Extracting Nanopublications from IR Papers*

Aldo Lipani ✉, Florina Piroi, Linda Andersson, and Allan Hanbury

Institute of Software Technology and Interactive Systems (ISIS)
Vienna University of Technology, Austria
{surname}@ifs.tuwien.ac.at

Abstract. The published scientific results should be reproducible, otherwise the scientific findings reported in the publications are less valued by the community. Several undertakings, like myExperiment, RunMyCode, or DIRECT, contribute to the availability of data, experiments, and algorithms. Some of these experiments and algorithms are even referenced or mentioned in later publications. Generally, research articles that present experimental results only summarize the used algorithms and data. In the better cases, the articles do refer to a web link where the code can be found. We give here an account of our experience with extracting the necessary data to possibly reproduce IR experiments. We also make considerations on automating this information extraction and storing the data as IR nanopublications which can later be queried and aggregated by automated processes, as the need arises.

1 Motivation

A core activity of Information Retrieval (IR) research is the experimental work that consists of testing and tuning software systems to extract and output information related to a given input. Every year, new results on various retrieval algorithms are published, articles describing the experimental results in varying degrees of detail. A large amount of data and corresponding data analysis is generated for each such experiment. If interested to reproduce an IR experiment, the necessary details to implement and prepare the experiment are often not easy to obtain.

Reproducible research and experiment replication is already a known issue which is specifically addressed in various research areas like statistics [12], life sciences (e.g. [9]), computational linguistics [10], or programming [14]. In the IR evaluation community, even though researchers are aware of this issue, the subject is not being dealt with in a systematic way. A few island solutions to publish evaluation experiments are available. EvaluatIR is a system where scientists can upload their experiments on a standard (TREC) test collection to compare them

* This research was partly funded by the Austrian Science Fund (FWF) project number P25905-N23 (ADmIRE).

with existing experiments [2]. A permanent URL also gives later access to the details of the uploaded experiment, which could, thus, be used in citations. DIRECT [6] has been used to manage the CLEF evaluation campaigns. It evolved to include a library of CLEF experiments that, in theory, could be citable and re-used.

Both of the mentioned systems concentrate mostly on the experimental data, with very little information about the necessary software setup, like the parameters of the indexing components or of the result ranking component, that would allow a researcher to replicate the experiment. RunMyCode, myExperiment, and OpenML are examples of environments dedicated to sharing not only data but also to sharing code and algorithms. RunMyCode is a computational infrastructure created to give researchers the space where experimental data, code and other data associated with an article can be published [13]. The open science platform for machine learning, OpenML, allows users to submit their results, code, or data, which is then organized such that it can be later searched and re-used via web APIs [11]. myExperiment, a Virtual Research Environment, uses a social web approach to the sharing and the collaboration on experiments and workflows [5], as well as to the publication of citable Research Objects with executable components [4].

Despite these impressive efforts, collecting results for a larger number of IR experiments, over a period of time, to establish genuine advances in this research area, is strenuous. Also, the connection to research publications that describe the experiments is not evident, and experiment provenance, in absence of an explicitly stated author or research institution, is often lost. Additionally, at least for publications in the IR evaluation domain, the experiment definitions are often incomplete, system definitions are omitted or incomplete as authors consider them implicit or uninteresting, or due to lack of printed space.

Ideally, publications describing IR systems, data, and experiments are accompanied by further data items describing the particular implementation details relevant to anyone that wishes to reproduce the reported experiments. One proposal to make such details available to the community is via IR nanopublications [7], which give researchers the possibility to publish additional details about their experimental data, details which do not have space or would not fit conceptually in a printed publication. Nanopublications give researchers the opportunity to share statements about data, code, and experiments, annotated with references, and to establish authorship, in a citable way. Nanopublications can be, later, automatically queried and explored to gain further insights into the IR domain.

But plenty of important and valuable research is already published in conference proceedings, workshops, or journals. We would like to make that data available and machine processable. We explore in this paper ways to extract data from publications, data that would support reproducing IR experiments, and store it as IR nanopublications. We describe our efforts in processing a small set of IR workshop papers to make initial observation sets that can be later evolved into automatic data extraction from IR papers.

2 Manual Nanopublication Extraction

Our experimental setting is that of a PhD student reading research papers describing IR systems and experiments, who tries to extract the necessary information to reproduce the retrieval experiments reported in the paper.

For this experiment, we chose to manually examine the workshop proceedings of the first mathematical retrieval challenge, NTCIR-10 Math Task [1]. This task is dedicated to exploring the information retrieval solution space for information access in a corpus with a high level of mathematical content. The NTCIR-10 Math test collection contains 100.000 documents from mathematics, physics and computer science. The queries contained not only keywords, but also mathematical formulae with wildcards as query variables. The retrieval solution chosen by the participants to this challenge included creating specific indexing and searching algorithms for the mathematical formulae in the corpus.

The Math pilot task had two subtasks, the Math Retrieval subtask and the Math Retrieval Understanding subtask. The first one is a question-and-answer task that has three different search scenarios: Formula Search where the query is a formula, Full-Text Search where the query is defined as a combination of keywords and formulae and Open Information Retrieval where a query is a free text that includes request expressed in natural language and formulae.

There were six groups that submitted experiments to this task, and we took a close look at their workshop notes[1]. Our interest was to extract all the reported details of the retrieval systems so that—should the need or wish arise—we could reproduce the systems and the experiments.

Since this was a pilot task, i.e. no previous editions of such an evaluation challenge were organised, there were no previously available relevance assessments, and no training data. Due to the specificities of the corpus data and of the given tasks, the participating teams addressed the subtasks with conceptually very different solutions, making the systems a good candidate for an initial observation set in our experiments. For example, in the Full Text Search subtask, one team used an extended Boolean retrieval model and then an exact match model in order to push up the documents that have the formula in the query, while another team used a non-indexed search model.

Generally, in our view, the architecture of an IR system has the following components: A document preprocessor, DP, an indexer, I, a search module, or a scorer, S, a query (pre)processor, QP, a reranker of the search results, R, a collection (pre)processor, CP, a result merger, M (see also Figure 1). Each IR system has a set of core components, which are marked on the figure with the label IRSystemCore. In this example, the core components include a ranker of the search results. An IR system may make use of several sets of core components, in which case a result merging component and possibly a re-ranking component are included in the IR system's flow.

[1] http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings10/NTCIR/toc_ntcir.html

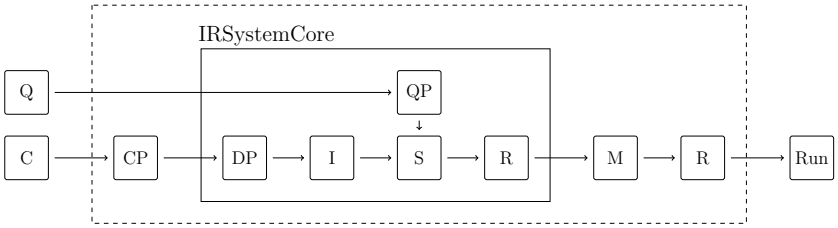

Fig. 1. Model of a retrieval system

We examine, thus, the selected workshop papers looking specifically for descriptions of the components in Figure 1. While collecting the information subject to our interest we made use of an IR Ontology designed by us. The ontology is used to describe the concepts involved in the IR system evaluation research [7]. From the three ontology categories, `EvaluationThing`, `IRSystemThing`, and `IRModelThing`, we use the concepts of the `IRSystemThing` and `EvaluationThing` categories to guide our manual information extraction from the workshop papers. We looked to find instances for the concepts described by the `Indexer`, `Preprocessing`, `Reranker`, `Merger`, `Scorer`, `Run`, `TestCollectionComponent` and `Topics` classes[2]. Figure 2 shows only the subset of the ontology that we use to identify the instances occurring in the papers. Close to the ontology classes, on their right side, we show the IR System component that we associate with the respective ontology class. To store the information extracted from the workshop papers we use the IR nanopublication format shortly described in [7]. A nanopublication is stored as one or more RDF named graphs which are RDF triples with an associated context. Each of the main components of a nanopublication, assertion, provenance, and publication details, is again defined as a named graph. To store the RDF triples we choose AllegroGraph[3] which is an environment optimized for storing graphs. Using a web interface to our AllegroGraph server instance[4] we can query the RDF triple storage using the SPARQL query language.

For the papers in our observation set we set out to extract the information describing the IR system components and create one nanopublication for each paper in the set.

This was done by the first author who achieved this in a two phase process. In the first phase an automated annotation of the IR system components was done, with the help of preliminary versions of the tools described in Section 3, used as standalone components. The annotation, however, did not detect all the instances and relationships of interest, therefore further manual work was required, which was done in the second phase.

[2] The ontology is available at: http://ifs.tuwien.ac.at/~admire/ir_ontology/ir
[3] http://franz.com/agraph/allegrograph/
[4] http://ifs.tuwien.ac.at/~admire/ir_ontology/agwebview/

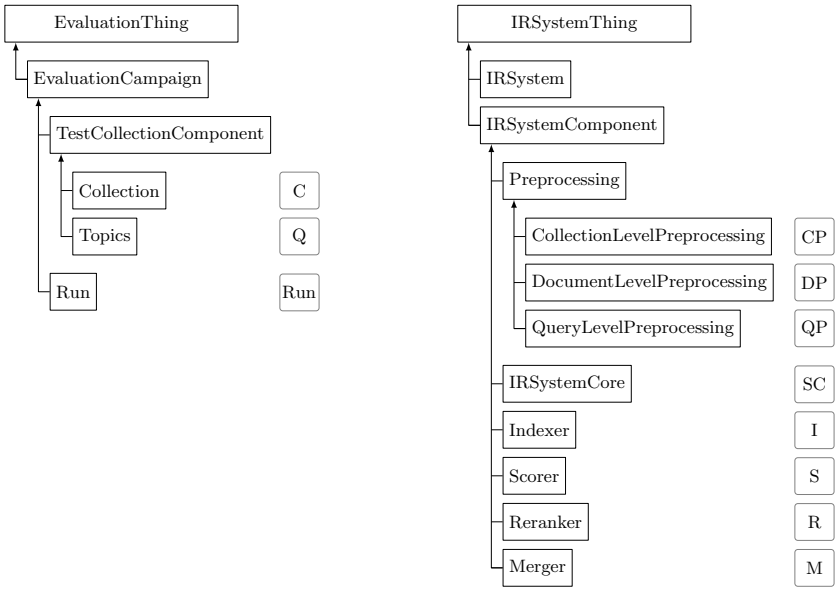

Fig. 2. IR Ontology Subset

In the second phase we defined a protocol that we applied, refined, and re-applied in our information extraction process, until we were satisfied with the outcome. The protocol contains three steps: the first one necessitated us to annotate all the potentially relevant information about the IR system described in the paper examined; the second step was dedicated to connecting the IR ontology classes correctly to the IR components identified in the paper; the final step of the protocol was to interconnect the system components identified in the previous steps, which actually translated to extracting relationships between the identified components, relationships defined in the IR ontology as class relationships.

We used the protocol described above iteratively, each iteration making use of the knowledge gained in the previous steps to extract more detailed information.

Extracting the information about the IR experiments (run descriptions) was straightforward, the test collection, the evaluation measures, the queries, the participant runs, etc. being clearly described in the task overview paper [1]. Identifying and extracting the IR system component descriptions was, however, more challenging.

We recall that one of the goals of this manual extraction was to establish whether a researcher would be able to reproduce the experiments reported in the papers. Being able to get enough information about the IR systems used is a prerequisite, and, during the information extraction described above, we collected availability information on each of the IR system components we looked for (those shown in Figure 1).

	CP	$SC_{formula}$					SC_{text}					M	R
		DP	QP	I	S	R	DP	QP	I	S	R		
1	∘	✓	✓	✓	✓	∘	•	•	•	•	•	✓	∘
2	∘	✓	✓	✓	✓	∘	✓	✓	✓	✓	∘	•	∘
3	•	•	•	•	•	•	∘	∘	∘	∘	∘	∘	∘
4	✓	✓	✓	✓	✓	∘	•	•	•	•	•	•	•
5	•	•	•	•	•	•	•	•	•	•	•	•	•
6	✓	✓	✓	•	•	∘	∘	∘	∘	∘	∘	∘	∘

Table 1. Availability of the component description for each IR system

Table 1 shows the result of our analysis. In this table each column represents an IR system component and each row represents a paper. Most of the retrieval solutions presented in the examined papers involve two IR system cores (SC)—one for the textual retrieval, one for the formula retrieval—which are corespondingly grouped in the table. The markings in each table cell indicate if, within the paper, there is enough information to reproduce the IR component (✓), if there is not enough information (•) or if the described IR system does not need that IR component (∘). Thereby, an IR system is reproducible if all of its components are reproducible.

What we notice out of Table 1 is that, for this observation set, the IR systems involved in the task participation are under-explained, and a researcher would not be able to re-implement whole systems without further digging into references, sending e-mails to researchers, etc.

3 Automating the Nanopublication Extraction

Manual extraction from papers of the information necessary to repeat some IR experiment is feasible when only a few papers are to be studied, like the case described in the previous section. If we are interested in developments over time, or comparisons of a large number of experiments, manual information extraction is tedious.

To avoid such tedious manual work, we devised a workflow that can be used to automate the extraction of IR related information from IR papers. The ultimate goal of this automated process is to have all information of interest stored in a machine processable format, for example as an IR nanopublication.

Figure 3 presents the main phases of the proposed workflow. We assume that the input to our workflow, the published articles, are stored in PDF format. First, we need to apply a PDF extractor which will transform the content of the PDF file into a format that can be handled by the tools that will be applied in the next steps. Then, a series of simple Natural Language Processing (NLP) tools are applied in order to identify sentences, parts of speech, tokens and chunks. The result of this processing forms the input to an Ontology-based Information Extraction (OBIE) component that iteratively fills the IR ontology with class instances. The last component of the workflow is a nanopublication extractor

that uses nanopublication templates and the IR ontology. The nanopublications thus created can be stored on a server that supports querying them.

In the following we describe the details and challenges of these phases:

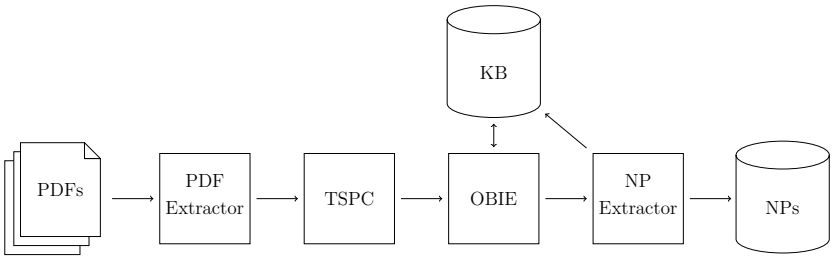

Fig. 3. Flow of the nanopublication extraction system

PDF Extractor: The papers are collected in PDF format are translated into XML format, following the Journal Article Tag Suite[5] (JATS) schema, with the open source toolkits like PDFBox of the Apache project. Using the JATS schema, images and tables are also extracted from the PDF files.

Applying NLP Tools, TSPC: The XML files extracted in the previous step are now passed to a series of NLP tools: **T**okeniser, **S**entence splitter, **P**art of Speech (PoS) Tagger, and a **C**hunker (TSPC). For this step we use the GATE Developer integrated environment annotation and NLP analysis framework [3], which is used also in the next step.

Ontology Based Information Extraction, OBIE: After the standard NLP annotations of the previous step, an ontology based information extraction [8] should be applied in order to enrich the existing IR ontology. Automatic ontology population is related to methods used in Information Extraction (IE) which extract pre-defined relations (defined in the IR ontology, in our case) from text. OBIE techniques are used to enhance domain knowledge or customized ontologies. The techniques involve the identification of different named entity types, of technical terms or relations. After normalisation of data, tokenisation, PoS tagging, etc., a recognition step follows where gazetteers can be combined with rule-based grammars, ontology design patterns, or by identifying pattern slots such as lexico-syntactic patterns.

In the training phase this is an iterative process where, in the first iteration, we identify instances and named entities (NE) for the IR ontology classes. In order to learn new named entities we use coreference resolutions to identify the attributes of each NE instantiating an IR ontology class and extract further inter-term relationships such as hyponyms and hypernyms. When new NEs are discovered, the knowledge base KB (which could be, for example, the IR ontology) is updated and the process restarts extracting new NEs. The process is repeated until a previously agreed on quality standard is reached. Updating

[5] http://jats.nlm.nih.gov

the knowledge base can be done with a tool like OwlExport [15] which is developed as a GATE resource to map existing NLP analysis pipelines into an OWL ontology.

The development of this workflow component must incorporate our experiences gained during the information extraction described in Section 2, particularly the steps of the protocol used in the second phase.

Nanopublication extractor: The last phase of the process is extracting nanopublications from either previously created knowledge bases (e.g., IR ontology) or from annotations created with GATE. Several nanopublication templates can be used in the extraction, depending on the information sought.

The nanopublications thus extracted are to be stored, then, into a searchable repository. For example, in our experiments we can use the query listing below to extract one nanopublication from our local AllegroGraph installation. The SPARQL query[6] asks to retrieve the anonymous graph of the nanopublication stored within the context defined by the unnamed prefix, and then, using its properties (hasAssertion, hasProvenance and hasPublicationInfo) asks to retrieve the definition of the three sub graphs and their content.

```
prefix : <http://ifs.tuwien.ac.at/~admire/nanopubs/LarsonEtAl2013#>
prefix np: <http://www.nanopub.org/nschema#>
select ?G ?S ?P ?O where {
  {graph ?G {: a np:Nanopublication}} union
  {: np:hasAssertion ?G} union
  {: np:hasProvenance ?G} union
  {: np:hasPublicationInfo ?G}
  graph ?G {?S ?P ?O}
}
```

Listing 1.1. SPARQL query

4 Conclusions and Future Work

We have presented in this paper our experience with manually processing a small set of papers describing IR systems in order to extract information necessary for experiment replication. This being a tedious work, we also present a methodology to automate this process, where the final output is to be a large collection of IR nanopublications, stored in a database.

Our work is currently at an early stage. Extending the, now partial, implementation of the workflow described in Section 3 has a high priority in our future work. We plan to use the the manually created nanopublications as ground truth in the experiments evaluating the workflow and its concrete components, moving towards quantitative experiments with other available corpora in the IR domain.

One of the most important issues for the scope of this paper is that the IR systems referred to in IR papers are under-described. There are generally too few details in the publication to allow a researcher to reproduce the reported experiments without further digging for system details. We recommend that

[6] http://ifs.tuwien.ac.at/~admire/nanopubs/queries/LarsonEtAl2013

researchers make the 'boring', non-publishable details of their work available to the community. This can be done by publishing their code, or by publishing additional data attached to the paper. It remains to be experimented with the type and amount of data attached to a paper such that reproducibility is made easy for the IR educated researcher. We are convinced that IR nanopublications can play a strong role in formalizing the description of IR systems and hence increase the availability and reproducibility of the research results.

References

1. A. Aizawa, M. Kohlhase, and I. Ounis. NTCIR-10 math pilot task overview. In *Proceedings of the 10th NTCIR Conference, Tokyo, Japan*, 2013.
2. T.G. Armstrong, A. Moffat, W. Webber, and J. Zobel. EvaluatIR: An Online Tool for Evaluating and Comparing IR Systems. In *Proceedings of the 32Nd International ACM SIGIR Conference*, SIGIR '09, pages 833–833, New York, NY, USA, 2009. ACM.
3. K. Bontcheva, V. Tablan, D. Maynard, and H. Cunningham. Evolving GATE to Meet New Challenges in Language Engineering. *Natural Language Engineering*, 10(3/4):349–373, 2004.
4. D. De Roure. Towards computational research objects. In *Proceedings of the 1st International Workshop on Digital Preservation of Research Methods and Artefacts*, DPRMA '13, pages 16–19. ACM, 2013.
5. D. De Roure, C. Goble, and R. Stevens. The design and realisation of the virtual research environment for social sharing of workflows. *Future Generation Computer Systems*, 25(5):561–567, 2009.
6. M. Dussin and N. Ferro. DIRECT: applying the DIKW hierarchy to large-scale evaluation campaigns. In R.L. Larsen, A. Paepcke, J.L. Borbinha, and M. Naaman, editors, *Proceedings of JCDL*, page 424, 2008.
7. A. Lipani, F. Piroi, L. Andersson, and A. Hanbury. An Information Retrieval Ontology for Information Retrieval Nanopublications. In *Proceedings of the CLEF 2014 Conference*. Springer, 2014. to appear.
8. D. Maynard, Y. Li, and W. Peters. NLP techniques for term extraction and ontology population. In *Proceeding of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, pages 107–127, 2008.
9. A. Nekrutenko and J. Taylor. Next-generation sequencing data interpretation: enhancing reproducibility and accessibility. *Nat Rev Genet*, 13(9):667–672, 2012.
10. T. Pedersen. Empiricism is not a matter of faith. *Computational Linguistics*, 34(3):465–470, 2008.
11. J. N. van Rijn, B. Bischl, L. Torgo, B. Gao, V. Umaashankar, S. Fischer, P. Winter, B. Wiswedel, M. R. Berthold, and J. Vanschoren. OpenML: A collaborative science platform. In H. Blockeel, K. Kersting, S. Nijssen, and F. Železný, editors, *Machine Learning and Knowledge Discovery in Databases*, number 8190 in Lecture Notes in Computer Science, pages 645–649. Springer Berlin Heidelberg, 2013.
12. V. Stodden. The reproducible research movement in statistics. *Statistical Journal of the IAOS: Journal of the International Association for Official Statistics*, 30(2):91–93, 2014.
13. V. Stodden, C. Hurlin, and C. Perignon. RunMyCode.org: A novel dissemination and collaboration platform for executing published computational results. Technical Report ID 2147710, Social Science Research Network, 2012.

14. J. Vitek and T. Kalibera. Repeatability, reproducibility and rigor in systems research. In *2011 Proceedings of the International Conference on Embedded Software (EMSOFT)*, pages 33–38, 2011.
15. R. Witte, N. Khamis, and J. Rilling. Flexible ontology population from text: The OwlExporter. In N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner, and D. Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA), 2010.