PMSC-UGR: A Test Collection for Expert Recommendation Based on PubMed and Scopus

César Albusac, Luis M. de Campos, Juan M. Fernández-Luna(✉), and Juan F. Huete

Departamento de Ciencias de la Computación e Inteligencia Artificial, ETSI Informática y de Telecomunicación, CITIC-UGR, Universidad de Granada, 18071 Granada, Spain
calbusac@ugr.es, {lci,jmfluna,jhg}@decsai.ugr.es

Abstract. A new test document collection, PMSC-UGR, is presented in this paper. It has been built using a large subset of MEDLINE/PubMed scientific articles, which have been subjected to a disambiguation process to identify unequivocally who are their authors (using ORCID). The collection has also been completed by adding citations to these articles available through Scopus/Elsevier's API. Although this test collection can be used for different purposes, we focus here on its use for expert recommendation and document filtering, reporting some preliminary experiments and their results.

Keywords: Test collection · Authors disambiguation
Expert finding · Document filtering · MEDLINE/PubMed · Scopus

1 Introduction

In most areas of science it is necessary to test the possible merits and advantages of new methods and techniques against the state-of-the-art, as well as to compare competing new methods among them. To do that it is necessary to have benchmark data available where we can compare the different alternatives in a controlled environment. This is also true in the field of recommender systems and within expert finding [2] (also known as expert search or expert recommendation), where the goal is to recommend persons (experts) given a topic of interest for a given user. A different, but formally very related problem, is that of document filtering [3,9], where the goal is to decide which users should receive (because they are interested in) a new incoming document from a document stream. In both cases we have a set of individuals (experts or users) which are characterized by some kind of profile[1], and a "query" (the topic of interest or the document to be filtered). The system must be able to decide which individuals

[1] Profile extracted, for example, from documents authored by this individual.

F. Herrera et al. (Eds.): CAEPIA 2018, LNAI 11160, pp. 34–43, 2018.
https://doi.org/10.1007/978-3-030-00374-6_4

are more related to this query, according to their profiles. In such a context, benchmark data consists in a test collection composed of documents which can be associated to their authors. In this paper we describe a new test collection, primarily based on MEDLINE/**P**ub**M**ed scientific documents but also using the **SC**opus tool from Elsevier, PMSC-UGR, which can be used for expert finding and document filtering but also for other tasks.

In developing the collection, we have invested considerable effort in the disambiguation of the authors names. This is important for expert finding because different experts may have exactly the same name (or the same surname and the same initials), or the name of a given expert can appear in different versions. The only way to avoid ambiguities is to use a unique identifier, like ORCID. The problem is that in the PubMed articles the authors are not always identified by their ORCID. We have used Elsevier's API to complete the collection by adding validated authors (i.e. unambiguously identified by their ORCID codes) to those articles that do not have them but it is checked that they should be.

Moreover, we have enlarged the collection by adding available citations (also through the Elsevier's API) to the PubMed articles. In this way we can use these citations to create graphs of either articles or authors. These networks can be used, for example, to enlarge/improve the profile of an author *au* by taking into account information about either other authors or articles that cite the articles authored by *au*.

Our collection contains neither explicit queries nor explicit relevance judgements, which are necessary to evaluate any proposed model. We propose to use a machine learning based methodology for the evaluation of the models, splitting the dataset into training and test set. Then, given an article in the test set, our objective is to find –from the information in the training data– those researchers that might be interested in this article. As query, we propose the use of information from the article, such as the title, the abstract or the keywords. With respect to the relevance judgements, we consider that the experts who are relevant for a query are own authors of the article and/or the authors citing this article. Note that by means of this approach not everyone who is interested in the paper is in the judgements list, but everyone on it has expressed his/her interest. So, we can evaluate a model over a large number of pairs (automatic query – implicit relevance judgements).

The remainder of this paper is structured as follows: Sect. 2 outlines related work. In Sect. 3 we give details about how the PMSC-UGR test collection was built. Section 4 describes how the collection can be used in the evaluation of expert search and document filtering methods, together with some preliminary results. Section 5 outlines other possible uses of the test collection. Finally, Sect. 6 contains the concluding remarks.

2 Related Work

The are several collections suitable for expert finding. For example, some of them are LExR [10], ArnetMiner [12], CERC [1] and W3C [7].

W3C collection includes W3C working groups members as candidate experts (a total of 1,092 experts), 331,037 documents (mainly emails) and 50 queries (the topics were the own working groups, and working group membership was considered as the ground truth). The CERC collection includes 3,500 Commonwealth Scientific and Industrial Research Organisation (CSIRO) employees as candidate experts, contains 370,715 documents and 50 topics (created by SCIRO Science Communicators, which include a short list of key contacts used as ground truth). ArnetMiner is a test collection focused in an academic setting, the candidate experts are computer science researchers. The publication data come from online databases including DBLP, ACM Digital library, Citeseer, and others. ArnetMiner is a large collection, it includes 1,048,504 researcher profiles and 3,258,504 publications, and also contains citations (although the reported number of queries used for expert finding is only 13). Also from an academic setting, the most recent test collection is LExR, where candidate experts are researchers working in Brazil. Initially it contains 206,697 researchers and 11,942,014 references to publications (although only 483,222 of these references include the abstract). The number of available queries for expert finding is 235.

As we shall see in the next section, PMSC-UGR is much larger than W3C and CERC (in both number of documents and experts). Also, the number of queries we can use in our collection (as many as documents in the test set) is much larger than those in all the other collections. Therefore, the results obtained using PMSC-UGR can generate more reliable conclusions, from a statistical point of view. Finally, other important points in favor of PMSC-UGR are that it includes citations (only ArnetMiner also contains citations) and the strict disambiguation process carried out, which avoids incorrect attribution of papers to authors.

3 Building the PMSC-UGR Test Collection

This section describes in some detail the steps followed to build our test collection, starting from the MEDLINE/PubMed collection, which contains articles for biomedical literature from MEDLINE. This collection is property of the US National Library of Medicine (NLM).

3.1 MEDLINE/PubMed Collection

The initial step was to download the complete colleccion of articles[2] from PubMed[3]. The download was carried out on June 8, 2017. In this collection, there are 892 files in XML format and inside each one, there are around 30,000 articles. In this way, the initial collection contained 26,759,991 articles, being possible to find a lot of information in each article. Some of the fields are:

[2] Although PubMed does not contain complete articles but references to articles, called citations, we will use the term articles to refer to these citations, and reserve the name citations to refer to other articles that cite in their bibliographic references a given article.

[3] ftp://ftp.ncbi.nlm.nih.gov/pubmed/baseline/.

- *PubMedID* is a unique identifier of the article from PubMed.
- *Journal* is the name of the journal where the article was published.
- *ArticleTitle* is the complete title of the article, in English.
- *Abstract* of the article.
- *AuthorList* contains information about the authors of the article. For each one, we can find:
  - *LastName* contains the surname or the single name used by an individual.
  - *ForeName* contains the remainder of name.
  - *Identifier* is a unique identifier associated with the name. The value in the Identifier attribute Source designates the organizational authority that established the unique identifier. For our purposes we will pay special attention to the ORCID identifier.
- *MeshHeadingList* is NLM controlled vocabulary, Medical Subject Headings (MeSH). It is used to characterize the content of the article, using descriptors from this thesaurus.
- *KeywordList* contains controlled terms in Keywords that also describe the content of the article.

Formats and meanings of all fields can be consulted at the official PubMed page[4].

3.2 Disambiguation of Author Names

In a context of expert recommendation, it is important to remove possible ambiguities between authors, in order to not to attribute articles to the wrong author or to lose articles from the right author. To do it, the only and efficient way is to use a unique digital identifier for each author. We decided to use the perhaps more widely accepted identifier for authors of scientific articles, namely the Open Researcher and Contributor ID (ORCID). Therefore, we restricted our PubMed collection to include only authors with an ORCID. There are 112,546 differents authors with ORCID. The problem is that in the PubMed collection frequently occurs that in an article the author appears with his ORCID whereas in another article (possibly older), perhaps of the same author, the ORCID does not appear. Therefore we are not completely sure that both articles belong to the same author. This fact would severely limit the completeness of our document collection, because we should discard a lot of articles where the ORCIDs of the authors do not appear, thus limiting the sources of information about the interests of these authors[5]. The goal is to be able to attribute an article appearing in PubMed to an author with an associated ORCID, although the ORCID does not appear in the original PubMed record. To deal with this problem we will try to solve some ambiguities by using another information source, namely the Scopus database through the Elsevier's APIs[6].

[4] https://www.nlm.nih.gov/bsd/licensee/elements_descriptions.html.

[5] In fact 26,661,157 articles in PubMed have not any ORCID.

[6] The data was downloaded from Scopus API between July 3 and September 27, 2017 via http://api.elsevier.com and http://www.scopus.com.

In order to do it, we need to associate the ORCID of an author with the digital identifier used in Scopus, namely ScopusID, and then search for the articles published by this author within Scopus, finally comparing these articles with those appearing in PubMed.

For each author with ORCID in PubMed we have carried out the following query to Scopus:

`http://api.elsevier.com/content/search/author?query=ORCID(AuthorORCID)&apiKey=yourApiKey`

where *AuthorORCID* is an ORCID and *yourApiKey* is a Key that can be generated once a user is registered in the Elsevier system.

This query obtains the ScopusID for each author as well as his name (given-name, surname and initials) as the result.

After this step, 21,048 different authors with ScopusID were obtained (those authors which can be simultaneously associated with an ORCID and a ScopusID). These authors will be the ones used primarily in this collection. We also included those authors with ORCID in PubMed but not appearing in Scopus (91,498) as a secondary collection[7].

In the next step, for each ScopusID, the following query was created to get information about the articles written by the corresponding author:

`http://api.elsevier.com/content/search/scopus?query=AU-ID(AuthorScopusID)&field=dc:identifier,dc:title,doi,pubmed-id&count=200&apiKey=yourApiKey`

where *AuthorScopusID* is the ScopusID previously obtained and the field parameter specifies wich article data we want to get: *dc:identifier* is the Scopus identifier for the article (ArticleScopusID); *dc:title* is the article title; *doi* is the digital object identifier of the article; *pubmed-id* is de PubMed identifier (if it exists).

In this way, for each author we have obtained all their articles (within Scopus). All this information has been stored in a csv file with the ORCID, ScopusID, ArticleScopusID, Title, doi and PubMedID.

Then, to try to assign validated authors to articles in PubMed where the ORCID of the authors does not appear, the following process was carried out: an index with the search engine library Lucene[8] was created and all the PubMed articles downloaded at the beginning were indexed. The only fields in these articles that were indexed (those which are necessary to our purposes) are Article title, Article authors and PubmedID. Next, for each author au with ScopusID and ORCID, we obtain the list l_{au} of their articles retrieved from Scopus (those safely assigned to au in Scopus) which can be used to complete the information of au in PubMed. Then, for each pair $(au, l_{au}(i))$ we perform the following process: If an article has PubMedID, then it is clear that it belongs to the PubMed collection and moreover it can be safely associated with the ORCID of the author. Otherwise, we will try to set such association algorithmically. Particularly, a query by the article title was run against the PubMed index. Then, focusing

[7] The reason is that for these authors we cannot obtain citations to their articles, so this secondary collection is larger but contains less information.

[8] https://lucene.apache.org.

only on the top 20 results we check whether both the title and the author field in the retrieved PubMed article match[9] the pair $(au, l_{au}(i))$. If we find such a match, we can also safely associate the ORCID of the author with this article.

After this process, there are articles and validated authors who wrote them. In total, there are 762,508 validated articles (each one stored in an independent file), which form our primary test collection. In fact we have been able to enlarge the initial set of validated (article, author) pairs from 161,609 to a total of 868,498. It is worth mentioning that 642 out of 21,048 authors being considered have not validated articles[10], so finally there are 20,406 authors in our collection. The distribution of the number of articles written by each author follows a power-law distribution, where there are lots of authors with few articles and few authors with lots of articles. For example, there are 1,033 authors who have written a single article, whereas at the other extreme there is a single author who has written 1384 articles.

3.3 Adding Citations

To complete our collection with citations (not available in MEDLINE/PubMed), we have used again the Elsevier's APIs. Starting from the PubMedID of each article in our collection, we needed to obtain an identifier called *eid* by means of the following query:

`http://api.elsevier.com/content/search/scopus?query=pmid(PubMedID)&apikey=YourApiKey&field=eid`

Once obtained the eid for a PubMed article, the list of citations of this article (appearing in Scopus) can be obtained using the following query:

`https://api.elsevier.com/content/search/scopus?query=refeid(eid)&apikey=YourApiKey&field=pubmed-id`

which returns the list of the PubMedIDs of these articles (we only keep those articles belonging to our collection). Finally, each file containing one article in our collection is enlarged by adding information about the articles citing it (PubMedIDs) as well as the corresponding authors (ORCIDs). In this way we can easily build a network of articles (with an arc going from article a to article b if article b is cited by article a) as well as a network of authors (with an arc from author a to author b if in one of the articles written by a is cited an article written by b, this arc can be weighted by the number of such citations).

We were able to consult the citations of 749,811 articles (the remaining articles did not possess an eid), and 509,202 of them had citations in Scopus. Therefore 66.78% of the articles in our collection have citations. The average number of validated citations per article is 7. The total number of citations found is 3,593,931. The number of citations per article also follows a typical power-law

[9] We do not require a perfect match, allowing an edit distance of 5 for title and 3 for author.

[10] This may happen, for example, when the articles (probably only one) in PubMed of an author (having ORCID and ScopusID) do not appear in the list of papers in Scopus written by this author.

distribution, where a few articles have many citations and many articles have few citations. For example, there are 116,365 articles having only one citation, and there is only one article having 713 citations.

4 Using the Collection for Expert Search and Document Filtering

In order to simulate a document filtering scenario, we will consider that each author in the collection is a possible user, the documents to be recommended/filtered are the articles in the collection and the queries representing them may be composed of their abstracts and/or their titles. In the expert finding scenario the experts to be recommended are the authors in the collection and the titles of the articles in the collection may be considered as the simulated topics of interest (queries). Obviously, the data used for gathering information about the authors (in order to learn about their interests and perhaps build their profiles) should be different from the data used for building the queries. In other words, a partition of the collection in training and test sets must be carried out. Given the size of the collection, using for example 80% of the articles for training and 20% for testing, we have more than 150,000 cases in the test set, which is a size large enough to allow extracting statistically significant conclusions from the experiments.

As we mentioned previously, our collection has not explicit relevance judgements stating which are the relevant authors given a query. However, we can establish two levels of implicit relevance judgements: authors and citers. For example, in the context of expert finding, it is reasonable to assume that given a topic of interest represented by an article, then the own authors of this article are relevant experts for the topic. In the context of document filtering, if the document to be filtered is an article (represented by its abstract), we can assume that the authors of this article, together with the authors of other articles citing it are the users interested in this document. It is true that these are not exhaustive relevance judgements, because probably there are more authors interested in the document than just their authors and their citers. Therefore we have an scenario where the implicitly fixed relevance judgements are correct but possibly incomplete. However, there are some previous studies [5,6,11] establishing that this situation of incompleteness of judgements does not represent a problem to reliably compare different systems, provided that the number of queries is large (and in our case it is quite large). For example, Carterette et al. [5] says that "evaluation over more queries with fewer or noisier judgments is preferable to evaluation over fewer queries with more judgments".

4.1 Building a Recommender/Filtering System Through an Information Retrieval System

In order to build a recommender/filtering system we can use essentially two techniques [4,8]: either information retrieval-based (IR) methods or machine

learning-based methods. Focusing on IR methods, we are going to use a rather simple approach (which could serve as baseline for other more sophisticated approaches), where we do not build an explicit profile for each expert/user. Instead, we are going to use an Information Retrieval System (IRS) indexing the (training) collection of articles. Then, following a document model-based approach [2], given a query (either the abstract or the title of a test article), the IRS will return a ranked list of the articles that better match this query. Then we replace each article in the ranking by their associated authors, in order to get a ranking of authors. As this ranking may contain duplicate authors (if two or more articles of the same author appear in the original ranking), we combine all the scores associated to the same author and rerank the list of unique authors according to the combined score. In the experiments reported in this paper we have used the maximum as the combination function.

4.2 Preliminary Results

We randomly partitioned the collection of articles into 80% for training and 20% for testing. To index the abstracts, they were previously preprocessed (removing stopwords and doing stemming). The time required to index the collection using Lucene was less than five minutes. In these experiments we have used the abstracts of the test articles as the queries, and the implementation in Lucene of the classical vector space retrieval model as the IRS.

We have used trec_eval[11], the standard tool used by the TREC community for evaluating an ad hoc retrieval run, which uses the ranked list of authors obtained by the system in response to a query and the relevance judgements corresponding to this query. Concerning this, we have evaluated the system using two different sets of relevance judgements: the authors which are relevant given a query associated to a test article are (1) the own authors of this article and (2) the authors of this article plus the authors of other articles that cite it (the citers).

We will focus on three performance measures: R-Precision (Rprec), Precision at 10 (P_10) and Recall at 10 (recall_10). Table 1 shows these measures for the two sets of relevance judgements.

Table 1. Results of the experiments.

Only authors			Authors + Citers		
Rprec	P_10	recall_10	Rprec	P_10	recall_10
0.5232	0.0927	0.7399	0.4270	0.1490	0.5713

We can observe that the results when using only the own authors as relevant are better than the case where we also use the citers (except in the case of

[11] http://trec.nist.gov/trec_eval/.

P_10). This fact is expected, as in the second case we have a greater number of relevant authors to identify, so that the problem is more difficult in some sense. Obviously, this is not true for P_10, as having more relevant authors also means that it is easier to find more within the top 10 results. In absolute terms, we believe that the results are not bad for a baseline approach. On the average we find one relevant author within the first 10 authors (P_10); also, around 75% of the relevant authors are found within the first 10 results (recall_10); when we recover a number of results equal to the true number of relevant authors (Rprec), we find around 50% of these relevant authors.

5 Other Use Cases of the Collection

- As all the articles have an associated journal, we could use titles and abstracts of articles to build and evaluate recommender systems of scientific journals [13], to help authors to find the more appropriate journals to publish their new papers (as for example Springer Journal Suggester and Elsevier Journal Finder do).
- Using MeshTerms (associated to articles and indirectly to authors) we could evaluate expert profiling techniques. Also (hierarchical and multilabel) text classification methods could be studied.
- As all the authors' names are unambiguously associated to the corresponding ORCIDs, we could study disambiguation methods, trying to distinguish between authors with the same names taking into account the text (title and abstract) of their articles.
- We could use the graphs connecting authors (or articles) by means of citations to enhance the profiles of authors (using information about either the authors or the articles that cite them), to explore hybrid content-based and collaborative recommender systems, or even to explore graph visualization methods. Also considering the citations we could compute some bibliometric measures that could be incorporated to the recommendation model in order to enhance the performance.

6 Concluding Remarks

In this paper we have described the building process of a new test document collection and some preliminary results obtained using it to evaluate expert finding and document filtering methods, although we have also outlined other possible uses of the collection. Our collection starts from the MEDLINE/PubMed collection of scientific articles, but also uses Scopus/Elsevier data with two purposes: disambiguate author names (using ORCID and ScopusID) and adding information about citations to the PubMed records.

Our PMSC-UGR collection is relatively large: it contains 20,406 authors, 762,508 articles and 3,593,931 citations. Although it does not include external relevance judgements, we can use the authors and the citers of articles (with either the titles or the abstracts of these articles acting as queries) to establish

implicit (but incomplete) relevance judgements. As reported in the literature, this incompleteness of the judgements is not a serious problem to compare the performance of competing systems given the large size of the collection (more than 150,000 queries if we use for example a 80%-20% partition in training and test articles).

For future work we plan to use the collection to evaluate different methods for expert finding and document filtering, also taking into account the citation graphs and the MeshTerms. We also plan to make PMSC-UGR available as a community resource.

Acknowledgment. This work has been funded by the Spanish "Ministerio de Economía y Competitividad" under project TIN2016-77902-C3-2-P, and the European Regional Development Fund (ERDF-FEDER).

References

1. Bailey, P., Craswell, N., Soboroff, I., de Vries, A.P.: The CSIRO enterprise search collection. In: SIGIR Forum, vol. 41, pp. 42–45 (2007)
2. Balog, K., Fang, Y., de Rijke, M., Serdyukov, P., Si, L.: Expertise retrieval. Found. Trends Inf. Retrieval **6**, 127–256 (2012)
3. Beel, J., Gipp, B., Langer, S., Breitinger, C.: Research-paper recommender systems: a literature survey. Int. J. Digit. Libr. **17**, 305–338 (2016)
4. Bobadilla, J., Hernando, A., Fernando, O., Gutiérrez, A.: Recommender systems survey. Knowl.-Based Syst. **46**, 109–132 (2013)
5. Carterette, B., Pavlu, V., Kanoulas, E., Aslam, J., Allan, J.: Evaluation over thousands of queries. In: Proceedings of the 31st ACM SIGIR Conference, pp. 651–658 (2008)
6. Carterette, B., Smucker, M.: Hypothesis testing with incomplete relevance judgments. In: Proceedings of the 16th ACM CIKM Conference, pp. 643–652 (2007)
7. Craswell, N., de Vries, A.P., Soboroff, I.: Overview of the TREC 2005 enterprise track. In: Proceedings of the 14th TREC Conference (2005)
8. de Campos, L.M., Fernández-Luna, J.M., Huete, J.F., Redondo-Expósito, L.: Comparing machine learning and information retrieval-based approaches for filtering documents in a parliamentary setting. In: Moral, S., Pivert, O., Sánchez, D., Marín, N. (eds.) SUM 2017. LNCS (LNAI), vol. 10564, pp. 64–77. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67582-4_5
9. Hanani, U., Shapira, B., Shoval, P.: Information filtering: overview of issues, research and systems. User Model. User-Adap. Inter. **11**, 203–259 (2001)
10. Mangaravite, V., Santos, R.L.T., Ribeiro, I.S., Gonçalves, M.A., Laender, A.H.F.: The LExR collection for expertise retrieval in academia. In: Proceedings of the 39th ACM SIGIR Conference, pp. 721–724 (2016)
11. Sanderson, M., Zobel, J.: Information retrieval system evaluation: effort, sensitivity, and reliability. In: Proceedings of the 28th ACM SIGIR Conference, pp. 162–169 (2005)
12. Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., Su, Z.: ArnetMiner: extraction and mining of academic social networks. In: Proceedings of the 14th ACM SIGKDD Conference, pp. 990–998 (2008)
13. Wang, D., Liang, Y., Xu, D., Feng, X., Guan, R.: A content-based recommender system for computer science publications. Knowl.-Based Syst. **157**, 1–9 (2018)