCG	Method	nDCG
lm-tot-fs	0.745	lm-txt	0.691	tfidf-con-fn	0.646
tfidf-txt-nam	0.728	lm-tot-fn	0.686	tfidf-con	0.637
tfidf-txt	0.715	lm-txt-fn	0.682	tfidf-con-nam	0.636
tfidf-tot-fs	0.713	tfidf-tot	0.661	lm-con-fs	0.606
tfidf-txt-fn	0.707	lm-tot	0.653	lm-con-fn	0.587
tfidf-tot-fn	0.706	tfidf-con-fs	0.649	lm-tot-nam	0.566
tfidf-txt-fs	0.705	tfidf-tot-nam	0.648	lm-con	0.555
lm-txt-fs	0.700	lm-txt-nam	0.647	lm-con-nam	0.502

although real, are not significant[7] enough, except for *tfidf-con-nam*, *lm-con* and *lm-con-fs*, which perform worse than *tfidf-tot-nam*, *lm-tot* and *lm-tot-fs* in Experiment 1. In Experiment 2, the differences among all language model cases are significant except for those between *lm-txt-fs* and *lm-con-fs*, *lm-txt-fn*/*lm-con-fn*, *lm-txt*/*lm-tot*, and *lm-txt-fn*/*lm-tot-fn*. On the contrary, the differences in the vector space model cases are not significant except for *tfidf-tot*/*tfidf-con*, and those between the methods involving *nam*.

To study the impact of feature selection (*fs*), the models based on frequently cited authors (*nam*) and the combination of both (*fn*), we fix the method and the type of information used. In general, the best results are obtained when feature selection is applied. It must be noted, however, that these differences are only significant in some cases: the *con* and *tot* cases of the language model based methods in Experiment 1, and the *tot* case of the language model based methods in Experiment 2. On the other hand, results obtained with the *nam* methods are worse than the original, with significant differences for the language model based methods in Experiment 2. As for *fn*, it usually improves the original results, but there is no significant evidence of this.

Finally, we compare the methods based on the vector space model with those based on language modeling. In Figures 1 and 2 we can observe that the former generally outperform the latter. Some methods based on language modeling (*lm-txt-fs* and *lm-tot-fs* in Experiment 1, joined by *lm-txt*, *lm-tot*, *lm-txt-fn* and

[7] In this work we consider a difference to be significant when $p < 0.05$ for the Mann-Whitney U test.

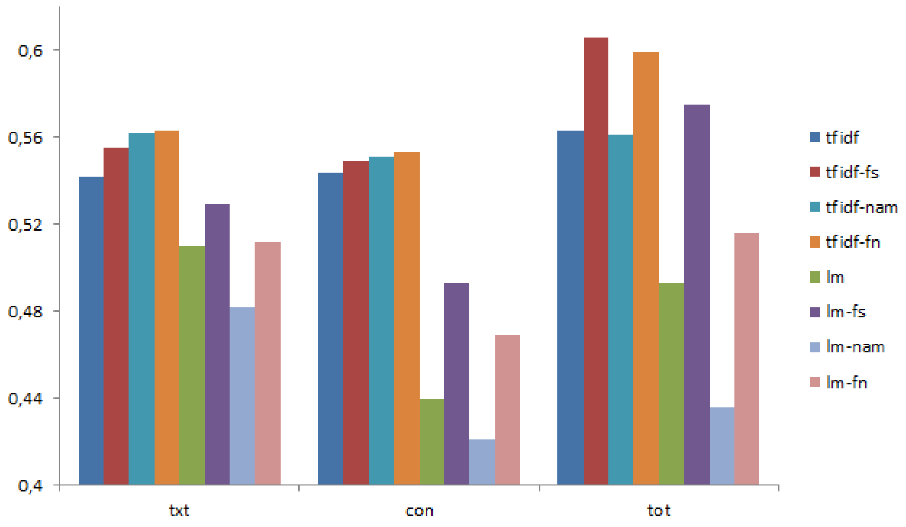

Fig. 1. Results for experiment 1; the Y-axis shows the nDCG, while the X-axis indicates the kind of information used

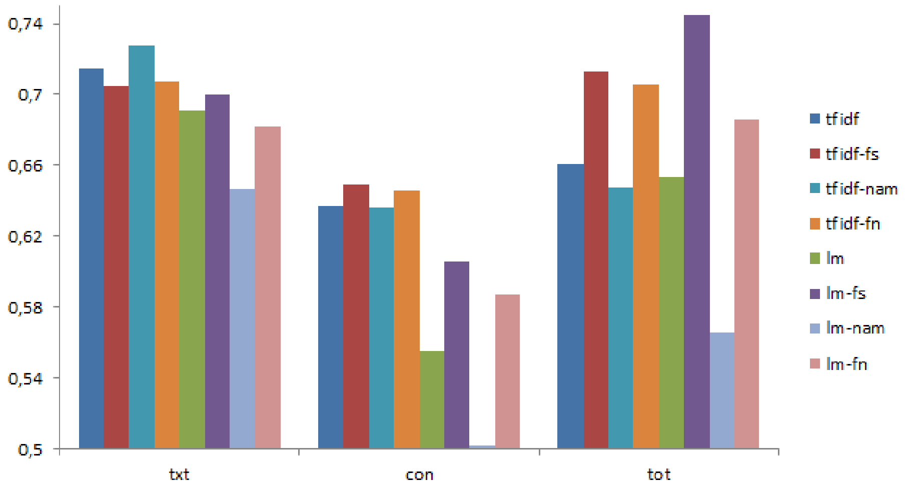

Fig. 2. Results for experiment 2; the Y-axis shows the nDCG, while the X-axis indicates the kind of information used

lm-tot-fn in Experiment 2) perform comparably to those based on the vector space model, but although both vector space model and language model based approaches can achieve good results, the former appear to be much more robust against changes in the particular way in which CFPs are modelled. In a comparison where the information type and the use of feature selection is fixed (e.g. we compare *tfidf-txt-fs* and *lm-txt-fs*) methods based on the vector space model significantly outperform those based on language modeling in many cases. In Experiment 1 this is the case of *txt-nam*, *txt-fn*, the *tot* methods except *tot-fs*, and all *con* methods. On the other hand, in Experiment 2 differences are significant for *txt-nam*, *tot-nam*, and the *con* methods except *con-fs*.

5 Conclusion

We have proposed and compared several content-based methods to match users with CFPs. We have studied the impact of the different types of information available, the accuracy of the models that represent such information, and the effect of feature selection on these models. Also, using the users' names and the names of the CFP members we have accessed the papers recently written by them to profile the users and to complete available information about the CFP respectively. Information about authors frequently cited by the users is also used to reflect the importance given by the users to the CFPs of conferences with people in the PC working in the same field and whose work they usually cite.

The results indicate that methods based on the vector space model are generally more robust, and achieve the best performance on this task. Both for vector space models and language models, feature selection improved the results.

Finally, we have also seen that although the abstracts of the papers written by the PC members can be helpful when combined with other information, the resulting models are not sufficiently accurate to be used on their own.

As mentioned in the introduction, we remark that content-based approaches alone do not suffice to cover all the aspects of the task of matching users and CFPs, as the relevance of a conference depends also on information not contained in the text of the CFPs. Therefore, the studied content-based methods should be complemented with other techniques, which provides an interesting starting point for future work. Collaborative filtering would be of great help; in this case a given CFP could be matched to a user because another user with similar interests attended a previous edition of that conference. Alternatively, a user could get a notification about a CFP because that given conference covers similar topics as another conference he attended in the past. Also, trust-based methods could reflect additional information not covered by collaborative filtering: a user could then be notified about a conference because a researcher he trusts is in the program committee, or because he trusts the conference given its impact on his research field.

References

1. Bogers, T., van den Bosch, A.: Recommending scientific articles using CiteULike. In: Proc. of the 2008 ACM Conf. on Recommender Systems, pp. 287–290 (2008)
2. Deng, H., King, I., Lyu, M.R.: Formal models for expert finding on DBLP bibliography data. In: Proc. of the 8th IEEE International Conference on Data Mining, pp. 163–172 (2008)
3. Järvelin, K., Kekäläinen, J.: IR evaluation methods for retrieving highly relevant documents. In: Proceedings of the 23rd Annual International ACM SIGIR Conf. on Research and Development in Information Retrieval, pp. 41–48 (2000)
4. Karimzadehgan, M., Zhai, C., Belford, G.: Multi-aspect expertise matching for review assignment In. In: Proc. of the 17th ACM Conference on Information and Knowledge Management, pp. 1113–1122 (2008)
5. McNee, S.M., Albert, I., Cosley, D., Gopalkrishnan, P., Lam, S.K., Rashid, A.M., Konstan, J.A., Riedl, J.: On the recommending of citations for research papers. In: Proc. of the 2002 ACM Conf. on Computer Supported Cooperative Work, pp. 116–125 (2002)
6. Pazzani, M.J., Billsus, D.: Content-based recommendation systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) Adaptive Web 2007. LNCS, vol. 4321, pp. 325–341. Springer, Heidelberg (2007)
7. Ponte, J.M., Croft, W.B.: A language modeling approach to information retrieval. In: Proc. of the 21st Annual International ACM SIGIR Conf. on Research and Development in Information Retrieval, pp. 275–281 (1998)
8. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management 24, 513–523 (1988)
9. Strohman, T., Croft, W.B., Jensen, D.: Recommending citations for academic papers. In: Proc. of the 30th Annual International ACM SIGIR Conf. on Research and Development in Information Retrieval, pp. 705–706 (2007)
10. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Advances in Artificial Intelligence 2009, 4:2–4:2 (2009)
11. Tang, W., Tang, J., Tan, C.: Expertise Matching via Constraint-Based Optimization. In: Proc. of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, vol. 1, pp. 34–41 (2010)
12. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Proc. of the 14th International Conference on Machine Learning, pp. 412–420 (1997)